

Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií
Ilkovičova 2, 842 16 Bratislava 4

NAVIGÁCIA V BUDOVE

BeaCode

Dokumentácia k dielu

Vedúca tímu: Mgr. Alena Martonová, PhD.
Členovia tímu: Bc. Augustín Peter, Bc. Veronika Balážová, Bc. Marek Bruchatý, Bc. Juraj Flamík, Bc. Ondrej Kipila, Bc. Sandra Kostova, Bc. Andrej Žlnka
Predmet: Tímový projekt
Školský rok: 2016 / 2017

Obsah

1	Úvod	1
2	Globálne ciele	3
2.1	Zimný semester	3
2.2	Letný semester	3
3	Celkový pohľad	4
4	Architektúra	11
5	Frontend.....	14
5.1	Android	14
5.2	iOS	14
5.3	Admin Web.....	15
6	Backend.....	17
7	Dátový model.....	18
8	Tímový server.....	20
9	Moduly systému	22
9.1	Diagram tried - Android.....	22
9.2	Diagram tried – iOS	24
9.3	Diagram tried – Admin Web.....	25
9.4	Zobrazenie zoznamu eventov	25
9.4.1	Android	25
9.4.2	iOS	26
9.4.3	Admin Web.....	26
9.4.4	Backend – API na odstránenie eventov, na odstránenie sekcie a na odstránenie exhibitu 27	
9.5	Vyhľadanie eventu.....	27
9.5.1	Android	27
9.5.2	iOS	28
9.6	Zobrazenie detailu eventu	28
9.6.1	Android	28
9.6.2	iOS	29
9.7	Zobrazenie detailu exponátu.....	29
9.7.1	Android	29

9.7.2	iOS	29
9.7.3	Admin Web – pridávanie a zobrazovanie obrázkov a mapy k exponátom a eventov	30
9.7.4	Backend – API na prácu s eventom a API na prácu s exhibitmi	30
9.8	Profil	31
9.8.1	Android	31
9.8.2	iOS	31
9.9	Výber exponátov	32
9.9.1	Android	32
9.9.2	iOS	33
9.10	Lokalizácia	33
9.10.1	Android	33
9.10.2	iOS	33
9.10.3	Admin Web – Parsovanie beaconov z mapy	34
9.10.4	Backend – API na vracanie mapy a polohy beaconov	34
9.11	Notifikovanie	35
9.11.1	Android	35
9.11.2	iOS	35
9.11.3	Admin Web	36
9.11.4	Backend	36
9.12	Poslanie spätnej väzby	37
9.12.1	Android	37
9.12.2	iOS	37
Príloha A: Testovanie		39
	Android	39
	Admin Web	49
	iOS	55
Príloha B: Inštalačná príručka		61
	Android	61
	iOS	61
	Admin Web	61
	Server	61

1 Úvod

Navigačné zariadenia sú veľmi populárnymi a spoľahlivými pomocníkmi, ako sa dostať do požadovaného cieľa. GPS je najrozšírenejšia technológia, ktorá nám pomáha zorientovať sa v priestore. V rámci budov je však GPS signál dosť slabý (často aj žiadny signál nie je) a preto sa potom táto technológia v zatvorených priestoroch nedá používať.

Okrem tohto základného problému sa návštevníci podujatí stretávajú s rôznymi ďalšími problémami, ktoré sú často spojené s informáciami, ich dostupnosťou, objaviteľnosťou, kvalitou a množstvom. Informácie o podujatiach a exponátoch dostupné pre návštevníka sú často rozsahovo limitované. Spôsoby a médiá, prostredníctvom ktorých sú distribuované (napr. tlačové správy, audio/vizuálne záznamy) sú obmedzené fyzickými rozmermi alebo majú rozsah prispôbený tak, aby naplnili iba základné informačné potreby návštevníkov za čo najkratší čas. Ďalším problémom je dostupnosť a objaviteľnosť informácií. Podujatí sa zúčastňuje veľké množstvo návštevníkov, čo predstavuje zhoršené podmienky pre nájdenie informačných letákov, tabuliek a iných informačných médií, ktoré poskytujú informácie o konkrétnych exponátoch. Často je to spôsobené najmä ich zlým umiestnením v priestore, prípadne tým, že sú distribuované nevhodnou formou. Iným problémom je zahltenie informáciami, ktoré nie sú pre návštevníka zaujímavé. Preferencie návštevníkov na obsah sa môžu výrazne líšiť, čo znamená, že návštevníkovi môžu byť v niektorých prípadoch prezentované informácie, o ktoré nemá záujem.

Zároveň, v súčasnosti vystavovatelia nemajú efektívne nástroje na zbieranie informácií od návštevníkov, ako aj o samotnom podujatí. Spätná väzba od návštevníkov je väčšinou zbieraná prostredníctvom papierových formulárov, alebo elektronických formulárov rozosielaných dodatočne prostredníctvom mailov. Pre návštevníkov sú takéto spôsoby nepraktické, čo sa často premietne na nízkej účasti pri vyjadrení spätnej väzby. Zber informácií o samotnom podujatí ako celkový počet návštevníkov na exhibíciách alebo návštevnosť jednotlivých exponátov je komplikovaný, pretože neexistuje jednoduchý spôsob automatického zberu týchto dát.

Prehľadná a ľahko dostupná navigácia vo verejných priestranstvách je jedným z faktorov používateľského komfortu, o ktorý by sa mali majitelia výstav snažiť. Riešenie problému s navigáciou a ostatných spomínaných problémov poskytuje naša aplikácia BeaCode.

Aplikácia slúži na indoor navigáciu pomocou technológie Bluetooth LE beacon-ov. Zobrazuje polohu návštevníka na mape podujatia, polohu exponátov, ktoré si zvolil a zároveň aj návštevníka prevedie po vyznačených exponátoch s využitím navigácie. Počas pohybu návštevníka po podujatí aplikácia poskytuje informácie o jednotlivých exponátoch práve v čase, keď sa návštevník pri nich nachádza. Tieto informácie môžu byť ľubovoľného formátu od textu, zvukového záznamu až po video záznam. Organizátor podujatia má možnosť úpravy tohto obsahu podujatia. Aplikácia tiež poskytne používateľom prehľad o výstavách, ktoré sa uskutočnia v najbližšom čase.

Výhoda nášho návrhu je v tom že, riešenie aplikácie sa dá jednoducho preniesť na akúkoľvek inú budovu, v ktorej sa výstavy uskutočnia, či už múzeá, školy, nákupné centrum alebo iné budovy. Aplikácia bude dostupná na oboch mobilných platformách: Android a iOS.

2 Globálne ciele

2.1 Zimný semester

Hlavným cieľom v rámci zimného semestra bolo vytvorenie základnej kostry aplikácie pre Android a iOS a rovnako aj základnej kostry webovej časti aplikácie – Admin Web.

Nakoľko v rámci práce pracujeme pre väčšinu z nás s novou technológiou (beacony), tak ďalšou dôležitou časťou v zimnom semestri bolo naštudovanie danej technológie. Aplikácia je dostupná pre obe mobilné platformy: Android a iOS, takže bolo potrebné preštudovanie najvhodnejších knižníc, ktoré sa budú dať používať, t.j. ktoré budú umožňovať zachytiť signály z majáčikov.

Cieľom pre zimný semester bolo vytvoriť prototyp, v ktorom fungujú základné obrazovky v aplikácii a získavanie dát pre jednotlivé obrazovky je prepojené s backendom.

Úspešné dokončenie predchádzajúcich cieľov prináša aj to, že každý člen získal a následne si zlepšil organizačné a pracovné schopnosti. Práca v tíme nás naučila aj lepšej tímovej komunikácii a spolupráci.

2.2 Letný semester

Hlavným cieľom pre letný semester bolo dokončenie aplikácie do takej fázy, aby mohla byť prezentovateľná na IIT.SRC a aby v nej boli dokončené všetky funkcionality v súvislosti s používateľským rozhraním a navigáciou.

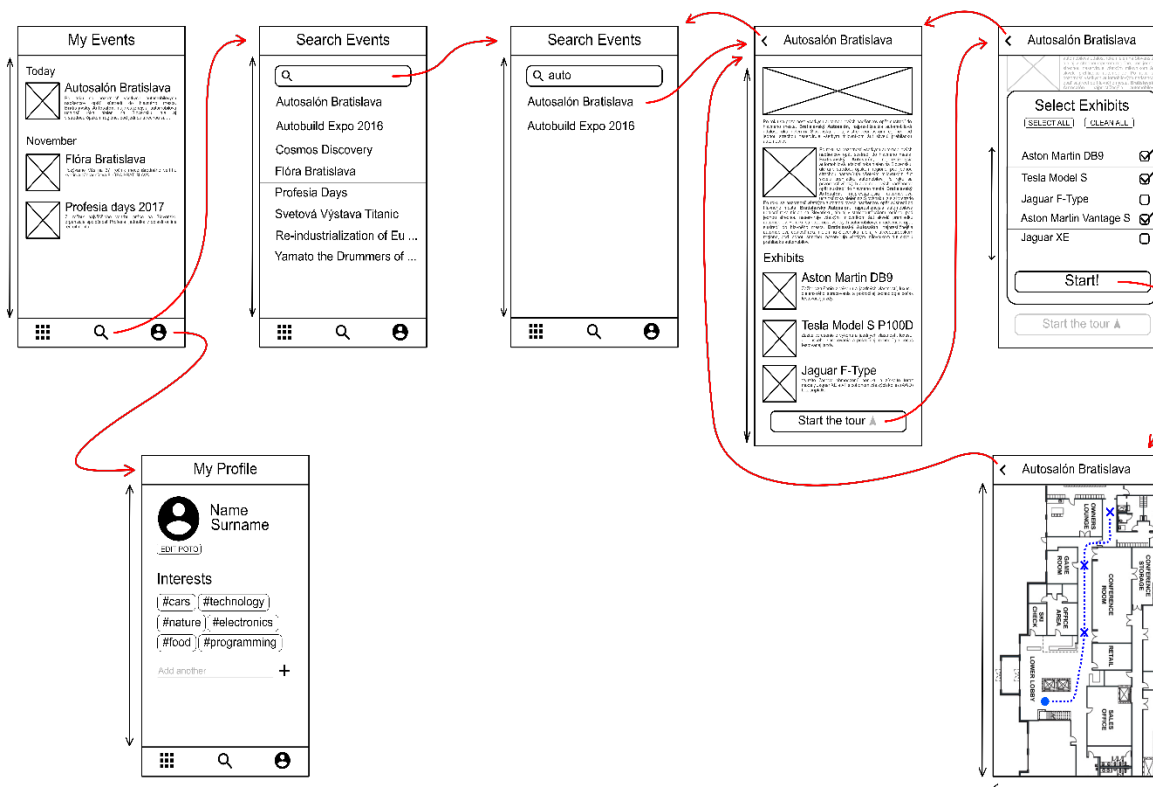
Na začiatku letného semestra bolo našou prioritou upraviť používateľské rozhranie, aby bolo user-friendly. Okrem toho bolo potrebné aj to, aby sme zjednotili dizajn pre Android a iOS.

Následne najväčšia práca v letnom semestri sa týkala lokalizovania exponátov a používateľa (návštevníka výstavy) pomocou beaconov na mape. Okrem toho sme vytvorili aj notifikácie s podrobnými informáciami o jednotlivých exponátoch, ktoré sa používateľovi zobrazia vtedy, ak má o daný exponát záujem a priblíži sa k nemu.

Okrem toho sa pracovalo aj na webovom rozhraní pre vystavovateľa, ktorý si pomocou tohto Admin Webu môže prispôsobiť obsah danej výstavy.

3 Celkový pohľad

Na úspešné vytvorenie našej aplikácie bolo potrebné najprv navrhnuť low-fidelity prototyp, na základe ktorého sa potom vyvinie aj high-fidelity prototyp aplikácie. Na nasledujúcom obrázku je rozpracovaný low-fidelity návrh, ktorý obsahuje základne obrazovky, ktoré umožňujú splnenie funkcionality.



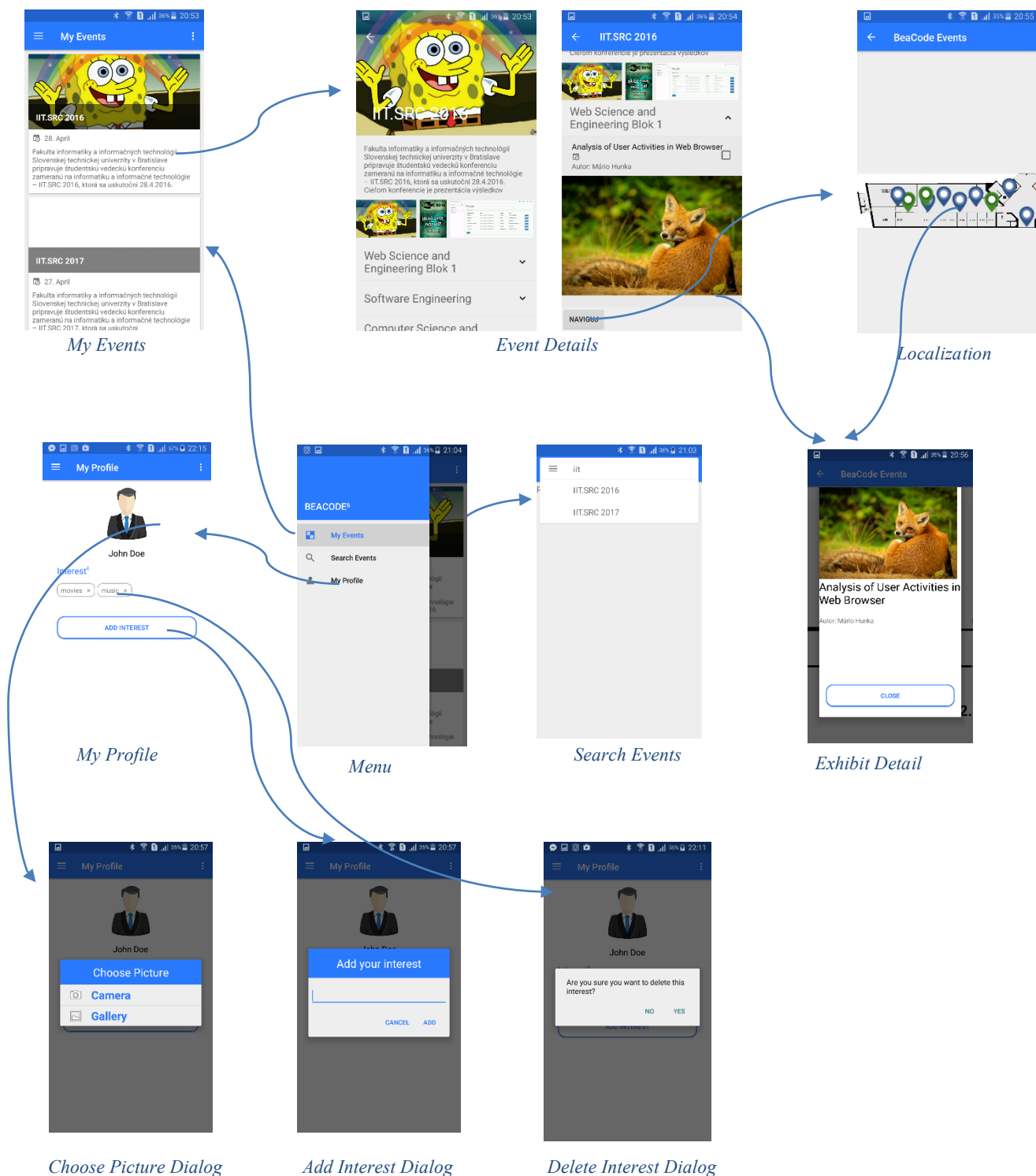
Obr. 3.1 Low – fidelity prototype.

