Sem vložte zadání Vaší práce.

České vysoké učení technické v Praze
Fakulta informačních technologií
Katedra softwarového inženýrství

Diplomová práce

Hra pro mobilní zařízení s rozšířenou realitou a podporou hry více hráčů

Bc. Jakub Baierl

Vedoucí práce: Ing. Dominik Veselý

23. června 2015

Poděkování

V této sekci bych rád poděkoval Ing. Dominiku Veselému za dozor nad touto prací, skvělé a bezproblémové vedení a za pomoc v těžších místech této práce i samotné aplikace. Dále bych rád poděkoval Ing. Martinu Kačerovi, Ph.D. za ujetí se oponentury, Ing. Josefu Gattermayerovi za připomínky a rady týkající se textové části, a v neposlední řadě také mé babičcce za korekturu celé práce. Dále bych také rád poděkoval firmě Ackee s. r. o. za poskytnutí zázemí při vytváření aplikace, konkrétně poskytnutí developerských účtů a dalších nutností pro testování aplikace. Nakonec bych rád poděkoval celé své rodině a přítelkyni za podporu po celou dobu studia.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou, a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu), licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 23. června 2015 .....

Odkaz na tuto práci

Baierl, Jakub. *Hra pro mobilní zařízení s rozšířenou realitou a podporou hry více hráčů.* Diplomová práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2015.

Abstrakt

Tato práce si klade za cíl provést důkladnou analýzu prostředí a technologií pro vytvoření mobilní aplikace s podporou hry více hráčů, využívající rozšířenou realitu. Dále tato práce bude obsahovat návrh a samotnou realizaci této aplikace a také webového serveru s REST API, pro ukládání všech informací a statistik o hrách a uživatelích. Práce má také za cíl analyzovat a navrhnout jednoduchý responsivní web, který bude využívat nejnovější technologie jako například nástroje Gulp, Grunt, Bower, NPM či ReactJS, kteté budou také podrobeny analýze.

Klíčová slova iOS, hra, mobilní aplikace, multiplayer, Frontend, Backend, REST API, detekce obličeje, webové technologie

Abstract

Conclusion of this thesis is making thorough analysis of enviroment and technologies for creating mobile app supporting multiplayer connectivity and using augumented reality. This thesis also contains concept and implementation of mobile app and Backend server with REST API support for saving all information and statistic about games and users. Next point of this thesis is to

analyze and realize responsive web which is using some modern technologies and tools like Gulp, Grunt, Bower, NPM or ReactJS that will be analyze too.

Keywords iOS, game, mobile app, multiplayer, Frontend, Backend, REST API, face detection, web technologies

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Fakulta informačních technologií má již mnoho studentů, a tedy mnoho rozličných témat. Avšak herní průmysl má mezi závěrečnými prácemi velmi malé zastoupení. Proto jsem velice rád, že mám tu možnost toto procento podpořit a vytvořit práci jak zábavnou, tak i poučnou. Tato práce začala vznikat během předmětu BI-IOS, vedeným Ing. Dominikem Veselým, kdy jsem se rozhodl tento nápad realizovat jako semestrální práci. Tehdy práce obsahovala spíše nápad než řádnou realizaci, ale i tak velice kladně zapůsobila na školní komisi [1] s následnou nabídkou, jestli bych nechtěl toto téma zpracovat jako diplomovou práci. Práce má obsahovat analýzu nástrojů pro tvoření her na mobilním telefonu s rozšířenou realitou a podporou více hráčů a následnou realizaci a otestování. Dále se práce bude věnovat i webovým technologiím, konkrétně bude obsahovat webovou aplikaci tvořenou podle nových trendů s využitím nejnovějších technologií.

Kapitola 1

Cíl práce

Tato práce má jako hlavní cíl přiblížit zejména problém rozšířené reality a na konkrétním případě ukázat jedno z mnoha jejích použití. Práce bude obsahovat analýzu daného problému a následnou implementaci. Tato práce by měla sloužit i jako pomocný materiál pro další potenciální studenty, kteří se tímto tématem budou chtít také zabývat. Práce bude dále poskytovat informace o návrhu uživatelského prostředí aplikace a webového serveru, který bude sloužit k ukládání výsledků a dalších informacích o uživatelích a samotných hrách. Dále se čtenář může dozvědět plno informacích o nejnovějších technologiích pro vytváření webové aplikace, včetně balíčkovacích systémů Bower, Grunt a NPM, které běží na poměrně mladém programovacím a skriptovacím jazyce NodeJS.

Kapitola 2

O rozšířené realitě (Augumented Reality)

Název rozšířená realita se používá pro rozšíření obrazu reálného světa o digitální objekty, popisky, interakce a dalších prostřednictvím fotoaparátu daného zařízení.

2.1 Jak AR funguje

Kamera zařízení snímá obraz reálného světa, detekuje umístění a orientaci v prostoru. Po vyhodnocení detekce renderuje další digitální audio či video informace. Toto vše se samozřejmě děje v reálném čase, což přidává na skutečnosti celé scény. U mobilních zařízení se spolu s AR využívá často GPS, která detekuje polohu uživatele, a pak například vyhodnocuje, kdy se co má provést nebo objevit na displeji [2].

2.2 Využití

Rozšířená realita je poměrně nová technologie, tudíž se očekává, že se bude čím dál tím více rozšiřovat a bude vznikat více a více aplikací, které ji používají. Rozšířená realita se používá v dnešní době v mnoha oborech. Například v oboru vzdělávání je její použití velmi zajímavé, kdy učebnice mohou být "vylepšeny"o prvky, které jsou vidět pouze přes čočku fotoaparátu. Dále se AR silně využívá v reklamě a marketingu, kdy jsou různé markery (speciální obrázky, které aplikace umí rozpoznat) otištěny v tištěných produktech a například uživatel jejich rozpoznáním může docílit nějaké výhody. Samozřejmě velké využití je i v herním průmyslu, kde můžeme docílit například přidání 3D objektů na různá místa, či v tomto případě vytvoření real-time RPG střílečky.

Kapitola 3

Analýza

V této kapitole je popsána podrobná analýza stávajících aplikací, jsou zde podrobně analyzovány použité nástroje a technologie jak pro mobilní tak i pro webovou aplikaci. Podrobněji rozebrány jsou i uživatelské role a různé případy užití. Analýza byla provedena i skrz verzovací nástroje pro možnost backtrackingu či týmového rozšíření a nástroje pro testování API modulu.

3.1 Rešerše existujících produktů

3.1.1 AR Missile - Automatic Target Tracking

Tato hra dokonale používá rozšířenou realitu ve světě mobilních her, protože zdatně spojuje skutečný svět s 3D grafikou. V této hře můžete ničit cokoliv, co si vymyslíte, a na co následně namíříte. Hra pak obstará efekt výbuchu či zničení zadané budovy, či místa [3].

3.1.2 AR Invaders

AR Invaders je hra, která vás donutí postavit se za planetu Zemi a bojovat proti jiné civilizaci, která se snaží naší planetu okupovat. Střílečka, která využívá rozšířenou realitu pro zobrazení cizích předmětů (létající talíře, UFO rakety), které musíte zničit [4].

3.1.3 Real Strike

Další zkoumanou hrou je Real Strike. Herní prostředí je velmi dobře zpracované. Obsahuje několik různých zbraní i brnění a poskytuje několik různých typů vidění, jako termální vidění, přiblížené vidění a další. Bohužel tato hra nabízí jen hru pro jednoho hráče, což po nějakém čase, kdy vyzkoušíte vše, co hra nabízí, přestane být tolik zábavné [5].

3.1.4 Laser Tag!!!

Hra Laser Tag!!! je zatím asi nejpodobnější aplikaci, o které tato práce pojednává. Používá rozšířenou realitu pro nasimulování reálného hraní známe hry Laser Tag. Tuto hru může také hrát i více hráčů. Hra používá pro detekci interní svítilnu zařízení. Bez toho bohužel hra nefunguje, což není úplně optimální, vzhledem k velkému poklesu baterie spolu s používáním zmiňované svítilny [6].

3.2 Výběr platformy

Vzhledem k tomu, že práce vznikala při programování aplikací pro operační systém iOS, rozhodl jsem se pro programovací jazyk obj-c, pro který se zároveň i ukázalo, že existuje nejvíce volně dostupných technologií. Dále i kvůli dlouhodobějšímu zájmu o vytvoření aplikace pro systém iOS. Výše zmíněné hry jsou preferované také pro tento systém, avšak několik z nich je poskytováno i pro operační systém Android. Musím v této záležitosti být nestranný, a tak při rozhodování rozhodovaly i nástroje pro tvorbu her na obou platformách. Nejlepším herním frameworkem pro systém Android platformu je v současné době Xposed framework, který je ale bohužel dostupný až pro Android verze 5 a požaduje vysoké technické dovednosti zařízení. Navíc je tento framework prozatím velice mladý a autoři varují, že při použití může aplikace často vyvolat restart zařízení, či může zůstat viset na logu (tzv. BootLoop). Při další analýze byla objevena služba Play Games Services for Android, který poskytuje skvělé prostředí pro tvorbu multiplayerových her. Doporučuji tento oficiální článek společnosti Google Inc. o tvorbě aplikace pomocí této služby [7]. Pro systém iOS vývojáři vyvinuli skvělý herní framework GameKit [8], se kterým je vyvíjení daleko jednodušší a zábavnější. Společně s využitím GameCenter služby [9] bude aplikace rychleji hotova, a i když se budete snažit sebe víc, lepší systém LeaderBoardu a Achievementů za takto krátký časový úsek, který je vymezen pro Diplomovou práci nejspíše nenapíšete, pokud ovšem nejste potomek Steva Wozniaka.

