

**Testovanie používateľského zážitku pomocou
sledovania pohľadu**

Dokumentácia riadenia

Vedúci tímu: Ing. Róbert Móro

Členovia tímu: Bc. Peter Kiš, Bc. Matej Kucek, Bc. Viktória Lovasová,
Bc. Peter Kušnír, Bc. Marek Šulek, Bc. Šimon Valíček, Bc. Martin Svrček

Akademický rok: 2014/2015

Obsah

1	Úvod.....	1
1.1	Roly členov a podiel práce	1
1.1.1	Manažérske činnosti členov tímu	1
1.1.2	Podiel práce na dokumentácii	2
1.2	Aplikácia manažmentov	2
1.3	Sumarizácia šprintov	3
1.3.1	Šprint 1	4
1.3.2	Šprint 2	5
1.3.3	Šprint 3	7
1.3.4	Šprint 4	9
1.3.5	Šprint 5	11
1.3.6	Šprint 6	13
1.3.7	Šprint 7	14
1.3.8	Šprint 8	16
1.3.9	Šprint 9	16
1.4	Použité metodiky	17
1.5	Globálna reprotrospektíva.....	18
1.5.1	Aspekty projektu, s ktorými sme boli spokojní.....	18
1.5.2	Aspekty projektu, ktoré treba zmeniť?.....	19
1.5.3	Nové a potrebné aspekty projektu	19
2	Zoznam kompetencii tímu.....	21
3	Metodiky	22
3.1	Manažment riešenia chýb	22
3.1.1	Úvod.....	22
3.1.2	Vytvorenie nového objektu – chyby v TFS	22
3.1.3	Odstránenie chyby.....	24
3.1.4	Zrevidovanie opravenej chyby	25
3.1.5	Nastavenie stavu chyby ako zatvorený	25
3.2	Dekompozícia používateľských príbehov na úlohy v nástroji TFS	26
3.2.1	Úvod.....	26
3.2.2	Metóda funkcionálnej dekompozície používateľských príbehov.....	27
3.2.3	Riešenie špeciálnych udalostí.....	31
3.2.4	Zoznam bežne chýbajúcich úloh	32

3.3	Metodika používania softvérového nástroja na správu verzií zdrojového kódu	33
3.3.1	Vymedzenie pojmov	33
3.3.2	Preambula.....	33
3.3.3	Metodické pokyny k odovzdávaniu zdrojového kódu	33
3.3.4	Metodické pokyny k vetveniu zdrojového kódu	34
3.3.5	Súvisiace zdroje.....	35
3.4	Metodika prehliadky kódu.....	36
3.4.1	Úvod.....	36
3.4.2	Prehliadka kódu.....	36
3.5	Písanie zdrojových kódov v jazyku C#	41
3.5.1	Predslov	41
3.5.2	Pri tvorbe názvoslovía	41
3.5.3	Pri písaní komentárov.....	41
3.5.4	Formátovanie zdrojových textov	42
3.5.5	Používanie dátových typov... ..	43
3.5.6	Ukázkový zdrojový kód	44
3.6	Metodika integrácie	45
3.6.1	Vymedzenie metodiky.....	45
3.6.2	Požiadavky pred začiatkom integrácie	45
3.6.3	Postup.....	45
3.7	Metodika testovania.....	49
3.7.1	Vymedzenie pojmov	49
3.7.2	Predslov	49
3.7.3	Vytvorenie projektu s testami	49
3.7.4	Písanie testov.....	51
3.7.5	Spúšťanie testov	53
4	Export evidencie úloh.....	55
4.1	Šprint 1	55
4.2	Šprint 2	56
4.3	Šprint 3	59
4.4	Šprint 4	63
4.5	Šprint 5	64
4.6	Šprint 6	65
4.7	Šprint 7	66
4.8	Šprint 8	67

4.9	Šprint 9	67
5	Webové sídlo projektu	68

Príloha A: Testovanie v jazyku Javascript – JASMINE

Príloha B: Zápisy zo stretnutí

Príloha C: Analýza pýtača v ALEFe

Príloha D: Experiment

1 Úvod

Tento dokument predstavuje podrobnú dokumentáciu k riadeniu tímového projektu Testovanie používateľského zážitku pomocou sledovania pohľadu, ktorému sa venuje sedemčlenný tím študentov s názvom Team Focus.

Projekt nadväzuje na prácu minuloročného tímu Carrots, ktorý v rámci projektu Sledovanie pohľadu pri používaní aplikácií vytvoril základnú infraštruktúru systému pomenovanom TrackView. Nakoľko sa minuloročnému tímu podarilo vytvoriť prostredia na správu používateľov, projektov a sedení, ako aj infraštruktúru pre zber a spracovanie dát z eyetrackerov, rozhodli sme sa v našom projekte zamerať na problematiku zjednodušovania, vyhodnocovania a vizualizácie zbieraných dát. Rovnako tak sme si dali za cieľ vytvoriť prepojenie nami vyvíjaného systému s existujúcimi webovými stránkami a systémami pomocou nášho API. Toto prepojenie hodláme demonštrovať na fakultnom edukačnom webovom systéme Alef.

Nasledujúce kapitoly popisujú Team Focus a jeho snahu o prácu v agilnom prístupe vývoja softvéru. Úvodná kapitola popisuje jednotlivých členov tímu, ich manažérske role a postupy, ktorými sa snažia riadiť pri vývoji. Rovnako sú v prvej kapitole popísané jednotlivé šprinty, silné a slabé stránky tímu pri dodržiavaní agilného prístupu, ako aj vízia riešenia akútnych problémov. Druhá kapitola pojednáva o kompetenciách jednotlivých členov tímu, ich programátorských zručnostiach, odborných schopnostiach, ako aj voľnočasových aktivitách. V tretej kapitole sú uvedené všetky metodiky, podľa ktorých sa Team Focus riadi pri vývoji, testovaní a údržbe vyvíjaného softvéru. Posledná kapitola obsahuje vyexportované backlogy tímu pre všetky dokončené šprinty.

1.1 Roly členov a podiel práce

V prvej časti kapitoly sme zaznamenali jednotlivé roly členov nášho tímu, tak ako sme si ich určili na začiatku projektu. V druhej časti sú opísané podiely práce členov tímu na jednotlivých častiach dokumentácie.

1.1.1 Manažérske činnosti členov tímu

V našom tíme máme jednotlivé činnosti rozdelené nasledovne:

- *Peter Kiš* - vedúci tímu, organizácia komunikácie v tíme, informovanie učiteľa o stave projektu
- *Matej Kucek* - zástupca vedúceho, udržiavanie informácií o stave projektu, integrácia softvéru
- *Marek Šulek* - monitorovanie, prehliadky vytváraného výsledku
- *Šimon Valíček* - manažment verzií, organizácia zdrojov, definovanie architektúry a rozsahu projektu
- *Viktória Lovasová* - riadenie procesu dokumentovania, identifikácia a riadenie chýb v softvéri
- *Martin Svrček* - plánovanie pre tím a jednotlivých členov tímu, vyhodnocovanie plnenia plánu a návrh úprav, identifikovanie a riadenie rizík

- *Peter Kušnir* - udržiavanie informácií o stave projektu, evidencia úloh, vyhodnocovanie testov

1.1.2 Podiel práce na dokumentácii

Dokumentácia riadenia

Tabuľka 1 znázorňuje prácu členov na jednotlivých kapitolách v dokumente Dokumentácia riadenia.

Tabuľka 1: Podiel práce na dokumentácii.

Úvod	Peter Kiš
Role členov tímu a podiel práce	Každý individuálne
Aplikácia manažmentov	Martin Svrček
Sumarizácia šprintov	Martin Svrček
Použité metodiky	Martin Svrček
Globálna retrospektíva	Martin Svrček
Zoznam kompetencií tímu	Peter Kiš
Metodiky	
Metodika písania zdrojových kódov v C#	Peter Kiš
Metodika integrácie	Matej Kucek
Metodika prehliadok kódu	Marek Šulek
Metodika manažmentu verzií	Šimon Valíček
Metodika manažmentu chýb	Viktória Lovasová
Metodika dekompozície používateľských príbehov na úlohy	Martin Svrček
Metodika testovania	Peter Kušnir
Export evidencie úloh	Matej Kucek

Dokumentácia inžinierskeho diela

Tabuľka 2 znázorňuje prácu členov na jednotlivých kapitolách v dokumente Dokumentácia riadenia.

Tabuľka 2: Podiel práce členov tímu na dokumente.

Úvod	Marek Šulek
Globálne ciele	Martin Svrček
Celkový pohľad na systém	Šimon Valíček
Modul API	Šimon Valíček
Knižnice JS	Martin Svrček
Agregácie	Peter Kiš, Matej Kucek

1.2 Aplikácia manažmentov

V rámci projektu máme rozdelené jednotlivé role a každý člen má za úlohu vytvoriť metodiky spadajúce do danej oblasti, ktoré nám pomôžu pracovať efektívne a kvalitne. V súčasnom

stave sme fungovali skôr na báze ústnej dohody ako sa budú jednotlivé procesy robiť, keďže ešte neboli spísané všetky metodiky.

V rámci manažmentu kvality bolo našou snahou vytvárať kód, ktorý bude čo najviac prehľadný a čitateľný a hlavne bez zbytočných chýb. V rámci celého projektu sa chceme zamerať práve na kvalitu kódu možno aj na úkor kvantity. V rámci tejto oblasti sa preto snažíme ku každej ucelenej funkcionalite vždy vykonávať aj jednotlivé úlohy zabezpečujúce kvalitu softvéru ako sú prehliadky kódu a testovanie. Toto pomerne jasne dokazujú úlohy zaznamenané v systéme TFS.

V kontexte určitých praktík v písaní kódu sme spoločne určili človeka, ktorý má pomerne dlhé skúsenosti s prácou v jazyku C# aby vypracoval metodiku na písanie kódu, ktorou sa riadime (viď 3.5 Metodika písania kódu v jazyku C#).

Pri práci na projekte využívame verziovací nástroj *Git*. Tento nástroj umožňuje určité špecifické aktivity ako vytváranie rôznych vetiev kódu, *commit* zmien v kóde a podobne, ktoré je však možné robiť rôznymi spôsobmi. Preto je veľmi dôležité, aby sme mali nejaké spoločné postupy, ktoré bude každý uplatňovať (napríklad aj pri názvoch), a bude tak jednoduché orientovať sa v rôznych verziách kódu. V súčasnosti sa snažíme pri spomínaných aktivitách využívať všeobecné postupy a vytvárať názvy, ktoré budú prehľadne opisovať danú vetvu alebo commit. V tomto kontexte už existuje aj metodika, ktorá opisuje všetky tieto podrobnosti (viď 3.6 Metodika manažmentu verzií).

V kontexte plánovania využívame všeobecné postupy (best practices) metodiky SCRUM pre kvalitné plánovanie jednotlivých šprintov. V tíme využívame nástroj TFS, do ktorého okrem iného zaznamenávame jednotlivé úlohy a ich plnenie. Jednou z dôležitých aktivít je dekompozícia používateľských príbehov, pri ktorej ako tím pomerne dobre stanovujeme jednotlivé úlohy. Toto je pekne vidieť aj na exporte úloh z TFS, kde sa často vyskytuje pomerne rovnaký vzor úloh:

- Analýza problematiky
- Implementácie v niekoľkých krokoch
- Prehliadka kódu a testovanie

Aj v tejto oblasti však už existuje metodika (viď 3.2 Metodika dekompozície používateľských príbehov na úlohy).

V rámci testovania musíme priznať, že sa nám príliš nedarí testovať náš kód. Je to hlavne z dôvodu malej skúsenosti s takýmto prístupom. Tento aspekt nám určite pomôžu zlepšiť aj jednotlivé dokumentácie k testovaniu (viď príloha A: Testovanie v jazyku javascript - JASMINE) a metodiky (viď 3.7 Metodika testovania).

Jednou z dôležitých aktivít je aj riešenie chýb, ktoré sa objavia. Je nutné postupovať určitým spôsobom, ktorý bude každý poznať a zabezpečí čo najlepšie vyriešenie daného problému (viď 3.13.1 Manažment riešenia chýb).

1.3 Sumarizácia šprintov

V tejto kapitole sú opísané a zhodnotené jednotlivé šprinty. Venujeme sa hlavne celkovému procesu práce na nich a splneniu všetkých stanovených úloh.

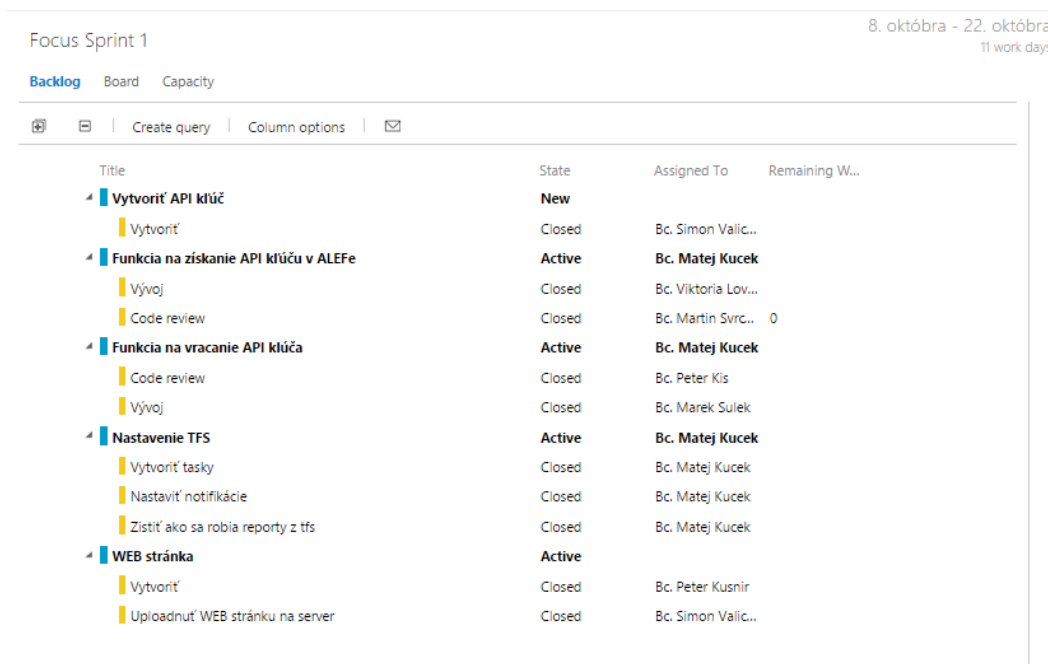
1.3.1 Šprint 1

Prvý šprint bol pre náš tím pomerne špeciálny, keďže pokračujeme v práci z minulého roku a základná infraštruktúra systému na sledovanie pohľadu už existuje.

Na jednej strane bolo preto našou úlohou hlavne pochopiť súvislosti a fungovanie aktuálneho stavu systému tak, aby sme mohli plynule pokračovať v jeho vývoji. Tomuto boli prispôsobené aj jednotlivé používateľské príbehy a súvisiace úlohy.

Na strane druhej bol tento šprint pre nás aj zoznamovacou fázou, keďže sme sa predtým niektorý príliš nepoznali.

Ako sme už spomenuli v rámci šprintu sa riešili základné úlohy na oboznámenie sa so systémom *Obrázok 1*.



Focus Sprint 1 8. októbra - 22. októbra
11 work days

Backlog Board Capacity

Create query | Column options | [Envelope icon]

Title	State	Assigned To	Remaining W...
▲ Vytvoriť API kľúč	New		
Vytvoriť	Closed	Bc. Simon Valic...	
▲ Funkcia na získanie API kľúčů v ALEFe	Active	Bc. Matej Kucek	
Vývoj	Closed	Bc. Viktoria Lov...	
Code review	Closed	Bc. Martin Svrc... 0	
▲ Funkcia na vracanie API kľúčů	Active	Bc. Matej Kucek	
Code review	Closed	Bc. Peter Kis	
Vývoj	Closed	Bc. Marek Sulek	
▲ Nastavenie TFS	Active	Bc. Matej Kucek	
Vytvoriť tasky	Closed	Bc. Matej Kucek	
Nastaviť notifikácie	Closed	Bc. Matej Kucek	
Zistiť ako sa robia reporty z tfs	Closed	Bc. Matej Kucek	
▲ WEB stránka	Active		
Vytvoriť	Closed	Bc. Peter Kusnir	
Uploadnúť WEB stránku na server	Closed	Bc. Simon Valic...	

Obrázok 1: Šprint 1 backlog.

Pretože sme ešte nepoznali nástroj TFS bolo nutné prácu s ním analyzovať a z tohto dôvodu sme ho vlastne pri zaznamenávaní práce na šprinte č.1 ani nepoužili. Všetky úlohy sme pridali do systému až na konci, keď už boli hotové čo zapríčinilo, že náš burndown graf je prázdny a neuvádzali sme ho preto ani v dokumente. V rámci prvého šprintu sa nám však podarilo spoločnými silami spraviť všetky úlohy. Jediným problémom bolo nasadenie stránky v rámci termínu pre predmet Tímový projekt. Tento problém bol však na strane servera a súvisel s presmerovaním stránky tímu z minulého roku.

V retrospektíve tohto šprintu sme určili, že bude potrebné hlbšie pochopiť problematiku celého systému, pretože väčšina z nás nemá vôbec predstavu ako to celé funguje vo vnútri. V tomto kontexte sme sa dohodli, že tomuto prispôbíme aj nasledujúce úlohy.

Na druhej strane sme sa zhodli, že sme sa počas tohto šprintu ako tím pomerne dobre spoznali. Odcenili sme takisto aj vzájomnú pomoc na úlohách.

1.3.2 Šprint 2

Na základe zistení zo šprintu 1 sme si stanovili, že vzhľadom na zložitost' systému bude potrebné venovať sa ešte hlbšie jednotlivým častiam systému prostredníctvom jednoduchších implementácií. S ohľadom na zistenia zo stretnutia so zákazníkmi sme si teda stanovili nasledujúce úlohy na ďalší šprint.

Focus Sprint 2 23. októbra - 5. novembra
10 work days

Backlog Board Capacity

Create query Column options

Title	State	Assigned To	Remaining W...
Pridanie sledovania pohľadu do administrácie	New	Bc. Matej Kucek	1
Code review	New		1
Vytvoriť ikonu na prepnutie do pohľadu administrácie	Closed	Bc. Matej Kucek	0
Commitnúť zmenu	Closed	Bc. Matej Kucek	0
Otestovanie	Closed	Bc. Matej Kucek	0
Volanie z ALEFu	New	Bc. Viktoria L...	1
Upravenie autentifikácie	Active	Bc. Viktoria Lov...	1
Autentifikácia cez API kľúč	New	Bc. Marek Sulek	9
Overenie API kľuca v DB	Closed	Bc. Marek Sulek	9
Pocuvanie eventov cez backbone JS a zvyraznenie elementov	Active	Bc. Martin Sv...	4
Analýza Backbone JS	Closed	Bc. Martin Svrc...	0
Implementácia časti na počúvanie/odchytávanie eventov	Closed	Bc. Martin Svrc...	1
Analýza práce s javascriptom	Closed	Bc. Martin Svrc...	1
Zvýraznenie elementu, na ktorý sa pozeráme	Closed	Bc. Martin Svrc...	1
Code review	New		1
Analýza testovania v javascripte	Closed	Bc. Martin Svrc...	0
Publikovanie eventov	New	Bc. Peter Kusnir	8
Zdokumentovanie fungovania rozšírenia do Chrome-u	Active	Bc. Peter Kusnir	4
Implementácia časti pre publikovanie eventov	Closed	Bc. Peter Kusnir	3
Code review	New		1
Zadefinovanie volania v API	New	Bc. Simon Val...	
Dorobit Web stránku	New	Bc. Peter Kusnir	

Focus Sprint 2 23. októbra - 5. novembra
10 work days

Backlog Board Capacity

Create query Column options

Title	State	Assigned To	Remaining W...
Pridanie sledovania pohľadu do administrácie	New	Bc. Matej Kucek	1
Code review	New		1
Vytvoriť ikonu na prepnutie do pohľadu administrácie	Closed	Bc. Matej Kucek	0
Commitnúť zmenu	Closed	Bc. Matej Kucek	0
Otestovanie	Closed	Bc. Matej Kucek	0
Volanie z ALEFu	New	Bc. Viktoria L...	1
Upravenie autentifikácie	Active	Bc. Viktoria Lov...	1
Autentifikácia cez API kľúč	New	Bc. Marek Sulek	9
Overenie API kľuca v DB	Closed	Bc. Marek Sulek	9
Pocuvanie eventov cez backbone JS a zvyraznenie elementov	Active	Bc. Martin Sv...	4
Analýza Backbone JS	Closed	Bc. Martin Svrc...	0
Implementácia časti na počúvanie/odchytávanie eventov	Closed	Bc. Martin Svrc...	1
Analýza práce s javascriptom	Closed	Bc. Martin Svrc...	1
Zvýraznenie elementu, na ktorý sa pozeráme	Closed	Bc. Martin Svrc...	1
Code review	New		1
Analýza testovania v javascripte	Closed	Bc. Martin Svrc...	0
Publikovanie eventov	New	Bc. Peter Kusnir	8
Zdokumentovanie fungovania rozšírenia do Chrome-u	Active	Bc. Peter Kusnir	4
Implementácia časti pre publikovanie eventov	Closed	Bc. Peter Kusnir	3
Code review	New		1
Zadefinovanie volania v API	New	Bc. Simon Val...	
Dorobit Web stránku	New	Bc. Peter Kusnir	

Focus Sprint 2 23. októbra - 5. novembra
10 work days

Backlog Board Capacity

Create query Column options

Title	State	Assigned To	Remaining W...
<ul style="list-style-type: none"> <ul style="list-style-type: none"> Code review Vytvorit ikonu na prepnutie do pohľadu administrácie Commitnúť zmenu Otestovanie 	New	Bc. Matej Kucek	1
<ul style="list-style-type: none"> <ul style="list-style-type: none"> Upravenie autentifikácie 	Active	Bc. Viktoria Lov...	1
<ul style="list-style-type: none"> <ul style="list-style-type: none"> Overenie API kľuca v DB 	Closed	Bc. Marek Sulek	9
<ul style="list-style-type: none"> <ul style="list-style-type: none"> Implementácia časti na počúvanie/odchytávanie eventov Analýza práce s javascriptom Zvýraznenie elementu, na ktorý sa pozeráme Code review Analýza testovania v javascripte 	Closed	Bc. Martin Svrc...	0
<ul style="list-style-type: none"> <ul style="list-style-type: none"> Implementácia časti pre publikovanie eventov Code review 	Closed	Bc. Peter Kusnir	3
<ul style="list-style-type: none"> <ul style="list-style-type: none"> Zdokumentovanie fungovania rozšírenia do Chrome-u 	Active	Bc. Peter Kusnir	4
<ul style="list-style-type: none"> <ul style="list-style-type: none"> Zadefinovanie volania v API 	New	Bc. Simon Val...	1
<ul style="list-style-type: none"> <ul style="list-style-type: none"> Dorobit Web stránku 	New	Bc. Peter Kusnir	

Obrázok 2: Šprint 2 backlog.

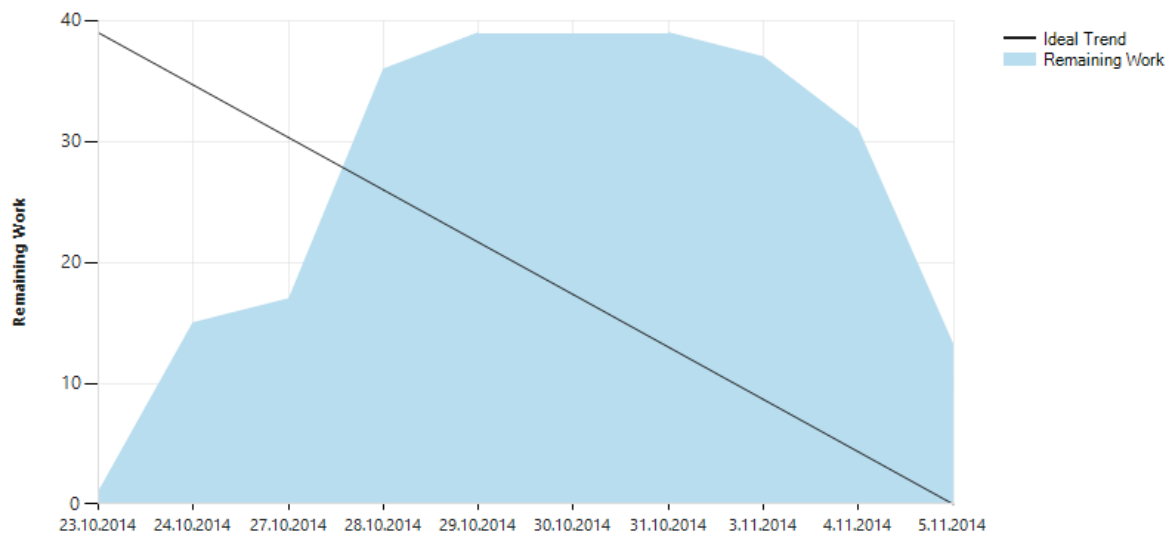
Počas druhého šprintu sme si zaznamenali všetky používateľské príbehy aj úlohy do systému TFS. Avšak niektorí členovia si nerozdelili im pridelené príbehy na úlohy dostatočne skoro a preto nám graf rástol v priebehu prvého týždňa (Obrázok 3). V rámci zadefinovaných úloh sa nám podarilo spraviť skoro všetky, avšak ukázalo sa, že sme si nechávali úlohy až na koniec šprintu (viď Obrázok 3) a preto sa nestihli dorobiť prehliadky kódu a dokumentácie k jednotlivým problematikám.

Jediným používateľským príbehom, ktorý ostal z veľkej časti nedorobný a preto musel byť presunutý do ďalšieho šprintu bolo:

- Počítanie agregácií

Dôvodom nedorobenia príbehu bolo zistenie, že je oveľa zložitejší ako sa predpokladalo a bude ho potrebné rozdeliť medzi viac ľudí.

V retrospektíve druhého šprintu sme určili ako najväčší problém nedostatočnú komunikáciu prostredníctvom nástroja HipChat. Určili sme si, že je potrebné aby sme viac komunikovali aby sme vedeli riešiť problémy v tíme čo najskôr. Identifikovali sme aj problém neskorého riešenia úloh a zhodli sa, že v budúcnosti je potrebné venovať sa úlohám priebežne. Výhodou bolo pre nás lepšie pochopenie problematiky na základe riešenia určených úloh a zorganizovanie stretnutia so zákazníkmi, kde sme si veľa vecí ujasnili alebo, na ktoré sme sa začali pozerat' inak.



Obrázok 3: Šprint 2 Burndown chart.

1.3.3 Šprint 3

V rámci tretieho šprintu sme si už zadefinovali pomerne reálne používateľské príbehy. V rámci príbehu *Analýza otázkovača v ALEFe* sa však v strede šprintu zistilo, že súčasná implementácia počúvania udalostí (event) prostredníctvom knižnice BackboneJS je z hľadiska prenositeľnosti na rôzne iné aplikácie zlým riešením. Preto bolo potrebné prerobiť implementáciu samotného počúvanie udalostí. Jednotlivé používateľské príbehy sú na Obrázok 4.

Focus Sprint 3 6. novembra - 19. novembra
3 work days remaining

Backlog Board Capacity

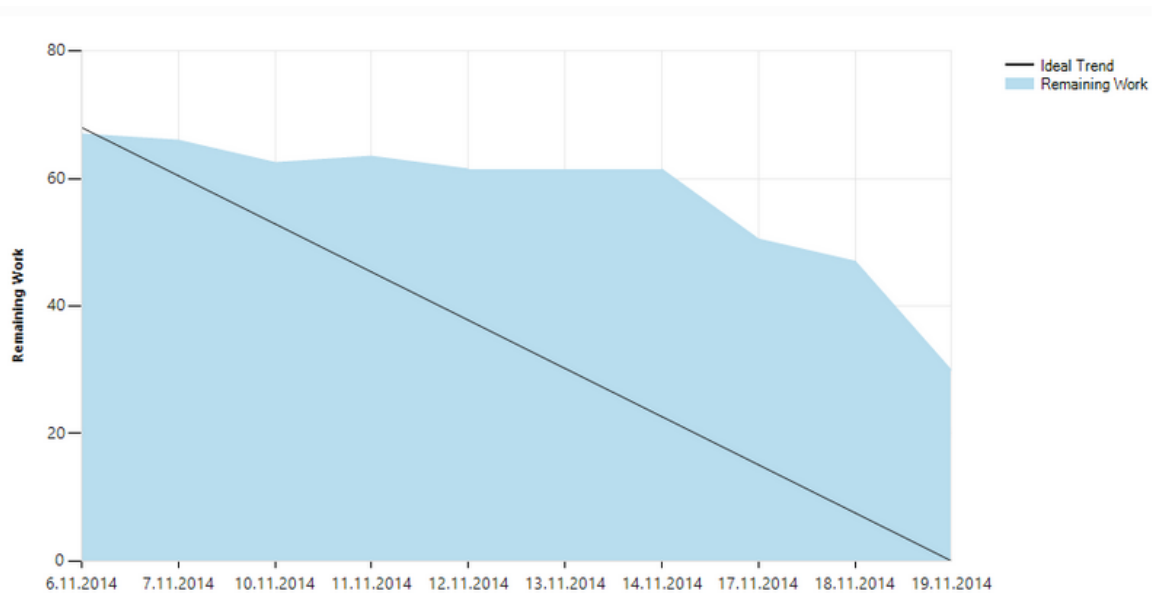
Create query Column options

Title	State	Assigned To	Remaining W...
Analýza fungovania oblasti záujmu na servery	New	Bc. Peter Kusnir	8
Zdokumentovanie rozšírenia pre Chrome	New	Bc. Peter Kusnir	3
Zdokumentovanie serverovej časti	Active	Bc. Peter Kusnir	5
Manažment projektu	New	Bc. Marek Sulek	13
Funkcia api Create project	Active	Bc. Marek Sulek	5
Funkcia api List projects	New	Bc. Marek Sulek	2
Funkcia api Get project by id project	New	Bc. Marek Sulek	2
Funkcia api Update project	New	Bc. Marek Sulek	2
Funkcia api Delete project	New	Bc. Marek Sulek	2
Manažment Gaze Trackingu v Alefe	New	Bc. Viktoria L...	13
Vytvorenie formulara na manazment s projektami	Closed	Bc. Viktoria Lov...	0
Vytvorenie resource projekt	Closed	Bc. Viktoria Lov...	0
Posielanie requestov na manazment s projektmi ku gaze trackin...	New	Bc. Viktoria Lov...	3
Otestovanie volani	New	Bc. Viktoria Lov...	3
Napisanie dokumentacie	New	Bc. Viktoria Lov...	5
Code review	New	Bc. Martin Svrc...	2
Zanalyzovať otázokvač v Alefe	New	Bc. Martin Sv...	6
Analýza otázokvača v ALEFe	New	Bc. Martin Svrc...	4
Zdokumentovanie ALEF otázokvača	New	Bc. Martin Svrc...	2
Počítanie agregácií	Active	Bc. Peter Kis	12,5
Code review	New	Bc. Matej Kucek	2
Úprava DB štruktúry	Closed	Bc. Peter Kis	0
Vytvorenie agregáčnych dopytov	Active	Bc. Matej Kucek	1
Analýza existujúcej aplikačnej a databázovej infraštruktúry	Closed	Bc. Peter Kis	0
Vytvorenie aplikačnej logiky	Active	Bc. Peter Kis	0,5
Refactoring existujúceho kódu	Closed	Bc. Peter Kis	0
Napisanie dokumentácie	New	Bc. Peter Kis	4
Testovanie	Active	Bc. Peter Kis	3
Otestovať priebornosť	New	Bc. Matej Kucek	2

Obrázok 4: Šprint 3 backlog.

V rámci tohto šprintu sa nám podarilo spraviť väčšinu úloh avšak niektoré sa taktiež nestihli čo je možné vidieť aj na burndown grafe (Obrázok 5). Jedným z príčin bola určite potreba práce na dokumentácii, ktorej sme museli venovať nejaký čas.

V retrospektíve tretieho šprintu bolo hlavnou témou ako brať do úvahy rôzne udalosti, ktoré môžu ovplyvniť daný šprint. V tomto šprinte bolo napríklad potrebné pracovať na dokumentácii. Táto úloha pomerne výrazne zasiahla do času, ktorý mal každý člen vymedzený na prácu na daných úlohách a príbehoch. V budúcnosti budeme chcieť takéto udalosti predikovať vopred aby sme sa mohli na ne pripraviť.



Obrázok 5: Šprint 3 Burndown chart.

1.3.4 Šprint 4

V rámci štvrtého šprintu sme sa pustili do vytvárania jednotlivých API volaní, ktoré budú tvoriť podstatnú časť našej ďalšej práce.

Takisto sme zistili, že otázkovač v systéme ALEF, ktorý sme analyzovali v treťom šprinte nie je úplne to čo potrebujeme a preto bolo potrebné vykonať analýzu v kontexte pýtača v systéme ALEF (viď príloha C: Analýza pýtača v ALEFe).

V rámci vytvárania agregácií bolo potrebné vykonať posledné úpravy, tak aby bola táto časť úplne dokončená. Na používateľskom príbehu *Počítanie agregácií* je vidieť ako sme zle identifikovali zložitosť. Malo to vplyv na nutnosť prenosu tohto príbehu do nasledujúceho šprintu.

Jednotlivé príbehy a úlohy k nim priradené sú zobrazené na Obrázok 6.