Z obrázka vidno, že základne položky menu sú Search events, My Profile a My Events. Po vybraní položky My Profile z menu sa zobrazí obrazovka, na ktorej sú údaje o používateľovi ako sú jeho meno a záujmy. Po kliknutí na My Events sa zobrazí obrazovka, na ktorej je prehľad o udalostiach, ktoré sa uskutočnia v blízkej dobe. Po kliknutí na tlačidlo Search Events v menu je dostupné vyhľadanie udalostí na základe zadania ich názvu. Po vybraní udalosti (napr. Autosálón Bratislava) sa objaví obrazovka, na ktorej sú informácie o tejto udalosti ako aj exponáty, ktoré sa dajú na tejto udalosti vidieť. Po kliknutí Start the tour z tejto obrazovky sa objaví nová, na ktorej sú dostupné všetky exponáty danej udalosti. Používateľ si tu môže vybrať tie, ktoré chce navštíviť. Potom môže stlačiť tlačidlo Navigation a vtedy navigácia na danej udalosti môže začať. Ak si nevyberie žiadne exponáty, tak automaticky bude prevedený cez všetky.

Aplikácia pre Android a iOS sa v dizajne mierne líšia. Na nasledujúcich obrázkoch sú zobrazené jednotlivé obrazovky pre iOS:



Na nasledujúcich obrázkoch sú znázornené jednotlivé obrazovky pre Android:



Z týchto odraziek je vidno že verzia na Androide sa zhoduje s prvotným low-fidelity prototypom, ktorý bol navrhnutý a ktorý je vyššie uvedený. Takže, základne položky menu sú My Events, Search events a My Profile. Po vybraní položky My Profile z menu sa zobrazí obrazovka, na ktorej sú údaje o používateľovi ako sú jeho fotku, meno a záujmy. Používateľ si fotku môže zmeniť po kliknutí na fotku s tým že sa mu zobrazí dialógové okna ohľadom tomu

či si chce fotku nahrať z galérie alebo si ju chce odfoťiť. Interest si používateľ môže pridať ak klikne na tlačidlo Add Interest a následne vyplní meno pre interestu ktorý chce pridať. Po kliknutí na My Events sa zobrazí obrazovka, na ktorej je prehľad o udalostiach, ktoré sa zo servera načítajú. Z obrázky je vidno že pre každú udalosť je poskytnuté jeho meno, obrazovka, dátum uskutočnenia a krátky popis. Po vybraní udalosti (napr. IIT. SRC 2016) sa objaví obrazovka, na ktorej sú informácie o tejto udalosti, kategórie udalosti, pričom každá kategória sa dá rozbaľiť s tým že po rozbaľení sa zobrazia všetky exponáty, ktoré sa dajú na tejto udalosti vidieť. Pri každom exponáte je checkbox ktorý používateľ v podstate označí v prípade keď má záujem o ním a má záujem sa navigovať do neho. Po kliknutí Naviguj sa používateľovi zobrazí mapa na ktorej modrou farbou sú označené všetky exponáty a zelenou sú označené tie ktoré používateľ označil. Po kliknutí na nejaký bod, prípadne po kliknutí na nejaký exponát z obrazovky MyEvents, sa používateľovi zobrazí okno s informáciami o danom exponáte. Po kliknutí na tlačidlo Search Events v menu je dostupné vyhľadanie udalostí na základe zadania ich názvu.

Čo sa týka aplikácie Admin Web, tá vyzerá nasledovne:

The screenshot shows the 'BeaCode Admin Tool' interface. On the left is a navigation menu with options: 'Podujatia', 'Používatelia', and 'Beacony'. The main content area is titled 'Podujatia' and contains a table labeled 'Zoznam podujatí'. The table has four columns: 'Názov podujatia', 'Dátum', 'Lokalita', and 'Opis'. Two rows are visible, representing the IIT.SRC 2016 and IIT.SRC 2017 events. Each row includes a 'Detail' button and a small 'x' icon. Below the table is a 'Pridať' button.

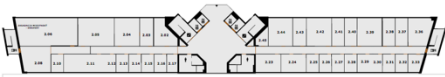
Názov podujatia	Dátum	Lokalita	Opis
IIT.SRC 2016	28.04.2016 00:00 - 28.04.2016 00:00	Bratislava	Fakulta informatiky a informačných technológií Slovenskej technickej univerzity v Bratislave pripravuje študentskú vedeckú konferenciu zameranú na informatiku a informačné technológie – IIT.SRC 2016, ktorá sa uskutoční 28.4.2016. Cieľom konferencie je prezentácia výsledkov výskumu študentov informatiky a informačných technológií vo všetkých troch stupňoch štúdia. Najlepšie príspevky dostanú ponuku na publikovanie vo vedeckom časopise Information Sciences and Technologies
IIT.SRC 2017	27.04.2017 00:00 - 27.04.2017 00:00	Bratislava	Fakulta informatiky a informačných technológií Slovenskej technickej univerzity v Bratislave pripravuje študentskú vedeckú konferenciu zameranú na informatiku a informačné technológie – IIT.SRC 2017, ktorá sa uskutoční 27. apríla 2017 Cieľom konferencie je prezentácia výsledkov výskumu študentov informatiky a informačných technológií vo všetkých troch stupňoch štúdia. Najlepšie príspevky dostanú ponuku na publikovanie vo vedeckom časopise Information Sciences and Technologies.

Search...

- Podujatia
- Používatelia
- Beacony

Sekcie podujatia

Mapa:
 Nie je vybratý žiadny súbor



Názov sekcie	
Web Science and Engineering Blok 1	<input type="button" value="Detail"/> x
Software Engineering	<input type="button" value="Detail"/> x
Computer Science and Artificial Intelligence	<input type="button" value="Detail"/> x
Computer Graphics, Multimedia and Computer Vision	<input type="button" value="Detail"/> x
Computer Networks, Computer Systems and Security	<input type="button" value="Detail"/> x
Intelligent Information Processing	<input type="button" value="Detail"/> x
TP Cup Competition Projects	<input type="button" value="Detail"/> x

Pridať obrázok k podujatiu:
 Nie je vybratý žiadny súbor



Search...

- Podujatia
- Používatelia
- Beacony

Web Science and Engineering Blok 1


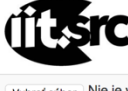

Názov:

Začiatok:

Koniec:

Miesto konania:

Opis:

Exponáty						
	Id	Názov exponátu	Beacon	Beacon Id	Poster	Notifikácia
✕	6	Analysis of User Activities in Web Browser	<input type="text"/>	Minor: 76	Vybrať súbor Nie je vybratý žiadny súbor <input type="button" value="Ulož poster"/> 	Cau
✕	4	Frequent Item Mining Comparison on Data Streams	<input type="text"/>	Minor: 74	Vybrať súbor Nie je vybratý žiadny súbor <input type="button" value="Ulož poster"/> 	
✕	8	Hybrid Personalized Explanation of Recommendations	<input type="text"/>	Minor: 78	Vybrať súbor Nie je vybratý žiadny súbor <input type="button" value="Ulož poster"/> 	

BeaCode Admin Tool

Search...

Podujatia

Používatelia

➕ Pridať používateľa

➖ Odstrániť používateľa

Beacony

Nové podujatie

Názov:

Začiatok:

Koniec:

Miesto konania:

Opis:

BeaCode Admin Tool

Search...

Podujatia

Vytvoriť podujatie

Odstrániť podujatia

Používatelia

➕ Pridať používateľa

➖ Odstrániť používateľa

Beacony

Nový exponát

Názov:

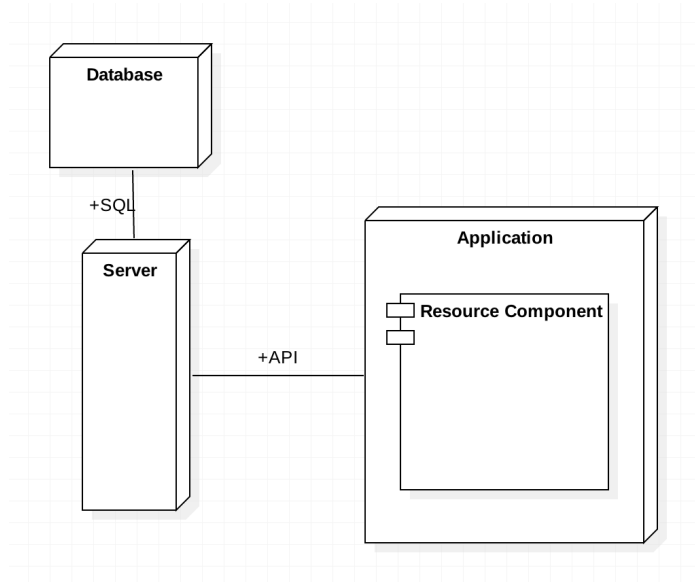
Opis:

Na prvej obrazovke „Podujatia“ sa nachádza zoznam existujúcich podujatí s možnosťou pozretia detailu a vytvorenia nového podujatia. Na nasledujúcej obrazovke je možné si pozrieť detail podujatia a je tu možnosť vytvoriť sekcie, pridať mapu a obrázky k podujatiu. Na ďalšom

obrázku vidíme obrazovku detail podujatia kde môžeme vidieť všetky exponáty, ich obrázky a môžeme k nim pridať obrázok a beacon. Na obrazovke „Nové podujatie“ je možné vytváranie nového podujatia. Na obrazovke „Nový exponát“ je možné vytváranie nových exponátov prislúchajúcich konkrétnemu podujatiu.

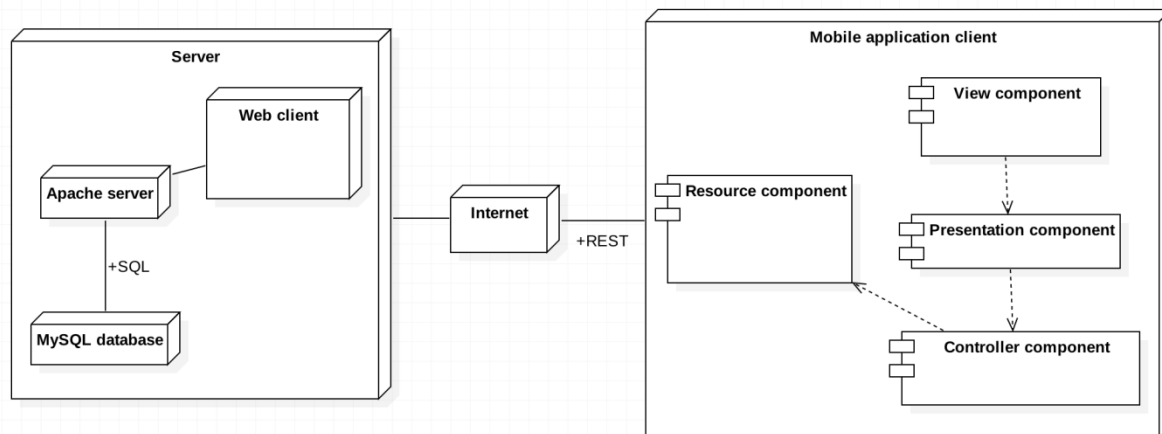
4 Architektúra

Na obrázku Obr. 4.1 je diagram nasadenia vyšších vrstiev. Diagram má za úlohu zobraziť komunikačný kanál medzi dôležitými komponentami architektúry. V prvom rade popisuje komunikáciu medzi databázou a server prostredníctvom SQL dopytov. Z druhej strany popisuje volania prostredníctvom API a na úrovni client-server.



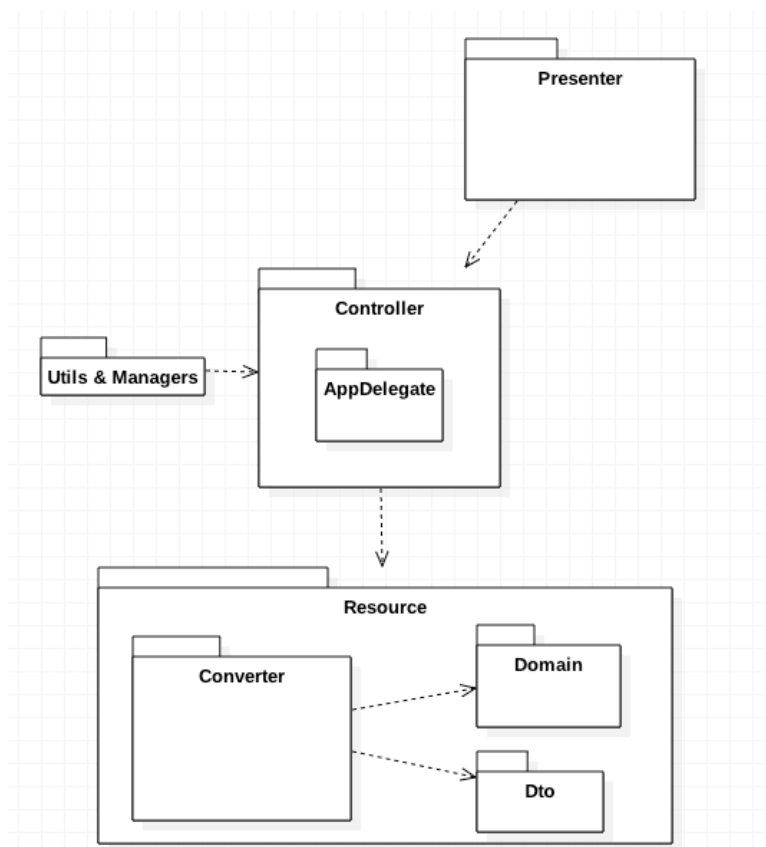
Obr. 4.1 Komunikačná vlna komponentov.

Obrázok Obr. 4.2 popisuje pomocou deployment diagramu kompletnú architektúru systému. Z predošlého diagramu je jasné, že sa jedná o client server komunikáciu. Prostredníctvom API sú spracovávané požiadavky z klientskej aplikácie (v našom prípade mobilnej). Klientská aplikácia pomocou Resource komponentu dokáže volať serverové požiadavky, ktoré sa spracujú a vracajú sa prostredníctvom formátu JSON, ktorý je v aplikácii konvertovaný pomocou controllerov na modelové objekty a následne vykreslené vďaka prezentačnému a zobrazovaciemu komponentu.



Obr. 4.2 Kompletná architektúra platformy.

Na ďalšom obrázku je diagram balíčkov. Z tohto diagramu je dôležité predovšetkým poukázať na balíček Resource. V rámci tohto balíčku sa tu nachádzajú ďalšie balíčky triedy, ktoré úzko spolupracujú so spomenutou serverovou komunikáciou. V prípade požiadavky sa zo servera vracia JSON, ktorý sa spracuje pomocou danej Dto (data transfer object) triedy. Následne je možné pomocou Converter triedy spracovať tento objekt a zkonvertovať do Domain (doménového) objektu, s ktorým je možné následne funkcionálne pracovať.



Obr. 4.3 Diagram balíčkov.

Aktuálne API, ktoré poskytuje server sú nasledovné:

API documentation body format: [JSON](#) request format: [json](#)

[Admin Web](#)
[App](#)

Admin Web

[Show/hide](#) [List Operations](#) [Expand Operations](#)

GET OPTI	/api/admin-web/beacons	Show beacons which are not linked to any exhibit.
PATCH OR	/api/admin-web/beacons/{beaconId}	Change beacon.
GET OPTI	/api/admin-web/events	Show events which were created by logged in user. Sorted by name ASC.
POST OPTI	/api/admin-web/events/new	Save new event for logged in user.
DELETE O	/api/admin-web/events/{eventId}	Delete given event.
GET OPTI	/api/admin-web/events/{eventId}	Show event which belongs to logged in user.
GET OPTI	/api/admin-web/events/{eventId}/exhibits	Show all exhibits which belongs to given event. Sorted by name ASC.
POST OPTI	/api/admin-web/events/{eventId}/exhibits/new	Save new exhibit for given event.
DELETE O	/api/admin-web/events/{eventId}/exhibits/{exhibitId}	Delete given exhibit.
PATCH OR	/api/admin-web/events/{eventId}/exhibits/{exhibitId}	Change given exhibit.
POST OPTI	/api/admin-web/events/{eventId}/exhibits/{exhibitId}/images/new	Upload new image for exhibit.
POST OPTI	/api/admin-web/events/{eventId}/images/new	Upload new image for event.
PUT OPTI	/api/admin-web/events/{eventId}/parse-beacon-svg	Parse and save beacons from map for given event.
POST OPTI	/api/admin-web/locations	Show all existing locations which start with given text. Sorted by name ASC.