3.3 Výběr technologie

Tato sekce popisuje analýzu výběru technologie pro jednotlivé části aplikace. Bylo nutné vybrat technologie pro spojení a komunikaci jednotlivých zařízení a s tím spojenou technologii přenosu informací. Pak bylo nutné vybrat technologii pro vytvoření serveru s podporou REST API, pro napojení se z aplikace, snadné posílání dat a následné uchovávání dat v databázi. Nakonec se musela vybrat technologie pro vytvoření webové aplikace. Jelikož tato práce spadá pod obor Webové inženýrství, bylo nutné analyzovat nejnovější technologie a vyzkoušet jejich použití.

3.3.1 Nástroje pro komunikaci mezi telefony

3.3.1.1 Multiplayer komunikace

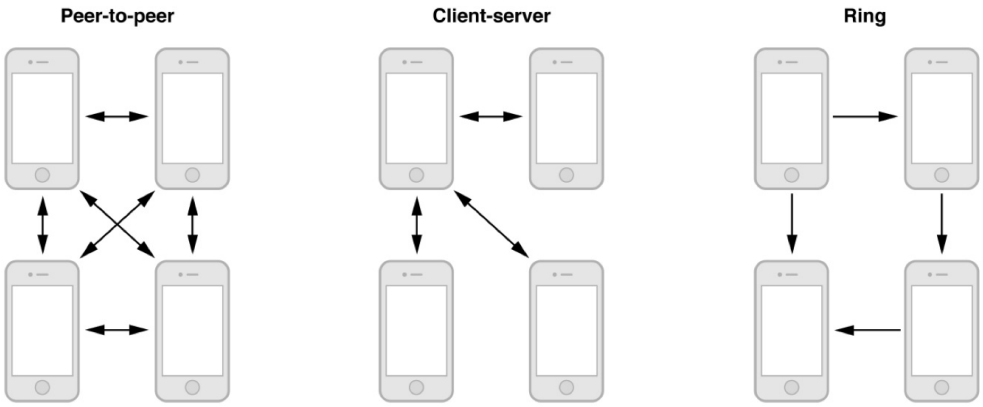
Komunikace v multiplayer hrách je většinou postavena na transportní vrstvě ISO/OSI modelu. Zde se nacházejí dva různé protokoly, TCP, což je spolehlivý protokol pro komunikaci a výměnu dat, protože se můžeme spolehnout, že data budou doručena v pořadí, ve kterém se pošlou. Zároveň tento protokol může být při špatném použití hodně pomalý. Druhý protokol UDP je naopak nespolehlivý, ale nabízí dobré prostředí pro výměnu dat s menším overheadem než TCP, tudíž rychlost přenosu bývá daleko rychlejší [10].

3.3.1.2 Problémy

Největší problém multiplayerových her je synchronizace několika stejných herních instancí. Kvůli nespolehlivosti sítí a občasné latenci je nutné velice dobře rozmyslet návrh a implementaci aplikace. Zaprvé je důležité zajistit všechny případy přerušení, protože je veliká šance, že nastanou, zvlášť u mobilního připojení. Zadruhé je velmi důležité dobře rozmyslet, jak často a v jaké struktuře posílat data na ostatní zařízení. Rozhodně není doporučeno posílat data vícekrát za sebou. Dále by se mělo docílit co nejmenší velikosti posílaného balíčku, proto je nutné rozmyslet jaké datové typy posílat.

3.3.1.3 Topologie propojení

Existuje více druhů topologií. Pro tento připad použijeme topologii Peer-to-Peer, která má sice největší síťový provoz, ale je velmi jednoduchá na implementaci . Další topologie jako Client-server či Ring ušetří síťovou komunikaci, avšak na implementaci jsou o dost složitější. Na obrázku 3.1 si můžete prohlédnout další typy topologií. Podrobnější rozdíly topologií vizte [11].



Obrázek 3.1: Nejpoužívanější typy topologií

3.3.1.4 Websockets

Tento nástroj je v současné době stále více používaný, jelikož přibývá více a více real-time aplikací. Websockets slouží k přímé real-time komunikaci browseru a iOS či Android clientů. Největší výhodou tohoto nástroje je využití stejného backendu pro všechny mobilní platformy. API je zde velmi jednoduše použitelné a navíc je zde velmi snadné přidat TLS šifrování, které se pro většinu real-time aplikací určitě hodí. V případě zájmu o použití Websocket doporučuji článek [12], kde je návrženo jednoduché spojení browseru skrze Javascript s mobilním clientem s využitím serveru psaného v jazyce Ruby. Návrh byl z menší části upraven a otestován. Dále je obecněji návrženo použití Websockets v konkrétní aplikaci.

Server
Server může být implementován v různých jazycích, zde konkrétně byla otestována implementace v jazyce Ruby s využitím gemu EM-Websocket. Jednoduchý server definuje prostředí, umožňující připojení se klientů do stejného kanálu, kam se následně přidávají i data, které někdo z klientů pošle. Následně se data pošlou všem klientům v kanálu. Když se klient odhlásí, je z kanálu smazán.

Browser
Je nutné připravit prostředí pro připojení se na server na konkrétním portu a následně naimplementovat callback funkce pro události navázání spojení, přijmutí zprávy a uzavření spojení. Dále je potřeba naimplementovat handlery pro poslání zprávy například webovým formulářem.

iOS Client
Nejvhodnější implementací je využití websocket clienta SocketRocket, jednoduše instalovatelného přes knihovnu Cocoapods, pomocí kterého se dá jednodušše doimplementovat SRWebSocketDelegate protokol. Ve výše zmiňovaném článku se můžete dočíst, které metody je nutné naimplementovat pro bezproblémový chod spojení a ošetření všech stavů. Implementace Android clienta je velmi podobná až na syntax jazyka, ve kterém je klient tvořen.

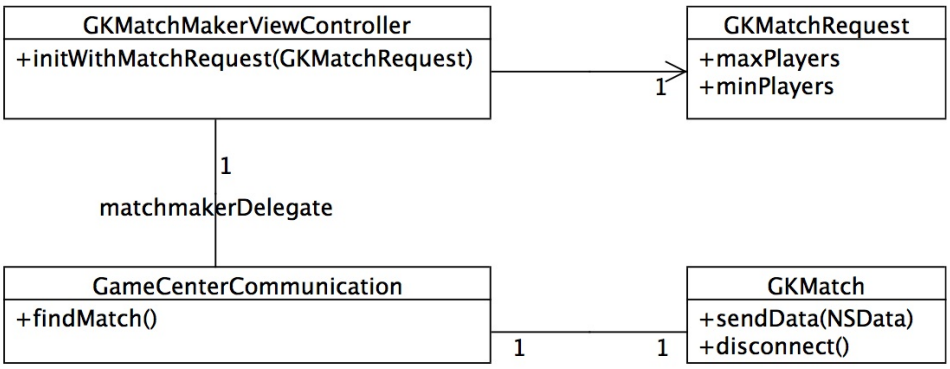
3.3.1.5 MultipeerConnectivity

MultipeerConnectivity je framework fungující od iOS 7. Nabízí použití pro odhalení zařízení v blízkosti a výměnu dat skrz Bluetooth či Wifi. Velkou výhodou je nezávislost na komunikační infrastruktuře, což znamená, že se uživatel nemusí starat o použitém Bluetooth ani Wifi v současně používané session. Navíc MultipeerConnectivity poskytuje podporu pro výměnu zpráv i dokonce streamovaných dat. Další výhodou je, že vývojáři odpadá starost o složité propojování zařízení přes transportní vrstvu, což je popsáno výše, protože to už

udělá framework za něj. Pro více informací o použití MultipeerConnectivity vizte [13]

3.3.1.6 Game Center

GameCenter [9] je oficiální služba vyrobená společností Apple Inc. Poskytuje herní prostředí pro pozvání přátel, vyzývání přátel ke hře, vytváření autozápasů, sbírání bodů, přidávání achievementů a zobrazování skóre v žebříčkách. Je použitelná jak pro zařízení iPhone tak i pro zařízení iPad a od verze OS X Mountain Lion je možné propojení systémů Mac a iOS. Vznikla společně s verzí iOS 4.1 a je součástí GameKit frameworku [8]. Dovoluje vývojářům ukládat základní informace o uživatelích jako Nickname, PlayerID, Body a Profil, což velice ulehčuje práci. Jsou zde tři hlavní třídy, která musí vývojář znát, aby dokázal vytvořit funkční prostředí. Jsou to třídy GKMatchRequest, která obsahuje důležité informace o tom, jak zápas má vypadat a kolik má obsahovat hráčů (min a maxPlayers). Dále třída GKMatchmakerViewController, která zobrazuje standardní ViewController pro hledání a pozvání dalších hráčů a následné hledání zápasů. Samozřejmě je možné tento ViewController vyměnit za svůj, pokud by standardní nevyhovoval. Třetí třídou je GKMatch, která poskytuje peer-to-peer síťové spojení mezi skupinou hráčů. Také představuje právě probíhající zápas a drží všechny informace o něm i o hráčích, kteří v něm účinkují a poskytuje i prostředí pro posílání a přijímání dat i hlasu mezi hráčemi. Na následujícím UML diagramu 3.2 je zobrazena struktura a interakce mezi těmito třídami. Třída GameCenterCommunication má referenci na GKMatchmakerViewController a GKMatch a ty jí pak delegují vzniklé události. Více o tvorbě aplikací s využitím služby GameCenter vizte [14].



Obrázek 3.2: GameCenter UML diagram tříd.

3.3.2 Server s REST API

Nutnou součástí této práce je vytvoření serveru s REST API, pro uchovávání výsledků, statistik a veškerých informací o uživatelích a hrách. Rozhodl jsem se použít jazyk PHP společně s Nette frameworkem a jako úložiště dat jsem vybral MySQL databázi.