Backlog Board Capacity

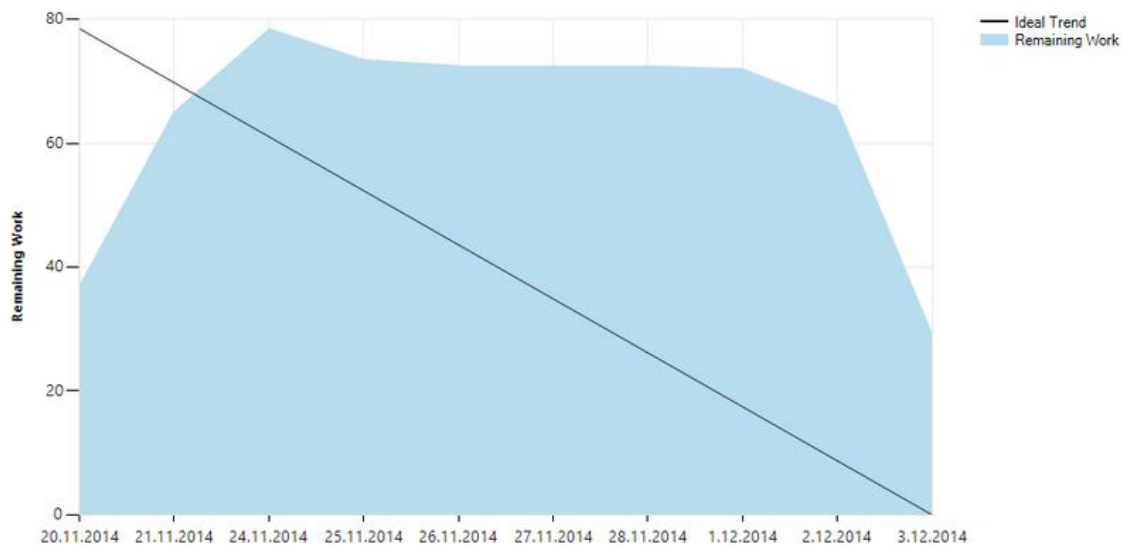
Create query Column options

Title	State	Assigned To	Remaining W...
Analýza a implementácia otázkovaca pre API	Active	Bc. Martin Sv...	1
Zdokumentovanie otázkovača v ALEFe	Closed	Bc. Martin Svrc..	0
Analyzovanie otázkovača v ALEFe	Closed	Bc. Martin Svrc..	0
Code review	New		1
Implementácia generovania otázky na základe prijatého eventu	Active	Bc. Martin Svrc..	0
API Agregácie	New	Bc. Simon Val...	8
specifikácia	New	Bc. Simon Valic..	2
implementácia	New	Bc. Simon Valic..	3
testy	New	Bc. Simon Valic..	2
code review	New		1
API AOI	New	Bc. Peter Kusnir	11
Create AOI	Closed	Bc. Peter Kusnir	4
Update AOI	Closed	Bc. Peter Kusnir	3
Read(Get) AOI	Closed	Bc. Peter Kusnir	2
Delete AOI	Closed	Bc. Peter Kusnir	2
Code Review	Closed	Bc. Marek Sulek	
Dokumentácia	Active	Bc. Peter Kusnir	
API Users	New	Bc. Marek Sulek	11
Implementácia	Closed	Bc. Marek Sulek	8
testy	Closed	Bc. Marek Sulek	
documentácia	Closed	Bc. Marek Sulek	3
code reilver	Active	Bc. Marek Sulek	
APIS Sessions	New	Bc. Viktoria L...	5
Implementácia	Closed	Bc. Viktoria Lov...	0
Testy	Closed		5
Dokumentácia	Closed		0
Počítanie agregácií	Active	Bc. Peter Kis	9
Code review	New	Bc. Matej Kucek	2
Otestovať priepustnosť	New	Bc. Matej Kucek	2
Dokumentácia	Closed	Bc. Peter Kis	0
Analýza	Closed	Bc. Peter Kis	0
Implementácia	Closed	Bc. Peter Kis	0
Testovanie	New	Bc. Peter Kis	5

Obrázok 6: Šprint 4 backlog.

V rámci štvrtého šprintu sa podarilo spraviť väčšinu úloh. Burndown graf však nevyzerá úplne ideálne (Obrázok 7: Šprint 4 Burndown chart Obrázok 7) a to z dôvodu neskorého zaznamenávania práce a ukončenia úloh v systéme TFS. Tento problém nás sprevádza počas všetkých šprintov a preto je potrebné aby sme to v budúcnosti zmenili.

V retrospektíve štvrtého šprintu sme sa rozprávali o dôležitosť zaznamenávania odporacovaného času do systému TFS. Zhodli sme sa taktiež na tom, že jednotlivé používateľské príbehy by bolo dobré dekomponovať na úlohy priamo na stretnutí a nie samostatne. Toto môže vyriešiť problém s nepresným určením zložitosti daného príbehu.



Obrázok 7: Šprint 4 Burndown chart.

1.3.5 Šprint 5

Týmto šprintom sme začali letný semester. Na začiatok sme si spravili zhodnotenie minulého semestra. Zhodli sme sa na tom, že sme prácu nemali dobre rozdelenú na prípady použita. Toto sme sa rozhodli zmeniť v ďalšom semestri.

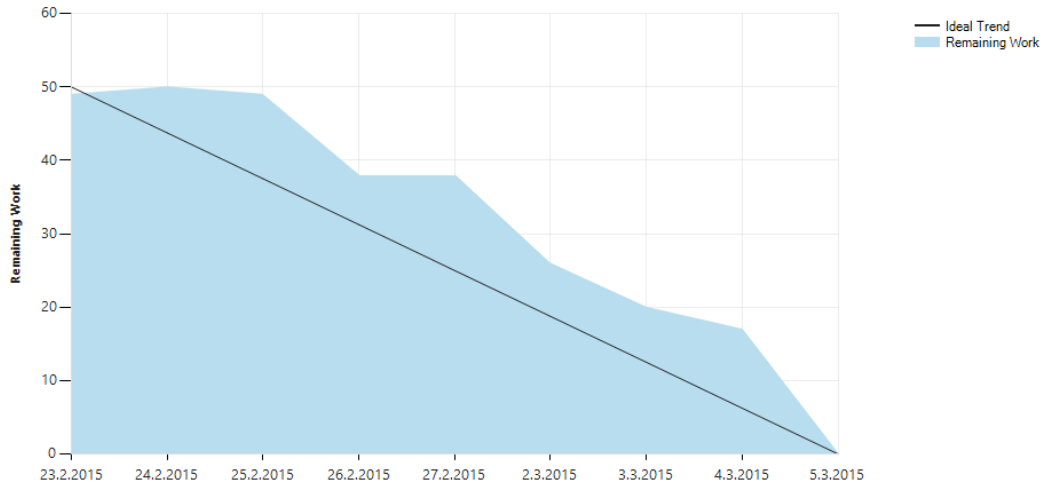
V prvom šprinte sme si určili potrebu presunu logiky z rozšírenia (do prehliadača) do JavaScript knižnice. Taktiež sme začali pracovať na vizualizácii dát. V tomto šprinte sme vytvorili prvé vykreslovacie funkcie pre webovú aplikáciu. Začali sme sa zaoberať aj ďalšou veľkou témou, ktorou boli fixácie a ich filtrovanie. Potrebovali sme nájsť riešenie alebo implementáciu generovania fixácií zo surových dát. S týmto súvisela aj úloha analýzy implementácie vzorov čítania. Zároveň sme vytvorili statickú stránku na účely plánovaného experimentu. Jednotlivé úlohy sú zaznamenané na Obrázok 8.

Title	State	Assigned To	Remaining W...
Presunut komunikáciu do JS lib	New	Bc. Marek Sulek	2
Analyza chrome extension	Closed	Bc. Marek Sulek	0
Presun logiky extensionu do js suboru	Closed	Bc. Marek Sulek	2
Volanie API z JS knižnice	Active	Bc. Peter Kusnir	
Grafické zobrazenie + filter (voliteľné)	Resolved	Bc. Peter Kis	
Analyza vhodnosti použitia d3.js a .Net chart control	Closed	Bc. Peter Kis	0
Vytvorit testovacie data	Closed	Bc. Peter Kis	0
Vytvorenie vykreslovacích funkcií	Closed	Bc. Peter Kis	0
Analyza použiteľnosti existujúceho kódu	Closed	Bc. Peter Kis	0
Doriešiť DB / Server	New	Bc. Simon Val...	
Otestovať zachytávanie eventov	Active	Bc. Viktoria L...	
Filter fixácií	Active	Bc. Martin Sv...	2
Analyza dostupných implementácií filtrov	Closed	Bc. Martin Svrc...	0
Analyza článkov	Closed	Bc. Martin Svrc...	2
Spracovanie zvolenej implementácie	Active	Bc. Martin Svrc...	0
Overenie správnosti implementácie	Active	Bc. Martin Svrc...	0
Zobrazenie počtu fixácií v tabulke	New	Bc. Matej Kucek	
Implementácia stiahnutia dát	New	Bc. Matej Kucek	0
Implementovanie naplnenia tabulky	New	Bc. Matej Kucek	0
Analyza predošlého kódu	Closed	Bc. Matej Kucek	0
Implementácia agregacných funkcií do pozadovaneho tvaru	Closed	Bc. Matej Kucek	0
Statická stránka s textom	Closed	Bc. Viktoria L...	
Analýza vzorov čítania	Active	Bc. Martin Sv...	2
Analyza dostupných riešení vzorov čítania	Closed	Bc. Martin Svrc...	2
Analyza vzorov čítania	Closed	Bc. Matej Kucek	0

Obrázok 8: Šprint 5 Backlog.

V rámci burndown chartu je možné vidieť plynulé riešenie všetkých úloh (Obrázok 9). Toto bolo zapríčinené aj začiatkom semestra, kedy sme nemali príliš veľa povinností do školy. Podarilo sa nám splniť veľkú časť úloh.

V retrospektíve sme sa rozprávali o dobre nastolenom trende v pracovnom nasadení a o potrebe udržania tohto trendu.

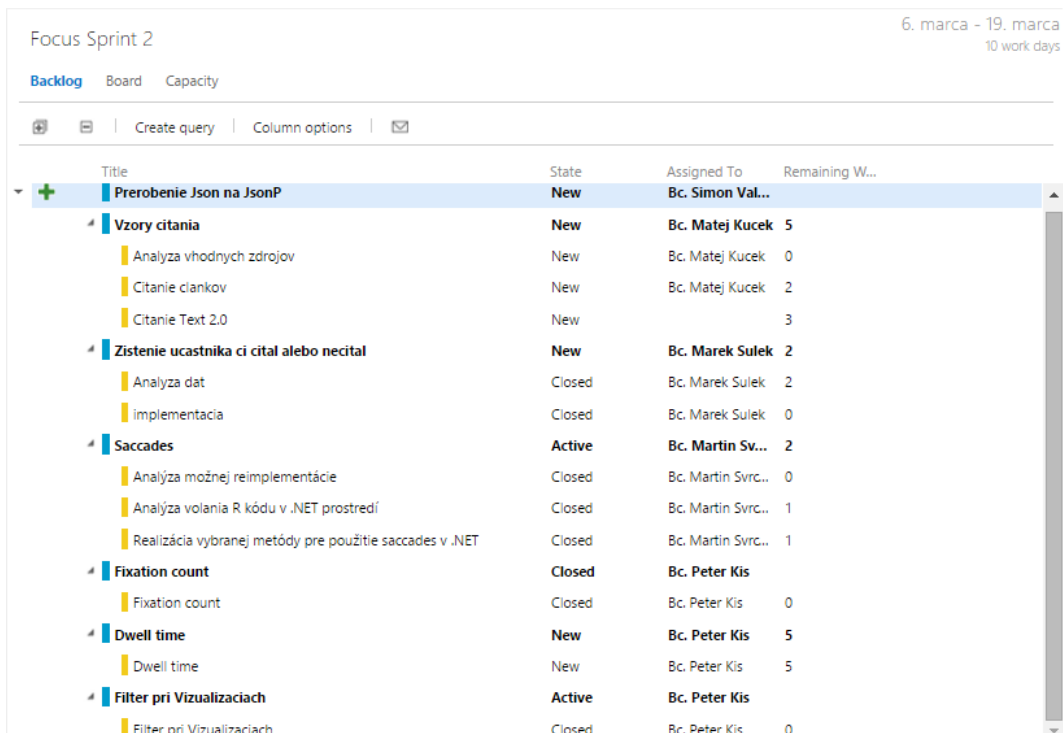


Obrázok 9: Šprint 5 Burndown chart.

1.3.6 Šprint 6

V predchádzajúcom úvodnom šprinte sme začali niektoré úlohy analýzou dostupných riešení v danej problematike. Preto bolo v tomto šprinte potrebné využiť získané znalosti a pustiť sa do implementácie jednotlivých úloh.

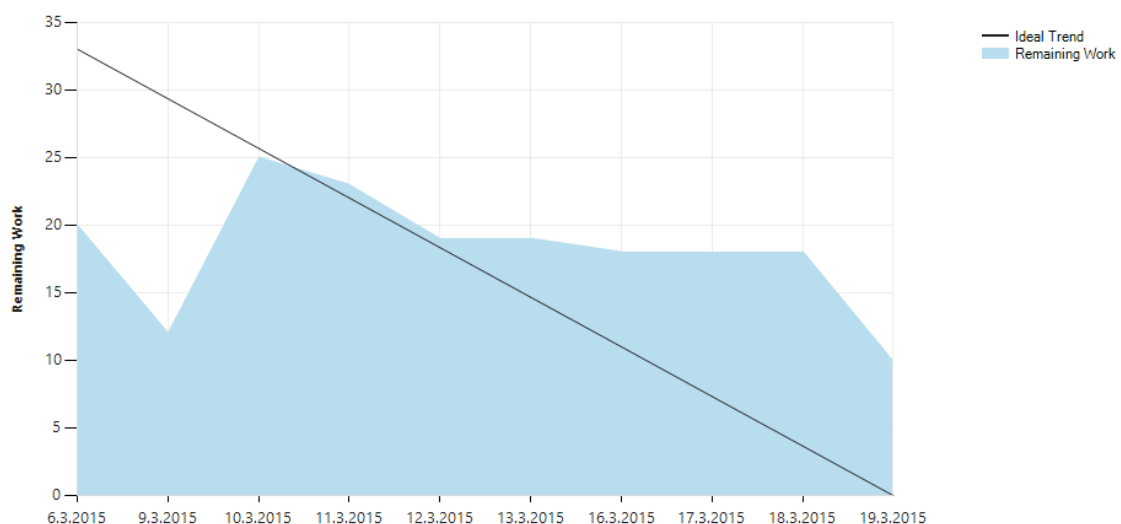
Posunuli sme sa pri implementácii detegovania vzorov čítania, kde sme našli vhodný algoritmus a rozhodli sme sa ju použiť a otestovať. V kontexte vizualizácií sme vytvorili jednoduchý filter na zobrazovanie rôznych dát a ich kombinácií. Vytvorené riešenie sme konzultovali a prerábali do čo najlepšej podoby. V kontexte filtrovania fixácií sme našli vhodné riešenie pre jazyk R. Keďže sme ho potrebovali využiť v C# prostredí bolo nutné implementovať volanie R funkcií, čo sa aj podarilo a po šprinte sme mali k dispozícii funkčné riešenie filtrovania fixácií. Jednotlivé úlohy sú zobrazené na Obrázok 10.



Obrázok 10: Šprint 6 Backlog.

Pri pohľade burndown chart (Obrázok 11) je možné vidieť pomerne dobré riešenie úloh. Podarilo sa vyriešiť skoro všetky úlohy, okrem zopár menších, ktoré sme z dôvodu ich menšej dôležitosti nestíhali dorobiť, keďže sme si ich nechali na až koniec.

V rámci retrospektívy sme zistili, že implementácia filtrovania fixácií bude dostupná aj v podobe IVT filtra, ktoré využíva aj konkurenčný softvér a dáva dobré výsledky. Preto bude možno dobré využiť tento algoritmus namiesto volania R funkcií.



Obrázok 11: Šprint 6 Burndown chart.

1.3.7 Šprint 7

Tento šprint začal rozhodnutím využiť IVT filter, ktorý dáva kvalitné výsledky ale bude ho potrebné reimplementovať, pretože ho máme dostupný iba v syntaxe Matlab.

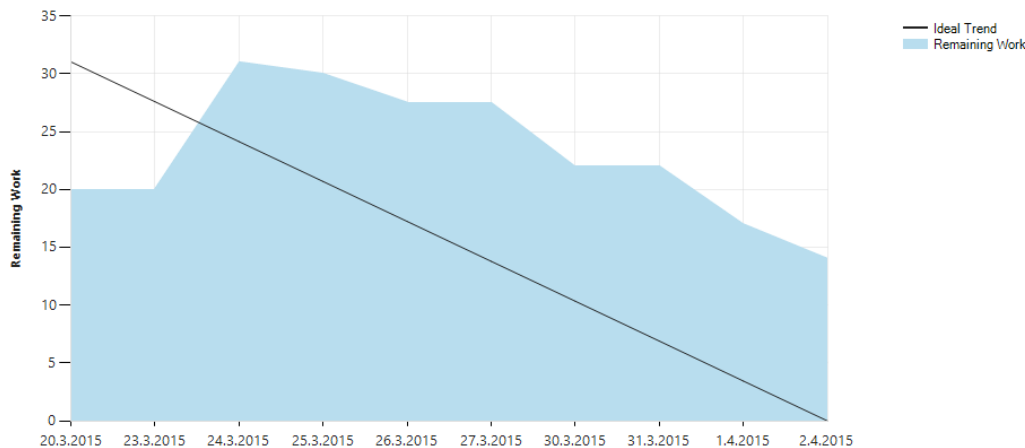
V rámci vizualizácií sme sa pustili do vytvárania samotných grafov, keďže už sme mali pripravené dáta aj formulár na ich výber. Takisto sa nám v tomto šprinte podarilo implementovať prvú verziu algoritmu detekcie vzorov čítania, ktorá dokázala etegovať čítanie a nečítanie. V rámci IVT filtra sme vytvorili prvú verziu tohto algoritmu v jazyku C#. Ďalšou veľkou oblasťou bolo generovanie upozornení. Túto oblasť bolo potrebné riešiť aj v rámci JavaScript knižnice aj v rámci servera. Taktiež sme naplánovali prvý experiment, ktorý mal prebehnúť v blízkej dobe. Na Obrázok 12 je možné vidieť jednotlivé úlohy.

Title	State	Assigned To	Remaining W...
Grafické štatistiky	Active	Bc. Peter Kis	
Vytvorenie grafov	Closed	Bc. Peter Kis	0
Implementacia algoritmu analyzy citania	New	Bc. Matej Kucek	3
Implementacia algoritmu rozoznania citania	Closed		0
Vylepsovanie a rozšírenie o ďalšie rozoznania	New		3
Parsovanie dat	Closed		0
Pocitanie fixacii v C#	Active	Bc. Martin Sv...	3
Analýza algoritmu I-V filtra	Closed	Bc. Martin Svrc...	0
Reimplementácia I-V filtra do syntaxe C#	Closed	Bc. Martin Svrc...	1
Otestovanie reimplementácie v C#	New	Bc. Martin Svrc...	2
JS API Trigger	New	Bc. Simon Val...	
Dopracovanie funkcionality JS kniznice	New	Bc. Marek Sulek	
vyriesenie cross domain volani	Active	Bc. Marek Sulek	
implementovanie funkcionality	Active	Bc. Marek Sulek	
Opravit session managment a spojzdnit klienta	New	Bc. Simon Val...	
Upozornenie na serveri	New	Bc. Peter Kusnir	
Vytvorenie upozornenia	Closed	Bc. Peter Kusnir	
Uprava upozornenia	Closed	Bc. Peter Kusnir	
Vytvorenie DB tabuiky a infrastruktury	Active	Bc. Peter Kusnir	
Zobrazovanie existujucich upozorneni	Closed	Bc. Peter Kusnir	
Mazanie upozorneni	Closed		
Stylovanie	Closed	Bc. Peter Kusnir	
Validacie formularov a dokoncovacky	Active	Bc. Peter Kusnir	
Naplanovanie + mail JT Data Creating	Closed	Bc. Viktoria L...	0,05
Uprava klienta + zmena URL	New	Bc. Simon Valic...	0,05

Obrázok 12: Šprint 7 Backlog.

Práca na tomto šprinte už neprebíhala tak hladko čo je možné vidieť aj na burndown charte (Obrázok 13). Súviselo to aj s povinnosťami v škole, ktorých pribudlo.

V rámci retrospektívy sme identifikovali veľkú zložitosť jednotlivých naplánovaných úloh a potrebu ich rozdelenia na menšie časti, ktoré bude možné za jeden šprint zrealizovať celé.

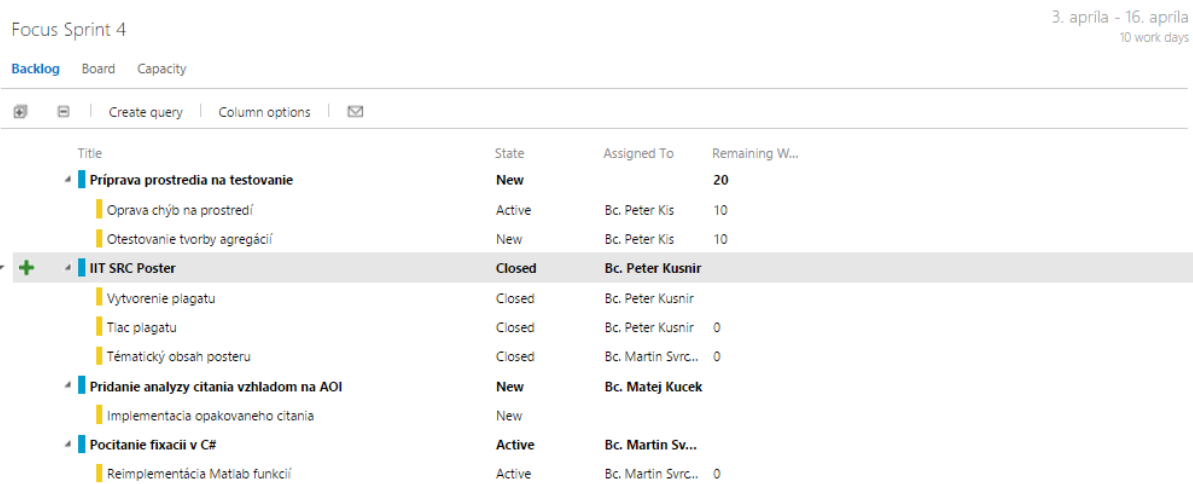


Obrázok 13: Šprint 7 Burndown chart.

1.3.8 Šprint 8

Tento šprint bol ovplyvnený vedeckou študentskou konferenciou IIT.SRC, na ktorej sme mali prezentovať naše riešenie. Bolo preto potrebné pripraviť sa na túto udalosť.

Venovali sme sa teda príprave posteru na konferenciu. Bolo potrebné vymyslieť obsahovú aj grafickú stránku posteru. Ďalší členovia tímu sa venovali vyhodnocovaniu vykonaného experimentu aby sme jeho výsledky mohli prezntovať aj na konferencii. V rámci vzorov čítania sa implementoval vzor opakovaného čítania. Pokračovalo sa tiež v reimplementácii Matlab funkcií do jazyka C#. Jednotlivé úlohyje možné vidieť na Obrázok 14.



Obrázok 14: Šprint 8 Backlog.

V kontexte retrospektívy sme sa bavili hlavne o ideách získaných na konferencii IIT.SRC a ich využitíu v našom projekte.

1.3.9 Šprint 9

V rámci šprintu 9 sme dokončovali všetky rozbehnutú úlohy. Dokončovali sa upozornenia v JavaScript knižnici aj na strane servera. Riešili sa problémy s rozbehnutím servera, ktoré sa ukázali ako pomerne veľké. V kontexte IVT filtra sa testovalo ako dobré sú výsledky.

V tomto prípade sa nepodarilo opraviť všetky problémy. Vzory čítania sa integrovali do celkového riešenia projektu. Jednotlivé úlohy sú na Obrázok 15.

Focus Sprint 5 17. aprila - 30. aprila
10 work days

Backlog Board Capacity

Create query Column options

Title	State	Assigned To	Remaining W...
Prepojenie implementácií upozorňovania	New	Bc. Marek Sulek	5
Implementácia upozorňovania (alert)	New	Bc. Marek Sulek	5
Dokončenie implementácie určovania fixácií	Active	Bc. Martin Sv...	
Doimplementovanie IVT filtra	Closed	Bc. Martin Svrc...	0
Otestovanie implementácie IVT filtra	Active	Bc. Martin Svrc...	0
opravenie agregácií a testovanie na eyetrackeri	New	Bc. Peter Kis	
integrácia analýzy čítania	New	Bc. Matej Kucek	

Obrázok 15: Šprint 9 Burndown.

1.4 Použité metodiky

V rámci práce na projekte sme vypracovali nasledovné metodiky:

- Písanie zdrojových kódov v jazyku C#
 - Metodika zahŕňa pravidlá pre tvorbu názvoslovia, písanie komentárov, formátovanie zdrojových textov a používania dátových typov.
 - Metodika neobsahuje popis syntaxe jazyka C#, návod na písanie optimálneho kódu, ani postupy pri písaní aplikačnej logiky.
 - Popísané pravidlá vychádzajú z oficiálnej príručky písania zdrojových kódov v jazyku C# od spoločnosti Microsoft.
 - Metodika integrácie Metodika opisuje postup ako nasadiť doimplementovanú funkcionálnu, poprípade zmenu, na server. Toto nasadzovanie sa týka iba vývojovej vetvy.
 - Postup je opisovaný v prostredí Visual Studio 2013 za použitia technológie Team Foundation Server.
- Metodika prehliadky kódu
 - Účelom tejto metodiky je definovať postup pri prehliadkach kódu, za cieľom aby zachovania dostatočná kvalita zdrojového kódu.
 - Metodika obsahuje postup, ktorý sa musí dodržiavať pri prehliadke kódu a opis práce s nástrojmi, ktoré sa pri prehliadke musia použiť.
- Metodika používania softvérového nástroja na správu verzií zdrojového kódu
 - Metodický pokyn k používaniu softvérového nástroja na kontrolu verzií zdrojového kódu.
- Manažment riešenia chýb
 - Metodika popisuje proces zaznamenávania a manažovania chýb prostredníctvom nástroja Team Foundation Server.
- Dekompozícia používateľských príbehov na úlohy v nástroji TFS
 - Princípom tejto metodiky je poskytnúť postup alebo metódu na spomínané funkcionálne dekomponovanie používateľských príbehov na menšie časti (úlohy).

- Táto metodika zahŕňa celý proces dekompozície používateľských príbehov od vytvorenia základne predstavy na tímovom stretnutí až po samotné zadávanie úloh do systému TFS.
- Metodika sa však nezaobera vytváraním používateľských príbehov ani správou úloh počas šprintu (napr. aktualizácia času a stavu úloh).
- Metodika testovania
 - Tento dokument popisuje proces vytvárania testovacích projektov a testov pre jednotlivé komponenty aplikácie napísanej v jazyku C#.
 - Zachytáva postup tvorby testu, jeho členenie, konvencie v pomenovávaní jednotlivých testovaných častí a spúšťanie testov.

1.5 Globálna reprotrospektíva

V tejto časti bližšie opíšeme určité aspekty projektu a práce na projekte z hľadiska ich prínosu a prípadných zmien, ktoré treba spraviť.

1.5.1 Aspekty projektu, s ktorými sme boli spokojní

- Vzájomná pomoc
 - Jednou z pozitívnych stránok našej doterajšej práce je pomerne dobrá spolupráca na projekte.
 - Jednotliví členovia si pri ich úlohách často pomáhali, hlavne vtedy ak už jeden mal skúsenosti s danou problematikou a pomohol ďalšiemu členovi s pochopením problému.
 - Tento aspekt je pre celkový úspech projektu nesmierne dôležitý a je potrebné, aby sme si ho uvedomili. Takisto je potrebné aby sme v takomto trende pokračovali, pretože to môže výrazne uľahčiť a aj zrýchliť jednotlivé aktivity.
- Správa úloh v TFS
 - Na začiatku (v priebehu 1 a 2 šprintu) sme si ako tím nie príliš dobre zaznamenávali prácu v systéme TFS a takisto sme mali problém s včasným rozdelením príbehov na úlohy.
 - V priebehu tretieho šprintu sa však toto výrazne zlepšilo. Každý z členov si hneď v deň stretnutia k novému šprintu vytvoril jednotlivé úlohy. Toto je z hľadiska sledovania správneho plnenia cieľov veľmi dôležité a taktiež sa môže pomerne skoro určiť a vyriešiť nejaký problém.
 - Takisto sme pomerne dobre zaznamenávali jednotlivé stavy plnenia úloh. V tomto kontexte je dobre, že každý člen zaznamenáva zmenu stavu danej úlohy čo možno najskôr. Pomôže to lepšej orientácii počas šprintu.
 - V rámci letného semestra sme si pravidelne zaznačovali vykonanú prácu a burndown charty vyzerali dobre. Tento trend sme si udržali ďalšie dva šprinty. Nový semester sme tak pokračovali v prístupe z konca zimného semestra.

1.5.2 Aspekty projektu, ktoré treba zmeniť

- Rozdelenie práce na šprinte
 - Z burndown grafov je možné vidieť, že sa nám často nepodarilo splniť všetky stanovné úlohy. Z veľkej časti to súviselo hlavne s neskorým začiatkom práce na danej úlohe alebo príbehu.
 - V tomto kontexte je preto potrebné, aby každý jeden člen vykonával všetky úlohy postupne od začiatku šprintu. Pomôže to vo viacerých oblastiach. Na jednej strane bude môcť byť daná úloha konzultovaná dostatočne skoro, ak sa objavia nejaké problémy. Na strane druhej sa zvýši kvalita implementácie, keďže ostane viac času na správne overenie riešenia.
- Špeciálne úlohy
 - Ďalšou oblasťou, kde je potrebné zmeniť prístup je vykonávanie špeciálnych úloh, ktoré v súčasnosti pomerne často vynechávame. Patrí sem:
 - Analýza problematiky
 - Testovanie
 - Prehliadky kódu
 - Dokumentácia k určitým uceleným celkom
 - Na začiatku celého projektu sme si povedali, že sa nechceme zamerať na kvantitu, ale hlavne na kvalitu. Dôležité je preto, aby bola každá jedna funkcionálna kvalitne otestovaná, aby bola spravená prehliadka kódu, a aby bola popri prípade zdokumentovaná celá problematika.
- Aktualizácia zmien v kóde do TFS (push)
 - Výrazným problémom je taktiež nedostatočné zdieľanie zmien prostredníctvom nástroja git.
 - Aktuálne toto možno nie je až tak veľký problém, keďže každý pracuje na pomerne oddelených častiach. Avšak ak bude potrebné pracovať a mať k dispozícii niektoré nové zmeny bude dôležité aby sa takéto aktualizácie robili čo najskôr po otestovaní.
- Potreba lepšieho manažmentu práce
 - V rámci projektu sa nám ukázal ako obrovský problém nerozbehávanie celého riešenia včas. Keď sme sa pokúšali sfunkčnit' server, vždy sa to nepodarilo a riešili sme niečo iné.
 - Takýto prístup, kedy nemôžeme overiť naše riešenie a ani nenasadzujeme nové funkcie je nie príliš dobrý. Nakoniec sa to ukázalo ako hlavný dôvod nesfunkčnenia celého riešenia. Projekt obsahoval obrovské množstvo častí, ktoré nikdy neboli spojené ani otestované na reálnom pokuse a pri ich rozbehávaní sa nám ani nepodarilo tieto problémy vyriešiť.

1.5.3 Nové a potrebné aspekty projektu

- Plánovanie úloh na stretnutí
 - V rámci plánovania sa nám stalo, že sme zle ohodnotili zložitosť jedného z používateľských príbehov. Konkrétne sa ukázalo, že daný príbeh je podstatne zložitejší ako sme si na začiatku mysleli. Nakoniec sa tento príbeh nepodarilo ukončiť v danom šprinte a musel sa presunúť do nasledujúceho.

- Aby sa predišlo takýmto problémom bolo by dobré, keby sme si jednotlivé používateľské príbehy rozdeľovali na úlohy priamo na stretnutí a následne spoločne ohodnocovali tieto úlohy z hľadiska zložitosti.
- Takýto prístup môže pomôcť aj lepšiemu pochopeniu daného príbehu.
- Možnosť simulácie
 - Veľmi dôležitou časťou je potreba vykonávania simulácie komunikácie medzi systémom na sledovania pohľadu a výučbovým systémom ALEF.
 - V tomto kontexte je potrebné nasadenie úprav v rámci ALEF implementácie na nejakú online inštanciu tohto systému.
 - Druhou možnosťou je využitie Sandbox inštancie určenej na podobné testovania.
 - Prepojenie s externou aplikáciou je jedným z našich najdôležitejších cieľov v projekte, a preto je potrebné aby bol tento problém čo najskôr vyriešený.

2 Zoznam kompetencii tímu

Peter Kiš

C, C#.Net, Asp.Net, javascript, xml, MySQL, MSSQL, Oracle DB, Unity3Dniekoľkoročná prax s vývojom hier, skúsenosti s vedením tímu - pol roka 2-4 ľudí

Matej Kucek

C/C++, Java, Ruby, Linux, MySQL, xml, html, android development. Vodcovské schopnosti, zodpovednosť, flexibilita, riešenie konfliktov, vo voľnom čase vývoj hier

Marek Šulek

C/C++, Java, Linux, JavaScript, MySQL, XML, HTML, android development, skúsenosti s prácou v tíme.

Šimon Valíček

C/C++, Java, HTML, CSS, JavaScript, PHP, MySQL, PostgreSQL, Android. Inštalácia a konfigurácia serverov na platforme Linux. Skúsenosti s prácou v tíme na malých a stredne veľkých projektoch.

Viktória Lovasová

C, java, ruby on rails, html, javascript, bootstrap, mysql. Zodpovednosť, komunikatívnosť

Martin Svrček

C, java, ruby on rails, HTML, CSS, JavaScript, MySQL, XML Menšie skúsenosti s grafickými návrhmi, záujem o User Experience (UX) a vizuálnu stránku softvéru.

Peter Kušnír

C, Java, HTML, CSS, JavaScript, PHP + Zend Framework, MySQL, PostgreSQL, menšie skúsenosti s knižnicami OpenGL a OpenCV, základy tvorby 3D modelov (Blender), kreslenie ako hobby.