App

[Show/hide](#) [List Operations](#) [Expand Operations](#)

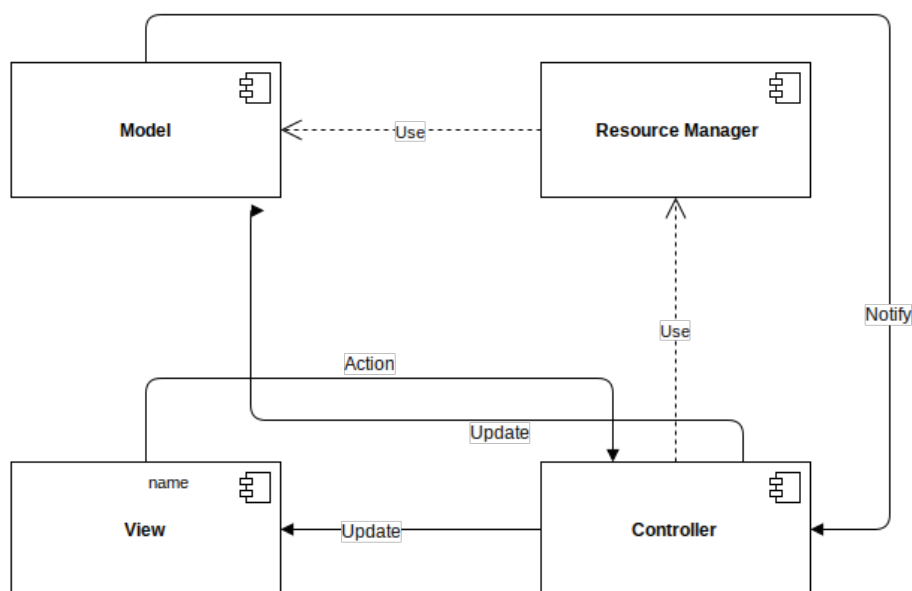
POST	/api/app/events	Show all existing events which start with given text. Sorted by name ASC.
GET	/api/app/events/{eventId}/exhibits	Show all exhibits which belongs to given event. Sorted by name ASC.
GET	/api/app/events/{eventId}/selected-exhibits-for-tour	Show selected exhibits for tour which belongs to logged in user. Sorted by systemCreated DESC.
POST	/api/app/events/{eventId}/selected-exhibits-for-tour/new	Save new selected exhibit for tour for logged in user.
DELETE	/api/app/events/{eventId}/selected-exhibits-for-tour/{selectedExhibitForTourId}	Delete selected exhibit for tour which belongs to logged in user.
GET	/api/app/interests	Show all interests which belongs to logged in user. Sorted by systemCreated DESC.
POST	/api/app/interests/new	Save new interest for logged in user.
DELETE	/api/app/interests/{interestId}	Delete interest which belongs to logged in user.
GET	/api/app/logged-in-user	Show logged in user.
POST	/api/app/logged-in-user/images/new	Upload new profile image for logged in user.
GET	/api/app/starred-events	Show events which were starred by logged in user. Sorted by systemCreated DESC.

Documentation auto-generated on Thu, 11 May 17 23:48:04 +0200

5 Frontend

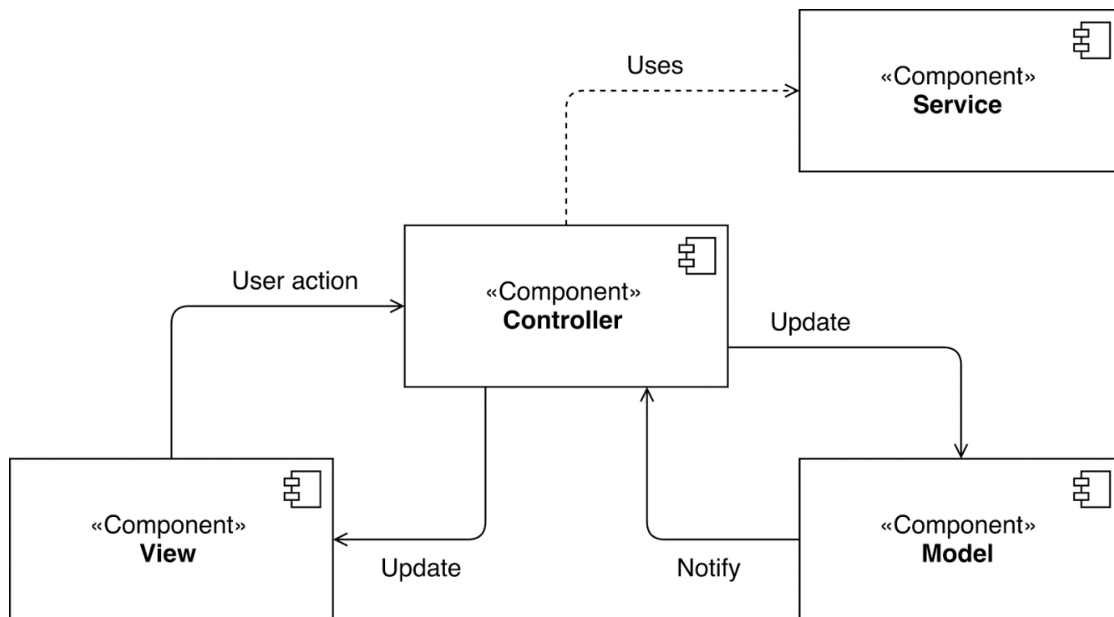
5.1 Android

Architektúra aplikácie je založená na modeli Model-View-Controller. Controller je reprezentovaný triedami Activity. Tieto aktivity môžu byť zároveň aj View (v prípade, že daná aktivita obsahuje len jednu obrazovku), alebo v prípade, že aktivita pozostáva z viacerých obrazoviek, tak je táto aktivita rozdelená do viacerých Views, ktoré predstavujú jednotlivé fragmenty. Ďalšou časťou aplikácie je Resource Manager, ktorý sa stará o komunikáciu so službami, ktoré poskytuje server prostredníctvom REST API. Zo servera sa pomocou resource managera načítajú dáta, ktoré sa lokálne ukladajú do entít, ktoré obsahuje balík Model. Ďalej sa s týmito údajmi pracuje lokálne a v prípade, že je potrebné ich zmeniť, sú pomocou resource managera znova odoslané na server.



5.2 iOS

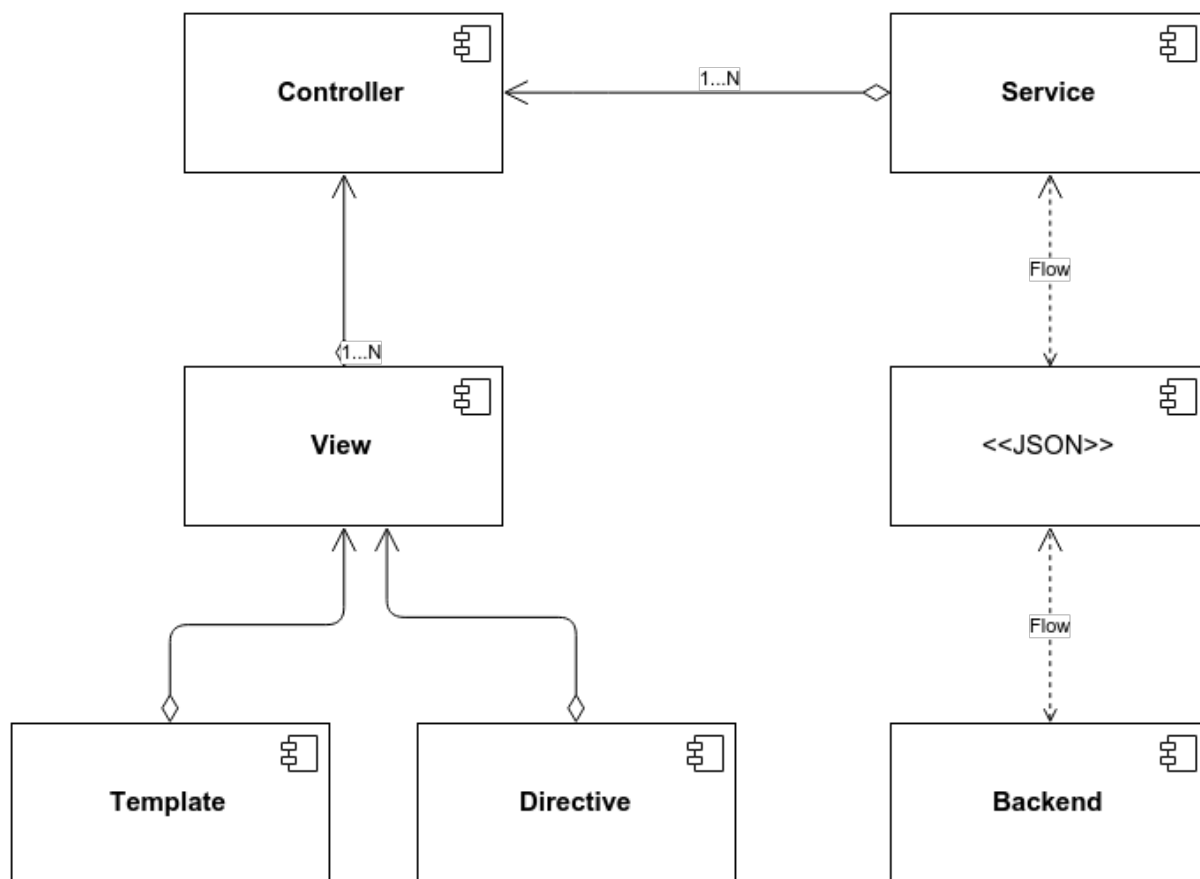
Diagram komponentov popisuje architektúru založenú na architektonickom štýle Model View Controller (Obr. 5.1). Ku každej obrazovke prislúcha jeden controller, ktorý aktualizuje view a zabezpečuje interakciu používateľa s aplikáciou. Controller na základe vstupov od view aktualizuje stav reprezentovaný modelom. Model na základe zmeny dát dokáže notifikovať controller, aby vykonal určité operácie nad view alebo v modeli samotnom. Controllery používajú služby, ktoré zabezpečujú komunikáciu cez REST rozhranie a sprístupňujú rozhranie pre komunikáciu s knižnicami zabezpečujúcimi lokalizačné služby.



Obr. 5.1 Diagram komponentov popisujúci architektúru systému.

5.3 Admin Web

Na obrázku (Obr. 5.2) je diagram komponentov opisujúci architektúru frontend Admin Webu. Frontend Admin webu je vyvíjaný technológiou AngularJS. Každé View v sebe môže agregovať template alebo direktívy. Každá stránka sa teda skladá z HTML kódu, ktorý vytvára šablóny. V samotnom HTML kóde sa nachádzajú direktívy, či už Angular direktívy, alebo nami vytvorené direktívy. Controller v sebe následne môže agregovať jeden, alebo viacero view. Komunikácia s backendom je zabezpečená pomocou REST angular služieb. Tieto služby sú agregované do controllers a dáta z backendu sa prenášajú vo formáte JSON.



Obr. 5.2 Diagram komponentov popisujúci architektúru frontendu Admin Web.

6 Backend

Backend je písaný v jazyku PHP pomocou frameworku Symfony. Štruktúra kódu je rozdelená do bundlov, pričom každý bundle je rozdelený do 4 hlavných typov tried:

- Controller
 - funkcionálnosť prijímania požiadaviek a vracania odpovedí
- Repository
 - funkcionálnosť práce s entitami, ktorá sa ich priamo týka
- Entity
 - opísané dátové entity, teda čo a ako je uložené v databáze
- Classes
 - iná funkcionálnosť v rámci projektu

Spojenie backendu s databázou je spravené cez Doctrine mapovač. To znamená, že kód nie je priamo viazaný na konkrétnu databázu.

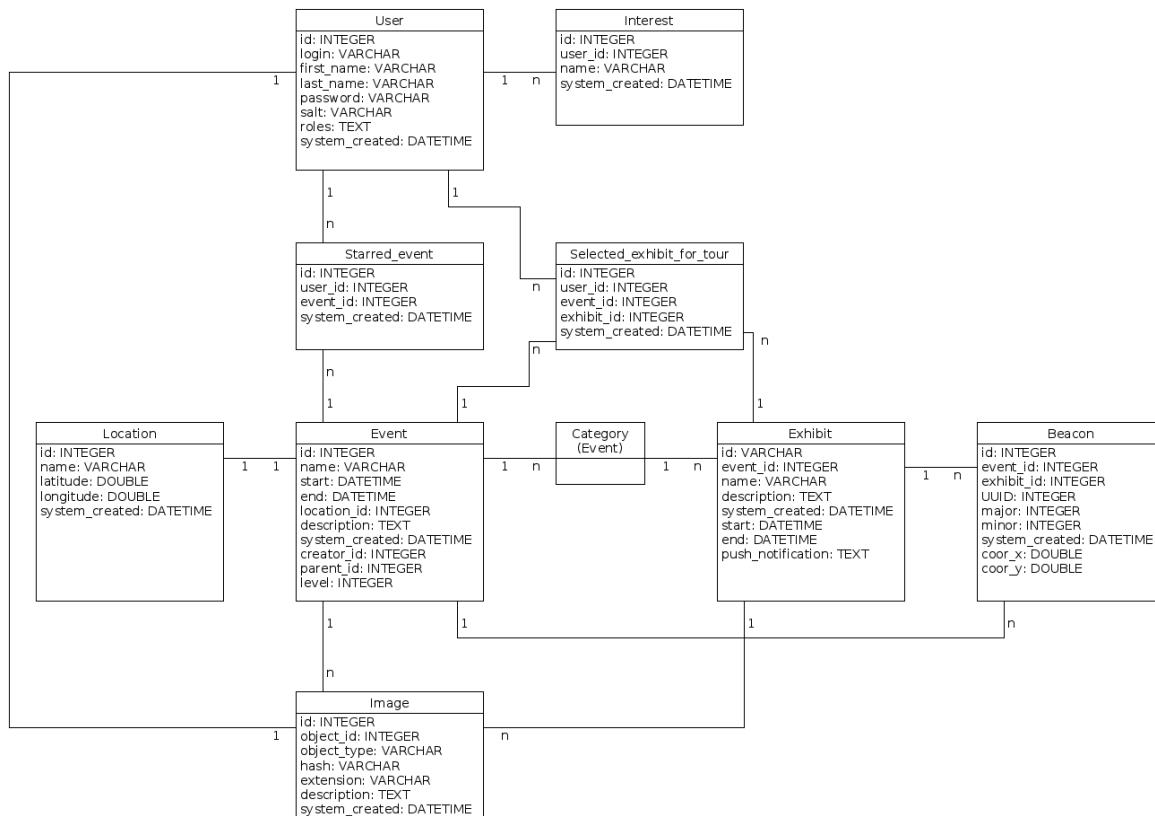
Backend je tvorený na štýl REST API. To znamená, že sú vystavené linky a metódy, ku ktorým môže vonkajší svet pristupovať a používať ich. Toto je zdokumentované v interaktívnej dokumentácii, ktorá je prístupná cez webový prehliadač, takže každý si môže pozrieť ako má požiadavka vyzeráť a čo má obsahovať. Dokonca si môže priamo v dokumentácii pomocou Sandboxu vyskúšať požiadavku vytvoriť a vidieť odpoveď zo servera.

Na deploy backendu na server používame nástroj Magallanes. Rozlišujeme 2 enviromenty: development a production. Nástroj dostane aktuálny kód na server, pričom ho tam uloží ako current release. Potom spustí potrebné nadefinované príkazy, ako update štruktúry databázy alebo upravenie práv súborov.