3.3.2.1 Nette Framework

Pro vytvoření serveru jsem se rozhodl použít Nette framework, což je open-source framework pro vytváření webových aplikací v PHP 5. Nette framework je v dnešní době nejpopulárnější framework u nás hlavně kvůli podpoře AJAXu [15], nabídky mnoha ladících nástrojů, možnosti použití Autoloaderu a mnoha dalších. Tento framework používá softwareovou architekturu MVC, která se skládá z modelů, přes které se přistupuje k jednotlivým entitám z databáze, presenterů, ve kterých se řídí veškerá logika aplikace a views, které řídí uživatelské rozhraní. Velkou výhodou je zde i rozlišení vývojového a produkčního režimu, což vývojářům ušetří mnoho času. Jelikož je tento framework velice populární, vznikají skoro každý den nové a nové komponenty pro snadnější práci s jinými používanými systémy (GoogleMaps, Paypal, Facebook ...). Dále je zde skvělá podpora pro vícejazyčnost aplikace a coolURL, které slouží k libovolnému nastavování konečných URL, což je zaprvé pro oko uživatele daleko přívětivější, ale navíc je to velmi důležité pro SEO [16].

3.3.2.2 MySQL Databáze

MySQL je multiplatformní databáze, která komunikuje přes jazyk SQL. Je snadno implementovatelná do všech systémů a nejčastěji se používá společně s PHP a Apache. Další důvody, proč je dobré použít tento typ databáze zde [17].

3.3.3 Webová aplikace

Prezentační část této aplikace jsem se rozhodl vytvořit taktéž v Nette frameworku. Webová část aplikace se tedy bude skládat z BackModulu (backendu), který bude mít na starost ukládání všech informací a statistik, poskytování REST API a FrontModulu (frontendu), který bude všechny informace prezentovat uživatelům. Frontend [18] aplikace bude vytvořen v technologiích HTML5, CSS3 s využitím AJAXU a JavaScriptu. Webová aplikace bude responsivní s využitím Bootstrapu a CSS preprocesoru. Tyto technologie budou dále popsány jen stručně, protože většina těchto technologií je velmi známá.

3.3.3.1 HTML5

HTML neboli Hyper Text MarkUp Language je označení pro značkovací jazyk, pomocí kterého se vytvářejí webové stránky. HTML5 se od staršího HTML

liší změnou struktury emelentů pomocí nových tagů **<section>, <article>, <main>, <header>, <footer>, <nav>** a plno dalších. Dále HTML5 umožňuje ukládání dat na lokální úložiště, tudíž je uživatel schopen prohlížet webové stránky i bez připojení k internetu. Používá dále i novou specifikaci dokumentu nazývanou DOCTYPE, která se nově zapisuje jako **<!DOCTYPE html>**. Dále má nově podporu multimediálního obsahu pomocí tagů **<video>, <audio>** a dalších. Výborná je i nová podpora vytváření a používání SVG objektů pomocí Canvasu [19].

3.3.3.2 CSS3

Kaskádové styly je jazyk, pomocí kterého můžeme stylovat veškeré elementy vytvořené jazykem HTML či XHTML. Již byly vytvořeny tři verze CSS a i poslední verze má v dnešní době téměř 100 % podporu ve všech webových prohlížečích. Kaskádové styly se skládají z definovaných pravidel, které obsahují velmi známou dvojici **selektor** a **seznam deklarací**. Nejnovější verze CSS3 se liší od předchozích podporou animací, zaoblenými tvary, 3D transformacemi či novými flexibilními bloky [20].

3.3.3.3 JavaScript

JavaScript je skriptovací jazyk, používaný při vytváření webových stránek. Stará se o obsluhu události například klikání elementů atp. Řadí se mezi objektově orientované jazyky podobně jako jazyky C++ či PHP. Používá dynamické přiřazování typů, tudíž stejné proměnné mohou obsahovat čísla i stringy, samozřejmě ale v různém čase. Dále podporuje možnost vytváření prototypů, které zde zastávají funkci tříd. V dnešní době jeho podpora ve webových prohlížečích je téměř 100 %, a tak se vývojáři nemusí obávat, že stránky nebudou fungovat.

3.3.3.4 Responzivní web design

Tato technika je postavená na způsobu stylování stránky tak, aby zobrazení stránky bylo vždy optimalizováno pro všechny druhy zařízení a rozlišení. Ve specifikaci CSS3 vznikla nová vlastnost Media Queries, která dokáže rozpoznat vlastnosti a rozlišení obrazovky daného zařízení a následně přizpůsobit danou stránku [21]. Pro plně responsivní web však pouze tato technika nestačí, jelikož by se muselo použít Media Queries na každé rozlišení a to je skoro nemožné. Proto se v současnosti nejvíce používá mix použití Media Queries a flexibilní struktury a obrázků. Tato technika je postavena na procentuálních šířkách, což zaručí optimalizaci obsahu. Tudíž Media Queries mají za úkol přizpůsobit strukturu elementů na různých typech zařízení, většinou pro stolní počítač, tablet a telefon. A optimalizaci pro rozlišení mezi těmito zařízeními zajistí právě flexibilní struktura.

3.3.3.5 Fluid grids

Populární technika je využití fluid grids systému. Je to technika, která obsah stránky formuje do sloupců, které pak rozmisťuje či jinak mění podle šířky obrazovky. Většinou se používá technika 12 sloupců, které se do sebe mohou i jakkoliv vnořovat [22].

3.3.3.6 Bootstrap

Rozhodl jsem se pro Frontend využít Twitter Bootstrap, jelikož tento volně dostupný balíček obsahuje všechny požadované nástroje pro vytváření moderního webu. Poskytuje podporu Technologií HTML, CSS, Javascriptu a dalších prvků, které jsou snadno použitelné na webu. Tento balíček poskytuje i podporu pro výše zmíněnou technologii fluid grids i media queries, tudíž responsivní web design se spolu s tímto nástrojem vytváří o mnoho jednodušeji. Dále tento balíček obsahuje nesčetné množství komponent, konkrétně různých předpřipravených tlačítek, formulářů, modálních oken, tabulek, ikonek či mnoho Javascriptových komponent jako dropdowny, popovery, alerty či různé obrázkové slideshow.

3.3.3.7 CSS preprocesor

CSS preprocesor je nástroj, který generuje optimalizované CSS soubory ze zdrojových kódů psaných volnější syntaxí. Velkou výhodou je možnost použití proměnných pro barvy či jakékoliv další vlastnosti a také použití tzv. mixinů [23]. Ty se hodí pro pro použití stejných vlastností u různých elementů vizte:

```
@mixin table-base {
  th {
    text-align: center;
    font-weight: bold;
  }
  td, th {padding: 2px}
}

#data {
  @include table-base;
}
```

Pro snadnější stylování a snadnější použití výše zmíněných nástrojů a Media Queries jsem se rozhodl využít nástroj SaSS, což je jeden ze tří nejznámějších CSS preprocesorů. Dalšími jsou LeSS a Stylus.

3.3.3.8 Další nástroje pro Frontend

Pro snadnější práci s různými nástroji a externími knihovnami je v dnešní době populární používat balíčkovací systémy pro Frontend. Konkrétně nástroje

Bower, GruntJS, Gulp či NPM. Použitím těchto nástrojů ulehčíme práci jak sobě tak hlavně dalším vývojářům se kterými spolupracujete. V současnosti je používání externích knihoven nezbytnou součástí Frontendu, a tudíž každý Frontend vývojář ztratí plno času s hledáním a includováním těchto knihoven. Právě tyto úkony za vás tyto nástroje zpracují. V další kapitole bude vysvětleno a ukázáno konkrétně jejich použití. Já jsem se rozhodl pro použití NPM, Boweru a Gruntu a tato kombinace bude pak přesněji popsána dále. Podrobnější analýza moderních nástrojů pro tvorbu Frontendu se vysktuje zde [24]

3.4 Uživatelské role + Use Cases

V této sekci je důležité vymezit všechny uživatelské role, které se v aplikaci mohou vyskytovat. Toto pomůže k analýze a návrhu práv na různé akce.

3.4.1 Mobilní aplikace

V mobilní aplikaci se budou vyskytovat jen dva typy uživatelů. Diagram si můžete prohlédnout na obrázku 3.3.

3.4.1.1 Přihlášený uživatel

Tuto roli dostane uživatel, který se po spuštění aplikace přihlásí do svého GameCenter účtu. S touto rolí má uživatel možnost hrát Single i Multiplayerovou hru a navíc všechny výsledky se ukládají na webovém serveru.