3 Metodiky

V tejto kapitole sú zaznamenané jednotlivé metodiky, ktoré vytvorili členovia nášho tímu s cieľom určenia spoločných princípov a postupov pri jednotlivých činnostiach v rámci práce na projekte. Všetky majú za úlohu poskytnúť čo najlepší pohľad na jednotlivé aktivity tak, aby sa vykonávali správne, kvalitne a hlavne jednotne.

3.1 Manažment riešenia chýb

3.1.1 Úvod

Táto metodika je určená pre tím číslo 16 – Focus a popisuje proces zaznamenávania a manažovania chýb prostredníctvom nástroja Team Foundation Server 2013, ktorý ako tím používame. V prípade, že je v projekte GazeTracking/Focus chyba, riadi sa tím touto metodikou.

3.1.1.1 Vymedzenie skratiek

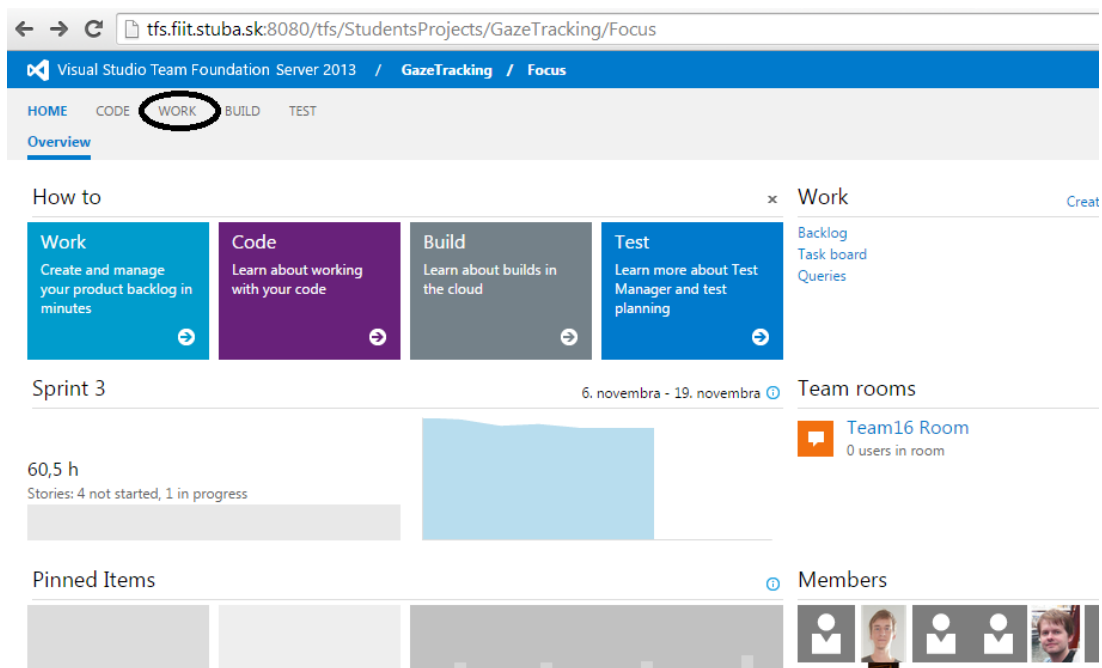
Team Foundation Server 2013 – TFS
Akademický informačný systém – AIS

3.1.1.2 Zoznam nadväzujúcich metodík a dokumentov

- Metodika – Metodika testovania
- Metodika – Manažment verzií
- Metodika – Prehliadky kódu

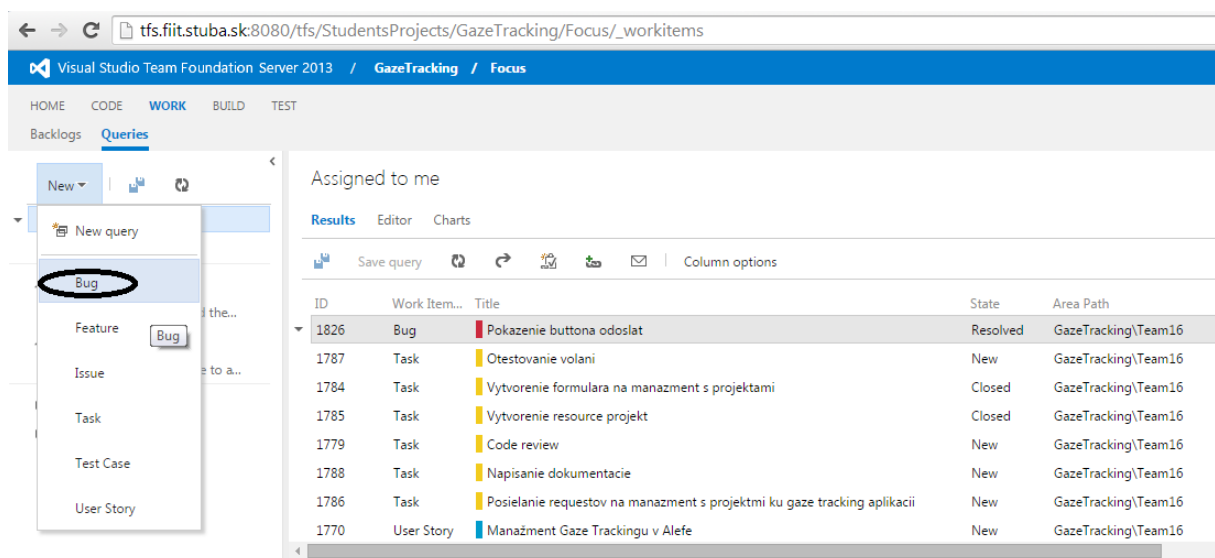
3.1.2 Vytvorenie nového objektu – chyby v TFS

1. Otvorenie internetového prehliadača, napísanie adresy: <https://tfs.fiit.stuba.sk/>
2. Prihlás sa s prihlasovacími údajmi ako do AIS
3. Klikni na Project collection
4. Vyber projekt GazeTracking/Focus
5. V hornom ľavom menu klikni na Work, pozri Obrázok 16:



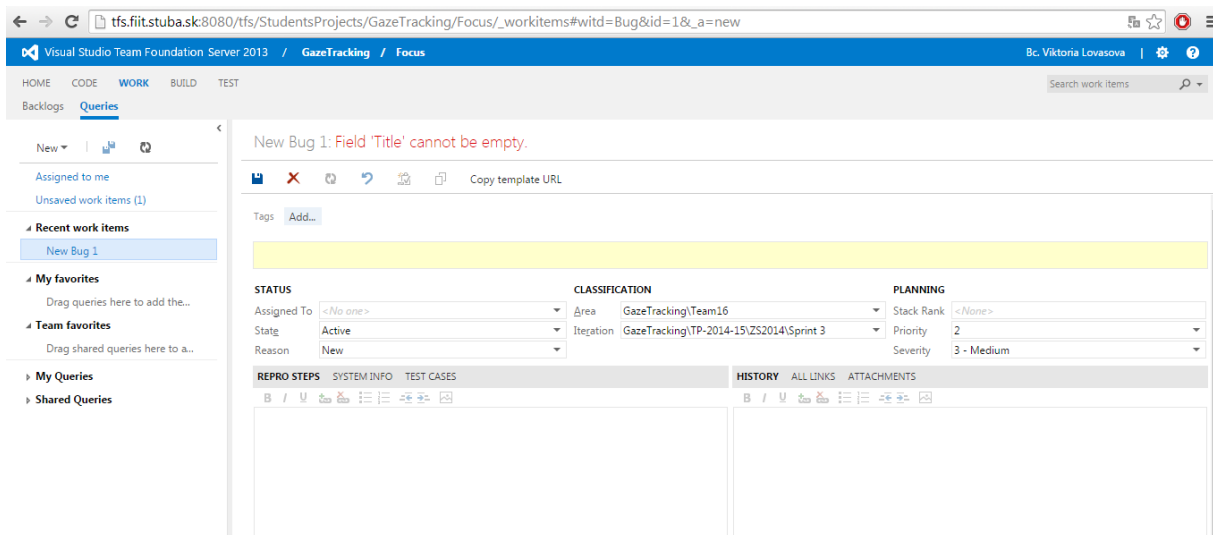
Obrázok 16: Kliknutie na možnosť work.

6. V menu Queries (dopyty) klikni na New (nový) a potom na Bug (chyba), Obrázok 17:



Obrázok 17: Kliknutie na vytvorenie novej chyby.

7. Zobrazí sa okno na vytvorenie novej chyby, pozri Obrázok 18:



Obrázok 18: Vytvorenie chyby v TFS.

8. Vyplň názov chyby, ktorý ju stručne a jasne vystihuje.
9. V prípade, že chybu budeš opravovať ty, vyber z možností svoje meno, ak nie, tak políčko nechaj prázdne.
10. Stav a príčinu chyby nechaj nezmenené.
11. V prípade, že je chyba závažná zmeň prioritu na 1.
12. Nakoniec dopíš komentár, v ktorom podrobne opíšeš ako chyba nastáva.

Potom ako je chyba vytvorená, príde používateľovi, ktorému bolo vyriešenie chyby pridelené, notifikácia na kontaktný email s informáciami, ktoré boli poskytnuté pri vytváraní chyby. Keď chyba nie je pridelená nikomu, príde notifikácia každému členovi z tímu, pričom si ju musí niekto zobrať na starosti a vyriešiť. V prípade, že si ju nikto nezoberie, prideliť ju niekomu vedúci tímu. Len vedúci tímu môže chybu prideliť inej osobe ako on je sám.

3.1.3 Odstránenie chyby

Aby sme vedeli, či je chyba odstránená a zároveň zvyšovali pokrytie funkcionality projektu testami, najprv si vždy vytvoríme test, ktorý bude chybu simulovať. Následne test spustíme a vidíme, že test padne. Keď chybu opravíme, spustíme test a bude úspešný, budeme vedieť, že sme chybu opravili.

3.1.3.1 Postup odstránenia chyby

1. Vytvorenie testu [1], ktorý simuluje danú chybu
2. Spustenie testu [1], výsledok – neúspešný test.
3. Oprava funkcionality v kóde (odstránenie chyby)
4. Otestovanie, či je chyba odstránená spustením testu, ktorý simuloval chybu, výsledok – úspešný test.
5. Potvrdenie zmien a nasadenie do vlastnej vetvy [2].
6. Nastav stav chyby na vyriešený (resolved).

3.1.4 Zrevidovanie opravenej chyby

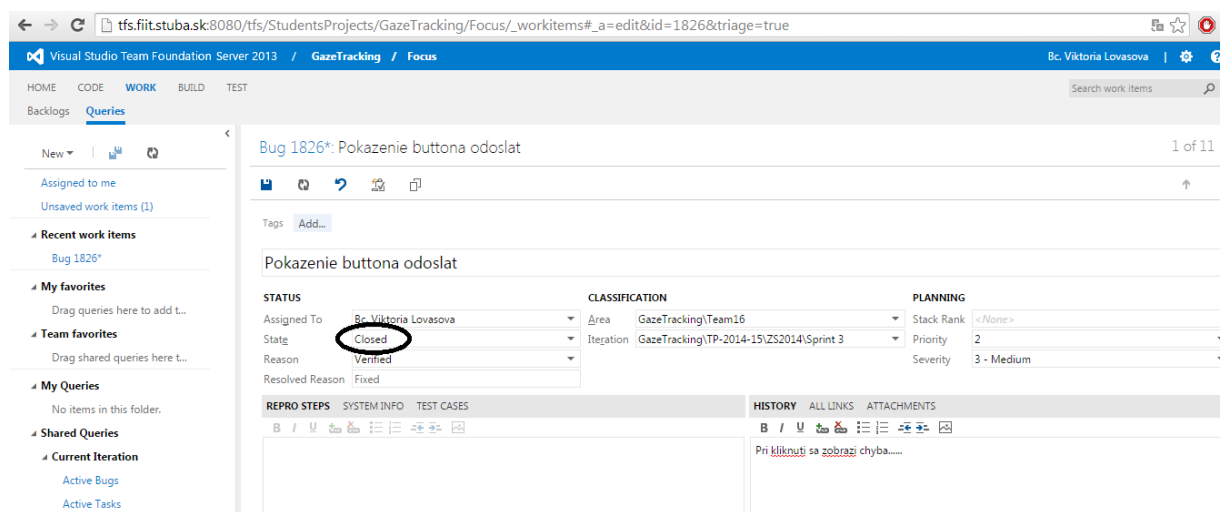
Aby bola daná oprava odsúhlasená aj ostatnými členmi tímu, musí ju prezrieť jeden člen z tímu, ktorý spraví code review, túto metodiku opísal Marek Šukek [3] a následne prezrieť a odsúhlasiť product owner.

3.1.5 Nastavenie stavu chyby ako zatvorený

Ak sa chyba vyrieši a aj tím ju skontroloval, je potrebné zmeniť jej stav na zatvorený.

3.1.5.1 Postup zmeny stavu chyby

1. Klikni na Work, Queries, Assigned to me. Zobrazia sa všetky úlohy, ktoré sú ti pridelené
3. Vyber chybu, ktorej chceš zmeniť stav
4. Otvorí sa nasledujúci formulár, pričom v stave vyber možnosť closed, Obrázok 19:



Obrázok 19: Zmenastavu chyby na zatvorený.

3.1.5.2 Stavy, v ktorých sa chyba môže nachádzať

Na začiatku, keď sa chyba vytvorí až pokým sa nevyrieši je v stave – aktívny. Do tohto stavu môže prejsť aj chyba, ktorú niekto znovu-otvorí. Keď ju člen tímu opraví, nastaví stav na vyriešený. Aby mohol byť stav chyby zatvorený, musia ju overiť v stave vyriešenom aj ostatní členovia tímu, až potom sa môže nastaviť na zatvorenú.

3.2 Dekompozícia používateľských príbehov na úlohy v nástroji TFS

3.2.1 Úvod

- Proces plánovania v našom tíme
 - Plánovanie práce v našom tíme - agilná metodika SCRUM.
 - Používateľské príbehy (ang. *user story*) si stanovujeme, plánujeme a rozdeľujeme spoločne na začiatku každého šprintu.
 - Ďalšou fázou je rozdelenie pridelených používateľských príbehov na jednotlivé úlohy (ang. *task*).
- Nástroj na manažment v tíme FOCUS
 - Naš tím v rámci agilnej metodiky SCRUM využíva pre manažment v tíme softvérový nástroj *Team Foundation Server* (TFS).
 - Celý proces funkcionálneho dekomponovania jednotlivých používateľských príbehov na úlohy sa vykonáva práve v tomto nástroji.
- Princíp metodiky (komu je určená, čo zahŕňa)
 - Princípom tejto metodiky je poskytnúť postup alebo metódu na spomínané funkcionálne dekomponovanie používateľských príbehov na menšie časti (úlohy).
 - Táto metodika je určená pre všetkých členov tímu FOCUS a budú s ňou pracovať počas celého procesu práce na projekte.
 - Táto metodika zahŕňa celý proces dekompozície používateľských príbehov od vytvorenia základnej predstavy na tímovom stretnutí až po samotné zadávanie úloh do systému TFS.
 - Metodika sa však nezaobera vytváraním používateľských príbehov ani správou úloh počas šprintu (napr. aktualizácia času a stavu úloh).
 - Aby bol celý proces dekompozície kvalitný a hlavne jednotný je potrebné aby sa presne dodržiavali všetky nasledujúce kroky.

3.2.1.1 Prvotná dekompozícia na tímovom stretnutí

- V prvom kroku si na tímovom stretnutí spoločne vytvorte základnú predstavu o aktivitách (úlohách) potrebných pre úspešné dokončenie daného používateľského príbehu.
- Usporiadajte tieto úlohy v čase. Pri ich usporiadaní je potrebné aby ste brali ohľad hlavne na závislosti medzi úlohami, keďže výstup jednej môže byť vstupom pre inú úlohu.
- Následne priradte ku každej úlohe *story point* na základe princípov metodiky SCRUM.
- Zložité úlohy, ktorým bolo priradené vysoké číslo musíte rozdeliť na menšie časti. Klasickou výnimkou takýchto úloh, ktoré nie je možné rozdeliť sú integračné úlohy.
- V prípade, že sa daná úloha skladá z viacerých aktivít vytvorte druhú úroveň podúloh. Toto však robí každý člen tímu individuálne. Avšak nikdy nerozdeľujte úlohy viac ako na dve úrovne.

Až keď máte takto definovanú a usporiadanú predstavu o tom čo bude potrebné spraviť a kedy to bude potrebné spraviť, môžete pristúpiť k samotnému vytváraniu úloh v TFS.

3.2.1.2 Vysvetlenie použitých pojmov

V rámci metodiky boli použité nasledovné odborné výrazy a skratky:

- Používateľský príbeh (ang. *user story*) – sú základné funkcie, ktoré musí systém poskytovať a predstavujú ucelenú funkcionalitu.
- Úloha (ang. *task*) – je jedna samostatná aktivita, ktorá sa musí vykonať v rámci práce na používateľskom príbehu.
- *Sprint backlog* – zoznam používateľských príbehov a prislúchajúcich úloh, ktoré by mali byť spravené na konci daného šprintu.
- *Story point* – jednotka obtiažnosti, ktorú tím priraduje k jednotlivým používateľským príbehom alebo úlohám.
- TFS (*Team Foundation Server*) – nástroj pre manažment v tíme používaný aj na zaznamenávanie používateľských príbehov a úloh a ich atribútov.

Celý dokument je rozdelený nasledovne. V kapitole 2 je opísaný celý postup funkcionálnej dekompozície používateľských príbehov na jednotlivé úlohy. V kapitole 3 sú popísané špeciálne udalosti, ktoré môžu nastať pri definovaní úloh. V kapitole 4 sú spomenuté úlohy, ktoré sa často zabúdajú pri dekompozícii uvádzať.

3.2.2 Metóda funkcionálnej dekompozície používateľských príbehov

Táto kapitola bude obsahovať podrobný opis jednotlivých krokov funkcionálnej dekompozície a metód ich vykonania aj s ohľadom na rôzne rozhodovania, ktoré je nutné v tomto kontexte spraviť.

3.2.2.1 Sprint backlog v TFS

V prvej fáze je potrebné mať otvorený adekvátny *sprint backlog* v TFS nástroji (Obrázok 20):

1. Prihláste sa do systému TFS.
2. Kliknite na projekt Gaze Tracking / Focus (bod 3 na Obrázok 20).
3. Kliknite na záložku Work (bod 1 na Obrázok 20) v ľavom hornom rohu obrazovky.
4. Zvoľte si adekvátny šprint, v ktorom chcete vytvárať úlohy. Zoznam šprintov sa nachádza taktiež na ľavej strane obrazovky (bod 2 na Obrázok 20).

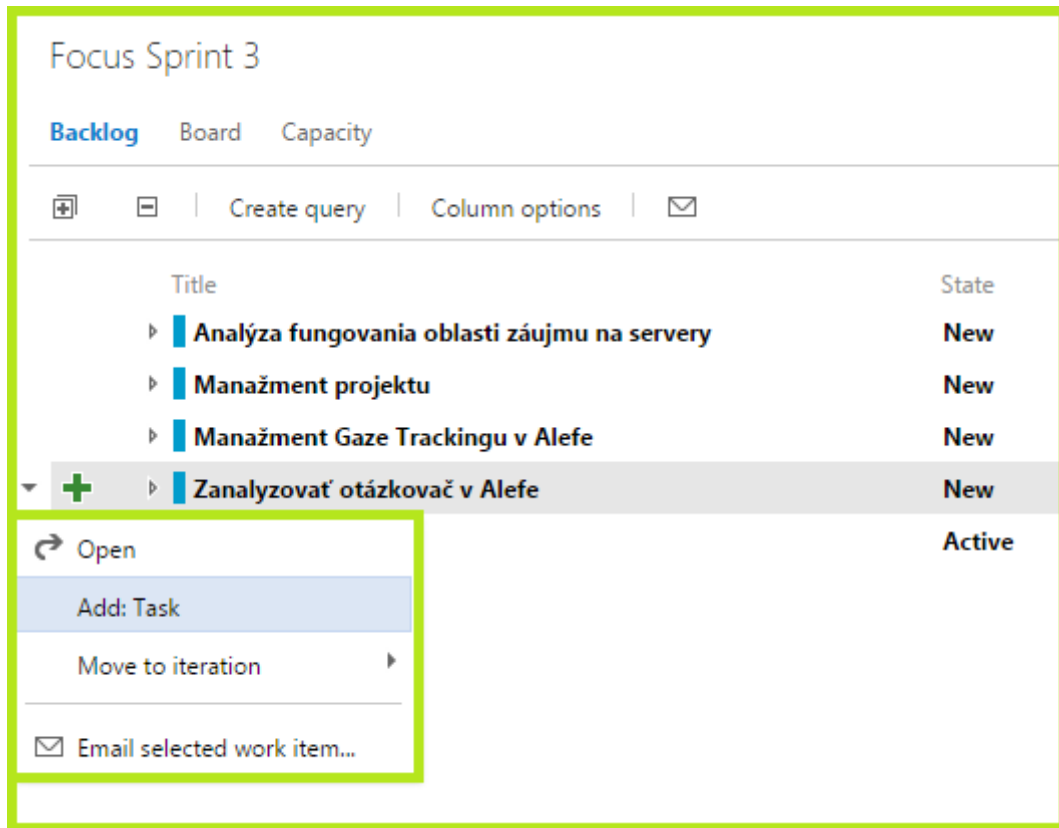
Title	State	Assigned To	Remaining W...
▶ Analýza fungovania oblasti záujmu na servery	New	Bc. Peter Kusnir	8
▶ Manažment projektu	New	Bc. Marek Sulek	13
▶ Manažment Gaze Trackingu v Alefe	New	Bc. Viktoria L...	13
▶ Zanalyzovať otázkoč v Alefe	New	Bc. Martin Sv...	13
▶ Počítanie agregácií	Active	Bc. Peter Kis	14,5

Obrázok 20: Sprint backlog.

3.2.2.2 Prístup formuláru na vytváranie úloh

Pridávanie úloh sa vykonáva prostredníctvom formuláru a preto bude prvým krokom prístup k takémuto formuláru:

1. Označte jeden z používateľských príbehov.
2. Kliknite na tento príbeh pravým tlačítkom myši.
3. Vyberte možnosť Add: Task (Obrázok 21) zo zobrazeného menu (alternatívou pre tento krok je stlačenie zeleného krížiku pri názve používateľského príbehu).



Obrázok 21: Otvorenie formulára.

3.2.2.3 Vytváranie úloh

Po úspešnom vykonaní všetkých týchto aktivít sa vám zobrazí formulár pre pridávanie úlohy (Obrázok 22). Vyplňanie jednotlivých boxov robte v uvedenom poradí aby sa predišlo zlým identifikáciám niektorých atribútov a z dôvodu predchádzaniu chybám, ktoré sa môžu takto včas identifikovať (napr. zistenie, že danú úlohu by bolo dobré rozdeliť na viac častí).

Postup:

1. Názov (ang. *Title*) – Ako prvé definujte názov úlohy, ktorý sa zadáva do prvého políčka (so žltým pozadím). Tento názov vždy uvádzajte v tvare slovesného podstatného mena (vytvorenie, zdokumentovanie, odoslanie, atď.). Vyhybajte sa príliš dlhým názvom a názvom so spojkou „a“. Takéto názvy naznačujú, že daná úloha je príliš veľká a musíte ju rozdeliť na menšie úlohy.
2. Tagy (ang. *Tags*) – Stlačte tlačítko Add..., ktoré sa nachádza nad políčkom pre zadávanie názvu úlohy. Pri tagoch vyberte jednu z ponúkaných možností alebo si zdefinujte vlastné tagy:
 - a. Klasickým príkladmi tagov, ktoré musí obsahovať každá úloha je typ tejto úlohy (napr. *Analysis, Design, Implementation, Testing*).

- b. Následne pridajte tagy opisujúce technológie, knižnice alebo aktuálne časti softvéru, s ktorými budete pracovať (napr. *coffee script*, *Backbone JS*, *database*).
 - c. Ďalšími dôležitými tagmi sú tagy predstavujúce kľúčové slová k danej úlohe (napr. *aggregation*, *AOIs*).
3. Status (ang. *Status*)
- a. Priradený k (ang. *Assigned To*) – Zadajte človeka, ktorý bude pracovať na danej úlohe. V tomto kontexte vždy zadávajte seba samého ako riešiteľa úlohy.
 - b. Stav (ang. *State*) – Zadajte stav danej úlohy. Ak úlohu práve vytvárate tak zadajte vždy stav *New*.
 - c. Dôvod (ang. *Reason*) – AK vytvárate úlohy tak táto položka bude mať vždy hodnotu *New*. Podobne ako stav má význam hlavne pri správe a aktualizácii úloh.
4. Plánovanie (ang. *Planning*)
- a. *Stack Rank* – Tento atribút podľa dokumentácie[1] predstavuje subjektívne hodnotenie obtiažnosti danej úlohy. Zadeľte túto položku vo formáte čísla z Fibonacciho postupnosti, ktorá jej podľa vás prislúcha. Je však dôležité aby ste tento atribút zadávali na základe porovnania jednotlivých úloh v danom používateľskom príbehu. Ak do tohto políčka zadáte príliš vysoké číslo, znamená to, že ste zle navrhli danú úlohu a je potrebné ju rozdeliť na menšie časti.
 - b. *Priority* (ang. *Priority*) – Následne zadajte prioritu danej úlohy vzhľadom na daný používateľský príbeh. Číslo 1 priradíte najdôležitejším úlohám.
 - c. *Activity* (ang. *Activity*) – Pri tejto položke zadajte typ aktivity, ktorá sa bude v rámci danej úlohy vykonávať. Podľa typu úlohy vyberte jednu z nasledujúcich možností: Nasadenie - *Deployment*, Návrh - *Design*, Vývoj - *Development*, Dokumentácia - *Documentation*, Požiadavky - *Requirements*, Testovanie – *Testing*.
5. Klasifikácia (ang. *Classification*)
- a. *Area* (ang. *Area*) – Toto políčko nemusíte zadávať. Tento atribút je preddefinovaný a predstavuje tím, ktorý na danom projekte pracuje.
 - b. *Iteration* (ang. *Iteration*) – Toto políčko nemusíte zadávať. Tento atribút predstavuje šprint, v ktorom sa daná úloha vykonáva.
6. Úsilie (hodiny) (ang. *Effort (hours)*)
7. Pôvodný odhad (ang. *Original estimate*) – V tomto políčku zadajte pôvodný časový odhad (v hodinách), ktorý bude predstavovať čas potrebný na dokončenie danej úlohy. Ak zadáte pôvodný odhad príliš vysoký, tak je potrebné rozdeliť danú úlohu do menších častí
- a. *Remaining* (ang. *Remaining*) – Následne zadajte zvyšný čas, ktorý vám zostáva aby bola úloha hotová. Pri vytváraní je tento atribút rovnaký ako pôvodný odhad.
 - b. *Completed* (ang. *Completed*) – Zadajte koľko času ste už strávili pri práci na danej úlohe. Pri vytváraní úlohy by mal byť tento čas „0“, keďže úlohy sa vytvárajú na začiatku šprintu pred začatím práce.
8. Opis (ang. *Description*) – V tejto časti napíšte stručný opis aktivity, ktorú budete vykonávať a popri prípade aj technológie, ktoré využijete v rámci danej úlohy.
- a. *Implementation* (ang. *Implementation*) – Túto časť nevyplňajte pri vytváraní úlohy.
9. História (ang. *History*) – Túto časť pri vytváraní úlohy nevyplňajte.

10. Po vyplnení všetkých potrebných atribútov skontrolujte všetky informácie a následne kliknite na tlačítko Save and close, ktorým uložíte danú úlohu. Jednotlivé vytvorené úlohy sa zobrazia priamo pod názvom používateľského príbehu (vid' Obrázok 23).

Obrázok 22: Formulár na pridávanie úloh.

Focus Sprint 3

Backlog Board Capacity

☰ | 📄 | Create query | Column options | ✉

	Title	State	Assigned To	Remaining W...
▶	Analýza fungovania oblasti záujmu na servery	New	Bc. Peter Kusnir	8
▶	Manažment projektu	New	Bc. Marek Sulek	13
▶	Manažment Gaze Trackingu v Alefe	New	Bc. Viktoria L...	13
▼ +	Zanalyzovať otázkoč v Alefe	New	Bc. Martin Sv...	13
	Analýza otázkoča v ALEFe	New	Bc. Martin Svrc...	4
	Odoslanie eventu otázkoča	New	Bc. Martin Svrc...	3
	Vygenerovanie otázky na základe prijatého eventu	New	Bc. Martin Svrc...	3
	Code review	New	Bc. Viktoria Lov...	1
	Zdokumentovanie ALEF otázkoča	New	Bc. Martin Svrc...	2
▶	Počítanie agregácií	Active	Bc. Peter Kis	14,5

Obrázok 23: Zobrazenie vytvorených úloh v časti Backlog.

3.2.3 Riešenie špeciálnych udalostí

3.2.3.1 Odstránenie úlohy

Odstránenie existujúcej úlohy je potrebné hlavne pri zmenách v procese alebo pri nových zisteniach, kedy sa príde na to, že danú úlohu už nie je potrebné riešiť.

1. Kliknite pravým tlačidlom na danú úlohu v časti Backlog.
2. Vyberte možnosť Open(Obrázok 21) v zobrazenom menu.
3. V zobrazenom formulári v časti Status kliknite na položku State a zvolte možnosť Removed.
4. Do časti History stručne popíšte dôvod prečo bolo potrebné zrušiť danú úlohu.

3.2.3.2 Rozdelenie úlohy na menšie časti

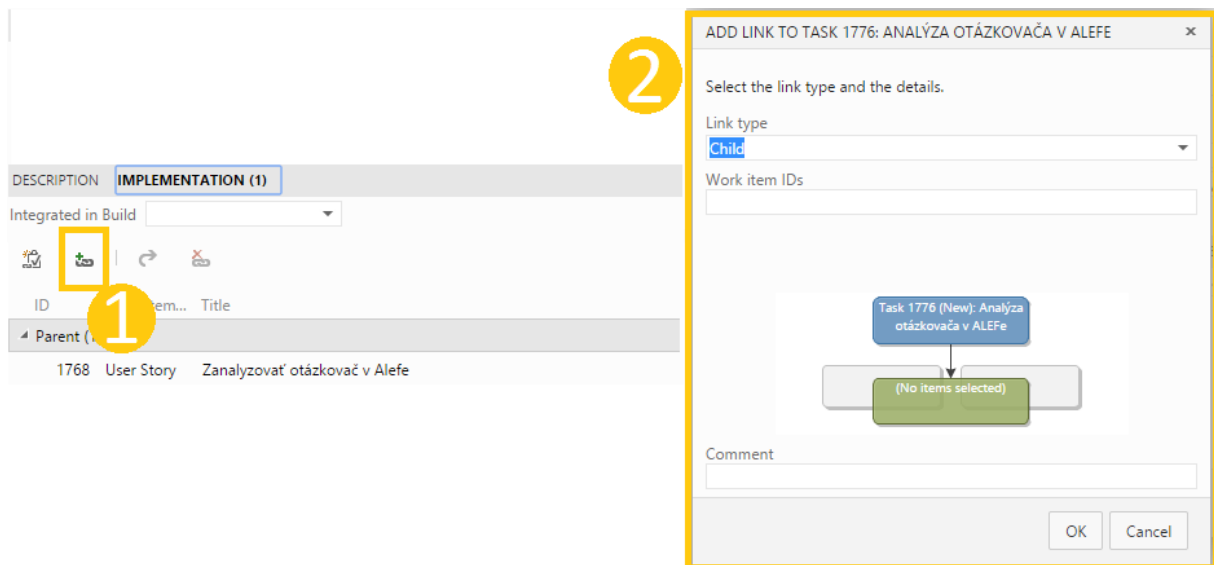
Rozdelenie úlohy je potrebné ak sa v priebehu práce na úlohe zistia nové skutočnosti, ktoré zvýšia zložitosť úlohy.

1. Určite si ako rozdelíte danú úlohu na viacero menších úloh.
2. Vykonajte udalosť *Odstránenie úlohy*.
3. Následne vytvorte novú úlohu podľa postupu v kapitole 2.3 *Vytváranie úloh*.

3.2.3.3 Závislosti úloh

Závislosti úloh pridávajúte až po vytvorení všetkých úloh v rámci používateľského príbehu.

1. Kliknite pravým tlačidlom na danú úlohu v časti Backlog.
2. Vyberte možnosť Open (Obrázok 21) v zobrazenom menu.
3. V časti Description vyberte záložku Implementation a v nej kliknite na ikonu pridania prepojenia (bod 1 na Obrázok 24).
4. V zobrazenom formulári (bod 2 na Obrázok 24) zadajte do časti Work item IDs identifikačné číslo úlohy, ktorá je závislá na aktuálnej úlohe a pridajte komentár do časti Comment s popisom danej závislosti.



Obrázok 24: Závislosti úloh.

3.2.4 Zoznam bežne chýbajúcich úloh

V tejto časti sú spísané základné úlohy, ktoré často chýbajú pri funkcionálnej dekompozícii používateľského príbehu. Pred tým ako skončíte s dekompozíciou používateľských príbehov tak sa pozrite na tento zoznam a skontrolujte či vám niektorá z nich v zozname úloh nechýba.

1. *Analýza problematiky* – Analýza oblasti, ktorá súvisí s riešením danej úlohy.
2. *Prehliadka kódu* – Kontrola kódu z hľadiska dodržiavania určitých pravidiel programovania.
3. *Testovanie* – Otestovanie vytvorenej funkcionality z hľadiska logiky riešenia.
4. *Integrácia* – Spojenie jednotlivých častí do jedného celku.
5. *Dokumentácia* – Zdokumentovanie práce na danej úlohe alebo používateľskom príbehu.