7 Dátový model

Dátový model pozostáva z týchto entít (Obr. 7.1):

- User
 - používateľ a informácie o ňom
- Interest
 - záujem používateľa, pričom používateľ môže mať viac záujmov
- Starred_event
 - ohviezdičkovaná udalosť, čiže udalosť, o ktorú používateľ prejavil väčší záujem, pričom takýchto udalostí si môže používateľ vybrať viacero
- Selected_exhibit_for_tour
 - exponát, ktorý si používateľ vybral, aby sa mu zobrazila navigácia k nemu, pričom používateľ si môže vybrať viacero exponátov
- Location
 - miesto konania udalosti s presnými špecifikáciami (napr. súradnice na mape), pričom sa toto miesto môže pre rôzne udalosti opakovať
- Event
 - udalosť (prehliadka, výstava)
- Exhibit
 - exponát, ktorý sa vyskytuje v udalosti, pričom udalosť môže mať viacero exponátov
- Beacon
 - BLE zariadenie umiestnené pri exponáte a informácie o ňom, pričom pri exponáte môže byť umiestnených viac takýchto zariadení
- Image
 - informácie o obrázku (napr. adresa fyzického súboru), pričom User môže mať jeden obrázok (profilová fotka) a Event a Exhibit môžu mať viacero obrázkov (fotky priestorov, exponátov)



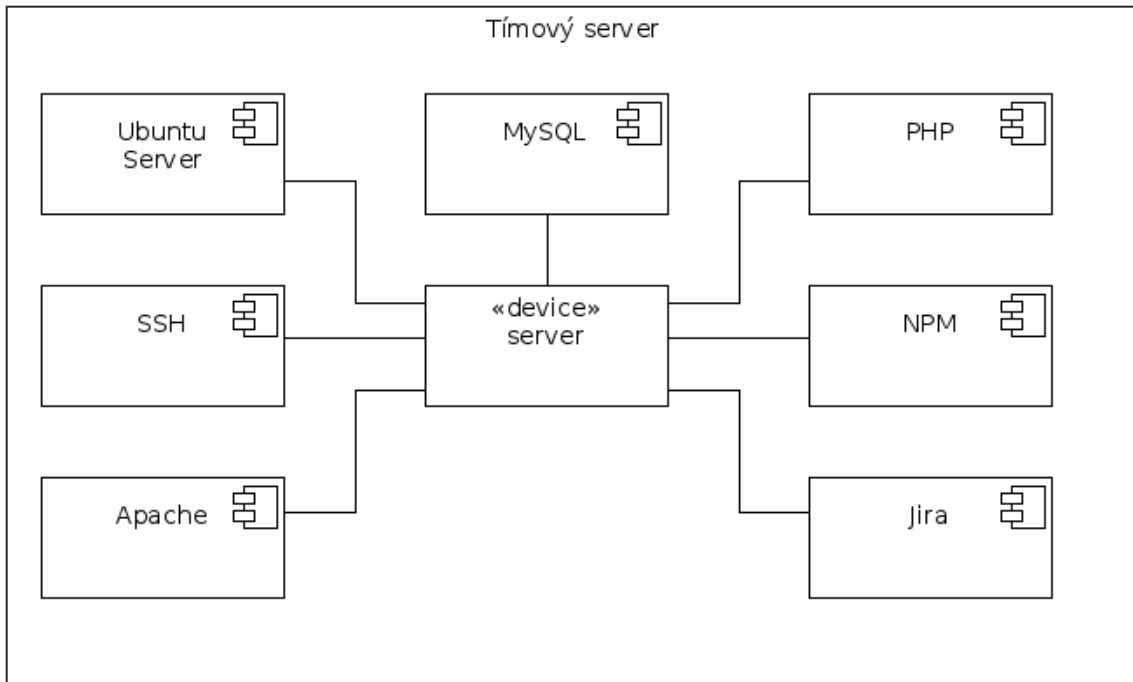
Obr. 7.1 Dátový model.

8 Tímový server

Na sprístupňovanie potrebných vecí online sme dostali k dispozícii tímový server, ktorý bol na začiatku prázdny. Bolo potrebné si nainštalovať a nakonfigurovať nasledovné veci, aby sme server mohli používať podľa potreby:

- Ubuntu Server 16.04.1 LTS (L)
 - operačný systém
- SSH
 - pre vzdialené pripojenie na server a vykonávanie zadaných príkazov
- Apache (A)
 - webový server pre sprístupnenie potrebných vecí cez webový prehliadač
- MySQL (M)
 - relačná databáza pre uchovávanie dát
- PHP (P)
 - kompilér pre spúšťanie zdrojových kódov, keďže backend je písaný v tomto jazyku
- NPM
 - node package manager pre správu javascriptových knižníc a webového servera, keďže admin web je písaný pomocou týchto nástrojov
- Jira
 - nástroj na manažovanie úloh

Na server sme potom vložili našu tímovú stránku, ktorá bola vďaka Apache prístupná cez webový prehliadač (Obr. 8.1).



Obr. 8.1 Diagram komponentov zobrazujúci tímový server.

9 Moduly systému

9.1 Diagram tried - Android

Celková funkcionálnosť BeaCode aplikácie sa nachádza v balíku *sk.beacode.beacoreapp*, v ktorom sú vnorené ďalšie balíky. Každý balík obsahuje triedy/rozhrania ktoré spoločne súvisia, presnejšie povedané ponúkajú podobnú funkcionálnosť. Základ aplikácie je vyvíjaný podľa vzoru Model – View – Controller. Tento vzor sme postupne prispôbili našej aplikácii tak, že bolo potrebné sa aj trochu vyhnúť z jeho pravidiel.

V balíku *models* sú prezentované objekty pre prenášanie konkrétnych dát. Napríklad tu sa vyskytujú triedy *Beacon* ktorá nesie informácie o majáčku, potom trieda *Event* ktorá nesie informácie o udalostiach, atď. Triedy *EventList* a *ExhibitList* ktoré sa v tom balíku tiež vyskytujú nám vrátia všetky udalosti a expozície, ktoré sú na serveri prítomné.

V balíku *managers* sa nachádzajú triedy, ktoré umožňujú prepojenie so serverom, načítanie údajov a posielanie údajov zo/do servera, vymazanie údajov. Napríklad trieda *ExhibitApi* je zodpovedná za všetky operácie, spojené s údajmi o exponáte, t.j. načítanie údajov o exponáte zo servera, posielanie údajov o exponáte na server atď.

Balíky *activities* a *adapters* v podstate predstavujú kontroléry (controllers). To znamená, že triedy/rozhrania v ňom pôsobia ako na model (models), tak aj na pohľady (views). Riadia tok dát do modelov, pričom robia aktualizáciu niektorých pohľadov pri zmene dát.

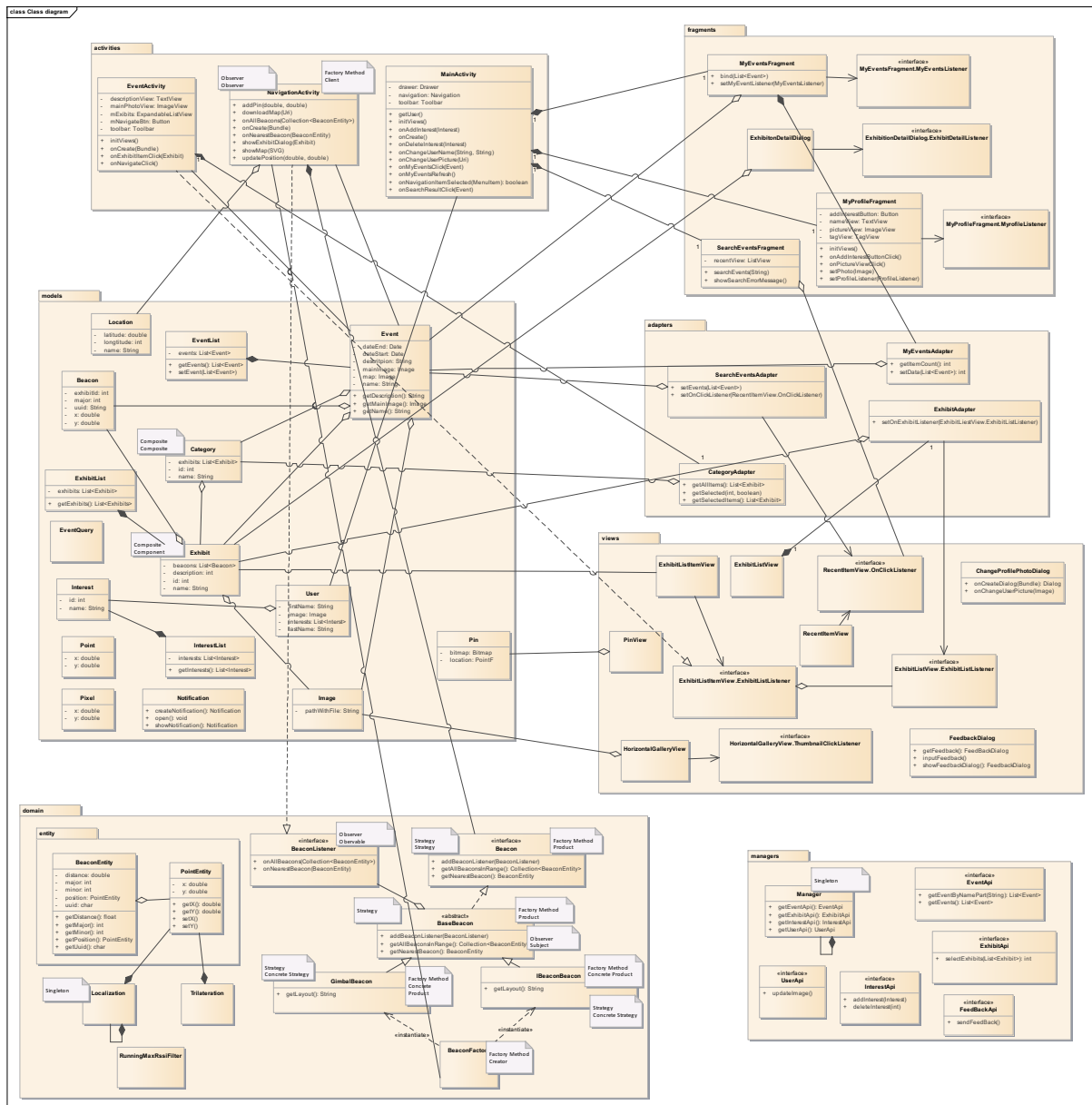
V balíkoch *fragments* a *views* sú zadefinované hlavné elementy obrazoviek našej aplikácie. Tiež je v ňom nastavená aj funkcionálnosť týchto elementov (napr. listeners). Konkrétnejšie povedané, vo *fragments* sa nachádzajú alert dialogy, ktoré sa v aplikácii vyskytujú a vo *views* sú rôzne komponenty, ktoré na tých obrazovkách sú. Obidva balíky tiež poskytujú logiku, ktorá sa veľmi prepletá s logikou balíkov *activities* a *adapters*. To znamená že v niektorých triedach tiež riadia tok dát do modelov a robia aktualizáciu niektorých pohľadov pri zmene dát.

V balíku *layouts* sú xml súbory, ktoré znázorňujú vzhľad všetkých obrazoviek aplikácie.

V balíku *values* sú zadefinované konštantné hodnoty, ktoré sa používajú v rôznych triedach aplikácie.

V balíku *drawable* sú súbory, ktoré znázorňujú vzhľad rôznych komponentov obrazoviek ako sú: tlačidlá, ikony atď.

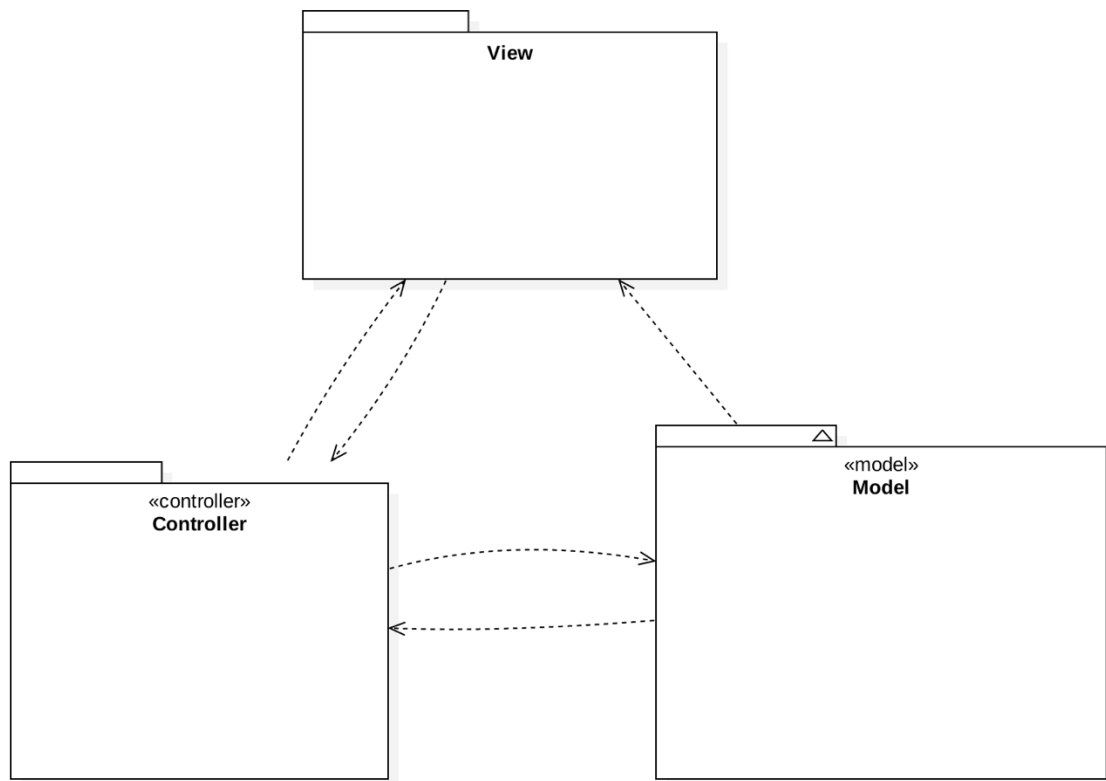
Na nasledujúcom obrázku je zobrazený diagram tried, na ktorom sú predstavené najdôležitejšie triedy z našej aplikácie a vzťahy medzi nimi



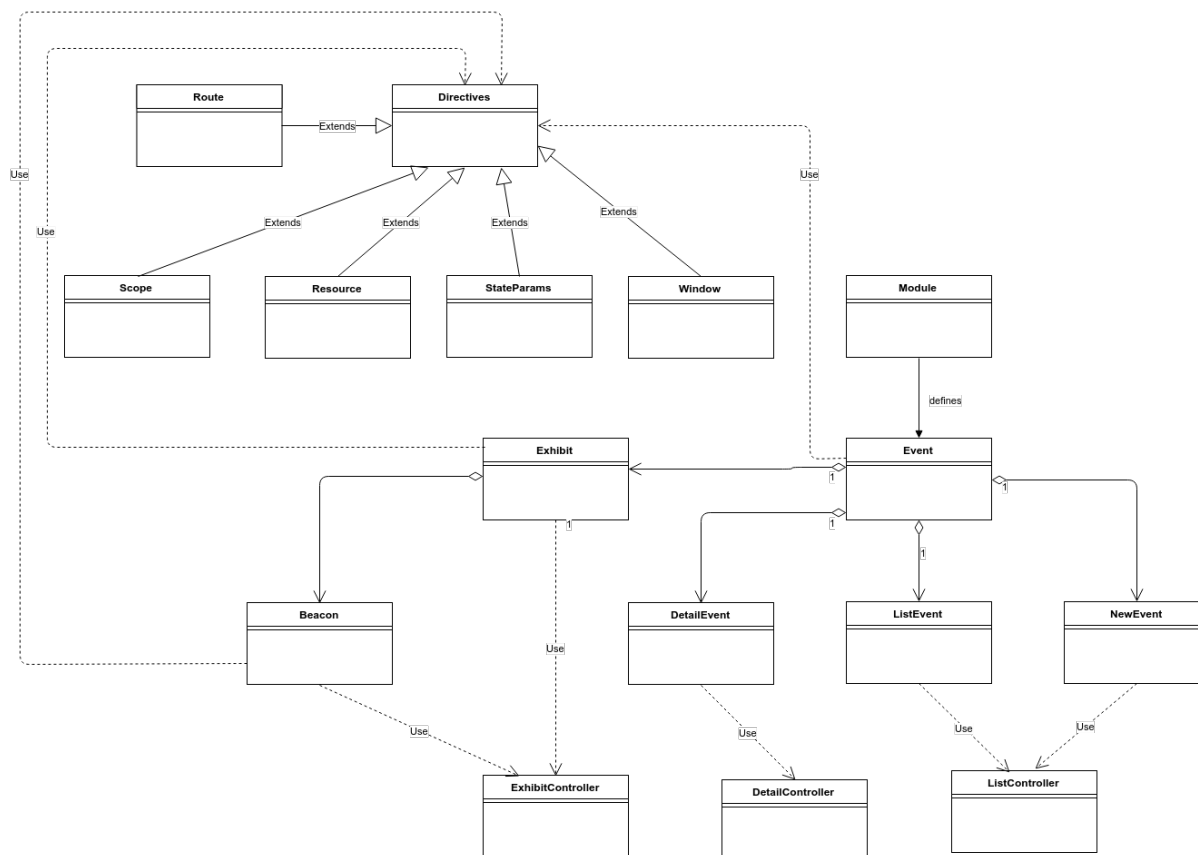
Obr. 9.1 Diagram tried aplikácie v Android.