3.4.1.2 Nepřihlášený uživatel

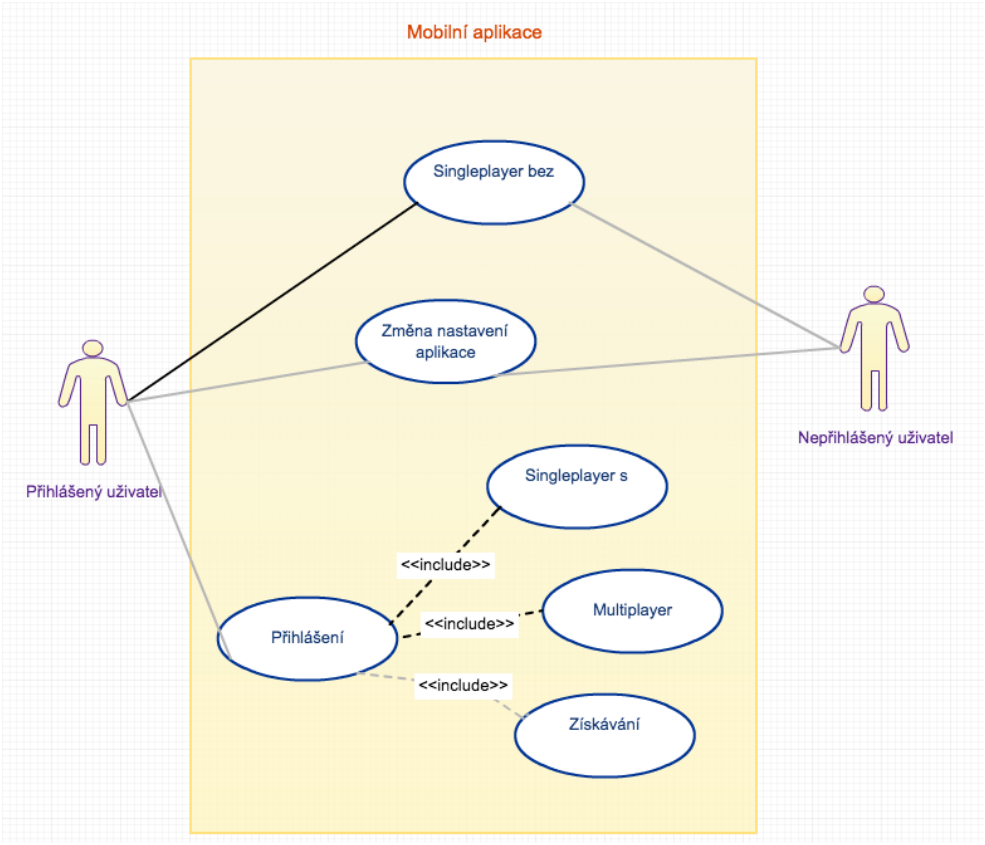
Tato role je přiřazena uživatelům, kteří ještě nemají GameCenter účet. Nemohou hrát Multiplayer hru, ale nemůžou být ochuzeni o Singleplayerovou hru, a tudíž mají možnost hru vyzkoušet se všemi výhodami, avšak výsledky se neposílají na server, protože hráč nemá žádnou identifikaci.

3.4.2 Webová aplikace

Na webu se budou vyskytovat prozatím také dva různé typy uživatelů. Do budoucna ještě bude nejméně jeden typ, a to Editor. Diagram si můžete prohlédnout na obrázku 3.4.

3.4.2.1 Nepřihlášený uživatel

Webová aplikace bude mít přihlašování pouze pro Adminy a Editory, tudíž role přihlášeného uživatele zde nebude. Nepřihlášení užvatelé budou mít však možnost listovat všemi záznamy o ostatních uživatelích či prohlížet statistiky



Obrázek 3.3: Diagram uživatelských rolí mobilní aplikace

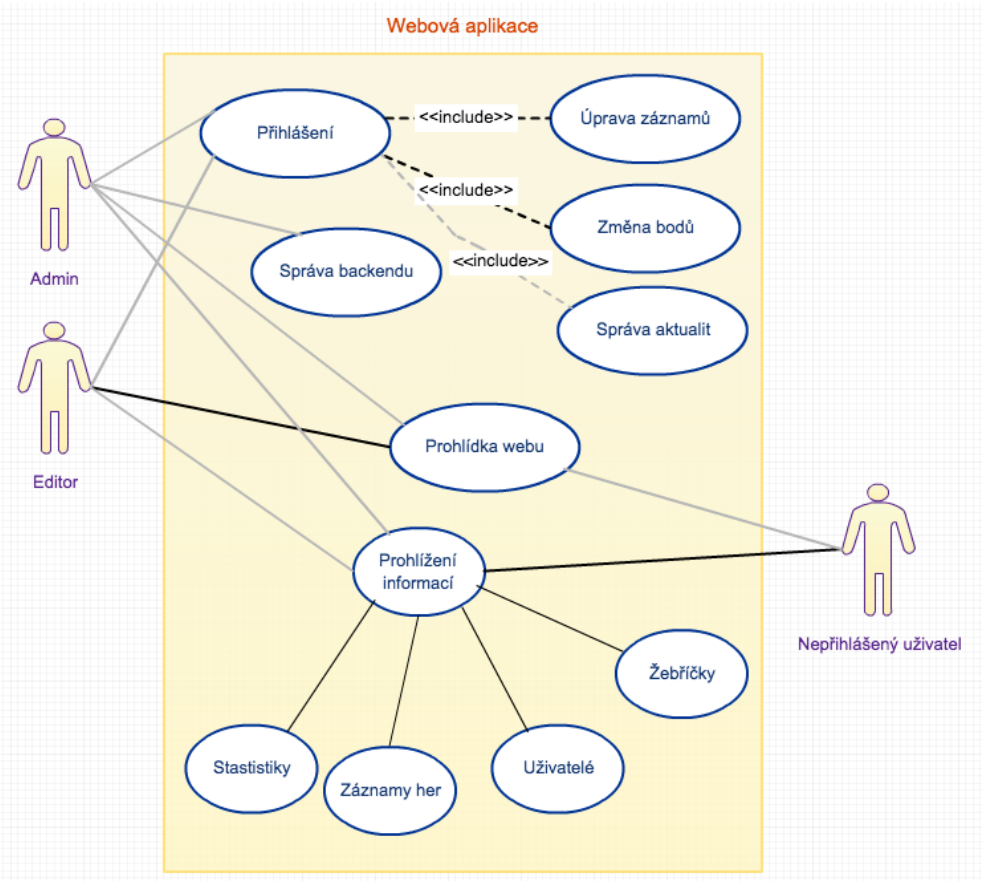
všech her. Bude zde i možnost vyhledat svůj profil a své statistiky, ale to pouze po zadání svého PlayerID spojeného s GameCenterem.

3.4.2.2 Admin

Admin bude mít na starost správu webu, statistik a informací o hrách. Bude mít i na starost správu serveru a celého backendu. Bude mít možnost uživatelům měnit práva či bodové ohodnocení či mazat libovolné záznamy.

3.4.2.3 Editor

Role editora bude sloužit k jednodušším úpravám na webu. Doplňování aktualit, novinek či změna bodových ohodnocení obyčejných uživatelů.



Obrázek 3.4: Diagram uživatelských rolí webové aplikace

3.5 Seskupený task list

V této sekci bych chtěl zmínit seznam nejzajímavějších a nejdůležitějších úkonů, které je možno napříč aplikací provádět. Pro zjednodušenou prezentaci dat jsem je rozdělil do dvou sekcí a navíc jsem zvolil úkony, řazené do skupin, aby bylo patrné, co se v jaké části může dělat.

V této tabulce 3.1 můžete vidět hlavní úkony, které je možné provádět v mobilní aplikaci.

A na této tabulce 3.2 můžete vidět hlavní úkony, které je možné provádět ve webové aplikaci.

Task list mobilní aplikace			
Menu	Nastavení	Singleplayer	Multiplayer
Přihlášení do GameCenteru	Odhlásit se	Míření	Poslat zprávu
Výběr hry	Zap./Vyp. zvuky	Střílení	Kontrola pořadí
Hledání hráčů	Zap./Vyp. hudbu	Přidělení bodů	
Výzva hráčů	Prohlídka leaderboardu	Výběr zbraně	

Tabulka 3.1: Hlavní úkony prováděné v mobilní aplikaci

Task list webové aplikace		
Landing page	Profil	Žebříčky a Informace
Úprava aktualit	Prohlídka profilu	Vyhledávání uživatelů
Prohlídka informací	Změna bodů	Změna řazení
Přihlášení	Změna údajů	Detail záznamu
Odhlášení		

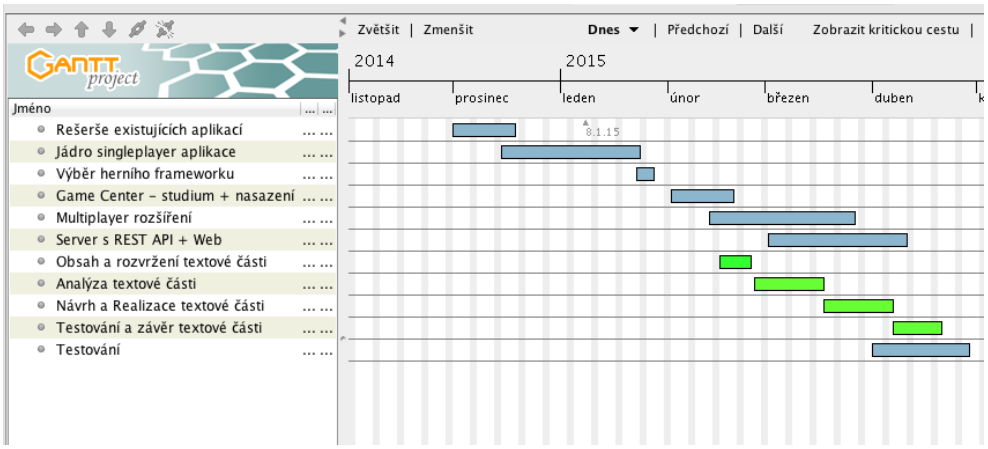
Tabulka 3.2: Hlavní úkony prováděné ve webové aplikaci

3.6 Harmonogram projektu

U každého většího projektu, diplomovou práci nevyjímaje, je velmi důležité před zahájením práce vytvořit harmonogram, který slouží ke snadnějšímu dodržení termínů rozdělením práce na menší úseky. Na následujícím grafu 3.5 je vidět harmonogram této diplomové práce, podle kterého jsem celou práci tvořil.

3.7 Verzovací nástroje

Jelikož chci tuto práci dále rozvíjet, je možné, že se zapojí i více lidí, a proto je nutné od začátku používat nějaký verzovací nástroj. Není podmínkou pouze využití v týmu, je to velmi výhodné i pro projekt o jednom člověku, díky uchování všech možných verzí projektu. Dva nejpoužívanější nástroje jsou Git a SVN, tudíž jsem nad nim provedl analýzu, která bude popsána dále, a nakonec si vybral nástroj Git, jehož výhody vysoko převyšují druhý zmíněný nástroj SVN.



Obrázek 3.5: Harmonogram projektu.