3.3 Metodika používania softvérového nástroja na správu verzií zdrojového kódu

3.3.1 Vymedzenie pojmov

Softvérový nástroj na kontrolu verzií zdrojového kódu

Týmto pojmom sa myslí taký softvérový nástroj, ktorý umožňuje udržiavať históriu jednotlivých verzií súborov so zdrojovým kódom a zároveň ich synchronizáciu s ostatnými členmi vývojového tímu. V prípade tohoto metodického pokynu bude pojednávané o nástroji GIT (ďalej len “nástroj na kontrolu verzií”).

Zdrojový kód

Pod pojmom “zdrojový kód” sa rozumejú súbory, ktoré obsahujú zdrojový kód implementovaného softvérového systému.

Repozitár

Pojmom “repozitár” je označovaná kópia zdrojového kódu, ktorá je spravovaná nástrojom na kontrolu verzií.

Vývojársky tím

Pojem “vývojársky tím” označuje členov tímu “Focus”, ktorý spolupracujú na vývoji softvérového systému, ktorého vyhotovenie je cieľom predmetu “Tímový projekt”.

3.3.2 Preambula

Tento metodický pokyn je určený na usmernenie práce vo vývojárskom tíme a týka sa všetkých jeho členov. Tento dokument určuje pravidlá pri používaní softvérového nástroja na kontrolu verzií zdrojového kódu. Zahrnuté tu sú najmä pokyny k odovzdávaniu a uverejňovaniu zdrojového kódu a manipuláciu s vetvami zdrojového kódu. V rámci tejto metodiky nie sú v žiadnom prípade pokryté pokyny pre vvytváranie a klonovanie repozitárov.

3.3.3 Metodické pokyny k odovzdávaniu zdrojového kódu

Uplatňovanie zmien v nástroji na kontrolu verzií

Zmeny v zdrojovom kóde je nutné priebežne uplatňovať v nástroji na kontrolu verzií, a to aspoň po dokončení implementácie funkcionality alebo odstránení chyby zaznamenatej v nástroji na sledovanie úloh [1], v ideálnom prípade však častejšie. Každá zmena zdrojového kódu uplatnená v nástroji na kontrolu verzií je sprevádzaná informatívnou správou. Táto správa je písaná výlučne v anglickom jazyku a jej maximálna dĺžka je obmedzená na 120 znakov. Sprievodná správa stručne identifikuje dotknutú časť zdrojového kódu a popisuje zmeny na nej vykonané.

Sprievodná správa má nasledovný formát:

[+/-/*] identifikácia dotknutej časti - popis zmien,

kde prvá časť v hranatých zátvorkách značí typ zmeny (+ — pridaná funkcionálnosť, - — odobraná funkcionálnosť, * — zmenená funkcionálnosť), druhá identifikuje dotknutú časť zdrojového kódu a tretia popisuje zmeny na nej vykonané.

Príklad:

[+] auth module - added session timeout check
[-] api module - removed user delete call
[*] api module - improved db connection settings

Uverejňovanie zmien do centrálneho repozitáru

Po každom uplatnení zmien v miestnom repozitári je nutné tieto zmeny uverejniť do vzdialeného spoločného repozitáru.

Zmeny do vzdialeného repozitáru uverejňuje každý člen tímu, prispievajúci do danej vetvy, a to výhradne do vzdialenej vetvy s rovnakým názvom, aký má miestna vetva obsahujúca uplatnené zmeny.

3.3.4 Metodické pokyny k vetveniu zdrojového kódu

Vytváranie vetiev

Pred začatím vývoja novej funkcionality je nutné vytvoriť v nástroji na kontrolu verzií novú vetvu zdrojového kódu. Každá vetva zdrojového kódu slúži na vývoj jedného uceleného bloku funkcionality. Do jednej vetvy môžu zasahovať zmenami viacerí členovia tímu, zodpovedný je za ňu však člen, ktorý je zodpovedný za vývoj funkcionality obsiahnutej v danej vetve.

Novú vetvu vytvára člen, ktorý je zodpovedný za implementáciu danej funkcionality a je ju možné vytvoriť výlučne z vetvy zdrojového kódu, v ktorej obsiahnutej funkcionality sa zmeny týkajú. V prípade potreby je možné do takto vytvorenej vetvy prebrať funkcionality z ďalších existujúcich vetiev, je však nutné dbať na stav a funkčnosť zdrojového kódu v požadovanej vetve. Po vytvorení novej vetvy je nutné túto vetvu uverejniť do vzdialeného spoločného repozitáru.

Každá vetva je pomenovaná názvom v anglickom jazyku. Názov čo najvýstižnejšie označuje funkcionality, ktorej sa týkajú zmeny zdrojového kódu v danej vetve. Názov je jednoslovný, v prípade potreby je možné medzery v názve nahradiť podtržníkmi. Maximálna dĺžka názvu vetvy je obmedzená na 20 znakov.

Príklad:

auth_module
api_module
frontend

Spájanie vetiev

Po dokončení vývoja funkcionality v niektorej z vetiev je potrebné tieto zmeny zlúčiť s vetvou, z ktorej bola táto vetva na začiatku vývoja vytvorená. Spájanie jednotlivých vetiev vykonávajú výlučne členovia vývojárskeho tímu, ktorý sú za dotknuté vetvy zodpovední.

Pred samotným spájaním vetiev je nutné prebrať zo vzdialeného repozitáru aktuálne verzie obidvoch dotknutých vetiev.

Pri spájaní vetiev zdrojového kódu môže dôjsť ku konfliktom medzi jednotlivými zmenami vykonanými na zdrojovom kóde. V niektorých prípadoch je tieto konflikty možné odstrániť automaticky za pomoci nástroja na kontrolu verzií. V opačnom prípade je nutné konflikty odstrániť ručne. Pri ručnom odstraňovaní konfliktov je potrebné použiť softvérový nástroj na tvorbu zdrojového kódu [2].

Pri odstraňovaní konfliktov je potrebné dbať na zachovanie funkcionality z obidvoch dotknutých vetiev. O prípadnom vynechaní funkcionality môže rozhodnúť výlučne člen tímu za danú funkcionality zodpovedný.

Po dokončení spájania vetiev je nutné na novo vzniknutom zdrojovom kóde vykonať jednotkové testy funkcionality. V prípade úspešného dokončenia testov je potrebné spojenú vetvu uverejniť vo vzdialenom spoločnom repozitári.

3.3.5 Súvisiace zdroje

1. Metodický pokyn k vytváraniu a správe úloh v softvérovom nástroji na manažment projektov
2. Metodický pokyn k písaniu zdrojového kódu v jazyku C#.

3.4 Metodika prehliadky kódu

3.4.1 Úvod

Účelom tejto metodiky je definovať postup pri prehliadkach kódu, za cieľom aby zachovania dostatočná kvalita zdrojového kódu. Metodika obsahuje postup, ktorý sa musí dodržiavať pri prehliadke kódu a opis práce s nástrojmi, ktoré sa pri prehliadke musia použiť.

3.4.2 Prehliadka kódu

3.4.2.1 Vymedzenie používateľov metodiky

Táto metodika je určená pre všetkých programátorov, ktorý budú písať nový kód alebo budú vykonávať prehliadku zdrojových súborov napísaných v jazyku C# pre project GazeTracking.

3.4.2.2 Súvisiace manuály

Metodika vychádza z nasledujúcich manuálov:

- **CodeReview Používateľská príručka projektu perconik**
- **Používateľské značky projektu perconik**

3.4.2.3 Použité pojmy

- Perconik - Nástroj na vytváranie a monitorovanie značiek v zdrojovom kóde
- Značka - Príkaz označujúci časť kódu, nesúci v sebe informáciu (z angličtiny tag)

3.4.2.4 Roly zahrnuté do prehliadky kódu

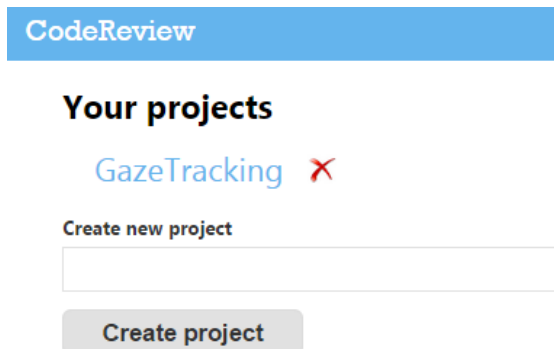
Tabuľka 3: Roly členov tímu.

Rola	Zodpovednosti
Manažér monitorovania projektu	Pridávanie používateľov do projektu v aplikácii perconik Vytýčenie cieľov prehliadok kódu Určenie osôb vykonávajúcich prehliadky kódu Výber používaných nástrojov
Prehliadajúci programátor	Identifikovanie problému v kóde Zapísanie príslušných značiek do perconic
Autor kódu	Úprava kódu na základe prehliadky Označenie kódu, ktorý je pripravený na prehliadku

3.4.2.5 Prehliadka kódu pomocou nástroja perconic

Prehliadka kódu sa uskutočňuje pomocou nástroja perconic, ktorý je dostupný na stránke <https://perconik.fiit.stuba.sk/CodeReview/Home/Index>. Do systému sa prihlasuje pomocou prihlasovacích údajov akademického informačného systému.

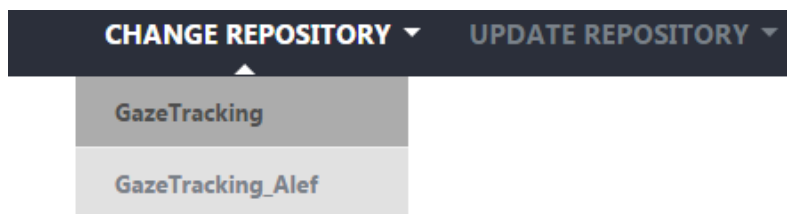
Prvé prihlásenie Po prihlásení kontaktujte Bc. Mareka Šulek pomocou správy v akademickom informačnom systéme na xsulek@fiit.stuba.sk pre pridanie do projektu GazeTracking. Po prihlásení a pridaní vášho účtu do projektu GazeTracking si vyberte projekt zo zoznamu projektov pod nadpisom Your projects (Obrázok 25)



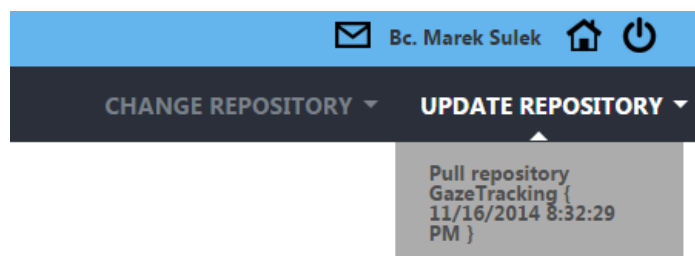
Obrázok 25: Výber projektu.

Výber a aktualizácia repozitára

Po výbere projektu si zvolte repozitár pomocou horného menu. Vyberte CHANGE REPOSITORY a volte GazeTracking (Obrázok 26). V ďalšom kroku si aktualizujte repozitár kliknutím na UPDATE REPOSITORY a následne vyberte Pull repository GazeTracking (Obrázok 27).



Obrázok 26 - Výber repozitáru

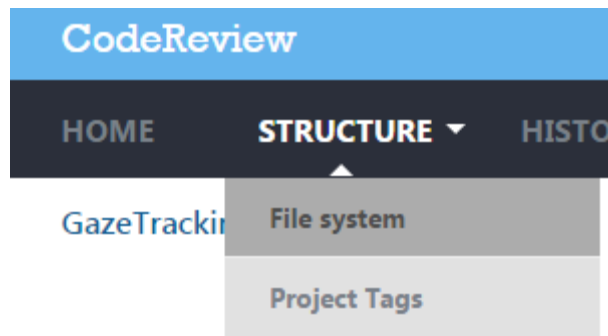


Obrázok 27: Pull repository.

Označenie kódu na prehliadku

Každý kód, ktorý je určený na prehliadku musíte označiť pomocou značiek v nástroji perconik. V menu nástroja perconik vyberte STRUCTURE a následne File system (Obrázok 28). Vyberte vetvu Team16 a kliknite na Change branch!. Pokračujte zobrazením vami vytvoreného kódu, pomocou stromového zoznamu projektu v ľavej časti obrazovky aplikácie

peconik (Obrázok 29). Následne sa zobrazí zdrojový kód. Označte časť kódu, ktorá je určená na prehliadku, zobrazí sa vám kontextová ponuka Add tag. Kliknite na ňu. Z combo boxu vyberte možnosť CODEREVIEW a do popisu napíšte TODO.15



Obrázok 28: Otvorenie súborov projektu.

Vyhľadanie kódu na prehliadku

V menu perconik vyberme STRUCTURE a klikneme na Project Tags (Obrázok 28). Vyberieme vetvu Team16 kliknutím na combo box Branches. Následne vyberieme typ značky CODEREVIEW z combo boxu type. Spustíme vyhľadávanie kliknutím na tlačidlo Submit. Zo zoznamu si vyberajte tie, ktoré v sebe obsahuje položku TODO(Obrázok 29). Vyberte z príslušného zoznamu, položku ktorú chcete prehliadnuť. Kliknutím na šípku sa vám zobrazí príslušný kód.

Kontrola kódu - pridanie značiek

Skontrolujte daný kód podľa konvencii ktoré sú v určene v týme. Kliknutím na čierny kruh, ktorý znázorňuje značku a výberom Delete zmažte značku. Ak kód nie je správny, pridajte značku pre označenie nesprávneho kódu. Vyznačte danú časť kódu a kliknite na Add tag. Vyberte možnosť CODEREVIEW. Prvá povinná položka je pre koho je značka určená. Vložte zavináč a vyberte člena tímu, ktorý ma kód opraviť. Napíšte #Category() a do zátvorky vložte atribút, čo je v kóde zlé. Zoznam všetkých atribútov a ich význam je v Obrázok 29. Nasleduje komentár, ktorý podrobnejšie opisuje chybu. V komentári nepoužívajte diakritiku! Príklad je zobrazený na Obrázok 30. Potvrďte Add tag.

GazeTracking - GazeTracking

List of project tags

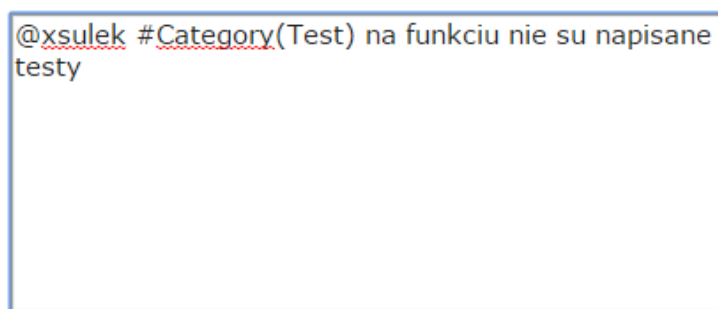
Filter
Branches: Team16 ▾ Type: CODEREVIEW ▾ Author:
All authors ▾ Submit

Files with tags

Api.svc.cs

- Bc. Marek Sulek: CODEREVIEW CODEREVIEW TODO | xsulek 2014-11-16T21:53:39.0448169Z ↻
- Bc. Marek Sulek: CODEREVIEW CODEREVIEW #Category(ErrorHandling) Skuska | xsulek 2014-11-16T16:36:54.3820796Z ↻
- Bc. Marek Sulek: CODEREVIEW CODEREVIEW @xsulek #Category(Test) na funkciu nie su napisane testy | xsulek 2014-11-16T...

Obrázok 29: Nájdenie súboru v štruktúre.



Obrázok 30: Príklad značky.

3.4.2.6 Atribúty

- Category – Kategória pripomienky
- Documentation – dokumentácia; kód je nedostatočne zdokumentovaný
- Name – zlé pomenovanie; premennú/triedu/... je potrebné premenovať
- Threading – porušenie pravidiel práce s vláknami
- Test – ku kódu je potrebné doplniť testy
- Logging – chýbajúce/nevhodné zaznamenávanie udalostí
- Recommendation – odporúčanie na zlepšenie
- Configuration – konštanty by mali byť definované v konfiguračných súboroch
- Logic – chyba v logike riešenia/algoritmu
- Security – porušenie bezpečnostných pravidiel
- API – nevhodné použitie knižnice, služby, ...
- DataStructure – nevhodné použitie dátovej štruktúry
- ErrorHandling – porušenie pravidiel spracovania výnimiek a chýb
- Readability – zle čitateľný kód. Napr. z dôvodu použitia príliš dlhýchkonštrukcií, nevhodného formátovania, ...

- Style – všeobecné porušenie štýlu písania zdrojového kódu
- Async – zlé spracovanie asynchrónnych udalostí
- Scope – nevhodné použitie modifikátora prístupu
- Perf – možný problém s výkonnosťou
- CVE – typ zraniteľnosti podľa CVE (<http://www.cvedetails.com>). V prípade, že je známa konkrétna zraniteľnosť, môže byť jej identifikátor uvedený v atribúte CRV.
- DoSo CodeExecution
- Overflow
- MemoryCorruption
- SqlInjection
- XSS
- DirectoryTraversal
- HttpResponseSplitting
- BypassSomething
- GainInformation
- GainPrivileges
- CSRF
- FileInclusion

3.5 Písanie zdrojových kódov v jazyku C#

3.5.1 Predslov

Táto metodika je určená pre členov tímu Team Focus, predovšetkým však pre tých, ktorí vytvárajú, upravujú a kontrolujú zdrojové kódy v jazyku C#. Metodika zahŕňa pravidlá pre tvorbu názvoslovia, písanie komentárov, formátovanie zdrojových textov a používania dátových typov. Metodika neobsahuje popis syntaxe jazyka C#, návod na písanie optimálneho kódu, ani postupy pri písaní aplikačnej logiky. Popísané pravidlá vychádzajú z oficiálnej príručky písania zdrojových kódov v jazyku C# od spoločnosti Microsoft ([C# Coding Conventions](#)). Niektoré pravidlá boli upravené pre potreby tímu Team Focus, nakoľko pri implementácii dorábame do už existujúceho kódu s určitými zažitými pravidlami a nechceme narušiť integritu zdrojových kódov.

3.5.2 Pri tvorbe názvoslovia ...

... sa musí:

- Všetky názvy píšete v angličtine.
- Všetky názvy musia mať samovypovedaciu hodnotu.
- Pre pomenovania tried, metód, konštánt a read only premenných používajte PascalCase notáciu, teda aby všetky slová tvoriace názov objektu začínali veľkým písmenom.
- Pre pomenovania argumentov metód, lokálnych a členských premenných používajte camelCase notáciu, teda aby všetky slová tvoriace názov objektu, okrem prvého, začínali veľkým písmenom.
- Všetky názvy rozhraní píšete s prefixom "I".
Skratky v názvoch používajte iba ak ide o všeobecne používané skratky, ako sú napr. url, xml, ftp,... alebo ak ide o projektové skratky ako sú napr. aoi, dto.
Všetky súbory so zdrojovými kódmi pomenúvajte rovnako ako ste v nich pomenovali zadanú verejnú triedu.

... sa nesmie:

- Zásadne nepoužívajte Maďarskú notáciu, teda nedoplňajte názvy premenných prefixom označujúcim dátový typ.
- Nepoužívajte v názvoch vlastné/vymyslené skratky.
- Neoddeľujte jednotlivé slová tvoriace názov objektu pomlčkami, podčiarkovníkmi
Žiadne názvy nepíšete VEĽKÝMI písmenami.

3.5.3 Pri písaní komentárov

... sa musí:

- Všetky komentáre píšete v angličtine.
- Všetky triedy a ich metódy okomentujte dokumentačnými XML komentármi opisujúcimi ich správanie/funkciu/zmysel.
- Všetky zložitejšie konštrukcie aplikačnej logiky doplňte o komentár, popisujúci funkčnosť danej konštrukcie.

- Všetky zakomentované kusy kódu doplňte o komentár, vysvetľujúci zmysel zakomentovania daného kódu.
- Všetky komentáre obsahujúce značky TODO, HACK, UNDONE doplňte o vysvetľujúci popis.
- Všetky komentáre píšete vždy hneď pred, alebo vedľa zdrojového kódu, ku ktorému sa daný komentár viaže.
- Samotný text komentáru vždy začnite veľkým písmenom

Medzi komentárovú značku a text komentáru vložte vždy jednu medzeru

... sa nesmie:

- Nepoužívajte vnhiezdené komentáre.

3.5.4 Formátovanie zdrojových textov ...

... sa musí:

- Na odsadzovanie blokov zdrojových kódov používajte 4 medzery.
- Otváracie a uzatváracie kučeravé zátvorky umiestňujete pri viac riadkových blokoch textu vždy na samostatný nový riadok.
- Deklarácie použitých knižníc musia byť od zvyšku zdrojových kódov oddelené jedným prázdny riadkom.
- Členské premenné uvádzajte vždy na začiatku tried.
- Členské premenné oddeľujte od metód triedy jedným prázdny riadkom.
- Jednotlivé metódy oddeľujte od seba presne jedným prázdny riadkom.
- Jednotlivé bloky kódu obsahujúce ucelenú logiku oddeľujte od ďalšieho kódu jedným prázdny riadkom.
- Jeden súbor so zdrojovými kódmi musí obsahovať najviac jednu definíciu verejnej triedy.
- Všetky parametre metód oddeľujte od seba čiarkou a medzerou tak, že za čiarka nasleduje vždy za parametrom a medzera nasleduje vždy za čiarkou.
- Vetvenia (if, else if) a cykly (for, while, foreach) oddeľujte od prislúchajúcich guľatých zátvoriek jednou medzerou.
- Každú podmienku pri vetvení obalujte do guľatých zátvoriek.
- Každý riadok súboru so zdrojovými kódmi musí obsahovať najviac 140 znakov.
- V prípade príkazov presahujúcich 140 znakov rozdeľte tento príkaz do viacerých riadkov tak, že na každý riadok umiestnite čo možno najviac ucelenú časť príkazu.
- Na každý riadok súboru so zdrojovými kódmi píšete najviac jeden príkaz.
- Na každý riadok súboru so zdrojovými kódmi píšete najviac jednu deklaráciu.

... sa nesmie:

- Na odsadzovanie blokov zdrojových kódov nepoužívajte symbol tabulátoru.
- Medzi deklarácie knižníc nepíšete nepoužité/zbytočné knižnice.
- Indexy v hranatých zátvorkách neoddeľujte od hranatých zátvoriek medzerou.
- Guľaté zátvorky neoddeľujte od parametrov medzerou.
- Meno funkcie a k nej prislúchajúce guľaté zátvorky s parametrami neoddeľujte medzerami.

3.5.5 Používanie dátových typov...

... sa musí:

- Používajte preddefinované primitívne dátové typy miesto ich systémových obalovacích objektov.
- Pri deklarácii lokálnych premenných používajte implicitný dátový typ var.
- Implicitný dátový typ var používajte aj pri deklarácii riadiacich premenných vo vnútri cyklov for a foreach.
- Implicitný dátový typ var používajte aj pre uchovávanie výsledku LINQ dopytov.
- V prípade potreby vrátenia viacerých hodnôt v metódach, používajte zložený dátový typ Tuple.
- V prípade veľkých reťazcov používajte objekt StringBuilder namiesto primitívneho dátového typu string alebo objektu String.

... sa nesmie:

- Nepoužívajte implicitný dátový typ var namiesto primitívnych dátových typov.
- Nepoužívajte implicitný dátový typ var ako dynamický dátový typ.
- Pri deklarácii premenných primitívnych dátových typov nepriradzuje premenným ich prednastavenú hodnotu.
- Zložený dátový typ Tuple nepoužívajte mimo návratovú hodnotu metód.

3.5.6 Ukázkový zdrojový kód

```
using System;
using System.Collections.Generic;
using System.Linq;
using ViewTracking.Infrastructure.Dto.Account;
using ViewTracking.Infrastructure.Dto.Message;
using ViewTracking.Infrastructure.Dto.Project;

namespace ViewTracking.Infrastructure.DataAccessLayer.AggregationRepository
{
    /// <summary>
    /// Contains methods for manipulating aggregations of gaze data
    /// </summary>
    public class AggregationRepository : GlobalRepository, IAggregationRepository, IDisposable
    {
        private readonly ModelContainer AggregationContext;

        /// <summary>
        /// AggregationRepository constructor
        /// </summary>
        public AggregationRepository()
        {
            AggregationContext = new ModelContainer();
        }

        /// <summary>
        /// Get AOI statistics for session between selected frames
        /// </summary>
        /// <param name="AOIId"></param>
        /// <param name="sessionId"></param>
        /// <param name="startFrame"></param>
        /// <param name="endFrame"></param>
        public void GetAOITimeStatistic(int AOIId, int sessionId, int startFrame, int endFrame)
        {
            // LINQ select to get AOI statistics from database
            var statistic = (from aggregation in AggregationContext.Aggregations
                            where aggregation.AreasOfInterestId == AOIId
                                  && aggregation.SessionsId == sessionId
                                  && startFrame > aggregation.FrameNumber
                                  && endFrame < aggregation.FrameNumber
                            group aggregation by aggregation.UsersId into b
                            select new
                            {
                                userId = b.Key,
                                duration = b.Sum(d => d.Duration)
                            }).ToList();

            for (int i = 0; i < statistic.Count; i++)
            {
                statistic[i].duration *= 0.9; // TODO optimisation and refactoring
            }
        }
    }
}
```

Obrázok 31: Vzorová ukážka.

3.6 Metodika integrácie

3.6.1 Vymedzenie metodiky

Metodika opisuje postup ako nasadiť doimplementovanú funkcionálnosť, poprípade zmenu, na server. Toto nasadzovanie sa týka iba vývojovej vetvy. Je určená pre vývojárov v Tíme 16. Postup je opisovaný v prostredí Visual Studio 2013 za použitia technológie Team Foundation Server.

3.6.2 Požiadavky pred začiatkom integrácie

Pred tým, ako začnete integrovať, tak je nutné mať splnené nasledujúce požiadavky. Ako prvé musíte mať úspešne skompilované lokálne riešenie s najaktuálnejšou verziou z developerskej vetvy. Ďalej je nevyhnutné mať otestovanú pridávanú funkcionálnosť a mať úspešný code review. Správny postup na code review sa nachádza v metodike Focus_Sulek_Metodika_Code_Review.

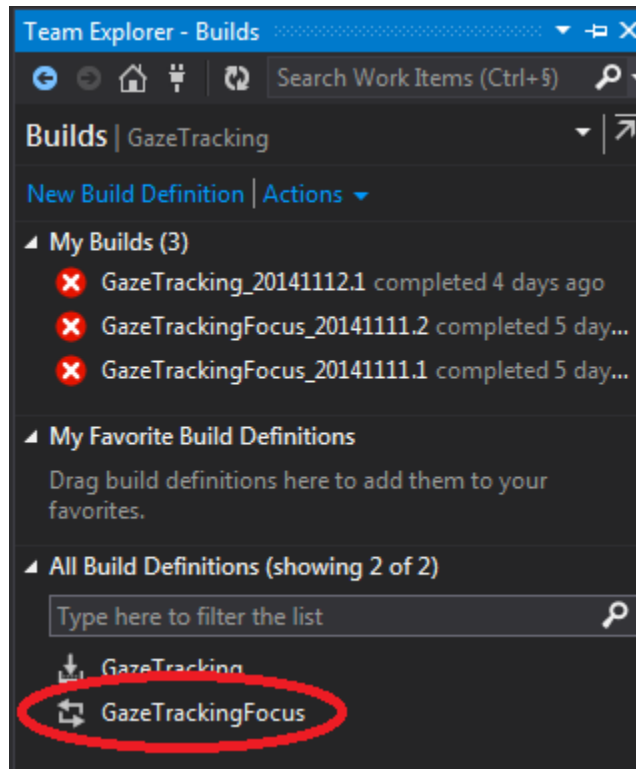
3.6.3 Postup

1. Pripojenie vetvy

Aby ste mohli napojiť vašu aktuálnu vetvu na jej nad-vetvu, tak musí byť daná vetva najskôr schválená vlastníkom produktu. V našom tíme sa vykonáva serverová kompilácia až na hlavnej vývojovej vetve. Vykonáva sa automaticky potom, ako sa na vetve vykoná zmena. Zmena znamená pripojenie vetvy, alebo “commit” zmeny. To znamená, že v prípade, že pripájate vetvu na hlavnú vývojovú vetvu, alebo vykonávate zmenu na hlavnej vetve, tak musíte skontrolovať správnosť a funkčnosť zmeny. Názov našej vývojovej vetvy je TeamFocus.

2. Kompilácia a testovanie

Ako bolo spomínané, kompilácia sa vytvára automaticky. Taktiež testovanie prebieha automatizovane. Kompilácia sa zaraďuje do poradovníku. V prípade, že chcete kompiláciu spustiť ešte raz tak musíte použiť nasledujúce prednastavené kompilácie vyznačené na *Obrázok 32*.

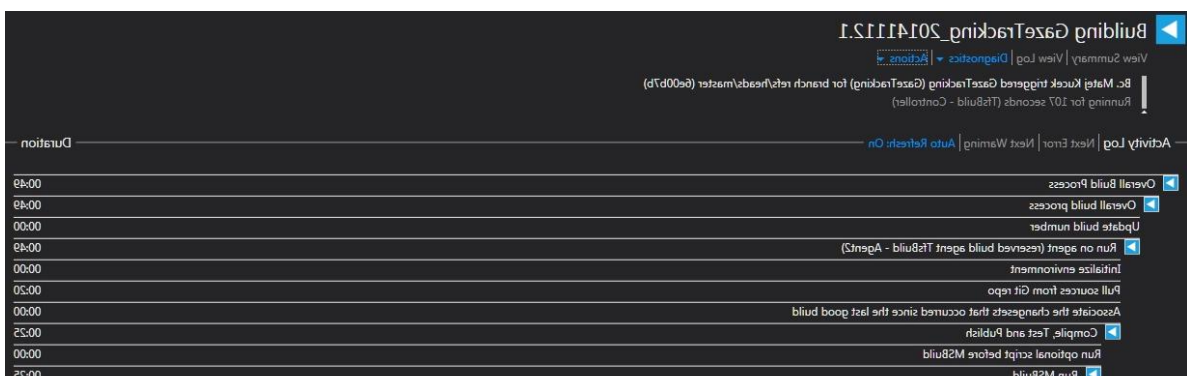


Obrázok 32: Prednastavené kompilácie.

Stav kompilácie potom sledujte v poradovníku kompilácií, alebo v jej detaľnom popise, ukážka poradovníku sa nachádza na Obrázok 33a ukážka priebehu kompilácie na Obrázok 34.



Obrázok 33 - poradovník kompilácií



Obrázok 34: Priebeh kompilácie.

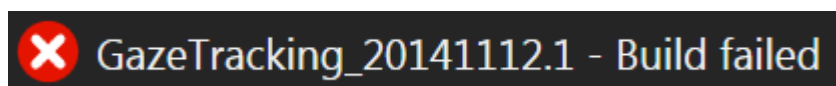
Pokiaľ výsledok ukáže, že kompilácia prebehla úspešne, tak je nasadenie novej verzie na vývojovú vetvu serveru úspešné a týmto ste skončili v rámci činnosti. V opačnom prípade pokračujte ďalším bodom.

3. Neúspešné nasadenie

Existujú iba dve možnosti pre neúspešne skompilovanie. Prvou je, že sa nepodarilo skompilovať riešenie a druhou, že nezbehli testy. V oboch prípadoch však platí, že nesmiete odkladať opravu závady na dlhšiu dobu ako je 1 hodina. V prípade, že to do tejto doby splniť neviete, tak musíte vývojovú vetvu navrátiť späť na poslednú funkčnú veriu.

4. Neúspešné skompilovanie

V prípade neúspešného skompilovania vám server zobrazí nasledujúcu chybovú hlášku (Obrázok 35):



Obrázok 35: Oznam o neúspešnej kompilácii.

Neúspech môže mať na svedomí viacero príčin:

- Medzi majoritné príčiny patrí chybný kód.
- Ďalšou notorickou chybou môže byť, že pri lokálnej kompilácii boli k dispozícii iné knižnice ako pri serverovej kompilácii. Knižnice môžu okrem toho chýbať, alebo mať inú verziu.
- Lokálna kompilácia prebiehala na inej verzii ako je vývojová. Preto fungovala lokálne, ale nie globálne.
- Rozdielne nastavenia

Ďalšie prípadné chyby sa nachádzajú v tomto odkaze: <http://www.timstall.com/2011/12/10-reasons-why-build-works-locally-but.html>

Čo robiť po identifikovaní chyby

V prípade, že identifikujete chybu, ktorá nie je charakteru chybného kódu, tak nesmiete samozvane sa snažiť opraviť chybu nahrávaním knižníc, alebo iným zásahom do serverovej časti. Je vyžadované, aby ste v takejto situácii kontaktovali buď vlastníka produktu, alebo manažéra integrácie.

Ak je chyba v zdrojovom kóde tak ju môžete opraviť bez upovedomenia. Odporúča sa však konzultovať s autorom daného kódu, ak ním nie ste vy.

1. Nespravené testy

Globálne najčastejšiou príčinou neúspešných testov býva chybný kód. Okrem toho môže byť problémom nezhoda v lokálnych a serverových testoch. Príčinou môže byť aj niektorá z vyššie uvedených, napríklad, že rôzne verzie knižnice majú obmenenú funkcionality rovnakej funkcie. Opäť ako v predošlom kroku, je zakázané zasahovať do knižníc samovoľne, ale treba dať upovedomenie kompetentnej osobe, pod ktorou rozumieme vlastníka produktu, alebo manažéra integrácie.

2. Čo robiť, keď sa chyba objaví až pri behu

Častokrát býva situácia, v ktorej kompilácia aj testovanie prebehne v poriadku, napriek tomu, že v kompilácii môže byť chyba. Takáto chyba vypláva na povrch až pri reálnom používaní. Opäť platí podobný postup ako v predošliach prípadoch s oboznámením situácie relevantných osôb. Ak pôjde o častú chybu, ktorá sa už viac krát opakovala, tak to nesmiete nechať len tým, že chybu odstránite, ale musíte buď vy pridať do tejto metodiky postup, ktorý zabezpečí prevenciu danej chyby, alebo vytvoriť test, ktorý by tejto chybe dokázal zamedziť. V prípade, že nie ste kompetentný, alebo z iného dôvodu nemôžete toto vykonať, tak musíte upovedomiť niekoho, kto toto urobí, alebo minimálne takúto prevenciu chyby zdokumentovať v podobe tasku na tfs.