9.2 Diagram tried – iOS

Keďže diagram class diagram pre iOS bol príliš obširny a pri priložení prílohy bol nečitateľný, vytvorili sme všeobecný class diagram pre náš projekt. Jednotlivé triedy majú teda rovnaké vzťahy, len ich je viac a majú iné názvy s duplikujúcou architektúrou tried.



9.3 Diagram tried – Admin Web



9.4 Zobrazenie zoznamu eventov

9.4.1 Android

Analýza: Súčasťou aplikácie je ponúknuť používateľovi zoznam udalostí, ktoré sa uskutočnia v blízkej budúcnosti.

Návrh: Pre realizáciu tejto funkcionality bolo potrebné, aby na API boli udalosti, ktoré sa budú konať. Údaje, ktoré pre každú udalosť majú byť na API sú názov udalosti, dátum, miesto diania a popis. Bolo potrebné tiež aj vytvoriť príslušné rozhranie pre frontend aplikácie, pomocou ktorého používateľ bude môcť príslušné udalosti pozrieť.

Implementácia: Vzhľad je zadaný v xml súborov fragment `_my_events` a `recycle_items`. Implementácia funkcionality (načítanie potrebných údajov, pridanie listenerov atď.) sa nachádza v triede `MyEventsFragment` a prepojenie medzi nimi je `MyEventsAdapter`. Model udalosti predstavuje trieda `Event` a `EventManager` definuje url API, z ktorého potrebné údaje sa načítajú.

Testovanie: Testovanie predstavovalo v podstate kontrolu, či údaje na frontende aplikácie sú rovnaké ako tieto, čo sú na serveri. Testy boli všetky ručne urobené

9.4.2 iOS

Analýza: Jedným z problémov zákazníka je spôsob upozornenia potencionálneho návštevníka podujatia o novom, nadchádzajúcom podujatí. Na tieto účely musia organizátori pre každé podujatie vyčleniť značné finančné prostriedky. Klasické druhy reklamy (rádio, televízia, papierové brožúry) nie sú dostatočne efektívnym riešením.

Návrh: Navrhovaným riešením tohoto problému je vytvorenie databázy podujatí, ktorá bude návštevníkom a osobám, ktoré majú o podujatie záujem prístupná cez mobilnú aplikáciu pre iOS a Android. Prehľadná obrazovka, ktorá zobrazuje podujatia podľa preferencie používateľa aplikácie a zároveň ich usporadúva vzostupne podľa času konania a lokality.

Implementácia: Vytvorená bola obrazovka My Events, ktorá zobrazuje náhľad pre každé podujatie ktoré vyhovuje preferenciám používateľa. Každý náhľad obsahuje obrázok podujatia, názov, čas konania a krátky popis.

Testovanie: Prebehlo manuálne testovanie funkcionality obrazovky ktoré odhalilo, že obrazovka je funkčná a spĺňa definované požiadavky. Súčasťou nasledujúcich šprintov bude aj testovanie používateľského zážitku.

9.4.3 Admin Web

9.4.3.1 Pridanie sekcií podujatia

Analýza: Z analýzy vyplynula potreba rozdeľovať podujatie na viacero sekcií, keďže napr. IIT.SRC, na ktorej plánujeme BeaCode testovať, je rozdelená na viacero blokov, čiže sekcií.

Návrh: Pre realizáciu je potrebné vytvoriť službu na BE na vytváranie nových oblastí. Ďalej je potrebné implementovať nové obrazovky v admin webe. Jednu, v ktorej budú zobrazené všetky sekcie k podujatiu, ďalšiu na vytváranie nových sekcií a ďalšiu na detail každej sekcie, kde budú zobrazené základné údaje o sekcii a možnosť pridávanie exponátov do sekcie.

Implementácia: Vytvorili sme nový priečinok “categories”, v ktorom sa nachádzajú 3 súbory: categories-list.controller.js slúžiaci na správu funkcionality modelu, categories-list.html slúžiaci na zobrazenie všetkých sekcií a category-new.html slúžiaci na pridávanie novej sekcie. Detail sekcie je implementovaný v event-detail.html.

API:

- pridávanie nového eventu: */api/admin-web/events/new*
 - level: ‘0‘
 - parentId: ‘0‘
- pridávanie novej kategórie: */api/admin-web/events/new*
 - level: ‘1‘
 - parentId: id príslušného nadradeného eventu

Testovanie: Testovanie predstavovalo kontrolu funkcionality na admin webe – t. j. pridávanie nových sekcií a ich zobrazovanie. Testovanie bolo vykonané ručne.

9.4.3.2 Odstránenie exponátov, sekcií a podujatí

Analýza: V admin webe máme implementovanú funkcionálnosť pridávania podujatí, sekcií a exponátov. Niekedy je potrebné tieto objekty odstrániť, preto pridávame funkcionálnosť na ich odstránenie.

Návrh: Pre realizáciu je potrebné vytvoriť REST službu na BE na odstraňovanie podujatí, sekcií a exponátov. Ďalej je potrebné implementovať túto funkcionálnosť aj na FE. V UI pridáme ku každému objektu krížik a po kliknutí naň sa daný objekt vymaže.

Implementácia: Logika a volanie REST služby je implementované v príslušných javascript súboroch. Implementácia pre odstránenie sekcií je v súboroch `categories-list.controller.js` a implementácia na odstránenie podujatí a exponátov v `event-list.controller.js` a `event-detail.js`.
API:

- odstránenie podujatia: `/api/admin-web/events/{eventId}`
- odstránenie sekcie: `/api/admin-web/events/{categoryId}`
- odstránenie exponátu: `/api/admin-web/events/{eventId}/exhibits/{exhibitId}`

Testovanie: Testovanie predstavovalo kontrolu funkcionality na admin webe – t. j. odstraňovanie podujatí, sekcií a exponátov. Testovanie bolo vykonané ručne.

9.4.4 Backend – API na odstránenie eventov, na odstránenie sekcií a na odstránenie exhibitu

Analýza: V aplikácii sa pracuje s objektami ako event, sekcia, exhibit. Momentálne je možné dané objekty vytvárať. Je ale nutné vedieť dané objekty aj vymazávať. Preto je nutné pridať možnosť vymazania vybraného objektu.

Návrh: Treba vytvoriť API, ktoré dostane identifikátor vybraného objektu a podľa neho vybraný objekt reálne vymaže z databázy.

Implementácia: Sú vytvorené API pointy pre každý druh objektu. Každý API point očakáva id vybraného objektu. Následne sa vykoná vymazanie z databázy a na frontend sa vráti informácia o úspešnosti vymazania.

Testovanie: Poslanie niekoľkých requestov na vymazanie každého druhu objektu a sledovanie, či sa vybraný objekt vymazal z databázy a teda sa už na frontend nevracia.

9.5 Vyhľadanie eventu

9.5.1 Android

Analýza: Aby používateľ mohol jednoducho vyhľadať event, o ktorý má záujem, bolo potrebné vytvoriť možnosť hľadania udalostí na základe zadania ich názvu.

Návrh: Navrhovaným riešením je vytvorenie obrazovky, v rámci ktorej používateľ môže zadávať názvy udalostí, o ktoré má záujem a následne na obrazovke sa mu zobrazí zoznam vyhovujúcich udalostí.

Implementácia: Bola vytvorená obrazovka Search Event, ktorá umožňuje vyhľadanie udalostí podľa zadaného názvu. Na obrazovke sa zobrazí zoznam, ktorý vyhovuje zadanému výrazu. Okrem toho bola do obrazovky pridaná časť, v ktorej sa zobrazujú nedávne hľadania.

Testovanie: Testovanie prebehlo manuálne, to znamená, že sa manuálne kontrolovala funkčnosť a správnosť danej obrazovky podľa definovaných požiadaviek.

9.5.2 iOS

Analýza: Používateľ aplikácie má pri manuálnom vyhľadávaní výrazne sťaženú situáciu, kedy je nútený prezrieť si manuálne dlhé zoznamy podujatí, len aby našiel podujatie o ktorom chce získať ďalšie informácie.

Návrh: Navrhovaním riešením je vytvorenie obrazovky vyhľadávania, ktorá zobrazuje iba podujatia ktoré zodpovedajú výrazu zadanému používateľom aplikácie. Filtrovanie sa deje v reálnom čase a spúšťa sa vždy po zmene stavu zadaného výrazu.

Implementácia: Súčasťou implementácie bolo vytvorenie novej obrazovky, ktorá je dostupná z hlavného navigačného panelu aplikácie. Obsahuje vstupné pole, ktoré používateľ použije pre vloženie podreťazca, na základe ktorého chce uskutočniť filtrovanie podujatí. Filtrovanie sa spustí hneď ako používateľ prestane písať a zobrazí výsledky filtrovania na obrazovke.

Testovanie: Prebehlo manuálne testovanie funkcionality obrazovky ktoré odhalilo, že obrazovka je funkčná a spĺňa definované požiadavky.

9.6 Zobrazenie detailu eventu

9.6.1 Android

Analýza: Aby používateľ mohol zobrazit' detail eventu, bolo potrebné vytvorit' možnosť zobrazovania detailnej obrazovky pre event.

Návrh: Riešením je vytvorit' obrazovku detailu podujatia, ktorá bude zobrazovať všetky potrebné informácie o danom podujatí: sekcie patriace danému podujatiu, možnosť rozkliknúť každú sekciu pričom sa zobrazia exponáty patriace danej sekcii.

Implementácia: Do baličku model bola vytvorená trieda *Category* a do baličku manager bola vytvorená trieda *CategoryManager* použitá na načítanie údajov o kategórie (sekcii) zo serveru patriace príslušnému eventu. Trieda *CategoryAdapter* v baličku adapter prezentuje celkovú logiku súvisiace s kategóriami ako je napríklad rozčlenenie kategórie na exhibície a tiež tú sú zahrnuté komponenty patriace obrázky. Bol vytvorený súbor *item_exhibit_category.xml*, ktorý zobrazuje kategórie vybrateho eventu a tiež zobrazuje aj k vybratej kategórie patriace exhibície.

Testovanie: Testovanie predstavovalo v podstate kontrolu, či sa používateľovi zobrazia všetky udalosti, kategórie patriace danej udalosti ako aj exhibity ktoré patria pod danej kategórie.

9.6.2 iOS

Analýza: Je nevyhnutné dodať používateľovi aplikácie detailné informácie o podujatí ktoré ho zaujímajú. Takýmto spôsobom sa môže informovať o podujatí samostatne a zistiť viacej informácií, čo ho nakoniec môže presvedčiť k účasti.

Návrh: Riešením je vytvoriť obrazovku detailu podujatia, ktorá bude zobrazovať všetky potrebné informácie, ktoré by potencionálneho záujemcu o podujatie mohli zaujímať.

Implementácia: Bola vytvorená nová obrazovka, ktorá obsahuje podrobný textový popis podujatia spolu s titulnou fotografiou. Ďalej si tu môže používateľ prezerat' galériu fotografií a aj zoznam exponátov ktoré je možné na podujatí vidieť. Označením exponátov o ktoré má používateľ záujem je možné definovať zoznam exponátov pre ktoré bude zostavená trasa v časti navigácie. Navigáciu je možné spustiť priamo z obrazovky detailu podujatia.

Testovanie: Prebehlo manuálne testovanie funkcionality obrazovky ktoré odhalilo, že obrazovka je funkčná a splňa definované požiadavky.

9.7 Zobrazenie detailu exponátu

9.7.1 Android

Analýza: Aby si používateľ vedel o požadovanom exponáte pozrieť potrebné informácie, bolo potrebné vytvoriť obrazovku, na ktorej budú všetky podrobné informácie o exponáte.

Návrh: Navrhovaným riešením je dialógové okno, ktoré sa zobrazí po kliknutí na určitý exponát.

Implementácia: Po kliknutí na určitý exponát sa zobrazí dialógové okno s detailom exponátu, ktorý obsahuje názov exponátu, obrázok exponátu a popis daného exponátu.

Testovanie: Testovanie prebehlo manuálne, to znamená, že sa manuálne kontrolovala funkčnosť a správnosť danej obrazovky podľa definovaných požiadaviek.

9.7.2 iOS

Analýza: Rovnako ako pri detaile podujatia je potrebné používateľovi poskytnúť informácie o samotných exponátoch. Takto môžeme priblížiť používateľovi dianie na podujatí a predstaviť mu stručne ponúkané exponáty.

Návrh: Navrhovaným riešením je obrazovka detailu exponátu ktorá zobrazí údaje o exponáte v textovej podobe ako aj prostredníctvom priloženého obrázka.

Implementácia: Vytvorili sme obrazovku detailu exponátu ktorú je možné vyvolať priamo z obrazovky detailu podujatia. Obrazovka obsahuje fotografiu exponátu spolu s krátkym a stručným popisom.

Testovanie: Prebehlo manuálne testovanie funkcionality obrazovky ktoré odhalilo, že obrazovka je funkčná a splňa definované požiadavky.

9.7.3 Admin Web – pridávanie a zobrazovanie obrázkov a mapy k exponátom a eventov

Analýza: Pre plnohodnotné zobrazovanie podujatí a ich exponátov v našej aplikácii je potrebné v admin webe pridávať a zobrazovať obrázku k exponátom a podujatiam.

Návrh: Pre realizáciu je potrebné vytvoriť službu na BE na pridávanie obrázkov k podujatiam a exponátom. Pridávanie obrázkov k podujatiam bude implementované na obrazovke so zoznamom sekcií podujatia, kde budú obrázky aj zobrazované. Pridávanie obrázkov k exponátom bude implementované na obrazovke detailu sekcie, kde sa nachádza zoznam všetkých exponátov danej sekcie. V tomto sprinte budeme implementovať aj pridávanie a zobrazovanie svg mapy podujatia. Mapu je možné pridávať a odoberať na obrazovke so zoznamom sekcií príslušného eventu. Po pridaní mapy sa na obrazovke zobrazí iba miniatúra mapy, ktorá sa po kliknutí na obrázok celá zobrazí v novej záložke prehliadača. Možné je nahrávať iba mapy vo formáte svg. Mapa sa pridáva cez API na pridávanie obrázkov k eventom, len sa zmení objectType na event-map.

Implementácia:

Logika implementácie služieb sa nachádza v javascript súbore *categories-list.controller.js*. Nahrávanie obrázkov zabezpečuje metóda *uploadImages* a nahrávanie mapy metóda *uploadMap*. Implementácia UI je v súbore *categories-list.html*.

API:

- pridávanie obrázka k exhibitu: */api/admin-web/events/{eventId}/exhibits/{exhibitId}/images/new*
- pridávanie obrázka k eventu: */api/admin-web/events/{eventId}/images/new*
- pridávanie mapy: */api/admin-web/events/{eventId}/images/new?objectType=event-map*

Testovanie: Testovanie predstavovalo kontrolu funkcionality na admin webe – t. j. pridávanie novej mapy ba pridávanie obrázkov v exponátom. Testovanie bolo vykonané ručne.

9.7.4 Backend – API na prácu s eventom a API na prácu s exhibitmi

Analýza: V aplikácii je nutné vedieť pracovať s eventami a exhibitmi. Sú to základné objekty našej aplikácie. Je ich treba vedieť vytvoriť, nastaviť im informácie, obrázky. Ďalej je treba vedieť ešte pracovať s objektom kategória, pričom event obsahuje kategórie a tie obsahujú exhibity.

Návrh: Treba vytvoriť API na vytvorenie a získanie spomínaných objektov. Musí sa tu pracovať zo všetkými potrebnými informáciami.