3.7.1 Rozdíly nástrojů

Hlavním problémem nástroje SVN je nutnost připojení k internetu kdykoliv, když je potřeba něco "commitnout". S Gitem je možné commitovat i mergovat, aniž byste museli být připojeni k internetu. Dále velká výhoda je vytváření různých větví tzv. branchí, což jsou vlastně různé cesty jak od jednoho bodu můžeme pokračovat. Následně právě tyto větve můžeme mezi sebou spojovat, čemuž se právě říká mergovat. Nástroj SVN nám dovoluje tedy ukládat svou práci v různých verzích, což je určitě lepší než si vždy před větší změnou uložit práci v nějakém komprimovaném formátu s unikátním timestampem, ale rozhodně to není nástroj, který se v dnešní době hodí na projekt, na kterém se podílí více lidí. Avšak i Git má pár nevýhod, a to například velké množství dost špatně zapamatovatelných přepínačů, tudíž silně doporučuji před použitím načíst dokumentaci a udělat si na ně vlastní aliasy. Pokud by něko měl problém s ovládáním nástroje přes command linu, můžu vřele doporučit software SourceTree. Zde jsou rozdíly těchto nástrojů podrobněji rozebrány [25].

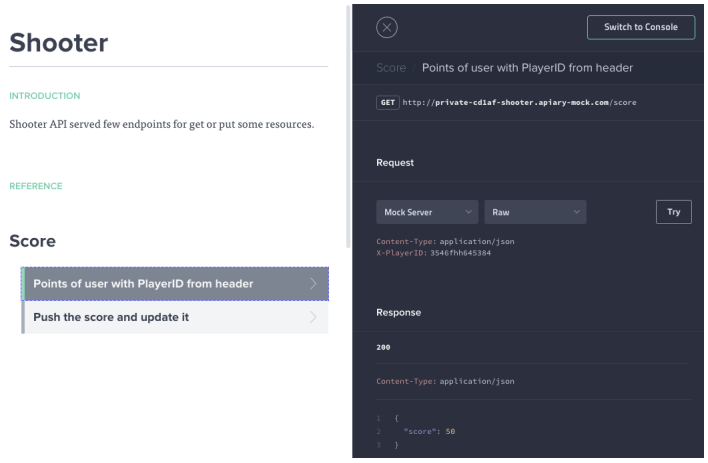
3.8 Nástroje pro testování API

Na harmonogramu projektu jste si mohli všimnout, že nejdříve jsem všechnu práci soustředil na mobilní aplikaci, tudíž byl potřeba nějaký nástroj, přes který bych byl schopen v krátkém čase vytvořit kvalitní a funkční nahrazení REST API serveru. Nejpoužívanějším nástrojem, který slouží k řešení právě tohoto problému je nástroj APIARY. Je to webová služba, ve které je možné vytvořit si stejné endpointy, na které se pak v aplikaci připojujete. Samozřejmě nástroj podporuje popsání libovolných typů metod (GET, PUT, ...) s povinnými i volitelnými hlavičkami či parametry. Tento nástroj umožňuje

psát k API i dokumentaci ve skvěle čitelném fortmátu. Vše se zde píše pod syntaxí API BluePrint Syntax, což je velice rychle pochopitelná a efektivní syntax [26]. Na obrázku 3.6 je vidět mnou napsaná metoda GET a PUT na resource Score. A na tomto dalším 3.7 pak můžete vidět přeloženou dokumentaci právě z BluePrint syntaxe, která už je připravena k použití.

```
FORMAT: 1A

# Shooter
Shooter API served few endpoints for get or put some resources.

# Score [/score]
### Points of user with PlayerID from header [GET]
+ Request (application/json)

    + Header

            X-PlayerID: 3546fhh645384

+ Response 200 (application/json)

        { "score": 50 }

### Push the score and update it [PUT]
+ Request (application/json)

    + Header

            X-PlayerID: 3546fhh645384
            X-Username: BorecekBaji

    + Body

            { "score": 60 }

+ Response 201 (application/json)

        { "playerID": "3546fhh645384", "score": 60 }
```

Obrázek 3.6: Apiary BluePrint Syntax editor pro vytvoření přístupových bodů.



Obrázek 3.7: Apiary dokumentace pro přehled použitých metod a přístupových bodů.

Kapitola 4

Návrh

V této kapitole je podrobně popsán návrh jednotlivých částí mobilní i webové aplikace. Tato kapitola může sloužit jako návod čtenářům, kteří by se chtěli věnovat podobnému tématu, proto zde budou popsány veškeré nástroje a technologie, které jsou v aplikaci použity a dále zde jsou vyobrazeny vytvořené wireframy. Použité nástroje, pokud bylo více různých možností, byly již vybrány v předchozí kapitole a důkladně rozebrány.

4.1 Mobilní aplikace

Hlavní jádro celého projektu je mobilní aplikace, proto byl návrh velice důležitý. Bylo nutné navrhnout komunikaci s fotoaparátem, samotné rozpoznávání obličejů, herní logiku a komunikaci mezi více hráči a dále komunikaci s webovým serverem, pro ukládání informací o hrách a hráčích.

4.1.1 Propojení s kamerou

Pro komunikaci s kamerou v zařízení byl použit objekt AVCaptureSession, který má za úkol kontrolovat tok dat z AV vstupů do všech výstupů. Tudíž je nutné definovat typy vstupů i výstupů, které chceme použít. Dále bylo nutné stanovit preset. Pro tuto aplikaci se nejvíce hodil AVCaptureSessionPresetPhoto, jelikož přenáší stejný obraz, který zachycuje kamera na rozdíl od presetu pro video, který má obraz přiblížený. Konkrétně v této aplikace bylo také důležité změnit orientaci obrazu z kamery pomocí parametru AVCaptureVideoOrientationLandscapeRight. Pro obsluhu výstupů bylo nutné vytvořit metodu, která asynchronně zachytává většinu snímků pořizovaných v reálném čase z kamery. Snímky, které se nezahodí jsou převáděny do objektu CIImage, který se zpracovává až po signálu "vystřelení".

4.1.2 FaceDetection

Detekce obličeje je jádro celé aplikace, protože právě od tohoto nástroje se celý nápad odvíjí. Rozhodl jsem se použít existující nástroj vytvořený přímo oficiálním zdrojem. Konkrétně je to nástroj Core Image, který nabízí jak převod jednotlivých snímků do něměnných CIImages, tak i objekt CIDetector, který dokáže detekovat z lidského obličeje všechny důležité prvky. Tudíž v asynchronním výstupu si ukládám jednotlivé snímky a převádím do CIImage a po vystřelení pouštím na právě uložený CIImage detekci, které mi vrátí pole objektů CIFeatures, ve kterém jsou všechny informace, které jsou potřeba. Na následujícím obrázku 4.1 jsou vidět hlavní CIFeatures, které byly používány. Konkrétně jsou to metody hasLeftEyePosition, hasRightEyePosition, hasMouthPosition a objekt FaceBounds. Dále je možné ještě využít plno dalších metod např.: hasSmile, leftEyeClosed atp.

4.1.3 GameCenter

V předchozí kapitola byla tato služba rozebrána a obecně popsána. Zde bude důkladně dovysvětlena a bude ukázáno i její konkrétní použití v této práci. Bude i důkladně rozebrán a ukázán celý herní cyklus z hlediska této komponenty, což by mělo sloužit jako tzv. kuchařka dalším studentům, kteří se tomuto tématu budou chtít věnovat.

4.1.3.1 Jak to funguje?

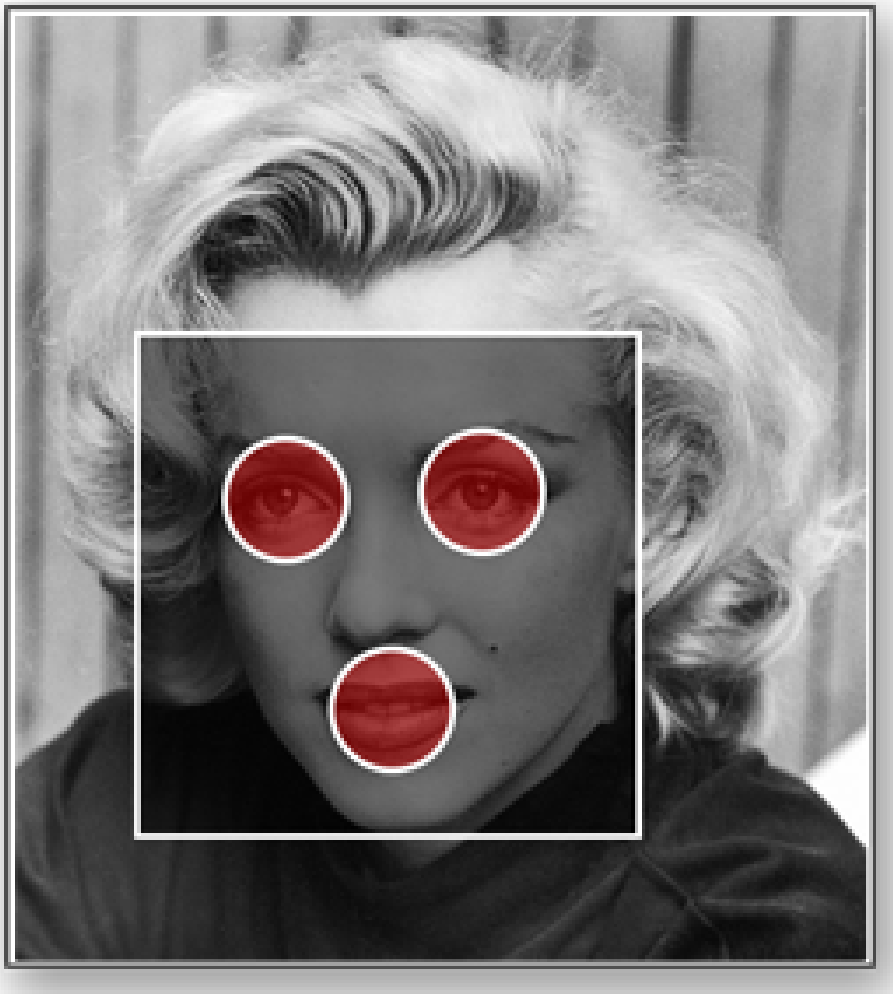
Každé zařízení od firmy Apple má již nainstalovanou aplikaci GameCenter, ve které si vytvoří účet a ten používá napříč všemi aplikacemi, které toto prostředí nabízí. Tento způsob obsluhování aplikací z jednoho účtu je velmi přehledný, jelikož pak různé akce jako výzvy ke hrám, schvalování přátel atp, může uživatel konat z jednoho místa.