4. Integračné testy

Integračné testy nie sú povinnosťou, preto ich zatiaľ ani nemáme vytvorené. Avšak v prípade, že máte pochybnosti o funkčnosti, alebo si myslíte, že integračné testy sú potrebné, prípadne vhodné tak môžete tieto testy navrhnúť a vytvoriť, pod podmienkou, že oboznámite manažéra integrácie, alebo vlastníka produktu o vami pridávaných testoch. Okrem vytvorenia musíte aj zdokumentovať tieto testy do tejto metodiky. Pod dokumentáciou rozumiem opísať možnosť a význam využitia týchto testov.

5. Nasadenie

Nasadenie prebehne automaticky potom ako sa úspešne dokompiluje. Nie je preto nutné vyvíjať žiadnu nadbytočnú aktivitu.

3.7 Metodika testovania

3.7.1 Vymedzenie pojmov

TDD – Vývoj riadený testami (z angl. test driven development) je jeden z prístupov k tvorbe softvéru

AAA – z angl. Arrange-Act-Assert (vo voľnom preklade stanov-konaj-osveč) je vzor bežne používaný na písanie jednotlivých metód unit testov

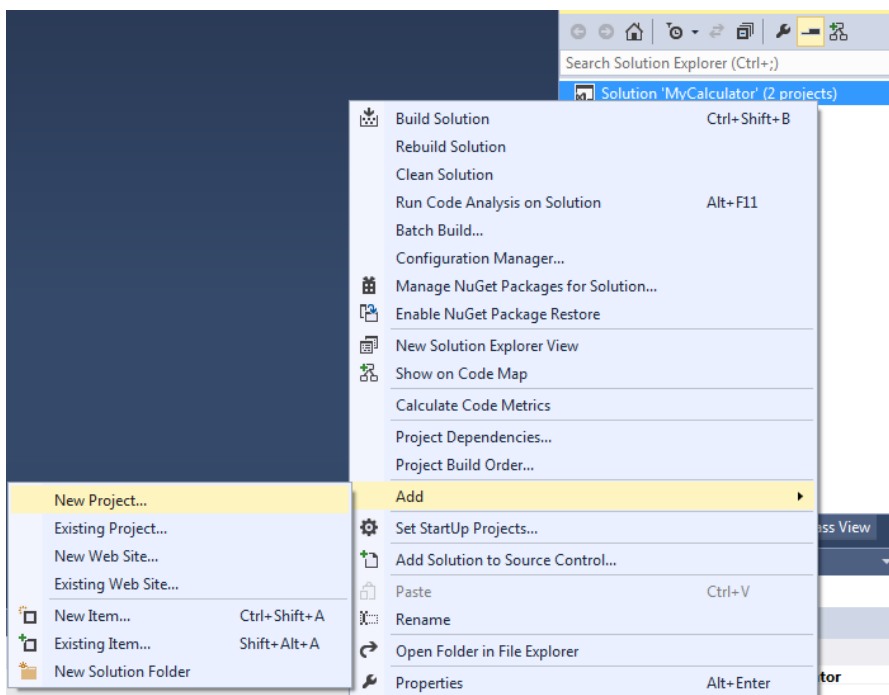
3.7.2 Predslov

Pri vývoji softvéru sa bude tím riadiť metódou TDD, ktorá sa opiera o opakovanie veľmi krátkych vývojových cyklov. Najprv vývojár napíše automatizované testy (na začiatku neúspešné), ktoré definujú potrebu zlepšenia alebo pridania funkcionality. Následne vytvorí minimálne množstvo kódu, ktorým je schopný úspešne zvládnuť vytvorené testy. Tento dokument popisuje proces vytvárania testovacích projektov a testov pre jednotlivé komponenty aplikácie napísanej v jazyku C#. Zachytáva postup tvorby testu, jeho členenie, konvencie v pomenovávaní jednotlivých testovaných častí a spúšťanie testov. Nasledujúce pravidlá sú záväzné pre členov tímu, ktorí sa podieľajú na vývoji častí aplikácie písanej v jazyku C# a využívajú vývojové prostredie Microsoft Visual Studio 2013. Nástrojom na tvorbu testov je Microsoft Unit Testing Framework, ktorý je súčasťou zvoleného vývojového prostredia.

3.7.3 Vytvorenie projektu s testami

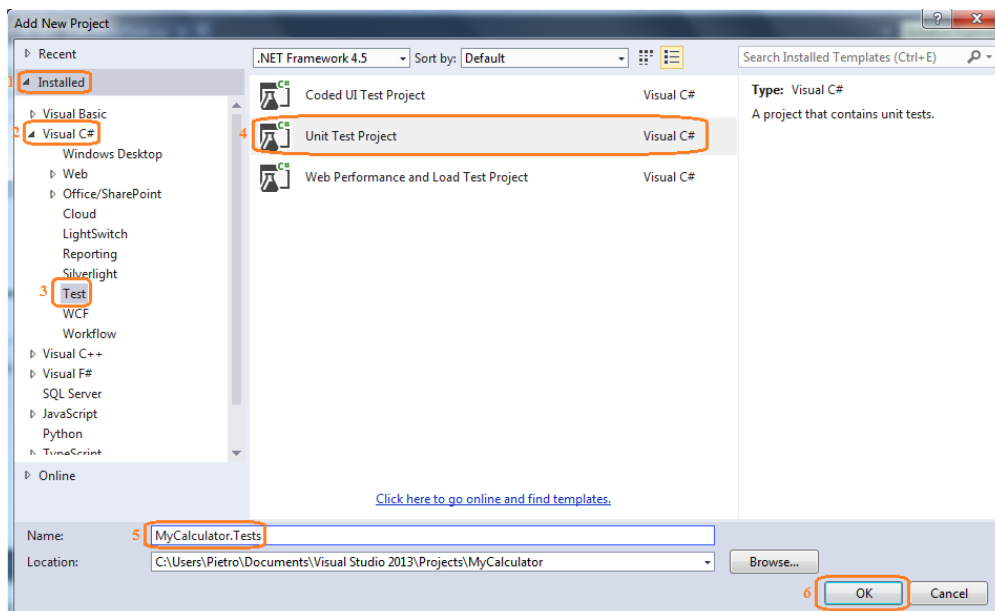
Štruktúra projektu s testami odráža existujúcu štruktúru testovaného projektu. Dbajte na to pri vytváraní hierarchie zdrojových súborov v testovacom projekte.

1. Pre existujúce riešenie (solution), kliknite pravým tlačidlom myši na názov riešenia v prieskumníkovi. V zobrazenej ponuke zvolte postupne možnosť *Add New Project...*



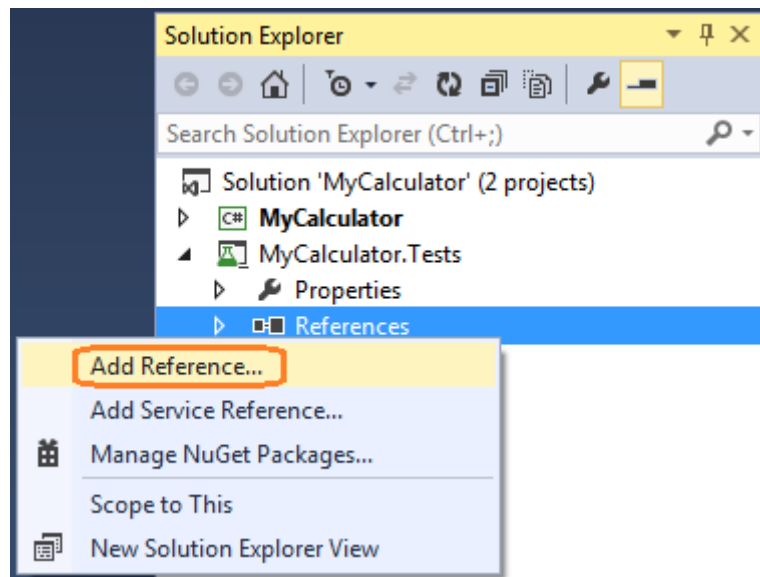
Obrázok 36: Pridanie nového projektu.

2. V zobrazenom okne zvolíte v ľavej časti sekciu *Installed*. Pokračujte výberom jazyka používaného v testoch, v našom prípade *Visual C#*. V zvolenej sekcii vyznačte podsekcii *Test* a z ponúknutých možností zakliknite *Unit Test Project*. Pomenujte vytváraný projekt s testami nasledovne <názov testovaného projektu>.Tests. Dokončíte vytváranie projektu kliknutím na tlačidlo *OK* v pravom dolnom rohu okna.



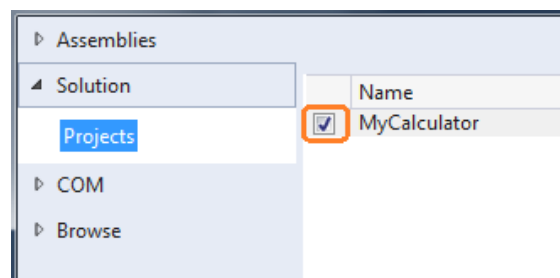
Obrázok 37: Vytváranie projektu.

3. Vo vytvorenom projekte pridajte referenciu na projekt, ktorý chcete testovať. Odkaz na projekt s testami v prieskumníkovi rozkliknite, kliknite pravým tlačidlom na odkaz *Reference* a zvolte možnosť *Add Reference...*



Obrázok 38: Pridanie referencie na projekt.

4. V ľavej časti zobrazeného okna rozkliknite položku *Solution* a zaznačte položku *Projects*. Z ponúknutých projektov zaškrtnite ten, pre ktorý chcete vytvárať testy. Výber potvrdíte kliknutím na tlačidlo *OK* v pravom dolnom rohu okna.



Obrázok 39: Výber projektu na testovanie.

3.7.4 Písanie testov

Snažte sa testami pokryť čo najväčšiu časť testovaného projektu, avšak zamerajte sa na funkcie aplikačnej logiky a funkcie nevyhnutné k chodu aplikácie.

1. Súbor s testami pomenujte podľa konvencie <názov testovaného súboru>Tests. Rovnako postupujte aj pri pomenovávaní testovanej triedy, čiže <názov testovanej triedy>Tests. V prípade konkrétnej metódy používajte meno testovanej metódy rozšírené o názov testovanej situácie, teda <názov testovanej metódy><názov testovanej situácie>. V prípade testovania základnej funkcionality metódy použijte ako názov testovanej situácie slovo *Test*.

- Do testovacieho súboru nezabudnite uviesť direktívu na názvoslovie testovaného súboru použitím kľúčového slova *using*.
- Nad názov testovacej metódy uveďte atribút metódy *[TestMethod]*. Keďže môže byť na jednu metódu napísaných viac testov, pod atribút *[TestMethod]* uveďte atribút *[TestCategory("<názov testovanej metódy>")]* v uvedenom tvare, ktorým budete značiť príslušnosť testovacej metódy k testovanej metóde. V prípade, že výsledkom metódy má byť vyhodenie výnimky uveďte nad funkciu atribút testovacej metódy *[ExpectedException(typeof(<názov výnimky s ktorou má skončiť testovaná funkcia>))]*.

```

[TestMethod]
[TestCategory("Divide")]
[ExpectedException(typeof(DivideByZeroException))]
public void DivideByZero()
{
    #region Arrange
    var calculator = new Calculator();
    #endregion

    #region Act
    calculator.Divide(10, 0);
    #endregion
}

```

- Písanie testu na konkrétnu metódu vykonávajúce podľa vzoru AAA. Každú metódu rozdeľte do sekcií v poradí *Arrange*, *Act*, *Assert*. Sekcie ohraničte použitím príkazu *#region <názov sekcie>* a *#endregion*.
- V sekcii *Arrange* inicializujte objekty a premenné, s ktorými sa bude v metóde pracovať. Ako prvé dve premenné v sekcii uvádzajte premenné *result* (na konci obsahuje výsledok testu) a *expected* (inicializovaná na očakávaný výstup testu). Vynechajte riadok a uveďte a inicializujte zvyšné objekty a premenné. V prípade, že výsledkom metódy má byť vyhodenie výnimky táto sekcia nebude obsahovať premenné *result* a *expected*.
- V sekcii *Act* volajte testované metódy so stanovenými parametrami z prvej časti.
- V sekcii *Assert* overte, či testovaná metóda vracia očakávaný výsledok. Vykonajte porovnanie očakávanej a reálne vrátenej hodnoty pomocou volania *Assert.AreEqual(expected, result)*; V prípade, že výsledkom metódy má byť vyhodenie výnimky táto sekcia ostane prázdna.
- Pri písaní kódu testov sa riadte pokynmi a odporúčaniami metodiky Písanie zdrojových kódov v C#¹.

¹dostupné v kapitole 4 v rámci dokumentu o riadení nachádzajúcom sa na stránke <http://labss2.fiit.stuba.sk/TeamProject/2014/team16is-si/> v sekcii Dokumenty.


```

using MyCalculator;

namespace MyCalculator.Tests
{
    [TestClass]
    0 references
    public class CalculatorTests
    {
        [TestMethod]
        [TestCategory("Add")]
        0 references
        public void AddTest()
        {
            #region Arrange
            int result = 0;
            int expected = 3;


            var calculator = new Calculator();
            #endregion

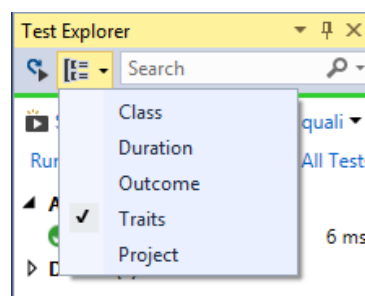
            #region Act
            result = calculator.Add(1, 2);
            #endregion

            #region Assert
            Assert.AreEqual(expected, result);
            #endregion
        }
    }
}

```

3.7.5 Spúšťanie testov

1. Všetky napísané testy môžete vidieť po skompilovaní v prieskumníkovi testov. V prípade, že prieskumník nemáte zapnutý, kliknite v menu vývojového prostredia Microsoft Visual Studio na položku *Test*, vyberte možnosť *Windows* a zvolte *Test Explorer*.
2. Keďže boli testy pri vytváraní zaraďované do kategórií, kliknite v prieskumníkovi testov v ľavom hornom rohu na šípku vedľa tlačidla  a zakliknite možnosť *Traits*. Existujúce testy sa zoskupia podľa priradenej kategórie.



Obrázok 40: Zoskupenie testov.

3. V prieskumníkovi testov kliknite v ľavom hornom rohu na možnosť **Run All**, alebo spustíte testy podľa potreby jednotlivo, pravým kliknutím na požadovaný test a výberom možnosti **Run Selected Tests**.
4. Výsledky testov môžete vidieť po ich vykonaní v prieskumníkovi testov. V prípade chyby v niektorom z testov sú podrobnosti o chybe zobrazené v prieskumníkovi testov v časti *Summary*. Ak ste pri testovaní objavili chybný test k časti kódu ktorej nie ste autorom, vytvorte o tejto chybe záznam. Návod ako postupovať nájdete v metodike Manažment chýb².

²dostupné v kapitole 4 v rámci dokumentu o riadení nachádzajúcom sa na stránke <http://labss2.fiit.stuba.sk/TeamProject/2014/team16is-si/> v sekcii Dokumenty.

4 Export evidencie úloh

4.1 Šprint 1

Tabuľka 4: Ukončenie 1. šprintu.

ID	Work Item Type	Title	Assigned To	State
1520	User Story	WEB stránka		Active
1522	User Story	Funkcia na vracanie API kľúča	Bc. Matej Kucek	Active
1523	Task	Code review	Bc. Peter Kis	Active
1524	Task	Vývoj	Bc. Marek Sulek	Active
1525	Task	Vytvoriť	Bc. Peter Kusnir	Active
1526	User Story	Funkcia na získanie API kľúču v ALEFe	Bc. Matej Kucek	Active
1527	Task	Vývoj	Bc. Viktoria Lovasova	Closed
1528	Task	Code review	Bc. Martin Svrcek	New
1529	User Story	Vytvoriť API kľúč		New
1530	Task	Vytvoriť	Bc. Simon Valicek	Closed
1531	Task	Uploadnuť WEB stránku na server	Bc. Simon Valicek	New
1532	Task	Nastaviť notifikácie	Bc. Matej Kucek	Closed
1533	Task	Nastaviť continuous integration	Bc. Matej Kucek	Active
1534	Task	Zistiť ako sa robia reporty z tfs	Bc. Matej Kucek	Active
1621	User Story	Dorobit Web stránku	Bc. Peter Kusnir	New
1622	Task	Dorobit features na vysej urovni	Bc. Peter Kusnir	New

4.2 Šprint 2

Tabuľka 5: Polovica 2. šprintu.

ID	Work Item Type	Title	Assigned To	State
1621	User Story	Dorobit Web stránku	Bc. Peter Kusnir	New
1622	Task	Dorobit features na vysej urovni	Bc. Peter Kusnir	Closed
1623	User Story	Počítanie agregácií	Bc. Peter Kis	Active
1624	User Story	Zadefinovanie volania v API	Bc. Simon Valicek	New
1625	User Story	Autentifikácia cez API kľúč	Bc. Marek Sulek	New
1626	User Story	Publikovanie eventov	Bc. Peter Kusnir	New
1627	User Story	Odozva ALEFU		Removed
1628	User Story	Pocuvanie eventov cez backbone JS a zvyraznenie elementov	Bc. Martin Svrcek	Active
1629	User Story	Volanie z ALEFu	Bc. Viktoria Lovasova	New
1630	User Story	Pridanie sledovania pohľadu do administrácie	Bc. Matej Kucek	New
1656	Task	Code review		New
1657	Task	Úprava DB štruktúry		New
1658	Task	Vytvorenie agregáčnych dopytov		New
1659	Task	Analýza existujúcej aplikačnej a databázovej infraštruktúry	Bc. Peter Kis	Active
1660	Task	Vytvorenie aplikačnej logiky		New
1667	Task	Code review		New
1668	Task	Vytvoriť ikonu na prepnutie do pohľadu administrácie	Bc. Matej Kucek	New
1676	Task	Analýza Backbone JS	Bc. Martin	Active

			Svrcek	
1677	Task	Pridanie controllera	Bc. Martin Svrcek	New
1678	Task	Implementácia tela controllera na počúvanie/odchytávanie eventov	Bc. Martin Svrcek	New
1679	Task	Analýza práce s javascriptom	Bc. Martin Svrcek	New
1680	Task	Zvýraznenie elementu, na ktorý sa pozeráme	Bc. Martin Svrcek	New
1681	Task	Code review		New
1682	Task	Napísanie testov	Bc. Martin Svrcek	New
1683	Task	Upravenie autentifikacie	Bc. Viktoria Lovasova	Active
1684	Task	Zdokumentovanie fungovania rozšírenia do Chrome-u	Bc. Peter Kusnir	Active
1685	Task	Implementacia casti pre publikovania eventov	Bc. Peter Kusnir	New
1686	Task	Code review		New
1687	Task	Refactoring existujúceho kódu	Bc. Peter Kis	Closed

Tabuľka 6: Ukončenie 2. šprintu.

ID	Work Item Type	Title	Assigned To	State
1621	User Story	Dorobit Web stránku	Bc. Peter Kusnir	New
1622	Task	Dorobit features na vysej urovni	Bc. Peter Kusnir	Closed
1623	User Story	Počítanie agregácií	Bc. Peter Kis	Active
1624	User Story	Zadefinovanie volania v API	Bc. Simon Valicek	New
1625	User Story	Autentifikácia cez API kľúč	Bc. Marek Sulek	New
1626	User Story	Publikovanie eventov	Bc. Peter Kusnir	New
1627	User Story	Odozva ALEFU		Removed

1628	User Story	Pocuvanie eventov cez backbone JS a zvyraznenie elementov	Bc. Martin Svrcek	Active
1629	User Story	Volanie z ALEFu	Bc. Viktoria Lovasova	New
1630	User Story	Pridanie sledovania pohľadu do administrácie	Bc. Matej Kucek	New
1656	Task	Code review		New
1657	Task	Úprava DB štruktúry	Bc. Peter Kis	Active
1658	Task	Vytvorenie agregáčnych dopytov	Bc. Peter Kis	Active
1659	Task	Analýza existujúcej aplikačnej a databázovej infraštruktúry	Bc. Peter Kis	Closed
1660	Task	Vytvorenie aplikačnej logiky	Bc. Peter Kis	Active
1667	Task	Code review		New
1668	Task	Vytvoríť ikonu na prepnutie do pohľadu administrácie	Bc. Matej Kucek	Closed
1676	Task	Analýza Backbone JS	Bc. Martin Svrcek	Closed
1677	Task	Pridanie controllera	Bc. Martin Svrcek	Removed
1678	Task	Implementácia časti na počúvanie/odchytávanie eventov	Bc. Martin Svrcek	Closed
1679	Task	Analýza práce s javascriptom	Bc. Martin Svrcek	Closed
1680	Task	Zvýraznenie elementu, na ktorý sa pozeráme	Bc. Martin Svrcek	Closed
1681	Task	Code review		New
1682	Task	Analýza testovania v javascripte	Bc. Martin Svrcek	Active
1683	Task	Upravenie autentifikácie	Bc. Viktoria Lovasova	Active
1684	Task	Zdokumentovanie fungovania rozšírenia do Chrome-u	Bc. Peter Kusnir	Active
1685	Task	Implementácia časti pre publikovania eventov	Bc. Peter Kusnir	New

1686	Task	Code review		New
1687	Task	Refactoring existujúceho kódu	Bc. Peter Kis	Closed
1688	Task	Commitnúť zmenu	Bc. Matej Kucek	Closed
1689	Task	Otestovanie	Bc. Matej Kucek	Closed
1690	Task	Overenie API kľuca v DB	Bc. Marek Sulek	Active

4.3 Šprint 3

Tabuľka 7: Polovica 3. šprintu.

ID	Work Item Type	Title	Assigned To	State	Tags
1623	User Story	Počítanie agregácií	Bc. Peter Kis	Active	
1656	Task	Code review	Bc. Matej Kucek	New	
1657	Task	Úprava DB štruktúry	Bc. Peter Kis	Closed	
1658	Task	Vytvorenie agregáčnych dopytov	Bc. Matej Kucek	Active	
1659	Task	Analýza existujúcej aplikačnej a databázovej infraštruktúry	Bc. Peter Kis	Closed	
1660	Task	Vytvorenie aplikačnej logiky	Bc. Peter Kis	Active	
1687	Task	Refactoring existujúceho kódu	Bc. Peter Kis	Closed	
1768	User Story	Zanalyzovať otázkoč v Alefe	Bc. Martin Svrcek	New	
1769	User Story	Manažment projektu	Bc. Marek Sulek	New	
1770	User Story	Manažment Gaze Trackingu v Alefe	Bc. Viktoria Lovasova	New	
1771	User Story	Analýza fungovania oblasti záujmu na servery	Bc. Peter Kusnir	New	
1774	Task	Zdokumentovanie rozšírenia pre Chrome	Bc. Peter Kusnir	New	

1775	Task	Zdokumentovanie serverovej časti	Bc. Peter Kusnir	Active	
1776	Task	Analýza otázkovača v ALEFe	Bc. Martin Svrcek	New	
1777	Task	Odoslanie eventu otázkovaču	Bc. Martin Svrcek	New	
1778	Task	Vygenerovanie otázky na základe prijatého eventu	Bc. Martin Svrcek	New	
1779	Task	Code review	Bc. Viktoria Lovasova	New	
1780	Task	Zdokumentovanie ALEF otázkovača	Bc. Martin Svrcek	New	
1781	Task	Analýza použitia Redis	Bc. Matej Kucek	Removed	
1784	Task	Vytvorenie formulara na manazment s projektami	Bc. Viktoria Lovasova	Closed	
1785	Task	Vytvorenie resource projekt	Bc. Viktoria Lovasova	Closed	
1786	Task	Posielanie requestov na manazment s projektmi ku gaze tracking aplikacii	Bc. Viktoria Lovasova	New	
1787	Task	Otestovanie volani	Bc. Viktoria Lovasova	New	
1788	Task	Napisanie dokumentacie	Bc. Viktoria Lovasova	New	
1789	Task	Code review	Bc. Martin Svrcek	New	
1790	Task	Napísanie dokumentácie		New	
1791	Task	Funkcia api Create project	Bc. Marek Sulek	Active	server
1792	Task	Funkcia api List projects	Bc. Marek Sulek	New	server
1793	Task	Funkcia api Get project by id project	Bc. Marek Sulek	New	
1794	Task	Funkcia api Update project	Bc. Marek Sulek	New	

1795	Task	Funkcia api Delete project	Bc. Marek Sulek	New	
1812	Task	Testovanie	Bc. Peter Kis	New	
1823	Task	Otestovat priepustnost	Bc. Matej Kucek	New	

Tabuľka 8: Ukončenie 3. sprintu.

ID	Work Item Type	Title	Assigned To	State	Tags
1623	User Story	Počítanie agregácií	Bc. Peter Kis	Active	
1656	Task	Code review	Bc. Matej Kucek	New	
1657	Task	Úprava DB štruktúry	Bc. Peter Kis	Closed	
1658	Task	Vytvorenie agregáčnych dopytov	Bc. Matej Kucek	Closed	
1659	Task	Analýza existujúcej aplikačnej a databázovej infraštruktúry	Bc. Peter Kis	Closed	
1660	Task	Vytvorenie aplikačnej logiky	Bc. Peter Kis	Active	
1687	Task	Refactoring existujúceho kódu	Bc. Peter Kis	Closed	
1768	User Story	Zanalyzovať otázkoč v Alefe	Bc. Martin Svrcek	New	
1769	User Story	Manažment projektu	Bc. Marek Sulek	New	
1770	User Story	Manažment Gaze Trackingu v Alefe	Bc. Viktoria Lovasova	New	
1771	User Story	Analýza fungovania oblasti záujmu na servery	Bc. Peter Kusnir	New	
1774	Task	Zdokumentovanie rozšírenia pre Chrome	Bc. Peter Kusnir	New	
1775	Task	Zdokumentovanie serverovej časti	Bc. Peter Kusnir	Active	
1776	Task	Analýza otázkoča v ALEFe	Bc. Martin	Closed	

			Svrcek		
1777	Task	Odoslanie eventu otázkovaču	Bc. Martin Svrcek	Removed	
1778	Task	Vygenerovanie otázky na základe prijatého eventu	Bc. Martin Svrcek	Removed	
1779	Task	Code review	Bc. Viktoria Lovasova	Removed	
1780	Task	Zdokumentovanie ALEF otázkovača	Bc. Martin Svrcek	Closed	
1781	Task	Analýza použitia Redis	Bc. Matej Kucek	Removed	
1784	Task	Vytvorenie formulara na manazment s projektami	Bc. Viktoria Lovasova	Closed	
1785	Task	Vytvorenie resource projekt	Bc. Viktoria Lovasova	Closed	
1786	Task	Posielanie requestov na manazment s projektmi ku gaze tracking aplikacii	Bc. Viktoria Lovasova	New	
1787	Task	Otestovanie volani	Bc. Viktoria Lovasova	New	
1788	Task	Napisanie dokumentacie	Bc. Viktoria Lovasova	New	
1789	Task	Code review	Bc. Martin Svrcek	New	
1790	Task	Napísanie dokumentácie	Bc. Peter Kis	Active	
1791	Task	Funkcia api Create project	Bc. Marek Sulek	Active	server
1792	Task	Funkcia api List projects	Bc. Marek Sulek	New	server
1793	Task	Funkcia api Get project by id project	Bc. Marek Sulek	New	
1794	Task	Funkcia api Update project	Bc. Marek Sulek	New	
1795	Task	Funkcia api Delete project	Bc. Marek Sulek	New	

1812	Task	Testovanie	Bc. Peter Kis	Active	
1823	Task	Otestovat priepustnost	Bc. Matej Kucek	New	
1826	Bug	Pokazenie buttona odoslat	Bc. Viktoria Lovasova	Closed	
1827	Bug	skuska	Bc. Marek Sulek	Active	
1828	Bug	skuska na najdenie chyby		Active	

4.4 Šprint 4

Tabuľka 9: Ukončenie 4 šprintu.

Project: GazeTracking Server: tfs.fiit.stuba.sk\StudentsProjects Query: Sprint 4 List type: Flat				
ID	Work Item	Title	Assigned To	State
1656	Task	Code review	Bc. Matej Kucek	New
1823	Task	Otestovat priepustnost	Bc. Matej Kucek	New
1845	User Story	Zobrazenie agregácií v časti Statistics	Bc. Matej Kucek	New
1846	User Story	APIS Sessions	Bc. Viktoria Lovasova	New
1847	User Story	API Users	Bc. Marek Sulek	New
1848	User Story	API AOI	Bc. Peter Kusnir	New
1849	User Story	API Agregácie	Bc. Simon Valicek	New
1851	Task	Analyzovanie kódu	Bc. Matej Kucek	Closed
1852	Task	Implementovanie prepojenia	Bc. Matej Kucek	Active
1853	Task	Implementovanie vizualizácie	Bc. Matej Kucek	New
1854	Task	Testovanie štatistik	Bc. Matej Kucek	New
1855	Task	Prehliadka kodu	Bc. Peter Kis	New
1859	Task	Create AOI	Bc. Peter Kusnir	Active
1860	Task	Update AOI	Bc. Peter Kusnir	Active
1861	Task	Read(Get) AOI	Bc. Peter Kusnir	Active
1862	Task	Delete AOI	Bc. Peter Kusnir	Active
1863	Task	Code Review		New
1864	Task	Dokumentacia	Bc. Peter Kusnir	New
1881	Task	Dokumentácia	Bc. Peter Kis	Closed
1882	Task	Analyza	Bc. Peter Kis	Closed
1883	Task	Implementácia	Bc. Peter Kis	Closed
1884	Task	Testovanie	Bc. Peter Kis	New
1885	User Story	Analyza a implementacia otazkovača pre API	Bc. Martin Svrcek	Active
1886	Task	Analyzovanie otázkovača v ALEFe	Bc. Martin Svrcek	Closed
1887	Task	Zdokumentovanie otázkovača v ALEFe	Bc. Martin Svrcek	Closed
1888	Task	Implementácia generovania otázky na základe prijatého eventu	Bc. Martin Svrcek	Active
1889	Task	Code review		New
1891	Task	Implementacia	Bc. Viktoria Lovasova	Closed
1892	Task	Testy		Closed
1893	Task	Dokumentacia		Closed
1895	Task	Implementacia	Bc. Marek Sulek	Closed
1896	Task	testy	Bc. Marek Sulek	Closed
1897	Task	documentacia	Bc. Marek Sulek	Closed
1898	Task	code reiew	Bc. Marek Sulek	Active
1907	Task	specifikacia	Bc. Simon Valicek	New
1908	Task	implementacia	Bc. Simon Valicek	New
1909	Task	testy	Bc. Simon Valicek	New
1910	Task	code review		New
1927	Task	Rozbehavanie lokalneho testovania web stranky	Bc. Matej Kucek	Closed

4.5 Šprint 5

Tabuľka 10: Ukončenie 5 šprintu.

Project: GazeTracking Server: tfs.fiiit.stuba.sk\StudentsProjects Query: sprint 5 List type: Flat					
ID	Work Item	Title	Assigned To	State	Tags
2110	User Story	Filter fixácií	Bc. Martin Svrcek	Active	
2111	User Story	Zobrazenie počtu fixácií v tabuľke	Bc. Matej Kucek	New	
2112	User Story	Doriešiť DB / Server	Bc. Simon Valicek	New	
2113	User Story	Grafické zobrazenie + filter (voliteľné)	Bc. Peter Kis	Resolved	
2117	User Story	Statická stránka s textom	Bc. Viktoria Lovasova	Closed	
2118	User Story	Analýza vzorov čítania	Bc. Martin Svrcek	Active	
2123	User Story	Otestovať zachytávanie eventov	Bc. Viktoria Lovasova	Active	
2124	User Story	Volanie API z JS knižnice	Bc. Peter Kusnir	Active	
2125	User Story	Presunúť komunikáciu do JS lib	Bc. Marek Sulek	New	
2130	Task	Implementácia stiahnutia dát	Bc. Matej Kucek	New	
2131	Task	Implementovanie naplnenia tabuľky	Bc. Matej Kucek	New	
2133	Task	Analýza dostupných implementácií filtrov	Bc. Martin Svrcek	Closed	analysis; fixation filter
2134	Task	Analýza článkov	Bc. Martin Svrcek	Closed	analysis; fixation filter
2135	Task	Spracovanie zvolenej implementácie	Bc. Martin Svrcek	Active	fixation filter; implementation
2136	Task	Overenie správnosti implementácie	Bc. Martin Svrcek	Active	fixation filter; testing
2137	Task	Analýza dostupných riešení vzorov čítania	Bc. Martin Svrcek	Closed	analysis; read patterns
2138	Task	Analýza predošlého kódu	Bc. Matej Kucek	Closed	
2139	Task	Analýza chrome extension	Bc. Marek Sulek	Closed	
2140	Task	Presun logiky extensionu do js suboru	Bc. Marek Sulek	Closed	
2144	Task	Analýza vhodnosti použitia d3.js a .Net chart control	Bc. Peter Kis	Closed	
2145	Task	Vytvoriť testovacie data	Bc. Peter Kis	Closed	
2146	Task	Vytvorenie vykreslovacích funkcií	Bc. Peter Kis	Closed	
2147	Task	Analýza použiteľnosti existujúceho kódu	Bc. Peter Kis	Closed	
2149	Task	Implementácia agregacných funkcií do požadovaného tvaru	Bc. Matej Kucek	Closed	
2193	Task	Analýza vzorov čítania	Bc. Matej Kucek	Closed	

4.6 Šprint 6

Tabuľka 11: Ukončenie 6 šprintu.