Implementácia: Sú vytvorené API pointy na vytvorenie a získanie objektov. Ak ide o vytvorenie, tak vytvorím nový objekt v databáze a nastavím mu dané informácie. Ak ide o získanie, tak vyberiem objekt z databázy a vrátim ho na frontend. Kategória je špeciálny typ eventu, takže sa s ňou dá pracovať podobne ako s eventom.

Testovanie: Poslanie niekoľkých requestov na vytvorenie a získanie objektov a sledovanie, či získané informácie sedia s poslanými informáciami.

9.8 Profil

9.8.1 Android

Analýza: V aplikácii používateľ má svoj profil, ktorý obsahuje jeho meno a priezvisko, fotku a jeho záujmy. Funkcionalita, ktorá aplikácia mu umožňuje, je v podstate pridanie/vymazanie svojich záujmov a zmenenie svojej fotky.

Návrh: Pre realizáciu tejto funkcionality bolo potrebné vytvoriť používateľské rozhranie do frontend aplikácie, ako aj API do backend-u. API musí umožňovať tieto funkcie:

- získanie údajov o používateľovi
- nahranie záujmov a fotky
- vymazanie záujmov a fotky

Implementácia: Vzhľad je zadaný v xml súborov *fragment_my_profile*, *fragment_dialog_change_profile* a *fragment_dialog_add_interest*. Implementácia funkcionality sa nachádza v triede *MyProfileFragment* a prepojenie medzi nimi je *MyEventsAdapter*. Model používateľa predstavuje trieda *User* a *UserManager* definuje url API, z ktorého potrebné údaje sa načítajú a do ktorého sa údaje nahrávajú. Bolo potrebné do build-gradle pridať TagView knižnic pre lepšie používateľské rozhranie.

Testovanie: Testovanie predstavovalo v podstate kontrolu, či údaje na frontende aplikácii sú rovnaké ako tie, čo sú na serveri. Počas testovanie sme prišli na to, že zmenenie fotky na frontende funguje, ale na API sa fotka nezmení. Je potrebné potom túto funkcionality dorobiť. Testy boli všetky ručne urobené.

9.8.2 iOS

Analýza: Odporúčanie podujatí na základe osobných informácií a preferencií používateľa môže výrazne uľahčiť spôsob, akým používateľ aplikácie dostáva informácie o dostupných podujatiach.

Návrh: Návrhom je vytvorenie obrazovky používateľa, ktorá bude slúžiť na zhromažďovanie informácií o používateľovi, jeho preferenciách a osobných údajoch. Obrazovku môžeme označiť ako profil používateľa, ktorý v aplikácii používa.

Implementácia: Vytvorili sme obrazovku profilu používateľa ktorá pozostáva z troch častí. Používateľ si v prvej časti môže definovať profilovú fotografiu, ktorú bude v aplikácii používať. Profilovú fotografiu je možné vybrať z galérie. Pod fotografiou má používateľ možnosť zadefinovať si svoje celé meno. Ďalšou časťou je sekcia kľúčových slov tzv. tagov. Tie predstavujú jednoslovné výrazy ktorými si vie používateľ definovať svoje preferencie a následne sú tieto tagy použité pre filtrovanie vhodných a pre daného používateľa zaujímavých podujatí.

Testovanie: Prebehlo manuálne testovanie funkcionality obrazovky ktoré odhalilo že obrazovka je funkčná a splňa definované požiadavky.

9.9 Výber exponátov

9.9.1 Android

Analýza: Súčasťou aplikácie je umožniť používateľovi vybrať exponáty, ktoré chce pozrieť ako aj umožniť mu zobrazenie označených exponátov na mape. Samozrejme pri exponátoch, ktoré si vybral má aj možnosť ich odznačiť. Aby to bolo možné je potreba vytvoriť API na prácu s označenými exponátmi: načítať označené exponáty pre daného používateľa, vymazať označené exponáty, ako aj pridať označene exponáty na API.

Návrh: Pre realizáciu tejto funkcionality boli na serveri vytvorené tri API pre načítanie označených exponátov, vymazanie a pridanie. Pre načítanie a pridanie exponátov je potrebné len poslať na API ID eventu a keď sa označuje exponát, tak je potrebné okrem ID eventu poslať aj ID exponátu, ktorý sa ma označiť.

Implementácia: Bol vytvorený interface SelectedExhibitsApi, pomocou ktorého je práca s označenými exponátmi umožnená. V podstate interface umožňuje načítanie, pridanie a vymazanie označených exponátov. Trieda Manager bola doplnená o funkciu getSelectedExhibitsApi. V baličku models bola vytvorená trieda SelectedExhibit, ktorá poskytuje dáta o označených exponátoch a bola vytvorená aj trieda SelectedExhibitList. Do nej sa uložia označene exponáty načítané z API.

Testovanie: Testovanie prebiehalo tak, že najprv bolo okontrolované to, či údaje, ktoré načítame sú naozaj tie údaje, ktoré sú na serveri. Úplné testovanie bolo v úlohe Zobrazenie označených exponátov na mape.

9.9.2 iOS

9.10 Lokalizácia

9.10.1 Android

Analýza: Súčasťou aplikácie je aj lokalizovanie používateľa. Preto bolo potrebné používateľovi ponúknuť mapu, na ktorej sa zobrazí jeho súčasná poloha a navyše aj poloha exponátov. Na začiatku bolo potrebné analyzovať už už existujúce riešenia. Bola urobená analýza nad diplomovou prácou Liskovca, kde bol popísaný spôsob, akým sa počíta vzdialenosť používateľa od beaconov a spôsob, akým sa dá robiť trilaterácia. Bolo potrebné aj analyzovať kód z bakalárskej práce Augustína, pretože v jeho práci bol rozoberaný rovnaký problém.

Návrh: Pre realizáciu tejto funkcionality bol otestovaný algoritmus na výpočet trilaterácie z bakalárskej práce Augustína. Potom bola vypočítaná vzdialenosť z beaconov k používateľovi použitím vzorca $D_{d0} * \text{Math.pow}(10, ((\text{rssi} - \text{RSSI}_{d0})/10 * N))$ a bol použitý `RunningMaxRssiFilter`, ktorý je urobený na základe `RunningAverageRssiFilter`. Použitím tohto spôsobu výsledky neboli lepšie ($n = \langle 2,4 \rangle$; $\text{RSSI}_{d0} = \langle -65, -75 \rangle$). Vzdialenosť vypočítaná použitím vzorca $D_{d0} * \text{Math.pow}(10, ((\text{RSSI}_{d0} - \text{tx_power})/10 * N))$, pričom bol použitý `RunningMaxRssiFilter`, ktorý je urobený na základe `RunningAverageFilter`. Ani s týmto výsledky neboli lepšie.

Implementácia: Na vypočítanie vzdialenosti je implementovaný `RunningMaxRSSi Filter`, pričom RSSI signál sa načíta 10krát za sekundu (použitím `updateScanPeriods` funkcie). Výpočet vzdialenosti je v triede `GimbalBeacon`, pričom vzdialenosť sa počíta na základe toho, čo je v Android - beacon knižnice – použitím metódy `getDistance()`. Konečné hodnoty pre x a y (súradnice používateľa) sa vypočítajú ako priemer posledných 20 hodnôt vypočítané použitím trilaterácie.

Testovanie: Testovanie prebiehalo tak, že boli načítané hodnoty z API a v škole na 3. poschodí bolo otestované, či poloha používateľa je správna.

9.10.2 iOS

Analýza: Súčasťou aplikácie je aj lokalizovanie používateľa. Preto bolo potrebné používateľovi ponúknuť mapu, na ktorej sa zobrazí jeho súčasná poloha a navyše aj poloha exponátov. Na začiatku bolo potrebné analyzovať už už existujúce riešenia. Skúsenosti stouť metódou mal v našom tíme Peter. Štúdiom bakalárskej práce sme dospeli k viacerým záverom a dokázali analyzovať problém lokalizácie. Potrebné bolo naštudovať aj framework s názvom `CoreLocation`.

Návrh: Pri návrhu sme využili sme využili poznatky, ktoré sme nadobudli z analýzy. Podarila sa nám zistiť, že framework ktorý sme vybrali rieši všetky potrebné náležitosti v rámci prepočítavania vzdialeností iBeacons.

Implementácia: Pri implementácii CoreLocation sme pridali triedu ktorá sa stará o lokalizáciu. Táto trieda bola naimplementovaná ako controller, ktorý obsahuje algoritmus trilaterácie. Na základe tejto triedy bolo možné implementovať jednoduchý lokalizačný modul. Jediným problémom bolo však menšie oneskorenie prepočítavania, keďže framework CoreLocation, ktorý sme využili vraciac vzdialenosti len raz za sekundu.

Testovanie: Testovanie prebiehalo tak, že boli hodnoty a mapa načítané staticky kvôli prezentácií. Testovanie prebehlo na spodnom poschodí FIIT.

9.10.3 Admin Web – Parsovanie beaconov z mapy

Analýza: Pre použitie lokalizácie a navigácie na mape je potrebné z svg mapy vyparsovať beacony.

Návrh: Pre realizáciu je potrebné vytvoriť službu na BE, ktorá bude vystavená pre admin web a bude slúžiť na vyparsovanie beaconov z mapy pridelenej určitému podujatiu. Služba bude vytvorená na BE a v admin web sa služba zavolá s príslušným ID podujatia, kliknutím na tlačidlo “Vyparsovať beacony z mapy”.

Implementácia: Logika implementácie služieb sa nachádza v javascript súbore categories-list.controller.js. Vyparsovanie beaconov z mapy a teda zavolanie služby z BE s príslušným ID podujatia je implementované v metóde *parseBeaconFromMap*. Tlačidlo, ktoré má v sebe implementovaný spúšťač tejto metódy sa nachádza v súbore categorie-list.html.

API:

- vyparsovanie beaconov z svg mapy: */api/admin-web/events/{eventId}/parse-beacon-svg*

Testovanie: Testovanie predstavovalo kontrolu funkcionality na admin webe a následné skontrolovanie na BE, či po zavolaní služby z BE s príslušným ID podujatia naozaj nastalo vyparsovanie beaconov z svg mapy. Testovanie bolo vykonané kliknutím na tlačidlo „Vyparsovať beacony z mapy“.

9.10.4 Backend – API na vracanie mapy a polohy beaconov

Analýza: K eventom je treba vedieť nahráť mapu a neskôr ju opätovne získať naspäť. Taktiež je nutné vedieť nahráť a nastaviť beacony (zatiaľ manuálne na backende z excel súboru) a neskôr ich vrátiť na frontend.

Návrh: Treba vytvoriť API na nahranie mapy. Nahranú mapu treba neskôr vracať k danému eventu. Treba spraviť parser, ktorý vyparsuje beacony z excelu a uloží informácie do databázy. Potom treba tieto informácie vracať k danému eventu.

Implementácia: Je vytvorený API point na nahranie mapy k eventu. Vracanie mapy k eventu je pridané k existujúcemu API pointu, ktorý vracia aj iné informácie o evente. Je vytvorený parser, ktorý dostane beacony z excelu (manuálne), vyparsuje ich a vloží ich do databázy.

Vracanie beaconov je tiež pridané k existujúcemu API pointu, ktorý vracia aj iné informácie o evente.

Testovanie: Poslanie niekoľkých requestov na nahranie a získanie mapy pre event. Spustenie parsera beaconov a sledovanie, či sa do databázy zapísali správne údaje. Poslanie niekoľkých requestov na získanie beaconov pre event.

9.11 Notifikovanie

9.11.1 Android

Analýza: Súčasťou aplikácie je zobraziť používateľovi notifikáciu o exponáte, ktorý je v jeho blízkosti, a o ktorý má záujem. Po prijatí notifikácie používateľ má možnosť otvoriť notifikáciu s tým, že sa mu zobrazia podrobné informácie o exponáte. Jedna notifikácia sa zobrazuje len pre jeden exponát. To znamená, že ak sa používateľovi už zobrazila notifikácia o tom, že je v blízkosti exponátu, tak potom sa mu tá istá notifikácia znova nezobrazí. Notifikácie sa môžu zobraziť len ak používateľ klikol na tlačidlo Naviguj.

Návrh: Pre realizáciu tejto funkcionality je potrebné najprv z api načítať, ktoré exponáty sú označené. Potom okontrolovať, ktoré z tých exponátov patria k beaconom, ktoré sú vo vzdialenosti od používateľa do 1m. Tiež musí byť aj kontrola, či už takáto notifikácia nebola už pushnutá, a ak áno, tak sa znova taká notifikácia nevytvorí. Ak používateľ otvorí notifikáciu, je potrebné, aby z api boli načítané údaje o exponáte, o ktorom notifikácia bola pushnutá.

Implementácia: V triede *Notification Manager* sú nastavené parametre notifikácii ako sú zvuk, čo má notifikácia obsahovať atď. V triede *NavigationActivity* v metóde *didRangeBeaconsInRegion* sa notifikácia vytvára. Tu je kontrola o tom, či sa má notifikácia pushnúť a je tu aj nastavenie o tom, čo sa má zobraziť používateľovi v prípade, keď notifikáciu otvorí.

Testovanie: Testovanie prebiehalo tak, že sme označili exponáty a potom sme sa približovali k tým beaconom, ktoré sú v podstate vybrané exponáty. Otestovali sme, či keď sme vo vzdialenosti menej ako 1m či sa notifikácia pushne a či po jej otvorení sa naozaj zobrazia informácie o exponáte. Bolo otestované aj to, či sa notifikácia zobrazí len raz.

9.11.2 iOS

Analýza: Dôležitou súčasťou aplikácie sú notifikácie, ktoré sú používateľovi poskytnuté v rôznych situáciách, podľa jeho aktuálnej polohy. Notifikácia samotná má pre používateľa nejakú hodnotu a nesie informácie o exponáte. V našom prípade je vhodné použiť existujúcu obrazovku detailu exponátu.

Návrh: Pre notifikovanie používateľa navrhujeme použiť jednoduchú obrazovku s detailom exponátu, ktorá bude zobrazená v prípade, že sa používateľ priblíži na určitú vzdialenosť k danému exponátu. Obrazovka bude pre daný exponát zobrazená vždy iba raz a pri opätovnom príchode sa nezobrazí. Opakujúca prezentácia okna by mohla používateľovi prekážať.

Implementácia: Vytvorili sme spúšťače ktoré v pravidelných intervaloch kontrolujú polohu najbližších majáčikov a ich identifikátory. V prípade že sa identifikátory zhodujú s niektorým z exponátov pre ktoré máme definované podrobnejšie informácie zhodujú a vzdialenosť exponátu presahuje určitú hodnotu (v našom prípade je to hodnota 2m), prezentujeme používateľovi detail tohoto exponátu. Zároveň je už navštívený exponát zaznamenaný a nemôže tak dôjsť k opätovnej prezentácii notifikácie používateľovi.

Testovanie: Testovanie prebiehalo v prostredí fakulty. Vytvorili sme simulovanú konferenciu s použitím šiestich majáčikov a demonštrovali sme si tak funkčnosť notifikácií. Notifikovanie fungovalo dobre, boli však zaznamenané drobné nepresnosti spôsobené pravdepodobne nepresným zameraním majáčikov v priestore.

9.11.3 Admin Web

Analýza: V mobilnej aplikácii je potrebné zobrazovať push notifikáciu pri priblížení sa k nejakému exponátu. Túto push notifikáciu je potrebné pridať k exponátu pri jeho vytváraní.

Návrh: Pre realizáciu je potrebné upraviť databázu a vystavenú REST službu, ktorá slúžila na vytváranie exponátov. Rovnako je potrebné vytvoriť REST službu na upravovanie a pridávanie notifikácií už pri vytvorených exponátoch.

Implementácia: Logika a volanie REST služby je implementované v príslušných javascript súboroch. Implementácia pre pridávanie nových exponátov a jej doplnenie o pridávanie notifikácií sa nachádza v súbore `exhibit-new.controller.js`. V UI sme doplnili textovú plochu pre zadávanie textu k notifikácie.