4.1.3.2 Hráč

Konkrétnímu hráči v této aplikaci se po autentifikaci přiřadí jeho hráčské údaje. Pro tento konkrétní případ bylo nutné ukládat PlayerID, což je unikátní identifikátor uživatele, který dále potřebujeme pro spojení s více hráči a rozlišení při odesílání a zápisu bodů na serveru. Dále LeaderboardID, což je unikátní řetězec znaků sloužící pro spojení hráče s konkrétním leaderboardem používaným v této aplikaci. GameCenter umožňuje v rámci jedné aplikace použít až 500 různých, v této aplikaci ale stačí pouze jeden.

4.1.3.3 Class GameHandler

Pro snadnější použití a lepší přehlednost jsem vytvořil třídu GameHandler, která obsahuje většinu nabízených metod od GameCenteru. Při spuštění apli-



Obrázek 4.1: Objekty CIFeatures, vrácené po detekci obličeje.

kace vytvořím singleton této třídy a s ním pracuji až do skončení celého herního cyklu. Tato pomocná třída obsahuje metody od autentifikace až po záverečné ukládání bodů na server.

4.1.3.4 Autentifikace uživatele

Služba GameCenter zajišťuje veškeré funkce potřebné k vytvoření herní aplikace, tudíž i autentifikace pomocí ní nebyla nijak obtížná. Je nutné autentifikaci provádět před spuštěním hry, jelikož údaje o hráči jsou potřeba k ukládání bodů a spojení s konkrétním leaderboardem, jak bylo již zmíněno výše.

4.1.3.5 Singleplayer vs. Multiplayer

Velice nutné bylo rozmyslet jaké rozdíly ve funkčnosti budou mezi single a multiplayerovou hrou a jak tyto dvě různé funkčnosti od sebe oddělit, aniž by vznikala duplicita kódu. Koncový návrh byl takový, že singleplayer je možný i pro nepřihlášené uživatele, avšak bez možnosti ukládání dosažených výsledků na server pro pozdější použití, ale jen na lokální úložiště. Dále nepřihlášený uživatel nemá přístup k leaderboardu a k celkovým statistikám. Uvidí pouze svoje skóre a své dosažené achivementy, které bylo nutné naimplementovat i lokálně a ne jen přes GameCenter, aby je i nepřihlášení uživatelé mohli získat. S přihlášením uživatel všechny tyto výhody nabírá a spolu s nimi má ještě možnost vytvořit multiplayerovou hru. Multiplayerová hra se dále liší od single nutností implementace metod pro vytvoření hry, pro vyhledání a spojení s ostatními hráči a hlavně metody pro posílání a přijímání dat mezi všemi hráčemi v podobě minimalistických packetů.

4.1.3.6 Spojení hráčů

Pro spojení s dalšími hráči a následné vytvoření hry je použita metoda findPlayers, které vytváří požadavek na vytvoření hry podle minimálního a maximálního počtu možných uživatelů a následně s nimi zajišťuje spojení. Je zde možnost vyzvat zpřáteleného hráče či spuštění automatické hry, kde je dosazen libovolný hráč hledající hru ve stejném čase.

```
-(void)findPlayers:(UIViewController*)uc{
    GKMatchRequest *request = [[GKMatchRequest alloc] init];
    request.minPlayers = 2;
    request.maxPlayers = 3;

    self.playersCount = request.minPlayers;
    players = [[NSMutableArray alloc] init];

    GKMatchmakerViewController *mmvc = [[GKMatchmakerViewController
        alloc] initWithMatchRequest:request];
    mmvc.matchmakerDelegate = self;
    [uc presentViewController:mmvc animated:YES completion:nil];
    gh = uc;
}
```

Je zde vytvořen delegát na herním controlleru GKMatchmakerViewController, který pak posílá zpět informace o výsledku spojení skrze tři metody. Konkrétně jsou to metoda matchmakerViewControllerWasCancelled(), která slouží jako obsluha události, kdy hráč hru sám ještě před vytvořením zruší, metoda didFailWithError(), která se zavolá při vzniku jakéhokoliv erroru a konečně metoda didFindMatch(), která je zavolána po úspěšném vytvoření hry i se všemi údaji o konkrétním zápasu (Match). V této metodě se všechny důležité

informace uloží a zavolá se hlavní controller, který už pouští hráče do samotné hry.

4.1.3.7 Obsluha skóre a achievementů

Třída GameHandler obsahuje i metody pro práci se skóre a dosaženými achievementy. Základní metoda showLeaderboardAndAchievements() přesouvá uživatele do obrazovky se zobrazením všech statistik jeho i ostatních hráčů, dosažených achievementů či bodovým ziskem a pořadím. Další metodou je reportScore(), což je metoda která slouží pro ukládání skóre na vzdálený server. Zde je nutné kontrolovat, jestli se hraje single nebo multiplayerová hra nebo jestli je hráč vůbec přihlášen, jinak se informace ukládají pouze lokálně. Naopak když hráč je přihlášen a je vytvořena hra více hráčů, je nutné všechny ostatní hráče obeznámit o změně bodového skóre daného hráče.

4.1.3.8 Práce s daty

Pro datovou komunikaci mezi uživateli bylo nutné vytvořit metody sendData a recieveData. Skrze tyto metody hráči posílají své skóre po jakékoliv změně, aby bylo možné vyhodnocovat v reálném čase pořadí jednotlivých uživatelů. Také bylo nutné vymyslet a vytvořit datovou strukturu, z níž bude vytvořen packet, který bude posílán. Proto byla vytvořena struktura MessagePacket, která obsahuje integer pro skóre hráče a pole charů o maximální délce 50, které slouží pro uložení playerID. Konkrétně metoda posílání dat vypadá takto:

```
- (void)sendData:(int)score
{
    NSError *error;
    struct MessagePacket msg;
    msg.score = score;
    [[[UIDevice currentDevice] name] getCString:msg.name maxLength:50
        encoding:NSUTF8StringEncoding];
    NSData *packet = [NSData dataWithBytes:&msg length:sizeof(msg)];
    [current_match sendDataToAllPlayers: packet withDataMode:
        GKMatchSendDataUnreliable error:&error];
    if (error != nil)
    {
        NSLog(@"error pri sendData - %@",error);
    }
    else{
        GKLocalPlayer *localPlayer = [GKLocalPlayer localPlayer];
        NSLog(@"%@",localPlayer.playerID);
        [self updateScores:localPlayer.playerID data:&msg];
    }
}
```

4.1.3.9 Pořadí uživatelů

Při hře více hráčů je nutné, aby každý hráč neustále viděl bodový stav a změny pořadí spoluhráčů nejlépe v reálném čase. Proto je po vytvoření samotné hry vytvořeno i pole, kde se pod jednotlivými playerID ukládají s nimi spojené bodové skóre. Vždy, když nějaký hráč přijme data, zavolá se metoda updateScore(), která má za úkol toto pole co nejrychleji seřadit a určit pořadí daného hráče vzhledem ke skupině.

4.1.4 CocoaPods

Pro jednodušší implementaci různých komponent je zde využíván systém CocoaPods, což je balíčkovací systém pro jazyk obj-c. Stejně tak jako v jazyce Ruby se používají gemy či v Javě se používá systém Maven, tak pro obj-c se používají právě Pody. Použití balíčkovacích systémů se řadí k modernímu vývoji aplikací hlavně kvůli jednoduchosti používání. Samozřejmě můžeme využít jiné metody jako kopírování kusů cizích kódů či stahování zkompilovaných knihoven z git repozitářů, avšak žádný jiný způsob nezaručí takový komfort jako balíčkovací systém. Konkrétně pro tuto práci se využívaly pody AFNetworking pro práci s API a Masonry pro snadnější práci s celkovým layoutem aplikace. Pro bližší informace o CocoaPods vizte [27].

4.1.4.1 Použití

CocoaPods, jak již bylo zmíněno velice urychlují práci, tudíž jsou i zcela jednodušše instalovatelné. Instalují se pomocí Ruby jako konkrétní gem jednoduchým příkazem gem install cocoapods. Pak je nutné vytvořit si v root složce projektu soubor Podfile, ve kterém se definuje, které pody chcete použít. V tomto konkrétním případě vypadá Podfile takto:

```
platform :ios, '7.0'
pod 'AFNetworking', '~> 2.4'
pod 'Masonry'
pod 'SVProgressHUD'
```

4.1.4.2 AFNetworking

AFNetworking je jeden z nejpoužívanějších a nejpopulárnějších open-source projektů v iOS a OS X vývoji. Tento nástroj slouží pro jednoduché použití HTTP metod jako GET, PUT, POST atp. Je zde i velmi jednoduchá definice vstupních či výstupních parametrů, datových typů či různých přidaných hlaviček [28]. Zde konkrétně bylo třeba naimplementovat posílání dat na server, kde se data následně uloží. Samozřejmě není možné posílat data při každé změně, aby nedošlo k přetížení serveru. V současné době je aplikace navržena tak, aby data posílala v daných časových intervalech, či při přerušení vzniklé

libovolným errorem či rukou uživatele. Konkrétní POST request zde obsahuje JSON pole se změněným skóre a dosaženými achievementy a dále hlavičky, které obsahují Username a playerID, podle čehož server pozná jakému uživateli aktualizovat údaje.