Project: GazeTracking Server: tfs.fiit.stuba.sk\StudentsProjects Query: sprint 6 List type: Flat					
ID	Work Item	Title	Assigned To	Status	Tags
2200	User Story	Filter pri Vizualizaciach	Bc. Peter Kis	Active	
2201	User Story	Dwell time	Bc. Peter Kis	New	
2202	User Story	Fixation count	Bc. Peter Kis	Closed	
2203	User Story	Saccades	Bc. Martin Svrcek	Active	
2205	User Story	Vzory citania	Bc. Matej Kucek	New	
2208	User Story	Prerobenie Json na JsonP	Bc. Simon Valicek	New	
2209	User Story	Zistenie ucastnika ci cital alebo necital	Bc. Marek Sulek	New	
2217	Task	Filter pri Vizualizaciach	Bc. Peter Kis	Closed	
2218	Task	Dwell time	Bc. Peter Kis	New	
2219	Task	Fixation count	Bc. Peter Kis	Closed	
2227	Task	Analýza vhodnych zdrojov	Bc. Matej Kucek	New	
2228	Task	Citanie clankov	Bc. Matej Kucek	New	
2256	Task	Analýza dat	Bc. Marek Sulek	Closed	
2257	Task	Analýza možnej reimplementácie	Bc. Martin Svrcek	Closed	analysis; reimplementation; saccades
2258	Task	Analýza volania R kódu v .NET prostredí	Bc. Martin Svrcek	Closed	.NET; analysis; R; saccades
2259	Task	Realizácia vybranej metódy pre použitie saccades v .NET	Bc. Martin Svrcek	Closed	.NET; implementation; saccades
2267	Task	Citanie Text 2.0		New	
2268	Task	implementacia	Bc. Marek Sulek	Closed	

4.7 Šprint 7

Tabuľka 12: Ukončenie 7 šprintu.

Project: GazeTracking Server: tfs.fiit.stuba.sk\StudentsProjects Query: sprint 7 List type: Flat					
ID	Work Item	Title	Assigned To	State	Tags
2204	User Story	Naplanovanie + mail JT Data Creating	Bc. Viktoria Lovasova	Closed	
2206	User Story	Upozornenie na serveri	Bc. Peter Kusnir	New	
2207	User Story	JS API Trigger	Bc. Simon Valicek	New	
2210	Task	Uprava klienta + zmena URL	Bc. Simon Valicek	New	
2300	User Story	Implementacia algoritmu analyzy citania	Bc. Matej Kucek	New	
2301	User Story	Opravit session managment a spojzdnit klienta	Bc. Simon Valicek	New	
2302	User Story	Dopracovanie funkcionality JS kniznice	Bc. Marek Sulek	New	
2304	User Story	Grafické štatistiky	Bc. Peter Kis	Active	
2305	Task	Vytvorenie grafov	Bc. Peter Kis	Closed	
2307	Task	Implementacia algoritmu rozoznania citania		Closed	
2308	Task	Vylepsovane a rozsirenje o dalsie rozoznania		New	
2330	Task	Analýza algoritmu I-V filtra	Bc. Martin Svrcek	Closed	analysis; fixations; I-V filter
2331	Task	Reimplementácia I-V filtra do syntaxe C#	Bc. Martin Svrcek	Closed	C#; fixations; implementation; I-V filter
2332	Task	Otestovanie reimplementácie v C#	Bc. Martin Svrcek	New	.NET; C#; fixations; I-V filter; testing
2403	Task	vyriesenie cross domain volani	Bc. Marek Sulek	Active	
2404	Task	implementovanie funkcionality	Bc. Marek Sulek	Active	
2405	Task	Vytvorenie upozornenia	Bc. Peter Kusnir	Closed	
2406	Task	Uprava upozornenia	Bc. Peter Kusnir	Closed	
2407	Task	Vytvorenie DB tabulky a infrastruktury	Bc. Peter Kusnir	Active	
2408	Task	Zobrazovanie existujucich upozorneni	Bc. Peter Kusnir	Closed	
2409	Task	Mazanie upozorneni		Closed	
2410	Task	Stylovanie	Bc. Peter Kusnir	Closed	
2411	Task	Parsovanie dat		Closed	
2412	Task	Validacie formularov a dokoncovacky	Bc. Peter Kusnir	Active	

4.8 Šprint 8

Tabuľka 13: Ukončenie 8 šprintu.

Project: GazeTracking Server: tfs.fiit.stuba.sk\StudentsProjects Query: sprint 8 List type: Flat					
ID	Work Item	Title	Assigned To	State	Tags
2299	User Story	Pocitanie fixácií v C#	Bc. Martin Svrcek	Active	
2423	User Story	Pridanie analyzy citania vzhľadom na AOI	Bc. Matej Kucek	New	
2424	User Story	IIT SRC Poster	Bc. Peter Kusnir	Closed	
2460	Task	Vytvorenie plagatu	Bc. Peter Kusnir	Closed	
2461	Task	Tlac plagatu	Bc. Peter Kusnir	Closed	
2470	Task	Implementacia opakovaneho citania		New	
2473	Task	Tématický obsah posteru	Bc. Martin Svrcek	Closed	IIT.SRC; Poster
2474	Task	Reimplementácia Matlab funkcií	Bc. Martin Svrcek	Active	.NET; C#; fixation filter; fixations; implementation; I-V
2480	Task	Príprava prostredia na testovanie	Bc. Peter Kis	New	
2481	User Story	Príprava prostredia na testovanie		New	
2482	Task	Oprava chýb na prostredí	Bc. Peter Kis	Active	
2483	Task	Otestovanie tvorby agregácií	Bc. Peter Kis	New	

4.9 Šprint 9

Tabuľka 14: Ukončenie 9 šprintu.

Project: GazeTracking Server: tfs.fiit.stuba.sk\StudentsProjects Query: sprint 9 List type: Flat					
ID	Work Item	Title	Assigned To	State	Tags
2487	User Story	integracia analyzy citania	Bc. Matej Kucek	New	
2488	User Story	opravenie agregácii a testovanie na eyetrackeri	Bc. Peter Kis	New	
2489	User Story	Dokoncenie implementacie urcovania fixácií	Bc. Martin Svrcek	Active	
2490	User Story	Prepojenie implementácií upozorňovania	Bc. Marek Sulek	New	
2491	Task	Implementácia upozorňovania (alert)	Bc. Marek Sulek	New	
2515	Task	Doimplementovanie IVT filtra	Bc. Martin Svrcek	Closed	C#; implementation; I-V filter; Matlab
2516	Task	Otestovanie implementácie IVT filtra	Bc. Martin Svrcek	Active	C#; I-V filter; Matlab; testing

V prílohe B sú umiestnené zápisy z jednotlivých tímových stretnutí.

5 Webové sídlo projektu

Webové sídlo je přiložené v .ZIP súbore.

Príloha A: Testovanie v jazyku javascript - JASMINE

1 Úvod

Čo je JASMIN ?

JASMIN je správaním riadený (behaviour-driven) rámec pre testovanie kódu v javascripte. Je vhodný pre vývoj riadený testami (test-driven development) a má syntax podobnú nástroju RSpec pre testovanie jazyka Ruby. Celý rámec JASMINE je možné stiahnuť na adrese: <https://github.com/pivotal/jasmine/releases>.

2 Základná štruktúra

Predstavme si funkciu v javascripte, ktorá vracia jednoduchý reťazec “Team Focus.” :

```
function focusName() {  
    return "Team Focus."  
}
```

Ak by sme pre takúto funkciu chceli otestovať jej správnosť test v JASMINE by mohol vyzeráť nasledovne:

```
describe("Focus Team Name", function() {  
    it("return Team Focus Name", function() {  
        expect(focusName()).toEqual("Team Focus.");  
    });  
});
```

Časť *describe* predstavuje nejaký komponent aplikácie, v našom prípade je to funkcia *focusName()*. “Focus Team Name” je len reťazec a nepredstavuje žiadny kód.

Časť *it* predstavuje špecifikáciu, je to vlastne funkcia, ktorá hovorí čo má daný komponent robiť [1]. V rámci časti *describe* môžeme pridať viacero špecifikácií. Časť *expect* predstavuje nejaké očakávanie, ktoré pracuje s hodnotou v zátvorke (v našom príklade návratová hodnota funkcie) a túto hodnotu následne nejakým spôsobom konfrontuje s funkciou nazývanou *matcher* [2] (v našom prípade *toEqual*).

3 Matcher

Ako už bolo spomenuté funkcia *matcher* vykonáva boolean porovnanie medzi aktuálnou hodnotou a očakávanou hodnotou [2]. To znamená, že pred každú takúto funkciu môžeme pridať *.not* ako negáciu (napríklad *.not.toEqual()*).

Dôležitou črtou týchto funkcií je, že existuje veľa predpísaných funkcií *matcher* ale taktiež si môžeme definovať aj vlastné.

Základnými príkladmi predpísaných funkcií *matcher* sú:

- `.toEqual()` - porovnanie či sú dva objekty rovnaké.
- `.toContain()` - porovnanie či jeden objekt obsahuje druhý objekt (reťazce, polia, ...).
- `.toBe()` - porovnanie či jeden objekt je ten istý ako druhý objekt.
- `.toBeTruthy()` - porovnanie či je niečo *true*.
- `.toBeFalsy()` - porovnanie či je niečo *false*.
- `.toBeDefined()` - porovnanie či je objekt definovaný.
- `.toBeUndefined()` - porovnanie či nie je objekt definovaný.
- `.toBeNull()` - porovnanie či je objekt NULL.
- `.toBeNaN()` - porovnanie či je objekt NaN (Not a Number).
- `.toBeGreaterThan()` - porovnanie či je objekt väčší ako druhý objekt.
- `.toBeLessThan()` - porovnanie či je objekt menší ako druhý objekt.
- `.toBeCloseTo()` - porovnanie či je číslo blízke inému číslu na stanovený počet desatinných miest.

Vlastné funkcie *matcher* [1]:

```
beforeEach(function() {
  this.addMatchers({
    toBeLarge: function() {
      this.message = function() {
        return "Expected " + this.actual + " to be
        large";
      };
      return this.actual > 100;
    }
  });
});
```

4 Ďalšie funkcie a črty

Rámec JASMINE má aj veľa ďalších užitočných črt, ktoré sa dajú efektívne využiť pri testovaní kódu vzhľadom na špecifikáciu.

4.1 Before and After

Klauzula *beforeEach* sa využíva ak chceme vykonať nejaký kód pred každou špecifikáciou (*it*). Na druhej strane klauzula *afterEach* sa využíva ak chceme nejaký kód vykonať po vykonaní každej špecifikácie.

Príklady [1]:

```
beforeEach(function() {
  employee = new Employee;
});
```

```
afterEach(function() {
    calculator.reset();
});
```

4.2 Zahniezdenie

Jednotlivé bloky *describe*(... môžeme do seba jednoducho zahniezdiť a vytvárať tak rôzne podskupiny. Príklad [2]:

```
describe("A spec", function() {
    var foo;
    describe("nested inside a second describe", function() {
        . . .
    });
});
```

4.3 Vynechávanie špecifikácií

Ak z nejakého dôvodu nechceme niektoré špecifikácie testovať, tak ich väčšinou zakomentujeme v kóde. JASMINE však má na tieto prípady vlatné riešenie. Stačí pridať pred každú sekciu *it* alebo *describe* písmeno “x” (napr. *xit* alebo *xdescribe*). Pri testovaní sa táto sekcia automaticky preskočí.

Príkaz *return*; zase preruší testovanie a ďalej už test nebude pokračovať [1]:

```
describe("I'm only going to run SOME of these", function() {
    it("will run this spec", function() {});
    it("will run this spec", function() {});
    return; // This will stop the function
    it("will not run this spec", function() {});
    it("will not run this spec", function() {});
});
```

5 Špióni

Špióni (spies) sú špeciálne črty rámca JASMINE, ktoré dokážu sledovať volania funkcií a aj ich atribúty. Existujú iba v rámci *describe* a *it* blokov a zrušia sa pokaždej ukočenej špecifikácii.

Obsahujú špeciálne funkcie *matcher* ako napríklad:

- *.toHaveBeenCalled*
- *.toHaveBeenCalledWith*
- *and.callThrough*
- *and.returnValue*
- *and.callFake*
- *and.throwError*
- *and.stub*

Samotného špióna môžeme vytvoriť nasledovne [1]:

```
var dictionary = new Dictionary;  
spyOn(dictionary, "hello");  
expect(dictionary.hello).toHaveBeenCalled();
```

V tomto prípade špión nahradí funkciu *hello* triedy *Dictionary*. A následne bude volanie tejto funkcie sledované.

V rámci tohto dokumentu som spísal základné informácie o testovaní javascript jazyka prostredníctvom rámcu JASMINE. Využil som pri tom dva zdroje, kde môžete nájsť viacero ďalších informácií.

6 Zdroje

- [1] <http://jasmine.github.io/2.0/introduction.html>
- [2] <http://it-ebooks.info/book/2085/>

Príloha B: Zápisy zo stretnutí

1 Zimný semester

1.1 Prvé stretnutie

- Stretnutie otvára Robo a oznamuje nám, že budúci týždeň bude musieť byť stretnutie v iný deň, pretože bude odcestovaný. Skúsi, či si ešte pamätá naše mená.
- Robo chce vidieť náš plagát, Martin Svrček mu ho ukáže na notebooku a Robo je spokojný, hovorí, že môže byť a aj názov tímu Focus sa mu páči.
- Začínajú sa riešiť organizačné veci, Robo navrhuje nech máme nejaké centrálné úložisko, kde si budeme ukladať dokumenty. Šimon odpovedá, že už sme si spravili Google drive. Robo povie, že môže byť a nech ho tam pridáme.
- Robo sa pýta ako okrem emailu medzi sebou komunikujeme, pretože to nestačí. Vie, že máme aj Facebook-skupinu ale radšej by bol keby to bolo niečo ako instant messaging, či niečo také nepoznáme z práce? Šimon navrhuje hipchat.
- Robo nám začína vysvetľovať jednotlivé funkcie rolí:
- **Náš tím by mal mať obsadene nasledujúce role:** Manažér komunikácie, Manažér dokumentácie, Vedúci tímu, Zástupca tímu, Manažér kvality: 2 ľudia môžu byť, jeden bude dozerat' nato ako sa budú robiť testy – pomocou tfs-ka, druhý bude mať na starosti code-review, to znamená, že každý kód budú vidieť dvaja ľudia a až potom pôjde na schválenie k Robovi. Tento proces by sa mal zachytiť v nejakej metodike manažérom kvality.
Manažér pre code versioning – napríklad v Gite, napísať aj postup ako to bude fungovať. Tieto metodiky môžeme využiť potom aj v manažmente.
Manažér plánovania: v podstate sa plánovanie robí spoločne, najprv sa stanovia featury, tie sa rozbijú na user stories a tie na tasky. Plánovanie sa robí pomocou metódy pokru, kedy spoločne prideme nato koľko story pointov obsahuje task.
- Manažér rizík a rozsahu – bude robiť aj manažér plánovania.
- Martin Svrček sa opýtal, kto sú naši zákazníci? Robo odpovedal, že máme možnosti:
 - možnosť, že to budú členovia fakulty - napríklad profesorka Bieliková, Michal Barla. Preto sa Robo pýta, či máme záujem o stretnutie s nimi, kde by sme sa dohodli o našich a ich predstavách, aby sme nezačali robiť niečo, čo oni nechceli.
 - plus sú to ľudia, ktorí používajú ux-lab, nejakí diplomanti a mohli by používať náš produkt
 - Robova funkcia je product owner.
- **Rozdelenie rolí:**
 - Vedúci tímu – Peter Kiš
 - Zástupca – Matej Kucek
 - Manažér komunikácie – Peter Kiš
 - Manažér kvality a integrácie – kvalita testovania – Peter Kušnír
 - integrácia – Matej Kucek
 - Git- Šimon Valíček
 - code review – Marek Šulek

Manažér plánovania a rozsahu- Martin Svrček

Manažér dokumentácie - Viktória Lovasová

- Dohodnutie sa na stretnutí so zákazníkmi, či máme záujem? Súhlasíme, Peter Kiš napíše prvý mail.
- Robo sa pýta, či chceme ísť na TP cup? Dve strany treba napísať a poslať do 28.10.2014. Prezentácia bude na IIT SRC – tam nejaké 4 strany po anglicky napísať. Hodnotenie bude na základe prezentácie a aj na základe výstupnej správy. Porota je z praxe a môžeme dostať spätnú väzbu. Záverom je, že sa ešte rozhodneme.
- Prichádza odborný praktikant Mišo a začína nám vysvetľovať projekt, na ktorý my nadväzujeme. Má tri komponenty: informačný systém, chrom-addon, firefox, dve zariadenia na sledovanie pohľadu. Server – obsahuje dve databázy: nosql ravenDB v c# písaná na framework .NET, podporuje JSONy tie sa posielajú, druhá je MSSQL: obsahuje dáta o používateľoch. Eyetracker posiela súradnice očí na server a takisto identifikuje element, ďalej aj focused application, kde sa nachádza kurzor, pracuje sa aj na tom kde sú kliky, takisto aj timestamps. Komunikácia je zabezpečená cez REST operácie. Zvonku je nedostupný gaze data service. Najvyššia úroveň:
 - Prepojenie cez api s aplikáciou ALEF:
Alef bude komunikovať s našou aplikáciou, bude môcť robiť nejaké akcie: reagovať na základe pohľadu a myši a odporúčať. Toto je najdôležitejšie, lebo to je použiteľné, nahradíme ALEF dáme tam niečo iné a bude to fungovať.
 - Štatistiky a vizualizácia
 - Podpora priebehu samotného testovania
Zadať pokyny, má nižšiu prioritu ako ostatné dve.

Prepojenie cez api s aplikáciou ALEF

Prebieha diskusia o features. Autentifikácia – posielanie v requeste apikey. ALEF si musí uložiť apikey, aby ho mohol posielat'. Potrebne získať aké sú zadané objects of interests. Get pre aoi koľkokrát sa tam používateľ pozrel, ako dlho sa tam pozerá, nie len kde sa pozerá, momentálne, xpath tiež môže vracat' kam sa pozerá. Šimon to celé zrušil lebo to je zlé pre biznis model. Preto sa bude posielat' len Get fix count, fix time -> rel, absolut. Uložiť v alefe apikey treba spraviť, na jeho strane nič nie je. V requeste posielam apikey, project_id, session_id, user_id.

Dohodli sme sa na nasledovných Features:

Features: Ukladanie Apikey v ALEFe

Vytvorenie projektu

Vytvorenie sedenia

Pridanie používateľov k projektu

Získavanie dát - aoi agregácie - pohľad, myš

Upozornenie/Prispôbenie obsahu(odporúčanie)

Štatistika/Vizualizácia

Robo navrhuje aj aby sme to dotiahli až na odporúčania v ALEFe.

Štatistiky a vizualizácia

Features:

Poskytnutie prehľadu – koľko tam je toho nazbieraného, project, session, user,
- počet, čas

Počítanie agregácií – nejaké predrátavanie by tam malo byť

(Vysvetlenie): ak sa niekam pozeral 1 s tak mi vytvor anotáciu(značku) viem si nastaviť aj že 5 sekúnd napríklad dá sa to nastaviť cez rozhranie. Mám objekty A1,A2,A3,... kam sa pozeral a koľko. Takže ak už robím agregácie tak z týchto anotácií. Takže aj pri prehľade sa dá z anotácií robiť aj agregácie. Priebežne aktualizovať anotácie, zvyšovať počet pozretí aby som mala predpripravené keď príde request. Teda mať tam nejaké countre .

Počet fixácií

Zmena v čase

Trvanie -> relatívne

->abolútne

Zachytávanie obrazovky-

(Vysvetlenie): robiť cez screenshot? Alebo video? Alebo stiahnuť stránku obraz(Mirror)

Heatmapy

Graf- poradie kam používateľ šiel

Zoom

Analýza pozerania na video

Šimon navrhuje personalizovanú reklamu – Robo hovorí, že to je super point na prezentáciu do TP cupu, nie len na výučbový systém ale aj na reklamu a bol by tam aj biznis model lebo že aj minulý rok sa sťažovali, že neboli stavané tak, že majú víziu posunúť to ďalej.

Minimálne prezentovať aj keď nie realizovať.

Ďalej nasledovalo plánovanie úloh do najbližšieho šprintu. Nabudúce si povieme v akom stave sú úlohy, na konci je retrospektíva, čo by sme mali zmeniť, čo bolo dobre. Potom je plánovanie nového šprintu.

Hotovo je keď:

Ten, kto implementuje hneď spraví aj testy, najprv by mal byť test a až potom implementácia, keď všetky testy prešli, kód spĺňa špecifikáciu, potom to preje k druhej osobe, ktorá bude zodpovedná za code review, dá mu feedback, ak je to ok tak to ide na kontrolu Robovi. Ako sa presne bude robiť code review si povieme nabudúce, podobne ako s gitom to je tam strom kde viem pridávať komentáre potom do kódu. Peter Kiš sa pýta, či bude code review robiť stále ten istý alebo môžu byť aj viacerí? Robo hovorí, že to môže byť jeden, ktorý bude potom implementovať menej a je dobrý v .NETE alebo budú dvojčičky a jeden implementovať a druhý code review robiť alebo pri každej úlohe sa určí kto robí review. Robo hovorí nech je úloh v šprinte menej ale aby sa to stihlo spraviť. Už po prvom týždni to má ísť ku code reviewerovi. K Robovi to môže ísť najneskôr na dokončení šprintu.

Úlohy:

- ALEF- vytvoriť funkciu v rámci gaze api, ktorý si alef vypýta apikey a tá funkcia v api vráti len obyčajné ok, nejaká jednoduchá komunikácia so slovom ok, a v alefe sa zobrazí modálne okno ok. User story: ALEF chce komunikovať cez api.
- Rozdelenie úloh - tasky:

- ALEF: Forknúť si git repozitár nad našou vlastnou verziou alefu, Rozbehať alef, Volanie cez api, Zobrazenie výsledkov, Testy.
 - api: Vytvoriť api kľúč pre ALEF - cez rozhranie spraviť. Vytvoriť volanie Support: rozbehať a nastaviť con. integration cez tfs
 - stránka tímu- jednoduchá, statická, vždy tam treba dať zápisnicu a dokumenty a plán
 - Zápisnica – všetky veci majú byť v tfs, teda aj zápisnica a exportovať sa vždy bude zápisnica z tfs
 - Budúce stretnutie bude viesť Viktória Lovasová, pretože písala túto dokumentáciu, povieme si čo sme mali spraviť.
 - Štúdium – ako sa pracuje s gitom, .net, prostredie (VS), Ruby on Rails, pozrieť dokumentáciu z min roku ako je zdokumentované api a databáza. Tieto úlohy spraví každý. Každý, za čo je zodpovedný začať nízko úrovňovo písať metodiku. Testovanie, integrácia a git je dôležité.
- Šimon sa pýta kedy bude budúce stretnutie. Bude vo štvrtok. Presný čas nám ešte Robo dá vedieť. Dohadovanie, kto bude čo robiť: stránka – do tfska nahodiť kto je zodpovedný, za task je zodpovedný vždy jeden, stránku robí Martin Svrček a Peter Kušník. Apikey spraviť pre ALEF- Šimon,
 - Api vytvoriť funkciu – Marek Šulek, codeReview: Peter Kiš.
 - Nastavovanie v tfs – Matej Kucek
 - ALEF – Viki a Robo , code review Martin Svrček
 - Ukončenie stretnutia.

1.2 Druhé stretnutie

- Viki spravila funkciu na preposielanie z ALEFu do GazeTrackingu, ale nefunguje zatiaľ funkcia na strane Gaze-u.
- Peter Kušník spravil stránku. Robo ->Big like. Treba prerobiť plán aby tam boli všetky hlavné úlohy a aj plán na semester s termínmi odovzdávania.
- Martin - - Plánovanie sa robí na konci šprintu ako retrospektíva plus ohodnocovanie obtiažnosti úloh.
- Matej - - Tasky v TFS by sa mali podrobnejšie trošku opísať. Nastaviť si notifikácie o úlohách. Ukladať si aj čas, ktorý sme venovali robote. Napísať aj keď sme riešili bugy a nestihli sme spraviť nejaký task.
- Marek - - Komunikácia – ALEF posielala requesty, aby sa identifikovala musí použiť API key. Marekova aplikácia by zachytila volanie a vráti hocijaký text. Funkcia už existuje – my potrebujeme aby sa prepojila s nami (vložíme tam URL alefu). Marekova API musí skontrolovať či ten prichádzajúci API kľúč je dobrý. Pri testoch musíme riešiť keď sa neprijme správny API kľúč. + API key v alefe hodiť do konfiguráku nejakého.
- Peter Kiš – vyriešil komunikačný kanál, dohodol stretnutie so zákazníkmi.

- Šimon pomáhal Marekovi generovať API key --- ďalšie úlohy: nahodiť stránku na web + vyriešiť API.

Ďalšie poznámky

Pozrieť si ako sa testuje v Railsoch aj v C# a RSpec.

Po každom šprinte by mala byť každá funkcia otestovaná a nasadená.

Všetko čo robíme by sa malo nasadzovať na našu vetvu a tam to bude bežať a naša práca sa mernie s prácou praktikantov až na konci.

Metodika na integráciu – pozrieť TFS integration.

Treba mať do konca šprintu spravené aj testy a spraviť aj review kódu. Aby sa stihol code review tak musí byť funkcia hotová trošku skorej.

V TFS nový repozitár na ALEF treba vytvoriť.

Code review:

- pozrieť aký kód bol pridaný vo Visual(pravé tlačidlo COMPARE)
- to čo je spravené či to robí to čo má robiť korektnosť a správnosť
- či to je spravené rozumne (konštanty, veci v konfiguráku, premenné) , či by som bol ja spokojný s kódom, zatiaľ to robiť cez komentáre ku tomu čo je zle alebo čo treba zmeniť
- podpora pre codereview v TFS možno vo Visualku , ale zatiaľ iba cez komentáre.
- Ten kto píše kód píše aj testy ale codereviewer následne kontroluje aj testy
- Robiť aj testy na výskyt bugu. – Toto môže robiť code reviewer a potom povie tomu kto to naprogramoval, že na tomto to môže padať.
- Každý by si mal pozrieť ako sa testuje.

Budúce stretnutie – okolo 3 hodiny

- Rozdeliť si nové zadanie úloh pre tím do budúceho cvičenia podľa upravenej prezentácie z prednášky TP.
- Rozhodnúť sa ako to bude s TPcupom.
- Vyhodnotenie šprintu – či to čo sme spravili funguje a či je to správne.
- Povieme čo bolo dobre a čo treba zmeniť.
- Continual Integration – keď ja niečo pushnem tak sa automaticky rozozná zmena a automaticky spustí všetky testy a ak je všetko zelené tak sa to nasadí ale ak nie je všetko zelené tak sa to musí opraviť --- lebo sa môže stať, že našou prácou sme mohli rozbiť alebo ovplyvniť niečo iné a Continual Integration tieto veci odhalí.
- Ak sa odhalí bug v robote predchádzajúceho tímu (lebo ten nemá spravené testy) tak treba spraviť test ktorý bude simulovať tieto bugy.
- Treba robiť refactoring kódu – aj odreportovať, že sme to spravili.
- Vygenerovanie reportu úloh z TFS.
- Product Backlog – treba si ho dopĺňať popri práci – ale nezasahovať do šprint backlogu počas šprintu. Môžeme pridávať úlohy kedykoľvek do product backlogu.

1.3 Tretie stretnutie

8:30 dostavili sa všetci. Môžeme začať.

V TFSku môžeme mať viacero repozitárov kvôli ALEFu, dá sa meniť stav tasku commitmi (aj ho zavrieť). O detailoch ešte príde mail.

Zhodnotenie týždňa

Viki: zavolała funkciu z API, ale nemá ešte hotové testy. Robo to kontroluje (nejak dlhšie to bežalo), ale nakoniec je spokojný.

Maťo Svrček: robil code review Viki, kontroloval zdrojáky aj konfiguračák, Robo má nejaké výhrady k názvu controlleru (do budúcnosti funkcionálnu umiestňovať do libky nie do controllera).

Marek: vytvoril nový projekt (solution) na testovanie API, lebo starý kód nebol použiteľný v tomto smere. S pomocou Šimona to nejak spravili, aby to šlo zavolať. Code review -> Peťo Kiš (nebolo veľmi čo opravovať), testy sa nestihli.

Matej: venoval sa nastaveniu continuous integration (Šimon že to donastaví), spravil nastavenie notifikácií, čo sa týka reportov tak k tomu sa mu nepodarilo nič nájsť. Robo k tomu niečo našiel, tak sa na to treba pozrieť. Report -> kto čo robí, v akom stave sú úlohy a čo sa bude riešiť.

Peťo Kušnir: dokončenie stránky (plán), task: dorobiť features na vyššej úrovni (termíny odovzdávania, vysokoúrovňový pohľad na veci čo ideme robiť; momentálny stav nie je úplne vyhovujúci, plán je náš backlog, čiže len termíny a fičúrky)

Potrebuje dohodnúť framework na testovanie pre C# (Šimon navrhuje Nspec, Peťo Kušnir Nunit), Rails je dohodnutý na Rspec.

Došiel Mišo, Šimon sa pýta ohľadom deployovania na server, oni to robia ručne; otázky na API: Šimon rozhodol, že je to nepoužiteľné tak ideme robiť API odznova. Čo sa týka stránky Šimon vybaví novú doménu v priebehu tohto týždňa, aby sa dali stránky odlišovať (stará a nová); chce tam nahodiť ftp.

Záver zo včera: hlavné veci pre tímák sú spraviť základné vizualizácie, začať zbierať viac dát (pohľad, myš, video); chystá sa projekt z UX labu, majú termín, aby bol sfunkčnený lab, možno v budúcnosti by sa dalo zohnať aj lepšie železo; ďalej sa bude riešiť integrácia s ALEFom, tiež boli rôzne nápady, posielalo by to aj nám aj klientovi (APIčku by bolo treba dorobiť server). Posledná vec je poskytovanie údajov učiteľom o tom, ktoré časti sú čítané apod. Mišo odchádza. Ak sa nájde niečo ďalšie, tak máme poslať mail.

Robo navrhol uzavrieť šprint. Každý týždeň by mal byť report a na konci šprintu by mala byť retrospektíva. Celkom zaujímavý spôsob ako to spraviť je, že povieme, čo z toho čo sme spravili už nerobiť, v čom pokračovať a čo začať robiť nové.

Začína Maťo Svrček. Navrhuje, aby sme ako tím lepšie pochopili, ako to celé funguje a navrhuje zaintegrovať tam všetkých členov. Bolo dobré, že sa nejak začalo, a že sa aj niečo spravilo. Navrhuje analýzu kódu, aby sme sa ako členovia lepšie oboznámili s fungovaním kódu.

Viki: nerobili sa testy a malo by sa to zmeniť. Páčilo sa jej, že sme spolupracovali na taskoch. Peťo Kiš: páčilo sa mu, že sme zorganizovali stretnutie so zákazníkmi, Šimon sa ukázal ako "ľahúň".

Maťo navrhuje ešte stanoviť si jasné metodiky.

Matej: páči sa mu atmosféra v tíme, a to že sme sa chválili :). Čo mu vadí, že by mohla byť väčšia aktivita na TFSku, písať si hodiny, čo sa týka visuálka, tak viac sa oboznámiť s projektom.

Šimon navrhuje kódovské stretnutia, kde by sme spoločne kódili.

Peťo Kušnir: páčila sa mu tímová práca a navrhuje pracovať na reálnych, aj keď jednoduchších taskoch.

Marek: oceňuje prácu v tíme a pomoc od Šimona. Negatívne hodnotí tímovú komunikáciu cez HipChat a odporúča označovať ľudí v komentoch a rýchlejšiu spätnú väzbu. Ešte spomenul, že veľa času strávil na aktuálnom tasku a veľmi mu to nešlo. Navrhuje začať makat' na ľahších, ale reálnych veciach.

Šimon: ++ ľudia v tíme a spolupráca, navrhuje začať makat' na reálnych veciach.

Robo: ++ že sme dokázali pracovať spolu po stretnutí s prof. Bielikovou, ale máme sa vyvarovať problémom s termínmi (stránka). Treba identifikovať potencionálny problém, aby sme si vedeli vytvoriť časovú rezervu.

Robo navrhuje spraviť lean canvas pre náš tímový projekt do najbližšieho stretnutia a nezabudnúť spraviť veci ešte v rámci tohto šprintu (aktualizovať web,...).

Uzatvorenie šprintu

Metodiky:

Vypracovať a hodiť do google docu kvôli pripomienkovaniu (má to byť tímová verzia, aby s ňou všetci súhlasili)

Založiť dokument (základ pre tímovú dokumentáciu) a na začiatok spraviť analýzu súčasného stavu (tiež ako google doc), aby sa rozobrala súčasná situácia a kvôli lepšiemu pochopeniu fungovania kódu (na úrovni tried)

Rozprava o paneloch: vytvorenie prezentácie o tom ako funguje produkt, ale aj o tíme, ako fungujeme v tíme, ako by to mohlo byť...

Roly:

Plánovanie pre tím a jednotlivých členov tímu: Maťo Svrček (metodika), inak prevažne spoločne v rámci tímu

Vyhodnocovanie plnenia plánu a návrh úprav: v priebehu a na konci šprintu priradiť taskom hodnoty, vytvoriť burndown chart => Maťo Svrček

Organizácia komunikácie v tíme: Peťo Kiš

Udržiavanie informácií o stave projektu: TFSko -> Matej, stránka aktuality -> Peťo Kušnir

Evidencia úloh: TFSko = Matej

Identifikovanie a riadenie rizík: odtrhnuté od reality pri tímákoch; identifikácia rizík, ohodnotenie pravdepodobnosťou, že nastane, cenou škody a navrhnuť riešenia, či im budem predchádzať, alebo ich budem ignorovať; vytvorenie nejakého plánovania akcií tímu (mal by sa na to niekto pozrieť a spísať to - prihlásil sa Maťo Svrček)

Zabezpečenie efektívneho znovu-použitia:

Monitorovanie prehliadky vytváraného výsledku: Robo posielal link na code review, Marek má pozrieť prezentáciu z prednášky o code review a vytvoriť projekt v Perconik-u už v tomto šprinte, aby sa robil code review v tomto systéme. Aby to do budúceho týždňa bolo nastavené.