API:

- úprava exponátu: `PATCH /api/admin-web/events/{eventId}/exhibits/{exhibitId}`

Testovanie: Testovanie predstavovalo kontrolu funkcionality na admin webe – t. j. vytváranie exponátov s push notifikáciou a následná kontrola v zozname exponátov, či sa tam notifikácia nachádza. Testovanie bolo vykonané ručne.

9.11.4 Backend

Analýza: Pri priblížení k exhibitu sa používateľovi pošle push notifikácia, ktorá mu dá krátku a rýchlu informáciu o danom exhibite. Je potrebné vedieť zadefinovať túto push notifikáciu pre každý exhibit.

Návrh: Do objektu exhibit treba pridať vlastnosť push notifikácia. Túto vlastnosť treba pri vytváraní exhibitu nastavovať. Taktiež je potrebné vytvoriť API point na update exhibitu, ktorý bude updatovať vlastnosti exhibitu a teda aj vlastnosť push notifikácia.

Implementácia: Do objektu exhibit je pridaná vlastnosť push notifikácia. Pri vytváraní exhibitu sa táto vlastnosť nastavuje. Je vytvorený API point na update exhibitu, ktorý očakáva id exhibitu a informácie o tom, čo a ako má zmeniť. Pre zmenu push notifikácie sa posielajú informácie v tvare `[{"op": "replace", "path": "/pushNotification", "value": "text"}]`.

Testovanie: Poslanie niekoľkých requestov na vytvorenie a updatovanie exhibitu a sledovanie, či sa správne nastaví vlastnosť push notifikácia a teda či sa aj správne posiela na frontend.

9.12 Poslanie spätnej väzby

9.12.1 Android

Analýza: Súčasťou aplikácie je zobraziť používateľovi notifikáciu na feedback. Notifikácia sa má zobraziť používateľovi vtedy keď odchádza od vybraného exponátu. Samotná notifikácia je na to aby umožnila používateľovi oceniť exponát o ktorý mal záujem a ktorý navštívil. Ocenenie je v podstate ocenie hviezdikami.

Návrh: Pre realizáciu tejto funkcionality je potrebné najprv okontrolovať ktoré exponáty sú vybrané. Potom sa má sledovať kontrola či používateľovi sa zobrazila push notifikácia o niektorých z tých exponátov. Ak áno, potom ma byť kontrola na to či sa používateľ vzdáľuje od tohto exponátu tj. či jeho vzdialenosť od exponátu o ktorom už bola zobrazená notifikácia sa zväčšuje a je viac ako 2,5m. V tomto prípade sa používateľovi zobrazí notifikácia o feedbacku pričom notifikácia o feedbacku o danom exponáte sa má zobraziť len raz.

Implementácia: V baličku *layout* je vytvorený *dialog_feedback.xml* súbor, ktorý v podstate predstavuje layout dialógu feedback. Tu je zadefinované rozmiestenie elementov pričom hlavný element je *Rating Bar* - hviezdinky pomocou ktorých sa robí ocenenie. V triede *NavigationActivity* v metóde *didRangeBeaconsInRegion* je logika na zobrazenie notifikácie feedbacku. Tu je kontrola toho, či sa zobrazila push notifikácia o vybranom exponáte a ak áno sa kontroluje či sa používateľ od neho vzdáľuje ($> 2,5m$). V tomto prípade sa zobrazí notifikácia feedbacku.

Testovanie: Testovanie prebiehalo tak že sme označili exponáty a potom sme sa približovali k tým becaonom, ktoré sú v podstate vybrané exponáty. Otestovali sme, či keď sme vo vzdialenosti $< 1m$ či sa notifikácia pushne. Potom sme sa vzdáľovali a keď sme boli vo vzdialenosti väčšia ako 2,5m tak sme testovali či sa zobrazí notifikácia o feedbacku, tj či v tom okne sa zobrazí meno exponátu ktoré sme navštívili spolu s rating star bar na ocenenie. Bolo otestované aj to, či sa notifikácia zobrazí len raz.

9.12.2 iOS

Analýza: Posielanie spätnej väzby je podobný prípad ako odosielanie notifikácií o exponáte. Ide o podobný problém s rozdielnym prípadom použitia. Z hľadiska implementácie je teda potrebné definovať iba formulár ktorý bude pri odchode od exponátu prezentovaný používateľovi.

Návrh: Navrhujeme vytvoriť pop-up okno ktoré bude obsahovať jednoduché oslovenie, otázku ktorá vyzve používateľa k reakcii a ukazovateľ spokojnosti v podobe radu hviezdíčiek, ktoré

budú symbolizovať úroveň spokojnosti návštevníka (stupnica 1 - nespokojný až 5 - najspokojnejší). Okno je zatvorené a odpoveď zaznamenaná pod potvrdení a zatvorení okna.

Implementácia: Bolo potrebné vytvoriť príslušné pop-up okno a definovať spúšťač ktorý, rovnako ako pri notifikáciách, sleduje relatívne vzdialenosti k najbližším exponátom. Spätná väzba je používateľovi prezentovaná iba v prípade, že sa pri exponáte nachádza prvý krát a iba v prípade, ak ešte spätnú väzbu k tomuto exponátu neobdržal. Zobrazenie sa udeje pri odchode od exponátu, konkrétne ak vzdialenosť prevýši 2 metre.

Testovanie: Testovanie prebiehalo v prostredí fakulty. Vytvorili sme simulovanú konferenciu s použitím šiestich majáčikov a demonštrovali sme si tak funkčnosť posielania spätnej väzby. Posielanie fungovalo dobre, boli však zaznamenané drobné nepresnosti spôsobené pravdepodobne nepresným zameraním majáčikov v priestore.

Príloha A: Testovanie

Android

Test name	Name	Step	Description (Design Steps)	Expected (Design Steps)	Status	Error Description	Status	Error Description
EV_001_A	Zobrazenie eventov	1	Predpoklady: - Používateľ spustí aplikáciu.	Aplikácia sa spustí bez chyby.	OK		OK	Občas trvá načítanie dlho.
		2	Používateľovi sa zobrazí zoznam udalostí.	Zoznam udalostí obsahuje: - názov udalosti - fotka udalosti - dátum udalosti - krátky popis udalosti	OK		OK	
EV_001_B	Zobrazenie eventov	1	Predpoklady: Aplikácia je spustená. Používateľ otvorí menu.	V menu sa používateľovi zobrazia položky: - My Events - Search Events - My Profile	OK		OK	Občas trvá načítanie dlho.
		2	Používateľ klikne na položku My Events	Zobrazí sa zoznam udalostí. Zoznam udalostí obsahuje: - názov udalosti - fotka udalosti - dátum udalosti - krátky popis udalosti	OK		OK	

EV _0 _02 _A	Podrobnosti o evente	1	Predpoklad: Sú zobrazené všetky udalosti. Používateľ klikne na jednu udalosť	Zobrazia sa podrobné informácie o udalosti: - názov udalosti - hlavná fotka udalosti - popis udalosti - fotky udalosti s možnosťou horizontálneho scrollovania - kategórie patriace pod udalosť - tlačidlo naviguj	O K	Popis udalosti sa nezobrazuje od začiatku	O K	
		2	Používateľ klikne na kategóriu	Zobrazí sa zoznam exponátov patriacich pod danú kategóriu. V zozname sú nasledovné údaje: - názov exponátu - časové informácie - autor - checkbox - tlačidlo naviguj	O K		O K	
EV _0 _03 _A	Výber exponátov	1	Predpoklad: Používateľ má zobrazenú základnú obrazovku o podrobnosti eventu. Používateľ klikne na prvú kategóriu.	Zobrazí sa zoznam exponátov, ktoré patria pod prvú kategóriu.	O K		O K	
		2	Používateľ zaklikne niektoré z exponátov, ktoré sa zobrazili.	Exponáty sa označia.	O K	Niekedy checkboxy blikajú (označene checkboxu blikajú a	O K	

						neoznačene sa označujú ako označené, potom bliknú a sa vrátia do neoznačene - správny stav)		
		3	Používateľ klikne znova na prvú kategóriu.	Kategória sa zatvorí a exponáty nebudú zobrazené.	O K		O K	
		4	Používateľ opäť klikne na prvú kategóriu.	Zobrazí sa zoznam exponátov, ktoré patria pod prvú kategóriu. Exponáty, ktoré používateľ predtým označil ostali označené.	O K		O K	
EV_03_B	Výber exponátov	1	Predpoklad: Používateľ má zobrazenú základnú obrazovku o podrobnosti eventu. Používateľ klikne na prvú kategóriu.	Zobrazí sa zoznam exponátov, ktoré patria pod prvú kategóriu.				
		2	Používateľ zaklikne niektoré z exponátov, ktoré sa zobrazili.	Exponáty sa označia.	O K		O K	
		3	Používateľ klikne na druhú kategóriu.	Prvá kategória ostane otvorená (zoznam exponátov je stále vidieť) a označené exponáty sú stále označené). Zobrazí sa zoznam exponátov, ktoré patria	O K		O K	

				pod druhú kategóriu.				
		4	Používateľ zaklikne niektoré z exponátov, ktoré sa zobrazili.	Exponáty sa označia.	OK		OK	
LO K_00 1_A	Lokalizovanie	1	Predpoklad: Používateľ označil niektoré exponáty. Používateľ klikne na tlačidlo Naviguj.	Zobrazí sa mapa, na ktorej červenou bodkou je označená poloha používateľa, zelenou sú označené výbrané exponáty (exponáty, ktoré používateľ označil) a modrou bodkou sú označené ostatné exponáty.	OK		OK	
		2	Používateľ sa začne pohybovať.	Červená bodka sa začne na mape hýbať podľa polohy používateľa.	1 / 2 OK	Poloha je správna keď sa používateľ pohybuje podľa osi x (tj chodí doľava-doprava) a podľa osi y (chodí hore/dole) červená bodka není úplne presná. Používateľ je pri stene (na dolnej strane) a jeho poloha na mape ohľadom osi y sa neustále mení (raz	1 / 2 OK	Polohav smere osi y nie je presná, skáče z miesta na miesto a nevyjadruje aktuálnu polohu používateľa.

						ukazuje že je v strede, raz že je dole).		
LO K_00 1_B	Lokalizovanie	1	Predpoklad: Používateľ neoznačil žiadne exponáty. Používateľ klikne na tlačidlo Naviguj.	Zobrazí sa mapa, na ktorej červenou bodkou je označená poloha používateľa, zelenou sú označené všetky exponáty.	ERROR	Všetky exponáty majú modrú farbu	ERROR	Všetky exponáty majú modrú farbu
		2	Používateľ sa začne pohybovať.	Červená bodka sa začne na mape hýbať podľa polohy používateľa.	OK		OK	
N OT_0 01_A	Notifikovanie	1	Predpoklad: Používateľ má zobrazenú mapu s exponátmi. Používateľ sa priblíži na vzdialenosť 1,5 metre k exponátu, ktorý je označený zelenou farbou.	Zobrazí sa push notifikácia o danom exponáte: - názov exponátu - začiatok popisu o danom exponáte	OK		OK	Niektoré beacons mali pravdepodobne horší signál, lebo k nim treba prísť až moc blízko (bližšie ako 1,5 metra)
N OT_0 02_A	Notifikovanie (feedback)	1	Predpoklad: Používateľ má zobrazenú mapu s exponátmi a už bol o zelenom exponáte notifikovaný.	Zobrazí sa push notifikácia s možnosťou vyjadrenia spätnej väzby	OK		OK	Niektoré beacons mali pravdepodobne horší

			Používateľ sa vzdáľuje od zeleného notifikovaného exponátu a je vo vzdialenosti nad 2 metre.	o danom exponáte: - hviezdičky vyjadrujúce spokojnosť - tlačidlo OK pre potvrdenie			signál, lebo od nich trebalo odísť viac ako na 2 metre.
INF_001_A	Informovanie o exponáte	1	Predpoklad: Používateľ má zobrazenú základnú obrazovku o podrobnosti eventu a rozbalenú prvú kategóriu. Používateľ klikne na názov alebo fotku exponátu v zozname.	Zobrazí sa dialógové okno s detailom exponátu: - názov exponátu - fotka exponátu - popis exponátu - tlačidlo Close	OK		OK
INF_001_B	Informovanie o exponáte	1	Predpoklad: Používateľ má mapu, na ktorej sú exponáty. Používateľ klikne na jednu bodku predstavujúcu exponát (červenú alebo zelenú).	Zobrazí sa dialógové okno s detailom exponátu: - názov exponátu - fotka exponátu - popis exponátu - tlačidlo Close Detail exponátu sa zhoduje s exponátom, na ktorý používateľ klikol.	OK		OK
INF_001_C	Informovanie o exponáte	1	Predpoklad: Používateľovi prišla push notifikácia o exponáte, ku ktorému sa priblížil na vzdialenosť 1,5 metra. Používateľ otvorí push notifikáciu.	Zobrazí sa dialógové okno s detailom exponátu: - názov exponátu - fotka exponátu - popis exponátu	OK		OK

				- tlačidlo Close Detail exponátu sa zhoduje s exponátom, ku ktorému sa reálne priblížil.				
V Y H_ 00 1_ A	Vyhľad ávanie	1	Predpoklady: Aplikácia je spustená. Používateľ otvorí menu.	V menu sa používateľovi zobrazia položky: - My Events - Search Events - My Profile	O K		O K	
		2	Používateľ klikne na položku Search Events.	Zobrazí sa: - poličko Search Events, do ktorého sa dá písať - zoznam udalosti, ktoré sú posledné vyhľadané (prázdny zoznam)	O K		O K	
		3	Používateľ klikne na poličko Seach Events	Zobrazí sa klavesnica.	O K		O K	
		4	Používateľ napíše iit.	Zobrazí sa drop down menu ktoré obsahuje položky: - IIT.SRC 2016 - IIT.SRC 2017	O K		O K	
		5	Používateľ vyberie položku IIT.SRC 2016	Zobrazia sa podrobné informácie o udalosti: - názov udalosti - hlavná fotka udalosti - popis udalosti - fotky udalosti s možnosťou	O K		O K	

				horizontálneho scrollovania - kategórie patriace pod udalosť - tlačidlo naviguj				
V Y H_00 1_ B	Vyhľadávanie	1	Predpoklady: Aplikácia je spustená. Používateľ otvorí menu.	V menu sa používateľovi zobrazia položky: - My Events - Search Events - My Profile	O K			O K
		2	Používateľ klikne na položku Search Events.	Zobrazí sa: - poličko Search Events, do ktorého sa dá písať - zoznam udalostí, ktoré sú posledné vyhľadane (prázdny zoznam)	O K			O K
		3	Používateľ klikne na poličko Search Events	Zobrazí sa klavesnica.	O K			O K
		4	Používateľ napíše udalosť a stlačí Go z klavesnici.	Zobrazí sa: - poličko Search Events, do ktorého je napísaná udalosť - prázdny zoznam	O K			O K
PR OF_0 01_A	Zobrazenie My Profile	1	Predpoklady: Aplikácia je spustená. Používateľ otvorí menu.	V menu sa používateľovi zobrazia položky: - My Events - Search Events - My Profile	O K			O K

		2	Používateľ klikne na položku My Profile.	Zobrazí sa: - meno používateľa - fotka používateľa - zoznam interestov používateľa - tlačidlo Add Interest	O K		O K	
PR OF _0 _02 _A	Úprava fotky	1	Predpoklady: Je zobrazená obrazovka MyProfile. Používateľ klikne na fotku používateľa.	Zobrazia sa možnosti odkiaľ si používateľ môže nahradiť fotku: -Camera -Gallery	O K		O K	
		2	Používateľ klikne na Camera.	Zobrazí sa mu Camera - možnosť fotiť.	O K		O K	
		3	Používateľ odfotí a klikne OK.	Zobrazí sa mu MyProfile obrazovka, pričom na mieste fotky používateľa je tá fotka ktorá bola odfotená.	O K		O K	
		4	Používateľ klikne na fotku používateľa a hneď potom na Gallery.	Zobrazí sa mu galéria fotiek.	O K		O K	
		5	Používateľ vyberie niektorú fotku z galérie a klikne OK.	Zobrazí sa mu MyProfile obrazovka, pričom na mieste fotky používateľa je tá fotka ktorú vybral.	O K		O K	
PR OF _0 _03 _A	Úprava záujmu	1	Predpoklady: Je zobrazená obrazovka MyProfile. Používateľ klikne na tlačidlo Add Interest.	Zobrazí sa mu dialogové okno Add your Interest do ktorého môže napísať svoj záujem	O K		O K	