4.1.4.3 Masonry

Masonry je další nástroj instalovatelný pomocí podů. Masonry se používá pro zjednodušení práce s AutoLayoutem pro jazyk obj-c. Pro jazyk SWIFT už toto použití není podporováno a je proto za Masonry nahrazení v podobě SnapKitu. Je možné představit si jeho použití jako microframeworku obalující AutoLayout, který poskytuje veškerou jeho funkčnost, ale s použitím vlastního DSL. Autolayout je nástroj sloužící k usnadnění přizpůsobení uživatelského prostředí aplikace pro různé rozlišení displeje. Umožňuje snadnější tvorbu vzhledu aplikace, kde se neměnné hodnoty (velikost, umístění..) získávají dynamicky ze systému či pomáhá aplikacím reagovat na změny vzhledu.

4.1.4.4 SVProgressHUD

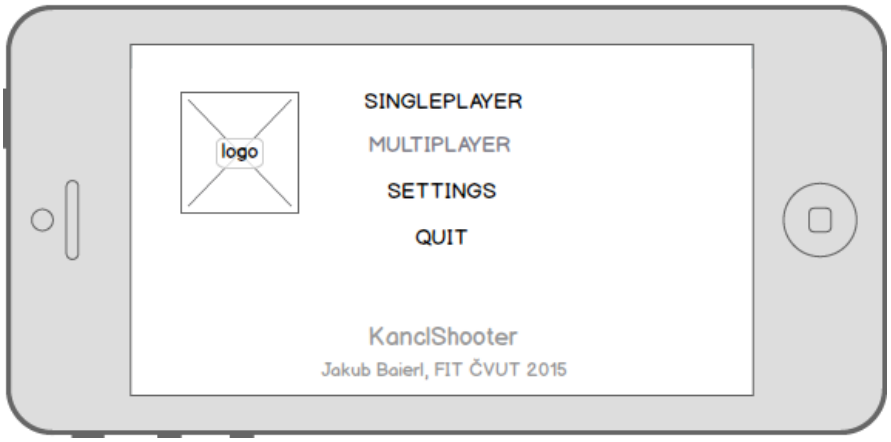
S touto komponentou se dokáže velmi jednodušše a čistě docílit zobrazení progresu (postupu) aplikace prostřednictvím loading screeny.

4.1.5 Wireframes

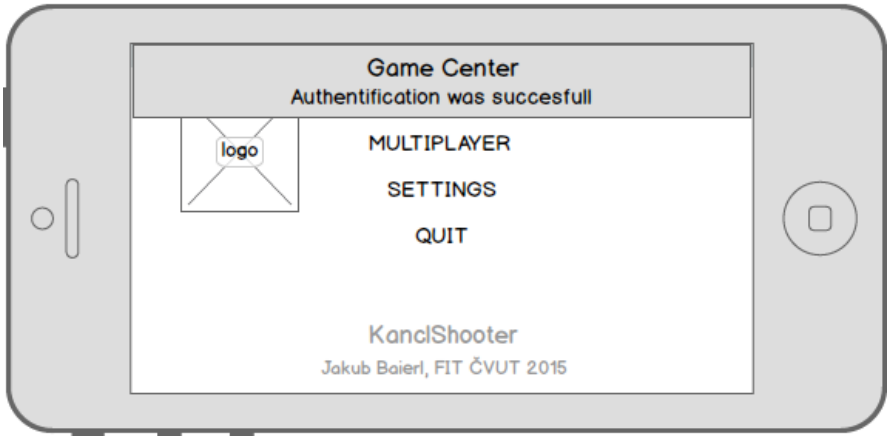
Wireframe neboli drátěný model aplikace se používá v mobilních či webových aplikacích pro návrh konečného vzhledu a rozmístění veškerých UI prvků. Používá se i pro definování funkčnosti veškerých prvků pro jednodušší pochopení či prezentaci ostatním členům týmu či samotným klientům. Správný wireframe by neměl obsahovat žádná média (obrázky, grapfické prvky), čili měl by se skládat pouze z čar, primitivních tvarů a textu. Neslouží ani jako koncový prototyp, který naopak od wireframu obsahuje veškerou grafiku a slouží jako koncový testovací produkt před samotnou implementací. Pro tvorbu těchto wireframů jsem použil nástroj Balsamiq Mockups 3, který má 30 denní trial verzi s možností využití 100 % funkčnosti. Použil jsem ho pro návrh mobilní i webové aplikace, jelikož poskytuje prostředí pro tvorbu návrhu pro obě tyto platformy. Vybral jsem pár hlavních obrazovek, pro které jsem připravil wireframy. Bylo nutné i počítat s prvky a funkčností, kterou dosazuje samotná služba GameCenter a nějak je do wireframů zapracovat.

4.1.5.1 Hlavní menu

Na obrázku 4.2 je vidět hlavní menu pro nepřihlášené uživatele. Je zde možnost zapnutí singleplayer hry (avšak bez trvalého ukládání výsledků), ale k multiplayer hře uživatel není puštěn.



Obrázek 4.2: Návrh hlavního menu aplikace pro nepřihlášeného uživatele.



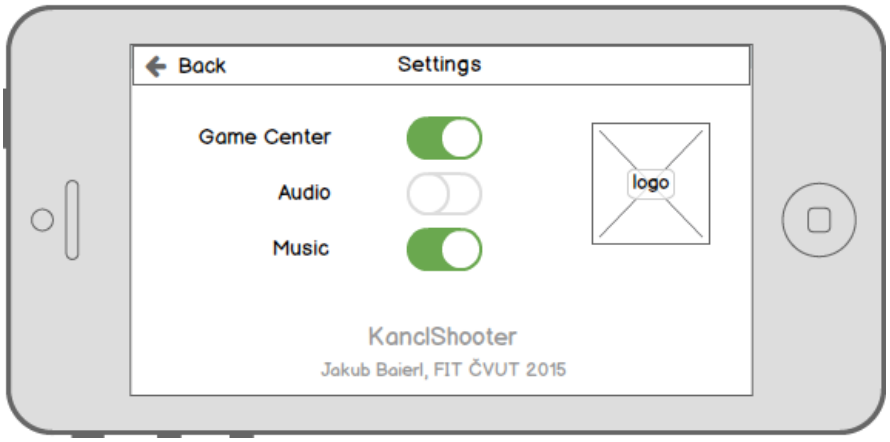
Obrázek 4.3: Návrh menu pro přihlášené uživatele.

Obrázek 4.3 už ukazuje stav přihlášení, které je zobrazeno horním panelem. Tento příznak úspěšného přihlášení se zobrazí buď bezprostředně po přihlášení nebo po opětovném zapnutí hry, kde přihlašovací údaje jsou stále uloženy v mezipaměti.

4.1.5.2 Nastavení

Mobilní aplikace má také pár jednoduchých nastavitelných parametrů, které jsou vidět zde 4.4, pro které také bylo nutno navrhnout prostředí a chování. Nejlepší možnost byla použít systémové Switch Buttony, na které jsou uživatelé iOS zařízení zvyklí. Je zde i možnost přihlášení a odhlášení se ze svého GameCenter účtu například, když na zařízení hraje někdo jiný.

V případě posunutí switch tlačítka u kolonky GameCenter do polohy "ena-



Obrázek 4.4: Návrh obrazovky pro nastavení aplikace.

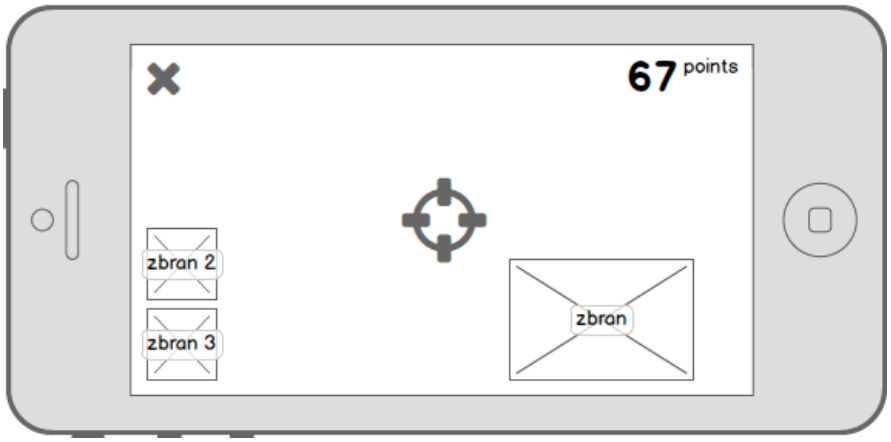
Obrázek 4.5: Návrh přihlašovací obrazovky.

ble", automaticky se spustí přesměrování na přihlašovací obrazovku GameCenteru pro zadání přihlašovacích údajů. Po úspěšné autentifikaci je uživatel přesměrován na hlavní obrazovku a může začít hrát.

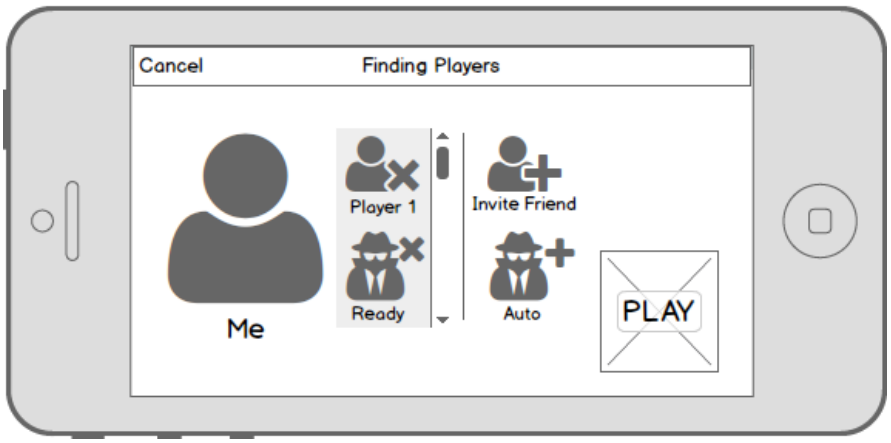
4.1.5.3 Hra

Samotná hra se rozděluje do tří různých typů. Může hrát nepřihlášený uživatel, který nemá možnost trvalého ukládání skóre, nemůže vidět žebříčky a informace o dalších hráčích ani nemá možnost hrát s více hráči najednou, což popisuje tento wireframe 4.6.