Vyhodnocovanie testov: každý píše testy sám

Identifikácia riadenia chýb v softvéri: ako zalogovať objavený bug, kto ho má riešiť, atď. Spíše Viki

Integrácia softvéru: viac menej hotové

Manažment verzií: Git, ako to používať, ako písať commit správy, ako sa dá automaticky uzatvárať úlohy a priradovať taskom commity

Požiadavky na zmenu: návrh na zmenu, ako vyhodnotiť požiadavku, ako ju zapracovať,...

TP cup:

národ je rozdelený, prebytočná robota za malý zisk.

Rozhoduje kvalita produktu, kvalita prezentácie, použiteľnosť v reálnom živote, biznis plán.

V zimnom semestri: prihláška, v letnom v marci 4 strany + 2 stranový abstrakt po anglicky, vytvoriť poster na IIT SRC a odprezentovať to, napísať na blog a predstaviť sa ako tím, spraviť video prezentujúce tím (60 sec), finálna prezentácia (povinná aj tak) + ak postúpime do finále tak 1 prezentácia navyše.

Čo nám to dá: skúsenosť súťažiť s niekým, podobné ako startup súťaže, prezentovať nápad a víziu.

Robo by bol za, máme sa rozhodnúť do večera že ako.

Šimon spravil veľmi dobré vysvetlenie aktuálneho stavu projektu s víziou na nasledujúce obdobie.

Budúce stretnutie (backlog rooming).

Rozdelenie úloh na nasledujúci šprint

AGREGÁCIE:

- počítanie agregácií z anotovaných dát -> vytvoriť tabuľku v MS SQL, rozšíriť workera (pri pridaní anotácie nech sa inkrementuje príslušný counter) - [zdokumentovanie fungovania Raven DB a aj workera]

API

- zadefinovanie volaní API -> identifikovanie základných call-ov (cez JSON) [vytvoriť projekt, vytvoriť session {v podstate podobné veci aké sú dostupné cez web}]
- autentifikácia cez API kľúč (cez databázu)
- volanie z ALEFu (skryť do libky), premenovať controller
- administrácia -> sledovanie pohľadu

REAL TIME

- add-on (Chrome) -> Publisher -> Event(x,y) OR Event(XPath)
- ALEF -> Subscriber (JavaScript) [format Publisher-Subscriber] -> handler (odozva ALEFu), AJAX
- Publikovanie Eventov -> JQuery \$('event').ontrigger(...); pozrieť BackboneJS framework; pridanie GazeTrackingController-u, ktorý by obsluhoval eventy ktoré prídu z JQuery
- počúvanie eventov cez backboneJS (spojiť so zvyrazňovaním elementov)

ODOZVA ALEFU

- zvyrazňovanie elementu v ALEFe, na ktorí sa užívateľ pozerá

Ohodnocovanie

úloh:

Autentifikácia cez API kľúč (20) - Marek

Volanie z ALEFu (3) - Viki

Administrácia (sledovanie pohľadu) (3) - Matej

Zadefinovanie volaní API (5) - Šimon

Agregácie (20) - Peťo Kiš

Publikovanie eventov (13) - Peťo Kušnir (pozrieť si dokumentáciu k add-onu a skúsiť spísať dokumentáciu, aby sa vedelo do budúcnosti)

Počúvanie eventov + zvýraznenie (13) - Maťo Svrček

Doplniť odhady časov (najlepšie ešte dnes).

Všetci sú už hladní a je tu zima. Je 12:15. Končíme a ideme jesť.

1.4 Štvrté stretnutie

Peťo Kušnir: spravil publikovanie eventov, nepodarilo sa nastaviť extension. Identifikoval si miesto. Nestihol dokončiť úlohu, bol v tom problém. Dnes to dokončí.

Treba spraviť dokumentáciu ku každému tasku, aby ostatní vedeli sa v tom orientovať.

Viktória – nevedela presne ako na to. Nepodarilo sa jej úplne dokončiť task.

Martin Svrček - Mal robiť zvýraznenie eventov. Podarilo sa mu to. Robo mal ešte pripomienky k tomu. Urobil analýzu testovania.

Peťo Kiš - nepodarilo sa mu úplne dokončiť task. Odhaduje to ešte na dosť dlho.

Maťo Kucek - Podarilo sa mu splniť task. Najviac sa mu zabralo rozbehať ALEF.

Marek Šulek - Air spec treba. Dokončil task ale nespravil testy.

Šimon - Spolupracoval na tasku s Marekom, čiastočne zadefinoval API, dnes to dorobí.

Zhrnutie šprintu

Peťo Kušnir - Navrhuje stretnutia kde sa spoločne programovalo.

Viki - tiež je za spoločné programovanie.

Martin - Spokojný.

Peťo - Jeho úloha bola, zle odhadnutá, venoval jej veľa času, potrebuje spolupracovníka. Jeho úloha bola pre viacerých ľudí.

Matej - Robilo sa mu dobre lebo dostal dobrý support.

Robo - Veľa času stále zaberajú informácie, treba si zdieľať informácie.

Marek - Nespokojný s efektivitou svojho programovania,

Šimon ukazoval navrhnuté **API**. Requery budú vo formáte post. Respony vo forme JSON.

Agregácie - Peťo navrhol ako vylepšiť databázu. Dohadovali sme sa ako bude navrhnutá databáza.

Real time komunikácia -

Alef

Pridelenie úloh

- Marek - Pridanie funkcionality do API, support od Šimona - 13 story points
- Šimon - Dozerať a pomôcť nad Marekom a Peťom.
- Martin - Analyzovanie otázkovača v ALEFE - 13 points
- Matej - Manažment a vytvorenie gazetracking projektu v ALEFE, volať funkcie API
- Peťo - Analýza

- Dokumentáciu, musí každý spísať.

1.5 Piate stretnutie

Peter Kiš - robili s Matejom dopyty. Vytváral testovacie dáta, agregáčné dopyty

Robov komentár - otestovať z pohľadu priepustnosti, kde sú limity riešenia, skúsiť si cez simulátor aspoň od jedného človeka či to bude stíhať. Musíte to niekde nasadiť.

Matej Kucek - Nevie sa pripojiť, Nedostal sa k analýze redisu, robil agregáčné dopyty

Robov komentár - heslo je nepoviem, spraviť si nový účet cez ktorý to budeme testovať, Redis nie je už relevantný,

Peťo Kušnir - dorobiť úlohu, začal so serverovou časťou, nestihol všetko

Robov komentár - keď si nie sme istí treba dať komentár, môžeme sa poradiť s minulými vývojármi, zakomentované kódy sú už asi nepotrebné, treba ich mazať a komentovať na úrovni

Šimon Valiček - pokračuje v práci, treba doriešiť deploy na stránku, skúsi to dnes pripomenúť,

Robov komentár - treba to doriešiť, sprístupniť stránku,

Marek Šulek - dorobil funkcie z minulého týždňa, pripravil si funkcie.

Robov komentár - oddeliť zobrazenie od controllera, pýtal sa ohľadne code review či sú všetci v perconiku.

Martin Svrček - dorobil view, opísal svoje problémy.

Viktória Lovasová - spravila manažovanie projektu gaze tracking, ale ešte to nekomunikuje a je to nekonzistentné jazykovo.

Robo rozprával o budúcnosti alefu.

Ďalej rozpráva o obsahu dokumentácie, akým činnostiam sa manažér venoval, podiel práce vyzerá ako jedna tabuľka kde sú vyjadrené pomer kto sa ktorej kapitole venoval. Aplikácie manažmentu tam opíšeme jednotlivé činnosti, v stručnosti zhodnotiť a odkázať sa na príslušnú metodiku. Sumarizácia šprintov - burndown chart, okomentovaný prečo tak vyzerá, opísať ako fungovalo plánovanie a priebeh. Ďalej zhodnotenie čo bolo dobre čo by sme chceli stihnúť.

Diskusia o metodikách.

1.6 Šieste stretnutie

Peter Kušnir:

-moderovať stretnutie

Peter Kiš:

-analyzoval, ako aplikácia komunikuje s RavenDB

-implementácia je neprehľadná, nepoužíva orm ale linq

-obsahuje veľa zakomentovaného kódu – nevieme, či je nefunkčný alebo nepotrebný

Robo:

- v prípade potreby je nutné urobiť markup existujúceho kódu
- pri implementácii nového kódu treba kód komentovať a generovať z neho dokumentáciu
- treba používať db migrácie

Matej:

- nastavoval query v tfs
- dorobiť do alefu nové view v administrácii
 - rozbehal si alef
 - nemá v ňom testovacie dáta

Robo:

- treba sa pridať do redminu alefu
- vie pomôcť s vysvetlením implementácie alefu

Marek:

- študoval rest služby

Robo:

- treba si pozrieť ako sa v súčasnosti realizujú dopyty na db

Viki:

- upravila api call z alefu
- vytvorila testy, ale ich funkčnosť je závislá na implementácii api
- ešte mala napísať metodiku testovania, čo nestihla.

Robo:

- testy by nemali priamo volať api ale mali by sa napísať mocky
- toto iste treba aplikovať pri testovaní funkcií, ktoré využívajú model
- treba si pozrieť dokumentáciu k Rspec
- pri zistení chyby pri testovaní je dobre vytvoriť nový task v tfs

Martin:

- robil analýzu implementácie zachytávania eventov zverejnených browser extensionom pomocou backbone.js.
- písal metodiku riadenia scrum.

Robo:

- treba si pozrieť repozitár alefu, kde je možné vidieť použitie backbone.js
 - backbone.js využíva model, ktorý komunikuje so serverom
 - view, ktorý zobrazuje dáta

Šimon:

- nová doména
- aliasy + deploymenty pre novú doménu
- vytvorenie kópie db
- rozpracovanie api

Robo:

- v db nie sú až tak hodnotné dáta

- špecifikáciu api je nutné stihnúť dokončiť do konca šprintu
- v 9. týždni sa odovzdáva priebežná dokumentácia k projektu, riadeniu a big picture
 - big picture
 - opis pôvodnej myšlienky služby
 - architektúra
 - moduly
 - prepojenie
 - komunikácia
 - výstupy analýzy jednotlivých modulov
 - súčasťou dokumentácie výsledky testov modulov
 - dokumentácia riadenia
 - big picture
 - role členov a podiel práce
 - zodpovednosti na systéme jednotlivých členov
 - aplikácie manažmentov
 - opis jednotlivých šprintov
- na testovanie budeme používať Nspec
- v rámci tp cupu budú k dispozícii mentoringy
 - prezentácia/business
- treba vytvoriť a vytlačiť lean canvas
- na code review treba používať perconik
 - Robo sa ozve Karolovi Rástočnému, aby nám v ňom vytvoril projekt.

1.7 Siedme stretnutie

Viktória Lovasová

naimplementované manažovanie projektov
testy nie sú.

Peter Kiš

stihli sa agregované dopyty, je to otestované
začal sa robiť manažér, ktorý zbiera dáta a vytvára agregácie
testovať cez simulátor, ale on nie je prepojený s anotáciami

Peter Kušnír

vytvoril dokumenty na zdokumentovaný kód
treba otestovať nejako.

Šimon Valíček

bojoval so serverom, písal na fakultný mail ale neodpísali
nemal iný task.

Marek Šulek

spravil tasky lokálne
nemá testy
ani code review

Martin Svrček

počúvanie v alefe javasript
otestovanie
analýza otázkovača

Retrospektiva

Robo

dúfa že, na konci každého šprintu bude vždy nasadené testy sa spustia.
Je koniec šprintu a tasky nie sú úplne skončené
snažiť sa uzavrieť
hlavne testy.

Martin Svrček

rozdeľovať tasky už na stretnutí, nie len user story
na budúci týždeň v stredu prejdeme si ďalšie user stories a ohodnotíme to aj na ďalšom
stretnutí už len vybrať a rozbiť na tasky a takto by sme to stíhali.

Marek Šulek

viazne komunikácia.

Začiatok 4. šprintu

Plánovanie

API

Sessions

Users: previazať usera so systémom druhým, vytvoriť používateľa a zapamätať idčko
AOI

Agregácie

get fix count, fix duration

AGREGÁCIE

prerobiť tabuľku v časti statistics
a zobrazíť pre týchto používateľov takéto počty
kľudne aj viac tabuliek
nemusí to byť user friendly

Vytváranie agreácií

JS KNIŽNICE

evaluation questions
budú volané z kódu
alebo definovať v administrácii custom pravidlá
urobiť trigger cez tú otázku.

AOI

URL

ako to funguje, ignorujem všetko za / napr že dám tam * - zanalyzovať či to tak
funguje, ak áno či to anotačný worker zohľadňuje.

xpath

nevrátiť len jeden ale viac.

1.8 Osme stretnutie

Šimon:

došpecifikovanie API, pretrvávajúce problémy s pridelením domény

Peter Kušnir:

vytvoril základnú kostru API volaní pre oblasti záujmu, otázky ohľadom typu návratových hodnôt, potrebuje ešte došpecifikovať niektoré volania

Marek:

Začal robiť na API funkciách pre správu userov. Upravil celkovú štruktúru API a zmenil návratové hodnoty funkcii na DTO príslušného modelu + http status kód, pomáhal Šimon. Napísal aj testy na niektoré vytvorené funkcie.

Viki:

Mala robiť API funkcie pre sessiony. Vytvorila základné funkcie podľa špecifikácie, pomáhal Marek.

Maťo Svrček:

Robí na analýze a implementácii otázkovača v Alefe a jeho prepojenie so systémom cez api. Podarilo sa mu zdokumentovať otázkovač v Alefe, vytvoril o tom dokumentáciu a začal s implementáciou.

Matej Kucek:

Mal spraviť zobrazovanie agregácií vo webovej aplikácii. Mal problémy s testovaním kódu keďže to u seba nevedel rozbehať. Došiel Mišo a pomohol mu s tým a aj rozprával o tom ako testovať aplikáciu.

Peťo Kiš:

Podarilo sa mu dokončiť počítanie agregácií.

1.9 Deviate stretnutie

Viki:

Dorobila API funkcie pre správu sedení, nestihla napísať testy, kód skontroloval Marek v rámci pair programming-u.

Matej:

Podarilo sa mu rozbehať webovú aplikáciu u seba aj keď musel následne riešiť nejaké problémy s prihlásením. Ako riešenie si musel rozbehať u seba MySQL databázu v ktorej sú záznamy o užívateľoch a preto sa mu nepodarilo úplne dokončiť vizualizovanie agregácií na webe

Peťo Kiš:

Dokončenie práce na počítaní agregácií, spísanie dokumentácie a testovanie priepustnosti.

Peťo Kušník:

Doplnil funkcie do API pre správu oblasti záujmu, zmenil návratové hodnoty funkcií podľa dohodnutého štandardu, začal na testoch ale nestihol dokončiť.

Marek:

Dokončil API funkcie pre správu userov, napísal testy a menil celkovú štruktúru projektu API. Pomáhal Viki s implementáciou.

Maťo Svrček:

Implementácia generovania otázky na základe prijatého eventu.

Šimon Valíček:

Trošku s meškaním ale predsa, večné problémy so serverom, došpecifikoval API volania, stretol sa s Petrom Lackom kvôli doméne, aj tak sa s tým veľmi nepodarilo pohnúť.

Semestrálna retrospektíva:

Podarilo sa spraviť počítanie agregácií a základ pre API, poopravovali sa niektoré chyby v kóde, robilo sa na prepojení s Alefom, od ktorého sa kvôli budúcim zmenám nakoniec upustilo, tím sa uchytil v téme, komunikácia bola na dobrej úrovni ale treba sa vyvarovať situáciám kedy sa týždeň kvôli nejakej chybe niekto nevie pohnúť a tím sa o tom dozvie až na stretnutí, zlepšiť prácu v kolektíve, stretnutia aj mimo vyhradený rozvrh a riešenie problémov spoločne, zlepšiť písanie testov a vykonávanie prehliadok kódu, neriešiť úlohy na poslednú chvíľu, kladne je hodnotená ústretovosť predošlého tímu a ochota pomôcť a poradiť pri vzniknutých chybách alebo nasadzovaní a testovaní aplikácie.

Plánovanie posledného týždňového šprintu:

Dokončenie rozrobených úloh z predchádzajúcich šprintov, doplnenie testov, spísanie dokumentácie riadenia a dokumentácie o inžinierskom diele a nasadenie celého systému na server.

1.10 Stretnutie so zákazníkmi

- Prepojenie s ALEFom , Záznam z kamery – problémom je synchronizácia, spôsob odosielania , Treba dbať na čo sa používa eye-tracker.
- Biznis problém
- MB – Posielať dáta klientovi, (Šimon – „prídeme s tým o biznis“)-> MB - neprídeme s tým o biznis, lebo to môžeme nejako spoplatniť -> ??.
- Ak to budeme posielať na server tak to nebude škálovateľné -> ak bude 200 aplikácií s 200 používateľmi tak všetko bude tiecť na náš server a z neho a bude to problém. Ideou bolo aby tá druhá aplikácia nemusela robiť cez svoju databázu nejaké veci. Skôr investovať do toho aby to mohol klient použiť.
- M. Labaj - Ani to nemusí ísť vôbec nikam a môže sa to v javascripte spracovať. Javascript bude čakať na event a ak sa niečo bude sledovať (nejaký pohľad) tak bude reagovať nejako.

SOFTEC-FIIT_projekt

- M. Bieliková - Pozor na konkurenciu so Softecom -> s nimi sa bude robiť projekt na UX lab – budú tam ľudia aj od nás. Nemôžeme replikovať to čo je v požiadavkách na tento projekt.
- Minulý projekt robil spracovanie s webom -> ak by sme to dotiahli tak sa nebijeme so softecom.
- Pridanie poznámok k priebehu experimentov -> toto sa bude riešiť so Softecom.
- Kvalitatívne testovanie – napr. zadávanie úloh a pod. – máme toto vôbec riešiť ? – skôr nie, lebo to sa rieši so Softecom. Toto ani nie je potrebné , skôr sa to robí ako súčasť webu.

Vizualizácia

- aj pre jedného aj pre viac používateľov – nie on the fly ale prepočítané vopred
- heat mapy a obrazovka za tým, za grafom
- Vizualizácia pre učiteľa – **dôležité pre zákazníkov !!!** -> na základe toho by vedel že ktoré časti textu sú zlé. – systém by vedel pomôcť učiteľovi na základe získaných údajov zo sledovania – toto by bolo veľmi dôležité.

ALEF integration

- MB: dáta sa dostanú do ALEFu – a priamo v ALeFe vymyslieť scenár súvisiaci s pozorovaním
- ML: definovanie pravidiel (serverové, klientské) keď sa človek pozrie na LO tak sa ho spýtame prečo tam pozerá -> toto už funguje !!!
- Zamerať sa na webové aplikácie.
- Analýza portálu -> toto bola 1. naj stránka a na toto sa nepozeralo.

Zhrnutie

- Ako urobiť vyzbierané dáta užitočné – vizualizovať ich a prepojiť ich s aplikáciami.
- Orientovať sa na pomoc učiteľovi.

2 Letný semester

2.1 Prvé stretnutie

Robo otvára stretnutie. Rozpráva o minulom semestri, čo sa spravilo a že sme to nemali správne rozvrhnuté na use casy. Nasleduje plánovanie úloh do backlogu.

2.1.1 Plánovanie úloh do backlogu

Filter fixácií - Jedna možná úloha je pracovať s fixáciami – pozrieť sa na to aké sú možné implementácie - analýza, teda ešte nie implementovať hneď ale keď nájdeme tak to použiť potom.

Otestovať zachytávanie eventov v JS knižnici. Aj všetky ostatné veci, ktoré sa robili - doriešiť server.

Štatistiky – zistiť počet fixácií – Maťo už minulý rok začal pracovať na tom ale stránka nefungovala poriadne. Treba spraviť 2 typy zobrazení tabuľkové a grafické. Z agregácií sa zoberú dáta – počet fixácií pre oblasť záujmu zvolený projekt a spraviť tabuľku.

Graficky zobrazit' – to isté čo je zobrazené v tabuľke, oboznámiť sa s knižnicou NodeJS. Napr: budú defaultne zobrazení všetky ale bude sa dať vybrať nejaký filter, koho zobrazit'.

Filter - nebude v kóde napevno zvolený z akého projektu chcem robiť štatistiky.

Druhá metrika – „dwell time“ – ako dlho som bol pohľadom v danej oblasti.

Time to first fixation - čas za koľko bola zaznamenaná prvá fixácia.

Export obrázku - export grafu.

Simple ALEF - statická stránka, nebude ani menu ani nič, len zobrať nejaký text a naformátovať tak aby sa dal čítať aj nadpis. Ale aj vhodne štruktúrovať text. Aby aj html bolo správne. A vložiť JS knižnicu. Overiť cez simulátor či v reálnom čase zbieram eventy. A potom podfarbiť element nejaký keď sa naň pozerá používateľ. Neskôr bude aj jednoduché menu.

Analýza čítania vzorov - aké sú metriky, čo sa používa či nie sú implementácie už nejaké.

Zobierať dataset z pohľadov - vymyslieť task.

Vytiahnuť z API informácie - aké sú oblasti záujmu atď.

2.1.2 Zaradenie úloh do 1. šprintu

Filtrovanie fixácií - Matej Svrček

Spravenie tabuľky - Matej Kucek

Simple ALEF - Viktória Lovasová

Grafické zobrazenie - Peter Kiš

Vytiahnuť z API oblasti záujmu - Peter Kušnir

Analýza čítania vzorov - Martin Svrček

Presunúť komunikáciu z extension do JS knižnice - Marek Šulek

Dokončiť server - Šimon Valíček

Ďalšie stretnutie bude vo štvrtok o 16:00 v Jobsovom štúdiu.

2.2 Druhé stretnutie

Na predchádzajúcom stretnutí sme si stanovili predbežne úlohy na tento šprint, ktoré teda platia. Martin Svrček si má naštudovať problematiku filtrovania fixácií prečítaním rôznych odborných článkov. Matej Svrček má do webovej aplikácie dorobiť tabuľky, ktoré vizualizujú agregáčné dáta. Viktória Lovasová má vytvoriť statickú stránku, na ktorej bude neskôr spravený experiment. Má byť jednoduchá, bez menu. Peter Kiš má graficky znázorniť vo webovej aplikácii štatistiky z agregáčnych dát pomocou grafov. Peter Kušnir má vytiahnuť z api oblasti záujmu. Martin Svrček dostal ešte jednu úlohu o analýze čítania vzorov. Marek Šulek bude presúvať funkcionality z rozšírenia do JS knižnice, aby si používateľ nemusel inštalovať ešte aj rozšírenie. Šimon Valíček má sfunkčniť server aby sme mohli nasadiť projekt.

Filtrovanie fixácií - Martin Svrček
Spravenie tabuľky - Matej Kucek
Simple ALEF - Viktória Lovasová
Grafické zobrazenie - Peter Kiš
Vytiahnuť z API oblasti záujmu - Peter Kušnir
Analýza čítania vzorov - Martin Svrček
Presunúť komunikáciu z extension do JS knižnice - Marek Šulek
Dokončiť server - Šimon Valíček

2.3 Tretie stretnutie

Na tomto stretnutí sme si povedali, kto sa kam so svojou úlohou dostal.

- Máme nasadené na serveri.
- GoogleChrome extension treba ešte dopracovať.
- Statická stránka s textom je spravená.
- Otestovanie zachytávania eventov treba ešte dotestovať - treba klienta nasadiť na server.

2.4 Štvrté stretnutie

Každý povedal čo za posledný týždeň stihol urobiť.

Peťo_Kiš: je to na dobrej ceste, už je hotové zobrazovanie tabuliek. Ukazuje čo spravil.

Matej: Čítal články, má viac pozrieť Text 2.0

Marek: Dorobil selecty, keď je hotový môže pomáhať Šimonovi.

Šimon: Zistoval čo má robiť.

Peťo_Kušnir: Maťo (Kucek) mu dosť pomohol. Pri alertoch nastáva server error. Je to preňho neznáma krajina, Robo sa stotožňuje. Keď sa zdalo že už je všetkému koniec, tak prišiel Matej s nápadom na rebuild. Fungovalo to

Viki: Padá jej simulátor aj klient. Písala Jožkovi Tvarožkovi ohľadom experimentu. Vymýšľala úlohy na experiment.

Martin: Venoval sa Rku.

2.5 Piate stretnutie

Robo otvára stretnutie, ideme si povedať, kto ako dokončil svoje úlohy.

Martin Svrček rozpráva o článku na IIT SRC, čo treba poupravovať: pridať obrázok, ktorý vysvetľuje infraštruktúru nášho systému. Martin má spraviť samostatnú knižnicu na určovanie fixácií do C# z R-ka.

Peter Kiš spravil filter, dokončil úlohu, ku grafom sa ale nedostal. Tú úlohu si preniesie do ďalšieho šprintu. Ak by stihol tak by mohol spraviť aj export grafov ako obrázky.

Matej Kucek čítal si články o algoritmoch čítania. Ďalšia jeho úloha bude naimplementovať takýto algoritmus, ktorý bude vedieť povedať, či používateľ čítal, nečítal, nepochopil text. Ku koncu šprintu dostane dáta z experimentu a na nich odskúša algoritmus a dostane výstup.

Budúci týždeň v utorok treba ísť odprezentovať na PeWe seminár o našom projekte.

Šimon Valíček vyriešil volania cross-domén. Ďalej riešil API triggey, ale to sa mu presúva lebo nestihol.

Peter Kušnír mal robiť upozornenia, spraviť formulár na stránke ale ešte si to preniesie do nového šprintu.

Marek Šulek bude pokračovať vo svojej úlohe a otestuje predchádzajúcu úlohu.

2.6 Šieste stretnutie

Robo otvára stretnutie. Povieme si čo sme spravili za týždeň.

Peťo Kiš zapracoval na grafoch, Zatiaľ to vyzerá byť dobre. Už to len treba dokončiť.

Matej Kucek robil algoritmus čítania. Teraz to plánuje overiť. A ešte doimplementovať parametre.

Viktória dokončila prípravu experimentu, bude ho treba potom aj vyhodnotiť: počet fixácií, dĺžka, dwell time, spraviť prehľad toho experimentu zo štatistík v Tobii studiu.

Peter Kušnír robil vo webovej aplikácii ale nepohol sa.

Marek Šulek si pripravil nejaké screeny. To je všetko čo si predpripravil.

Šimon Valíček zistil čo je chyba a pracuje na tom.

Martinovi Svrčkovi sa pokazil počítač, preto nič nespravil.

Nabudúce bude stretnutie cez skype predbežne vo štvrtok o 09:00.

2.7 Siedme stretnutie

Zápisnica z dňa 2.4.2015. Stretnutie je cez skype.

Zúčastnení:

Marek Šulek: ✓

Viktória Lovasová: ✓

Peter Kušnír: ✓

Peter Kiš: ✓

Šimon Valíček: ✓

Róbert Móro: X

Martin Kucek: X

Martin Svrček: X

Každý napísal, čo za posledný týždeň stihol spraviť:

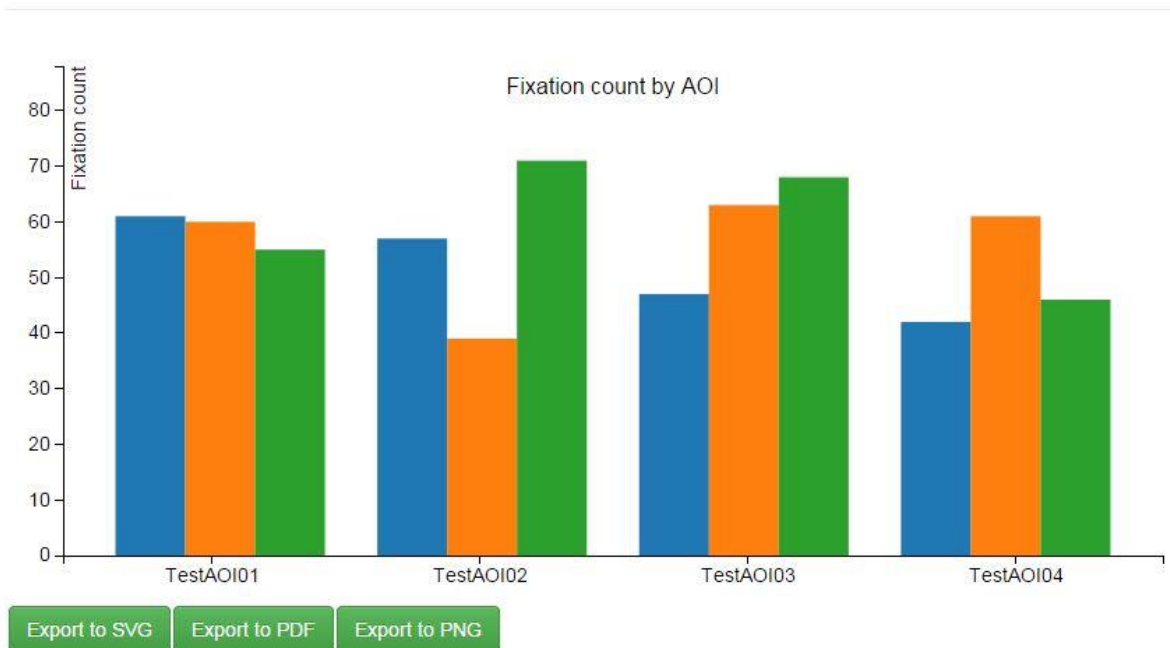
Marek Šulek: Spravil som prezenčku stretnutia. Vyriešil som z časti cross domain volania. Opravil som v našej knižnici volania na klienta, bola tam zlá adresa. Už sa dotahujú dáta kam sa používateľ pozerá pomocou simulátora ale neviem to rozparsovať lebo to ide do error vetvy, kvôli tomu, že teraz server vracia JSON namiesto JSONP, pretože extension do chrome umožňoval aj cross domain volania. Šimon sa o to postará. Musíme spoločne doriešiť gaze eventy. Inak kostra tasku je pripravená, čaká sa na url do api kde sa dotiahne xpath.

Viktória Lovasová:

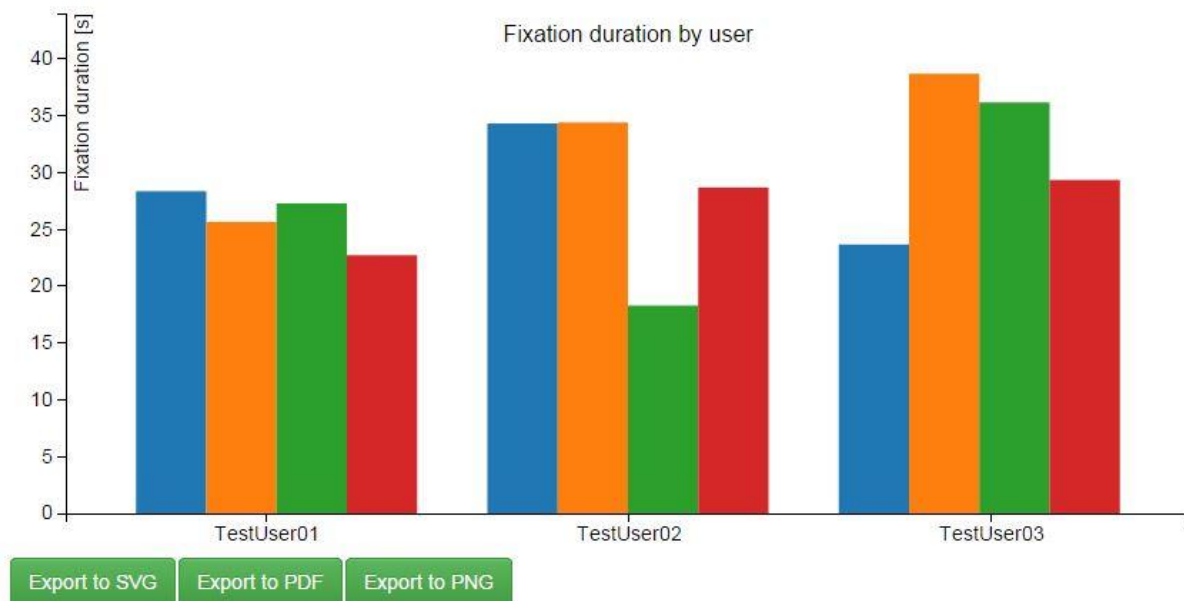
Spravila dokument o priebehu experimentu, ktorý sa konal 23.3.2014 v laboratóriu a vložila ho do drivu. Exportla dáta z projektu a poslala ich Matejovi Kuckovi, ktorý ich použil na analyzovanie vzorov čítania.

Peter Kiš:

Dokončil som vykresľovanie grafov, klasických bar chartov a aj grouped bar chartov:



Obrázok 1: Vizualizácia počtu fixácií.



Obrázok 2: Vizualizácia trvania fixácií.

Pridal som pod každý graf 3 tlačítka na export grafu do SVG/PDF/PNG.

Čo sa týka grafov, chýba dorobiť legendu. Export grafov ešte nie je úplne ok, grafická stránka sa exportne v pohode, opisy osí sú značne zmenšené do nečitateľnej podoby - treba poriešiť.

Šimon Valíček

Pomáhal som Marekovi s cross origin volaniami v js knižnici, navrhli sme spolu riešenie, ktoré spočíva v úprave serverovej časti klientskej aplikácie. Ďalej som pracoval na konfigurácii trubiek slúžiacich k pripojeniu databázy raven. Začal som implementovať api pre alerty.