		2	Používateľ klikne na prazdne poličko.	Zobrazu sa klavesnica.	O K		O K	
		3	Používateľ napíše musiv a klikne tlačidlo Add.	Zobrazí sa MyProfile obrazovka pričom tag music je v zozname interestov.	O K		O K	
		4	Používateľ dlho pridrží tag music.	Zobrazí sa dialogové okno ktoré sa pýta či chcete aby záujem bol vymazaný s možnosťou odpovedať Ano/Nie.	O K		O K	
		5	Používateľ stlačí Yes.	Zobrazí sa MyProfile obrazovka pričom tag music nie v zozname interestov.	O K		O K	

Admin Web

Test name	Name	Step	Description (Design Steps)	Expected (Design Steps)	Status	Error Description	Status	Error Description
EV_001_A	Vytvorenie podujatia	1	Predpoklady: - Používateľ spustí aplikáciu.	Aplikácia sa spustí bez chyby.	OK		OK	
		2	Používateľovi sa zobrazí zoznam udalostí.	Zoznam udalostí obsahuje: - názov udalosti - dátum udalosti - opis udalosti a lokalitu	OK		OK	
		3	Používateľ klikne na tlačidlo Pridať	Vyplní všetky dostupné údaje	OK		OK	
		4	Používateľ klikne na tlačidlo vytvoriť	Po vytvorení klikne na tlačidlo Podujatia a zobrazí sa mu zoznam podujatí s novým podujatím	OK		FAIL	Tester zabudol zadať dátum konca podujatia
EV_001_B	Vytvorenie sekcie	1	Predpoklady: Aplikácia je spustená exituje podujatie	Používateľ klikne na detail podujatia a zobrazí sa mu detail podujatia	OK		OK	

				a zoznam sekcií				
		2	Používateľ klikne na tlačidlo Pridať sekciu	Zobrazia sa položky sekcie a používateľ ich vyplní. Po vyplnení skontroluje, či sa sekcia pridala do zoznamu	FAIL	Tester zabudol zadať notifikáciu	OK	
EV_002_A	Vytvorenie exponátu	1	Predpoklad: Existuje aspoň jedno podujatie a podujatie obsahuje aspoň jednu sekciu	Používateľ si vyberie zo zoznamu podujatí jedno a klikne na Detail. Ďalej si vyberie jednu sekciu a klikne na Detail. Používateľovi sa zobrazí detail sekcie. Ak sa v nej nachádzajú nejaké exponáty, zobrazia sa v tabuľke na dolnej časti obrazovky	OK		OK	
		2	Používateľ klikne v tabuľke	Zobrazia sa položky exponátu,	OK		OK	

			exponátov na tlačidlo Pridať	používateľ ich vyplní a klikne na tlačidlo Pridať. Používateľ skontroluje zoznam exponátov, či sa tam nový exponát nachádza				
EV_003_A	Pridanie obrázku k exponátu	1	Predpoklad: Existuje aspoň jedno podujatie a podujatie obsahuje aspoň jednu sekciu a sekcia obsahuje aspoň jeden exponát	Používateľ sa nastaví do detailu danej sekcie, v ktorej sa nachádza exponát	OK		OK	
		2	Používateľ klikne na tlačilo Vybrať súbor pri exponáte, ku ktorému chce pridať obrázok.	Používateľ vyberie obrázok, ktorý chce pridať	OK		OK	
		3	Používateľ klikne na tlačidlo Ulož poster.	Po refresh stránky sa poster objaví pri danom exponáte.	OK		OK	

EV_003_ B	Pridanie obrázku k podujati u	1	Predpokla d: Existuje aspoň jedno podujatie	Používateľ sa nastaví do detailu podujatia				
		2	Používateľ klikne na tlačilo Vybrať súbor pri nápise Pridať obrázok k podujatiu	Používateľ vyberie obrázok, ktorý chce pridať	OK		FAI L	Obrázok sa nepridal
		3	Používateľ klikne na tlačidlo Ulož obrázok podujatia	Po refresh stránky sa obrázok podujatia objaví v zozname obrázkov	OK		OK	
MAP_001 _A	Pridanie mapy k podujati u	1	Predpokla d: Existuje aspon jedno podujatie	Používateľ sa nastaví do detailu podujatia	OK		OK	
		2	Používateľ klikne na tlačilo Vybrať súbor pri nápise Mapa.	Používateľ vyberie mapu, ktorú chce pridať.	OK		OK	
		3	Používateľ klikne na tlačidlo Ulož mapu	Po refresh stránky sa mapa podujatia zobrazí v hornej časti stránky	OK		OK	

LOK_001_B	Pridanie beaconu k exponátu	1	Predpoklad: Existuje podujatie so sekciou a exponátom	Používateľ sa nastaví do detailu podujatia	OK		OK	
		2	Používateľ vyberie pri konkrétnom exponáte z comboboxu jeden beacon a klikne naň	Po refresh stránky sa pri exponáte zobrazí Minor ID príslušného beaconu	OK		OK	
NOT_001_A	Odstránenie podujatia	1	Predpoklad: Existuje podujatie	Používateľ sa nastaví do zoznamu podujatí	OK		OK	
		2	Používateľ klikne na krížik pri príslušnom podujatí	Podujatie sa odstráni	OK		OK	
NOT_002_A	Odstránenie sekcie	1	Predpoklad: Existuje podujatie so sekciou	Používateľ sa nastaví do detailu podujatia	OK		OK	
		2	Používateľ klikne na krížik pri príslušnej sekcii	Sekcia sa odstráni	OK		OK	
INF_001_A	Odstránenie exponátu	1	Predpoklad: Existuje podujatie so sekciou a exponátom	Používateľ sa nastaví do detailu príslušnej sekcie	OK		OK	
		2	Používateľ klikne na krížik pri	Exponát sa odstráni	OK		OK	

			prísušnom exponáte					
--	--	--	-----------------------	--	--	--	--	--

iOS

<i>Test name</i>	<i>Name</i>	<i>Design Steps</i>	<i>Description (Design Steps)</i>	<i>Expected (Design Steps)</i>	Status	Err or Des cription	Status	Err or Des cription
I O S0 1	Zobrazenie podujatí	1	Predpoklad: Používateľ spustí aplikáciu.	Aplikácia sa spustí bez chyby.	O K	-	O K	-
		2	Používateľovi sa zobrazí zoznam udalostí.	Zoznam udalostí obsahuje: - názov udalosti - fotka udalosti - krátky popis udalosti	O K	-	O K	-
I O S0 2	Zobrazenie podujatí	1	Predpoklad: Aplikácia je spustená.	V ovládacom paneli sa používateľovi zobrazia položky: - My Events - Search - My Profile	O K	-	O K	-
		2	Používateľ klikne na položku My Events	Zobrazí sa zoznam udalostí. Zoznam udalostí obsahuje: - názov udalosti - fotka udalosti - krátky popis udalosti	O K	-	O K	-
I O S0 3	Zobrazenie detailu podujatia	1	Predpoklad: Sú zobrazené všetky udalosti. Používateľ klikne na jednu udalosť	Zobrazia sa detail podujatia: - názov udalosti - hlavná fotka udalosti - popis udalosti - fotky udalosti s možnosťou horizontálneho prehliadania - kategórie udalosti - tlačidlo naviguj	O K	-	O K	-
		2	Používateľ klikne na kategóriu	Zobrazí sa zoznam exponátov danej kategórie. V zozname sú položky s nasledovnými údajmi:	O K	-	O K	-

			- názov exponátu - krátky popis exponátu					
I O S0 4	Výber exponát	1	Predpoklad: Používateľ má zobrazenú obrazovku detailu podujatia. Používateľ klikne na prvú kategóriu.	Zobrazí sa zoznam exponátov, ktoré patria pod prvú kategóriu.	OK	-	OK	-
		3	Používateľ klikne znova na prvú kategóriu.	Kategória sa zatvorí a exponáty nebudú zobrazené.	OK	-	OK	-
		4	Používateľ opäť klikne na prvú kategóriu.	Zobrazí sa zoznam exponátov, ktoré patria pod prvú kategóriu.	OK	-	OK	-
I O S0 5	Zobrazenie detailu exponátu	1	Predpoklad: Používateľ má zobrazený detailu podujatia. Používateľ klikne na prvú kategóriu.	Zobrazí sa zoznam exponátov, ktoré patria pod prvú kategóriu.		-		-
		2	Používateľ vyberie niektorý z exponátov	Zobrazí sa detail exponátu.	OK	-	OK	-
I O S0 6	Lokalizovanie	1	Používateľ klikne na tlačidlo Naviguj.	Zobrazí sa mapa s polohou používateľa označená červeným terčom. Zobrazia sa exponáty označené modrými terčmi.	OK	-	OK	-
		2	Používateľ sa začne pohybovať.	Červený terč sa začne pohybovať na mape podľa polohy používateľa.	OK	-	OK	-
I O S0 7	Notifikovanie	1	Predpoklad: Používateľ má zobrazenú mapu s exponátmi. Používateľ sa priblíži	Zobrazí sa detail daného exponátu ktorý obsahuje: - obrázok exponátu - názov exponátu - popis exponátu	OK	-	OK	-

			na vzdialenosť 2 metre od exponátu.					
I O S0 8	Notifikovanie (feedback)	1	<p>Predpoklad: Používateľ má zobrazenú mapu s exponátmi ktoré už navštívil.</p> <p>Používateľ sa vzdáľuje od navštíveného exponátu a je vo vzdialenosti > 2 metre.</p>	Zobrazí sa notifikácia s možnosťou vyjadrenia spätnej väzby: - oslovenie s otázkou - ukazovateľ spokojnosti (hviezdy) - tlačidlo OK pre potvrdenie	OK	-	OK	-
I O S0 9	Zobrazenie detailu exponátu	1	<p>Predpoklad: Používateľ má zobrazený detail podujatia a rozbalenú prvú kategóriu.</p> <p>Používateľ klikne na exponát v zozname.</p>	Zobrazí sa obrazovka s detailom exponátu: - názov exponátu - fotka exponátu - popis exponátu	OK	-	OK	-
I O S1 0	Zobrazenie detailu exponátu	1	<p>Predpoklad: Používateľ má otvorenú mapu s exponátmi.</p> <p>Používateľ klikne na jeden terč predstavujúci exponát.</p>	Zobrazí sa obrazovka s detailom exponátu: - názov exponátu - fotka exponátu - popis exponátu Detail exponátu sa zhoduje s exponátom, na ktorý používateľ klikol.	OK	-	OK	-
I O S1 1	Zobrazenie detailu exponátu	1	<p>Predpoklad: Používateľovi sa priblížil k exponátu na < 2m a obdržal detail exponátu.</p>	Zobrazí sa obrazovka s detailom exponátu: - názov exponátu - fotka exponátu - popis exponátu Detail exponátu sa zhoduje s exponátom, na ktorý používateľ klikol.	OK	-	OK	-

I O S1 2	Vyhľadávanie	1	Predpoklady: Aplikácia je spustená.	V ovládacom paneli používateľ vidí: - My Events Search - My Profile	O K	-	O K	-
		2	Používateľ klikne na položku Search.	Zobrazí sa: - editovateľné textové pole Search - zoznam všetkých udalostí	O K	-	O K	-
		3	Používateľ klikne na políčko Search .	Zobrazí sa klávesnica.	O K	-	O K	-
		4	Používateľ napíše reťazec „iit“.	Položky ktoré nezačínajú rovnakým reťazcom sa stratia. Zoznam obsahuje iba položku: - IIT.SRC 2017	O K	-	O K	-
		5	Používateľ vyberie položku IIT.SRC 2017	Zobrazia sa detail podujatia: - názov udalosti - hlavná fotka udalosti - popis udalosti - fotky udalosti s možnosťou horizontálneho prehliadania - kategórie udalosti - tlačidlo naviguj	O K	-	O K	-
I O S1 3	Vyhľadávanie	1	Predpoklady: Aplikácia je spustená.	V ovládacom paneli používateľ vidí: - My Events Search - My Profile	O K	-	O K	-
		2	Používateľ klikne na položku Search.	Zobrazí sa: - editovateľné textové pole Search - zoznam všetkých udalostí	O K	-	O K	-
		3	Používateľ klikne na políčko Search.	Zobrazí sa klávesnica.	O K	-	O K	-
		4	Používateľ napíše reťazec „blabla“.	Položky ktoré nezačínajú rovnakým reťazcom sa	O K	-	O K	-

				stratia. Zoznam neobsahuje žiadnu položku.				
I O S1 4	Zobrazenie My Profile	1	Predpoklady: Aplikácia je spustená.	V ovládacom paneli používateľ vidí: - My Events - Search - My Profile	O K	-	O K	-
		2	Používateľ klikne na položku My Profile.	Zobrazí sa: - meno používateľa - fotka používateľa - zoznam záujmov používateľa - tlačidlo Add Interest	O K	-	O K	-
I O S1 5	Úprava fotky	1	Predpoklady: Je zobrazená obrazovka My Profile. Používateľ klikne na fotku používateľa.	Zobrazí sa galéria fotografií.	O K	-	O K	-
		2	Používateľ vyberie niektorú fotku z galérie.	Zobrazí sa mu obrazovka My Profile s novou fotografiou.	O K	-	O K	-
I O S1 6	Úprava záujmov	1	Predpoklady: Je zobrazená obrazovka My Profile. Používateľ klikne na tlačidlo Add Interest.	Zobrazí sa mu dialógové okno Add Interest do ktorého môže napísať svoj záujem	O K	-	O K	-
		2	Používateľ klikne na textové pole.	Zobrazí sa klávesnica.	O K	-	O K	-
		3	Používateľ napíše reťazec „cars“ a klikne tlačidlo Add.	Zobrazí sa obrazovka My Profile s pridaným záujmom v zozname záujmov.	O K	-	O K	-
		4	Používateľ vyberie tlačidlo Edit.	Zobrazia sa krížiky pri každom záujme v zozname.	O K	-	O K	-

		5	Používateľ klikne na krížik pre záujem „cars“.	Zobrazí sa My Profile obrazovka so zoznamom záujmov bez záujmu „cars“. Krížiky pri záujmoch v zozname sú stále prítomné.	OK	-	OK	-
--	--	---	--	--	-----------	---	-----------	---

Príloha B: Inštalačná príručka

Android

1. Nainštalovať Android Studio (<https://developer.android.com/studio/index.html>).
2. Na zariadení povoliť inštaláciu z neznámych zdrojov (*Settings – Security - Unknown sources*).
3. Otvoriť projekt v Android Studiu.
4. V menu vybrať možnosť Run.
5. Otvorí sa dialógové okno s výberom zariadenia, vybrať správne zariadenie.
6. Po potvrdení dialógu sa aplikácia nainštaluje do zariadenia a je automaticky spustená.
7. Pre správnu funkčnosť aplikácie je nutné mať zapnutý Bluetooth. Pre Android 6+ je ešte nutné povoliť všetky práva aplikácie (*App info – Permissions*).

iOS

1. MacOS 10.12 +, Xcode 8 + **required** for Alamofire 4.0
2. [Install Cocoapods](#)
3. Ignore Xcodes dialog to convert the sources to Swift 3, resolve issues like this:
 - Open "**Build Settings**" for the "**Pods**" project
 - Search for "**Legacy Swift Language Version**"
 - Change the value for the "**Use Legacy Swift Language Vrsion**" to "**No**"

Admin Web

- Uistite sa, že máte nainštalovaný bower, grunt-cli a npm globálne
- `$ sudo apt-get install npm`
- `$ sudo npm install -g grunt-cli`
- `$ sudo npm install -g bower`
- `$ cd `project-directory``
- bower install is ran from the postinstall
- `$ npm install`
- a shortcut for grunt serve
- `$ npm start`
- `$ npm run dist`

Server

1. instalacia ubuntu server:

<http://www.tecmint.com/ubuntu-14-04-server-installation-guide-and-lamp-setup/>

2. instalacia LAMP:

<https://www.unixmen.com/how-to-install-lamp-stack-on-ubuntu-16-04/>

3. zdrojový kód vložíme do priečinka `/var/www/html/`