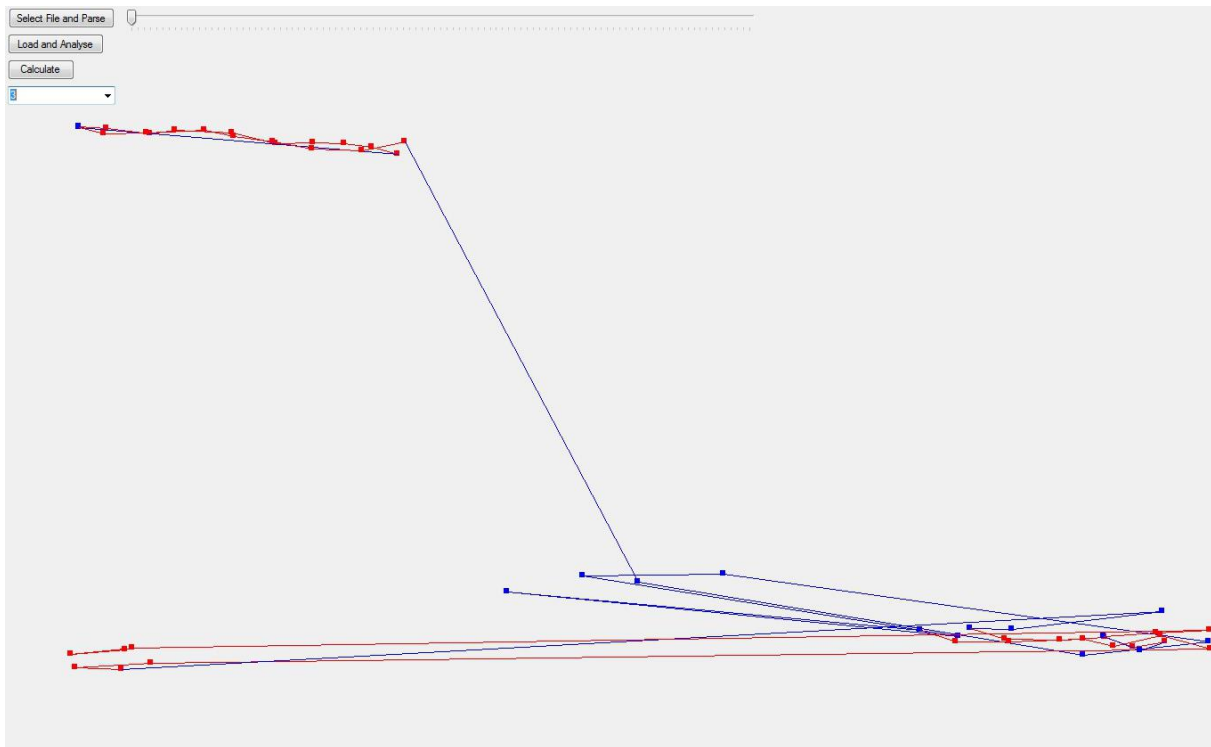
Peťo Kušnir

Robil som na webovej apke na podstránke s upozorneniami. Momentálne je možné vytvárať upozornenia, vylistovať si existujúce upozornenia pre jednotlivé oblasti záujmu aj pre celé sedenie a editovať existujúce upozornenie. Potrebujem ešte dorobiť mazanie upozornení a celú vytvorenú časť naštylovať + nejaké validácie.

Matej Kucek

Upravil som si dáta do potrebnej podoby. A testoval som na nich algoritmus detekcie čítania. Potom som následne doladľoval parametre a vylepšoval algoritmus. Vyzerá to, že to čítanie to vie rozlíšiť pekne, myslím si, že keby bola veľkosť písma pri experimente ešte väčšia, čiže by aj fixácie boli presnejšie tak by sa zvýšila aj úspešnosť algoritmu.

Červené body/čiary symbolizujú čítanie.



Obrázok 3: Čítanie textu.

Martin Svrček

Ja som pracoval a stále pracujem na tom prerobení funkcií I-VT filtra do C#. Keďže tie jednotlivé Matlab funkcie a C# funkcie nepoznám trvá to dlhšie. Vytvoril som základnú štruktúru pre dáta používané v algoritme. Taktiež som si vytvoril všetky funkcie a vzájomné volanie s atribútmi. Teraz dopĺňam ich telá.

2.8 Osme stretnutie

Rozdelenie úloh:

Matej Kucek:

Rozpoznať, či je to čítanie alebo sceming, pre každý odsek zagregovať štatistiky. Pri prezentácii môžeme takisto povedať o experimente, aké dáta sme nazbierali a kam to chceme posunúť ďalej. Ideálne spraviť veľký experiment v UX labe.

Peter Kiš:

Dorobiť grafy a agregácie.

Peter Kušnír:

Ešte doštýluje alerty, pozrie sa na poster, už budúci týždeň má mať nejaký návrh.

Viktória Lovasová:

Pripraviť obsah posteru na IIT SRC aj s Martin Svrčekom.

2.9 Deviate stretnutie

Peter Kiš sa má pozrieť prečo nejdú tie agregácie a vyskúšať to aj na eyetrackeri.

Martin Svrček má dokončiť implementáciu určovania fixácií.

Matej Kucek skúšal testovať svoj algoritmus, za úlohu ho má dokončiť a dať ho do infraštruktúry. Odovzdáva sa 25. mája celý projekt.

Marek Šulek má dorobenú knižnicu, má pozliepať viaceré veci.

Peter Kušnir spravil poster, dokončil alerty, zmergoval branche.

Šimon Valíček, robil API, len sú tam nejaké problémy, tak to ešte dokončí. Má pomáhať a asistovať pri nasadzovaní.

Viktória Lovasová pomáhala pri vytváraní obsahu na poster. Má si premyslieť lepší scenár pre tých 10 ľudí, dať tam skôr vzdelávacie testy, nebudeme to robiť v tobii studiu tak sa stránky môžu si skrolovať, môže byť aj viac textov. Pozorné čítanie nejako inak vymyslieť. Dať tam logickú úlohu, alebo naučiť sa definíciu to by bola spätná definícia, dať tam texty z alefu, z kurzu FLP. Štýlovanie jednoduché. Vytvoriť si projekt a spraviť oblasti záujmu.

2.10 Desiate stretnutie

Matej Kucek testoval ešte algoritmus, zaintegrovanie do infraštruktúry ešte má spraviť.

Martin Svrček dokončil algoritmy, ešte to bude musieť odskúšať.

Peter Kiš sa má pozrieť v tfsku na zdrojáky z minulého roku a porovnať prečo to nefunguje. Štatistiky ešte nefungujú.

Marek Šulek mal dorobiť alerty, spravil to. Ale treba to ešte trochu upraviť.

Šimon Valíček riešil sessiony. Skontroloval v databáze, či sedia agregácie a pomáhal Marekovi. Má dorobiť API pre Mateja.

Peter Kušnir robil JS knižnicu, chce dať do tej knižnice vytváranie záujmov lebo sa to dá robiť len cez rozšírenie.

Viktória Lovasová

Pripravila statickú stránku na experiment a navrhla prvé úlohy na experiment.

Záverečná správ má vplyv na hodnotenie, treba opísať aj ako sme to vyhodnocovali, aj ten prvotný experiment, ale tam nebol nikto externý a netestovali sme celú infraštruktúru. Takže je nutná podmienka spraviť aj druhý experiment.

2.11 Jedenáste stretnutie

Martin Svrček otestoval svoju metódu ale nedáva dobré výsledky.

Peter Kušnir pokračoval vo vytváraní oblastí záujmov, vyskytli sa nejaké problémy ale aj kontaktoval Roba. Do budúceho týždňa dokončí všetko.

Matej Kucek mal implementovať do systému vzory na čítanie, to spravil ale ešte nevyskúšal.

Marek Šulek dorobil JS knižnicu na sledovanie, či sa používateľ pozerá do oblasti záujmu.

Potom by trebalo spojiť Peťovu a Marekovu JS knižnicu.

Šimon Valíček pracoval s Peťom na add-onoch ale nejdú. Pridávanie záujmov funguje.

Peter Kiš skúšal sfunkčiniť klienta, správa sa podľa očakávaní, teda nefunguje.

Viktória Lovasová pripravila experiment, exportovala projekt z Tobii studia a dala ho na drive, odkiaľ ho potom môžeme importovať. Experiment má 5 úloh.

Teraz treba dokončiť úlohy a napísať záverečnú správu na TP cup.

2.12 Dvanáste stretnutie

Robo nám dával informácie k záverečnej správe – dokumentácii.

Dokumentácia

- Globálny pohľad:
 - Používateľská príručka, inštalačná príručka – rozšíriť ich príručku alebo sa odkázať na ich príručku.
 - Ak tam ešte nie je nič o experimente, tak to dajme do príloh
 - Aj diagramy komunikáciu cez api ako activity diagram alebo sekvenčný
 - aj CD sa má napáliť
 - trochu viac rozpísať

Kto čo spravil?

Peter Kiš: simulátor nefunguje lebo dáta sa nikam neukladajú, takže sa na to ešte musí pozrieť. Keď by to ale aj ukladalo do Raven databázy, tak agregované dáta sa neukladajú.

Martin Svrček ešte doopraví svoj skript.

Matej Kucek dorobil analýzu čítania a mal dorábať tú vrstvu nad tým, zatiaľ je tam dátová vrstva ešte dátové služby chýbajú, zle sa mu to zmergovalo tak ešte to musí docommitovať.

Peter Kušnir robil knižnicu JS.

Marek Šulek trochu refactorol kód, chcel zmergovať knižnicu ale ešte keď to Peťo nemal hotové tak to nemohol spraviť. Treba to teda zmergovať a zdokumentovať aj diagramy môžu.

Šimon Valíček dorábal do APIčka tie veci čo mu chýbali z minula, potom s Peťom skúšali nasadzovať sessiony, že kde je chyba. Tak teraz má zas Peťovi pomôcť a už to nejako dorobiť.

Experiment sa bude konať v ux labe, kde by bol spustený klient, JS knižnica v stránke by bola bez rozšírenia a taktiež by sa to nahrávalo aj Tobii Studiom. Overíme si, či fungujú tie veci, či sa generujú štatistiky, agregácie, či sa dali oblasti záujmu zdefinovať a nakoniec vygenerovať aspoň spätne do prezentácie záverečnej to napísať. Aj vizualizácia by sa dala spraviť, dala by sa stránka nejako číselne pri oblasti záujmu alebo podfarbilo, že s to podarilo nakoniec celé spojiť. Do Ravenu aspoň nech sa ukladajú dáta už vtedy sa dá ísť spraviť experiment.

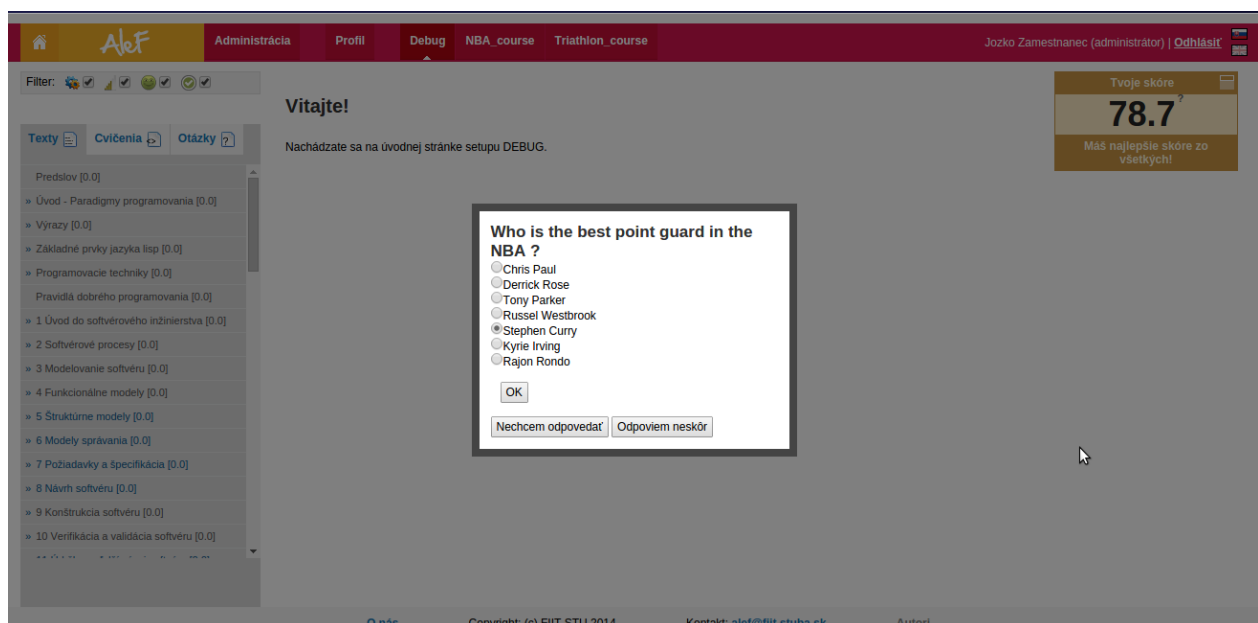
Na konci víkendu treba dať Robovi dať vedieť, v prípade že to bude hotové, treba sa dohodnúť s niekým z ux labu kto by nás tam pustil.

Príloha C: Pýtač v systéme ALEF (spätnoväzobné otázky)

1 Čo je to pýtač ? (evaluation_question)

V rámci ALEFu existuje otázkovač a pýtač:

1. otázkovač (CRANE) - Tento otázkovač systému ALEF poskytuje základné rozhranie pre zobrazovanie otázok, odpovedí a ich následné hodnotenie študentami. CRANE tento scenár rozširuje o možnosť odpovedania na otázky. Všetky otázky sú zobrazované ako samostatná html stránka. Bližšie informácie v časti Otázkovač v systéme ALEF (CRANE).
2. pýtač (spätnoväzobné otázky) - Tento pýtač umožňuje tiež zobrazovanie otázok a možnosť odpovedania na tieto otázky. Jeden z rozdielov je však v tom, že jednotlivé otázky sú generované ako samostatný formulár, ktorý sa zobrazí na aktuálnej html stránke (Obrázok C1).



Obrázok C1: Zobrazená otázka.

V celom nasledujúcom dokumente sa budeme venovať práve pýtaču zobrazovanému ako prídavný formulár v danom dokumente.

Možnosť takéhoto zobrazovania má jeden dôležitý význam z hľadiska možností v systéme ALEF. V kontexte presonalizovaného webu je takto možné reagovať na individuálne vzory správania sa používateľa a generovať mu na základe týchto vzorov jednotlivé otázky. Otvára sa nám tak možnosť:

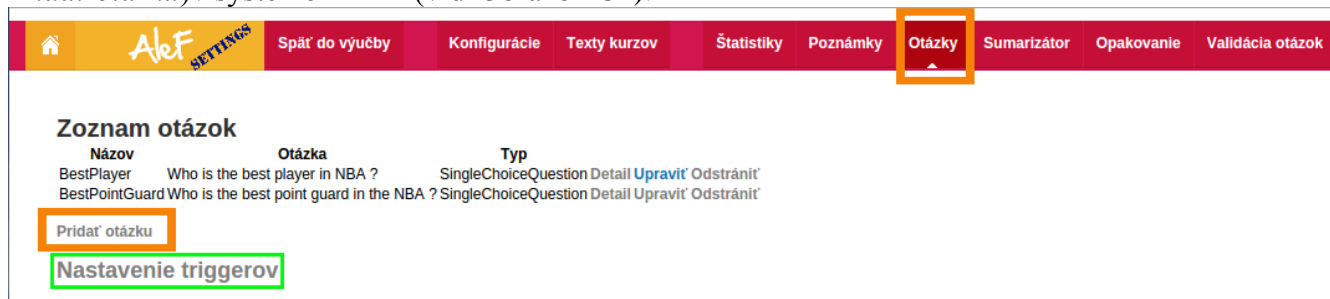
- Pomáhať študentom pri štúdiu
 - Tento scenár súvisí hlavne s možnosťou reagovať na určité akcie používateľa, ako napríklad dlhodobé zlé riešenie cvičení alebo dlhá doba strávená štúdiom jednej problematiky.
- Získavať spätnú väzbu na určité prvky systému
 - Tento scenár je veľmi dôležitý, lebo nám umožňuje získať informácie o niektorých prvkoch systému. Takisto môžeme zisťovať prečo daný používateľ nepoužíva niektoré nástroje v systéme ALEF.

2 Ako funguje takýto pýtač ?

Fungovanie pýtača je pomerne jednoduché. Jednotlivé otázky sú uložené v databáze, ktorá konkrétne obsahuje nasledujúce tabuľky:

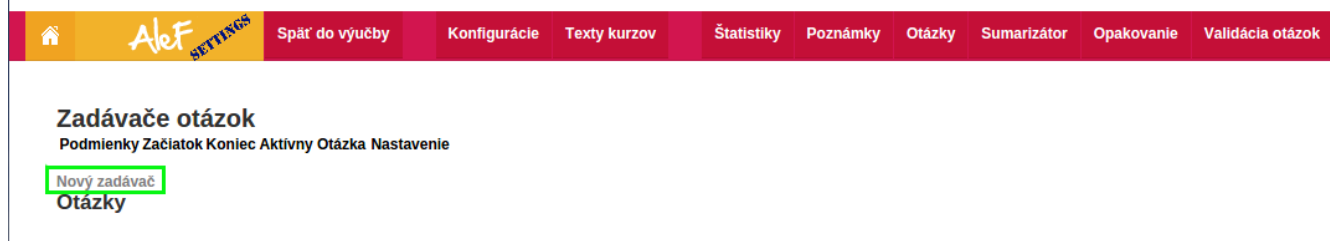
- EvaluationQuestion
 - Základná tabuľka pre otázky, pričom každá otázka sa skladá z: "id", "question_type", "content", "answers", "name", "group", "created_at", "updated_at".
- EvaluationQuestionQueue
 - Tabuľka obsahujúca otázky “čakajúce v rade” na to aby mohli byť zobrazené používateľovi.
- EvaluationQuestionTrigger
 - Tabuľka obsahujúca zavádzače pre jednotlivé otázky. Tieto zavádzače obsahujú pravidlá kedy sa môžu jednotlivé otázky zobraziť.
- UserAnsweredEvaluationQuestionRelation
 - Tabuľka obsahujúca záznam o tom, na ktoré otázky odpovedal daný používateľ.
- UserWasOfferedEvaluationQuestionRelation
 - Tabuľka obsahujúca záznam o tom, že danému používateľovi bola ponúknutá daná otázka.

Jednotlivé otázky sa dajú generovať prostredníctvom záložky *Administrácia->Otázky* (link *Pridať otázku*)v systéme ALEF (viď Obrázok C2).



Obrázok C2 - Pridanie otázky

Jednotlivé zavádzače sa taktiež pridávajú prostredníctvom záložky *Administrácia->Otázky* a následne je potrebné kliknúť na odkaz *Nastavenie triggerov* (Obrázok C2) a v ďalšej časti na odkaz *Nový zadávač* (Obrázok C3).



Obrázok C3: Pridanie zavádzača (trigger).

V Administrácii je možné upravovať, odstraňovať a pridávať jednotlivé otázky.

Pridanie otázky do “radu na čkanie” sa realizuje v časti *Detail* (Obrázok C2), kde je potrebné kliknúť na odkaz *Testovať*.

Takto pridaná otázka sa následne bude zobrazovať v systéme ALEF (Obrázok C1). A ako bolo spomenuté, podmienky jej zobrazovania môžeme určiť pomocou zavádzačov.

2.1 Vyvolanie otázky priamo z kódu

Ak by sme chceli vyvolať otázku priamo z kódu existujú na to dva spôsoby:

1. Jedna možnosť je vytvoriť *EvaluationQuestionTrigger*, v ňom sú podmienky, ktoré sa vyhodnocujú vtedy, keď sa zisťuje, či nie je pre používateľa nová otázka a ak sa splnia, položí sa otázka.
2. Druhá možnosť je rovno vytvoriť záznam *EvaluationQuestionQueue*, tam sa potom nič nezisťuje, ale pri najbližšej možnosti sa otázka (podľa priority vo fronte ak ich je tam viac) zobrazí.

3 Implementácia

Samotnú implementáciu pýtača môžeme rozdeliť o dvoch častí:

- Časť administrácie
- Časť generovania otázok - zobrazovanie otázok, odpovedanie a následné spracovanie odpovedí

3.1 Administrácia

V súbore *EvaluationQuestionsController* sa v prvej časti nachádzajú funkcie na administráciu otázok (pridávanie, odstraňovanie, upravovanie, a pod.).

V súbore *EvaluationQuestionTriggersController* sa nachádzajú funkcie na administráciu zavádzačov (pridávanie, odstraňovanie, upravovanie, a pod.).

V časti *views->evaluation_questions* a *views->evaluation_question->triggers* sa následne nachádzajú konkrétne GUI pomocou, ktorých je správa realizovaná.

3.2 Generovanie otázok

Veci spojené s generovaním otázok sa nachádzajú v druhej časti súboru *EvaluationQuestionsController*.

S generovaním otázok je spojené aj ajax volanie v súbore *evaluation_questions.js.erb*. O toto sa starajú funkcie *askForQuestions()* a *getEvaluationQuestionHTML()*.

V súbore *EvaluationQuestionsController* sú dôležité najmä metódy:

- *question* - funkcia, ktorá ako prvé kontroluje či sa už môže generovať ďalšia otázka (kontroluje či už ubehol dostatočný čas od predchádzajúcej otázky). V tomto kontexte sú volané funkcie.
 - *topmost_question_in_queue* - získanie prvého prvku z radu “čakajúcich”.
 - *offer_question* - ponúknutie otázky.Následne sa pozerá či už nejaká otázka “čaká v poradí” na zobrazenie. Treťou časťou je zistenie či si nejaký zavádzač žiada generovanie otázky.
- *answer* - funkcia na spracovanie odpovede na generovanú otázku.

Zisťovanie v metóde *question* sa vykonáva vždy ak je nanovo nahratá stránka alebo sa na pozadí už nahratej stránky zavolá pravidelný AJAX check.

Príloha D: Experiment

1 Experiment

Experiment sa uskutočnil v laboratóriu s 300 Hz eye-trackerom v pondelok 23.3.2015 o 10:00 hodine. Zúčastnili sa ho Robert Moro, Šimon Valíček, Peter Kušnir, Peter Kiš a Vikória Lovasová. Skladal sa zo 4 podúloh. Texty čítali 4 používatelia. Úlohy sa skladali z inštrukcií a z médií. V inštrukciách dostali pokyny a nasledovalo médium. Médium bola webová stránka, ktorú sme vytvorili na tento experiment. Dôvodom bolo, aby sme vedeli ľahko identifikovať jednotlivé časti a aby sa tam nachádzal len text a žiadne menu.

1. Úloha

Zadanie:

Prečítaj si pozorne prvý odsek. Druhý odsek preskoč a tretí si prečítaj nie pozorne (prebehni očami). Stlač medzeru na začatie nahrávania pohľadu.

Médium:

<http://team16.touch4it.sk/test2/first.html>

1 Pivo

Pivo je kvasený alkoholický nápoj horkej chuti vyrábaný v pivovare z obilného sladu, vody a chmeľu pomocou pivovarských kvasiniek (*Saccharomyces cerevisiae* sp.). Pivo sa varí už od nepamätí a je nemožné určiť miesto, kde bolo uvarené prvé pivo. Ako zem pôvodu se uvádza Mezopotámia a to približne už v 7. tisícročí pred n. l. Je však možné, že Sumeri pripravovali len kvas. Pivo je staroslovanské slovo, ktoré označovalo "nápoj najobytnejší a najrozšírejší".

Pivo je tradičným a populárnym nápojom, ktorý má na území bývalého Československa dlhú tradíciu. Zaraďuje sa medzi alkoholické nápoje s relatívne nízkym obsahom alkoholu (30–50 g v jednom litre). Okrem alkoholu obsahuje pivo tiež približne 2 000 ďalších látok. Obsahuje významné množstvo vody, takže ide o výrazne zvodňujúci nápoj, obsahuje tiež sacharidy v podobe tzv. "rýchlych kalórií", bielkoviny, horké látky chmeľu, polyfenolické zlúčeniny, oxid uhličitý, vitamíny a minerálne látky.



Kombinácia týchto zložiek dáva fyziologicky vyrovnaný roztok, ktorý je v rovnováhe s osmotickým tlakom krvi. Významné je zastúpenie minerálov, kde okrem draslíka a sodíka, ktoré sú v priaznivom pomere, môžeme nájsť tiež chloridy, vápnik, fosfor, horčík a kremík. Z vitamínov obsiahnutých v pive sú najvýznamnejšie vitamíny skupiny B – thiamin (3 % dennej spotreby v jednom litre piva), riboflavin (20 %), pyridoxin (31 %), niacin (45 %) a kyselina listová (52 %). Vitamíny skupiny B sú dôležité pre množstvo metabolických procesov (metabolizmus sacharidov, lipidov a aminokyselín), funkcií nervového systému a ďalšie. Jeden liter piva

Obrázok 1: Snímka obrazovky <http://team16.touch4it.sk/test2/first.html>.

2. Úloha

Zadanie:

Pod nadpisom číslo 2 Zloženie piva si prečítaj dvakrát vetu pod nadpisom a dvakrát úplne poslednú vetu.

Médium:

<http://team16.touch4it.sk/test2/second.html>

2 Zloženie piva

Pivo sa skladá zo 4 základných zložiek: vody, chmeľu, sladú a kvasiniek.

Voda

Voda je pri výrobe piva najdôležitejšia zo základných surovín, tvoriaca viac ako 90% nápoja. Pri výrobe sa využíva ako ingrediencia do samotného nápoja ale je potrebná tiež v procesoch, ktoré s výrobou súvisia. Z toho vyplýva, že pre výrobu piva je potrebné veľké množstvo vody. Na 1 liter vyrobeného piva je spotrebované štyrikrát až osemkrát viac vody. Jej použitie priamo ovplyvňuje kvalitu výsledného produktu. Pivovary sú tak budované v miestach, kde sú výdatné zdroje kvalitnej vody.

Chmeľ

Chmeľ je jedna zo základných surovín pre výrobu piva, ktorá mu dodáva charakteristickú horkastú chuť pomocou chmeľových živíc a chmeľovej arómy vplyvom silíc. Pre prípravu piva sa používajú samičie chmeľové hlávky, ktoré sa spracovávajú na tzv. chmeľové produkty. Ide o granulované prípravky z hlávok po usušení, rozomletí a následnej peletizácii.

Slad

Slad sa vyrába zo špeciálne vyšľachtených druhov obilia, najčastejšie jačmeňa či menej často zo pšenice, ktoré majú výrazný podiel na výslednej chuti piva, jeho farbe a aróme. Podľa farby sa rozlišuje svetlý a tmavý slad (plzenský a bavorský). Existujú aj slady špeciálne (karamelový, pražený, diastatický a pod.). Pre výrobu piva sa využíva jačmenné zrnko, tzv. obilka, ktorá sa skladá z obalu, zárodokú klíčka a endospermu. Pri slade sa následne sleduje hlavne klíčivosť a klíčivá energia.

Obrázok 2: Snímka obrazovky <http://team16.touch4it.sk/test2/second.html>

3. Úloha

Zadanie:

Nájdí v texte čo sú humolóny? Prečítaj odpoveď nahlas a stlač F10

Médium:

<http://team16.touch4it.sk/test2/third.html>

Kvasinky

Kvasinky sú mikroorganizmy používané v biotechnologickej výrobe piva - pivovarníctve. Podľa druhu (priebehu) fermentácie sú pre vrchné kvasenie (cca 20 °C; priebehom fermentácie kvasinky na hladine) používané kvasinky *Saccharomyces cerevisiae* ssp. *carlsbergensis* nazvanej na počesť ich objaviteľa Emila Christiana Hansena z dánskeho Carlsbergu, pre spodné kvasenie (5-10 °C; počas fermentácie kvasinky sedimentujú) *Saccharomyces cerevisiae*. Niektoré piva sú kvasené aj spontánnym kvasením, teda za použitia divokých kvasiniek a mikroorganizmov prítomných v mieste výroby bez dodatočného dodania čistej kvasinovej kultúry.

2.1 Chmeľ

Chmeľ je jedna zo základných surovín pre výrobu piva, ktorá mu dodáva charakteristickú horkastú chuť pomocou chmeľových živíc a chmeľovej arómy vplyvom silíc. Pre prípravu piva sa používajú samičie chmeľové hlávky, ktoré sa spracovávajú na tzv. chmeľové produkty. Jedná sa o granulované prípravky z hlávok po usušení, rozomletí a následnej peletizácii.

Pre výrobu slovenských a českých pív sa používa česká odroda chmeľu tzv. zatecký poloranný červeňák, ktorý býva radený medzi najkvalitnejšie na svete. Odroda patrí medzi staršie, ktoré síce nemajú taký výnos úrody ako novodobé sorty, plody sú však neporovnateľne kvalitnejšie. Radí sa do skupiny jemných aromatických chmeľov. Chmeľ je na území Česka pestovaný už od 8. storočia, na Slovensku od 11. storočia.

Chmeľ otáčavý

(*Humulus lupulus*) je kvitnúca rastlina z rodiny Cannabaceae. Vyskytuje sa najmä v Európe, západnej Ázii a severnej Amerike. Chmeľ predstavuje jednu z hlavných zložiek mnohých pív, a ako taký je široko kultivovaný pre použitie v pivovarníctve. Šišťičky chmeľu sú chemicky zložené, pretože obsahujú napr. alfa kyseliny známe ako humolóny. Jeho voňavé kvety totiž dodávajú horkosť a arómu, ktoré v pive vyvažujú sladkú chuť sladú a tiež podporujú konzervačné vlastnosti nápoja. H. lupulus obsahuje myrcen, humulén, xanthohumol, myrcenol, linalool, triesloviny a živice.

Obrázok 3: Snímka obrazovky <http://team16.touch4it.sk/test2/third.html>.

4. Úloha

Zadanie:

Nájdí v texte kde sa u nás vyskutojuje chmeľ. Prečítaj nahlas odpoveď a stlač F10

Médium:

<http://team16.touch4it.sk/test2/fourth.html>

Charakteristika rastliny

Chmeľ je trvác ovijavá bylina s pravotočivou, niekoľko metrov dlhou stonkou. Pomocou háčikatých chlupov, ktoré ju zdrsňujú, sa pridáva opory. Listy sú protistočné troj - až päťlaločné s pilkovitými okrajmi. Je dvojdómá, niektoré jedince majú iba samčie, iné iba samičie kvety. Samičie kvety sedia po dva v pazuchách žltkastých listenôv v krátkych klasoch a počas dozrievania sa menia na prevísnuté vajcovité šišťice. Samičie kvety v metlinách majú iba zelenkavé okvetie a päť tyčínok. Plod je nažka v zachovanom okvetí šišťice. Na okvetí aj nažkách je množstvo živcových žliazok obsahujúcich horkastú voňavú látku lupulín.



Vyskyt a pestovanie

U nás sa vo voľnej prírode vyskytuje často veľmi hojne vo vlhkých krovinách, húštinách a ješajinách. V pivovarníctve a na liečebné účely sa však využíva iba chmeľ pestovaný v chmeľniciach, ktoré sa zakladajú v teplejších oblastiach. Pestujú sa iba samičie rastliny, pretože iba tie sú cenné na ďalšie použitie.

Zber a úprava

Zberajú sa šišťice a chmeľové žliazky. Šišťice sa triajú na začiatku dozrievania a sušia na prevzdušnenom mieste alebo v umelých podmienkach pri teplote do 50°C. Chmeľové žliazky sa získavajú z úplne dozretých šišťíc, ktoré sa po usušení preosejú cez sito. Pri tomto postupe sa žliazky prepadnú cez otvory v site v podobe

Obrázok 4: Snímka obrazovky <http://team16.touch4it.sk/test2/fourth.html>.

2 Trvanie fixácií

1. Úloha

Tabuľka 1: Prvý odsek – čítanie.

Počet	273
Priemer [s]	0.30
Spolu [s]	83.25

Tabuľka 2: Tretí odsek – nepozorné čítanie.

Počet	218
Priemer [s]	0.25
Spolu [s]	35.25

V tejto úlohe mali používatelia prvý odsek prečítať pozorne a tretí odsek nepozorne. Vyplýva to aj zo štatistík, kde v prvom odseku trvali fixácie v priemere 0,30 sekundy a spolu trvali až 83,75 sekúnd. Pričom v tretom odseku trvala v priemere fixácia menej 0,25 sekundy a spolu len 35,25 sekúnd. Z celkovej dĺžky fixácií sa dá vidieť, že prvý odsek čítali oveľa pozornejšie.

2. Úloha

Tabuľka 3: Opakovanie – prvá veta.

Počet	84
Priemer [s]	0.33

Spolu [s]	27.33
-----------	-------

V druhej úlohe mali takisto pozorne čítať a to vyplýva aj z fixácií, kde bol priemerný čas fixácie 0,33 sekundy.

3. Úloha

Tabuľka 4: Hľadanie – druhý odsek.

Počet	105
Priemer [s]	0.21
Spolu [s]	21.53

Tabuľka 5: Štvrtý odsek – tu sa nachádzalo hľadané slovo.

Počet	250
Priemer [s]	0.27
Spolu [s]	67.16

V tretej úlohe mali hľadať zadané slovo, priemerná dĺžka fixácie v druhom odseku bola len 0,21 sekundy, čo naznačuje, že nečítali pozorne. V štvrtom odseku, kde sa už slovo nachádzalo a používatelia to už z kontextu zistili boli fixácie o málo dlhšie. Ich počet bol veľký v oboch prípadoch pretože prechádzali textom aj niekoľkokrát.

4. Úloha

Tabuľka 6: Hľadanie - druhý odsek – tu sa nachádzalo hľadané slovo.

Počet	142
Priemer [s]	0.25
Spolu [s]	34.97

Pri hľadaní v štvrtej úlohe bola dĺžka priemernej fixácie opäť pod 0,30 sekundy.

3 Záver

Z výsledkov sme zistili, že ak priemerná dĺžka fixácie presiahne 0,30 sekundy, tak môžeme prehlásiť, že používateľ čítal text pozorne. Tieto dáta následne využijeme na analyzovanie vzorov čítania.