Další typ hry je pro přihlášeného uživatele, který má všechny výhody, a tudíž může hrát singleplayer či vyzvat další hráče. Jak je vidět na dalším wireframu 4.7 je nutné před zapnutím multiplayerové hry se spojit s minimálně



Obrázek 4.6: Návrh herní obrazovky pro nepřihlášeného uživatele.

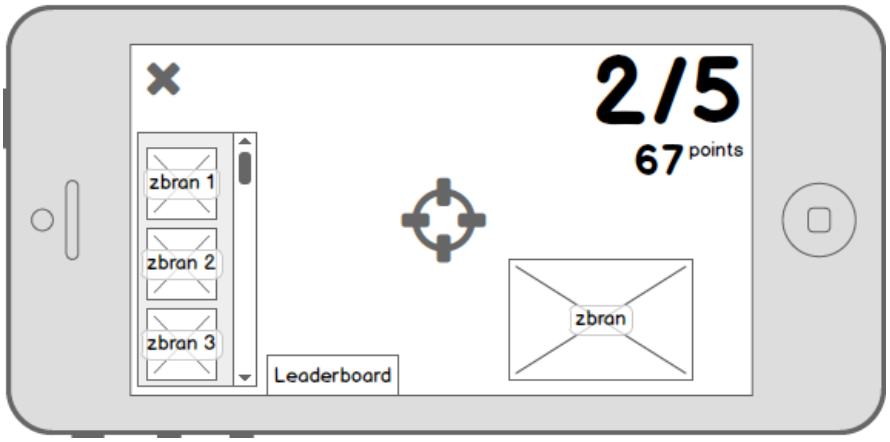


Obrázek 4.7: Návrh obrazovky pro nastavení multiplayer hry.

jedním dalším hráčem. Je možné vyzvat buď někoho ze spřátelených uživatelů či spustit automatickou hru, kde jsou spojeni náhodní hráči, kteří hledají hru. Na tomto 4.8 obrázku je vidět obrazovka při chodu multiplayer hry. Je zde jak odkaz na leaderboard, kde jsou přístupné dosažené achievementy, tak i příznak informující o pozici vzhledem k ostatním hráčům.

4.2 Webová aplikace

Návrh webové aplikace spočíval v jednoduchém spojení backendu a Frontendu, kde Backend se stará o komunikaci s mobilní aplikací a samotné zpracování přijatých dat a Frontend má funkci informační, tudíž hlavní jeho úkol je prezentace samotné aplikace a dat v uživatelsky přijatelné podobě.



Obrázek 4.8: Návrh herní obrazovky i se všemi prvky pro přihlášeného uživatele hrající s více hráči.

4.2.1 Backend

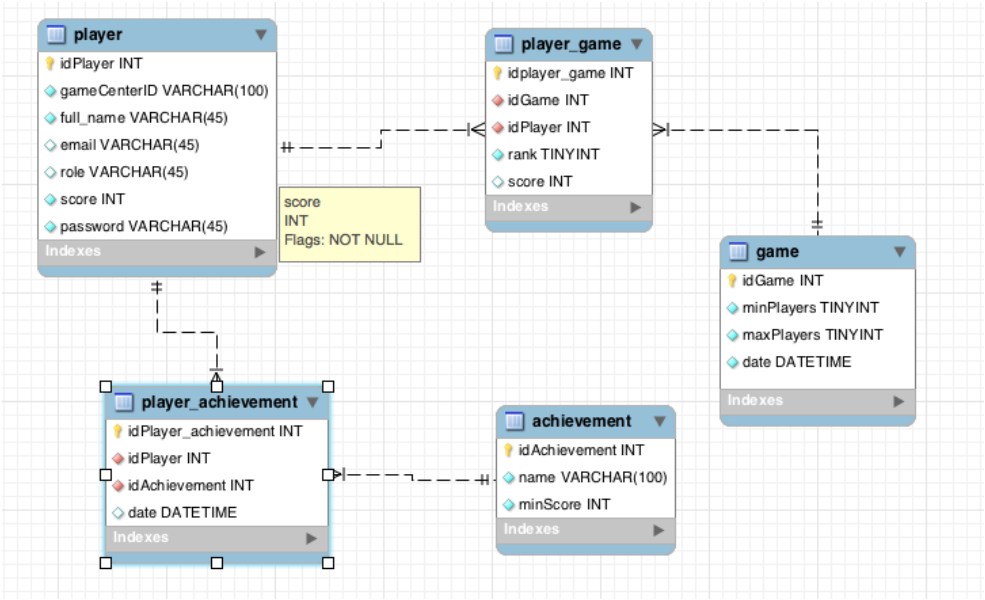
Backend, neboli zadní část aplikace, jak už bylo výše řečeno, se stará o spojení a zpracování dat a hlavní logiku aplikace. Bude vytvořen v jazyce PHP s použitím Nette frameworku s podporou MySQL databáze pro uložení dat. Jako hlavní návrhový vzor bude použit vzor MVC, který velmi čistě odděluje logickou část od prezentační.

4.2.1.1 MVC (MVP)

Nette framework velmi striktně využívá tento návrhový vzor MVC, proto bylo nutné se s ním seznámit a pochopit jeho funkčnost. MVC je většinou třívrstvý návrhový vzor, který rozděluje aplikaci na více vrstev, kdy nejčastěší použití spočívá v rozdělení na Model, View a Controller. Toto použití je velmi přehledné, jelikož už podle názvů každá vrstva zaštiťuje část aplikace. Model je vrstva, která se stará o zpracování dat, operace nad jednotlivými entitami a následné posílání strukturovaných dat do Controlleru. Prostřední vrstva nazývaná Controller se stará o hlavní funkčnost a logiku celé aplikace. Dostává předzpracovaná data z Modelu a následně s nimi provádí všemožné logické operace a následně data nejlépe v konečné podobě posílá do View. Vrsta View má pak za úkol data v koncové podobě prezentovat v uživatelsky přijatelné podobě koncovým uživatelům.

4.2.1.2 Databáze

Databáze je jakékoliv množství informací uložených v paměti. Tato datová základna slouží zde pro uchování všech informací o hrách, hráčích i výsledcích. V analýze bylo rozhodnuto použití MySQL databáze a v následující části



Obrázek 4.9: Diagram popisující datový model.

bude navržen relační datový model. Relační model je základ pro definici **relační databáze**. Tento typ databáze je založen na tabulkách a vztazích mezi nimi. Základem jsou relace, což jsou struktury tvořené dvěmy rozměry (hlavička, tělo). Sloupce zde nazýváme atributy, které mají definovaný datový typ a řádek je skrz všechny sloupce už jako jeden vlastní záznam uložených informací. Každá tabulka musí obsahovat primární klíč, který je definován jedním nebo skupinou atributů, což zapříčiňuje jednoznačnost záznamu. Dále tabulky mohou obsahovat cizí klíče, které definují vztahy s jinými tabulkami.

Normální formy
Proces normalizaci slouží pro označení optimalizace relací, tak aby se vyskytovalo jen co nejmenší množství redundantních dat. Pro označení různých stavů normalizace existují normální formy [29].

4.2.1.3 Datový model

Datový model byl navržen pomocí nástroje MySQLWorkbench, ve kterém se definují jednotlivé relace, atributy i datové typy atributů. Následně se z těchto relací dá velmi jednoduše a přehledně sestavit EER diagram, což je vylepšený ER model a slouží jako konceptuální návrh většinou relační databáze. Na tomto diagramu 4.9 jsou velmi dobře vidět všechny relace a vztahy mezi nimi, které budou dále podrobněji popsány.

Entita Player
Základní entitou je entita Player, popisující daného hráče. Obsahuje i unikátní gameCenterID, které je vždy posíláno z aplikace v hlavičce požadavku, podle kterého se vždy pozná o jakého hráče jde. Tato relace je spojena s relací Game a Achievement pomocí dalších vztahových tabulek, popsaných níže.

Entita Game
Entita Game slouží pro uložení informací o všech hrách a hráčích kteří danou hru hráli, kdo se jak umístil a kdy byla hra odehrána.

Entita Achievement
Tato entita slouží pro uchování informací o dosažených achievementech (úspěších), které jsou definovány pouze jménem a minimálním počtem bodů, nutných pro jeho získání.

Entita Player_Game
Tato entita slouží jako pomocné propojení tabulek Player a Game, jelikož jejich vztah je N:M. To znamená, že jeden hráč hrál více her a danou hru mohlo hrát více hráčů. Proto vznikla vztahová tabulka Player_Game, která popisuje který hráč danou hru hrál, kolikátý se umístil a kolik měl výsledné skóre.

Entita Player_Achievement
Tato entita, podobně jako předchozí, je vztahová entita mezi Player a Achievement, která popisuje kdy jaký hráč dosáhl nějakého achievementu.

4.2.2 RESTful API

Representational state transfer je způsob jak vytvářet, editovat či mazat záznamy v databázi pomocí HTTP požadavků. Je to velmi přehledný a dobře použitelný přístup ke zdrojům, což můžou být samotná data nebo data, která popisují stav, ve kterém se daná aplikace nachází. Všechny tyto zdroje mají unikátní URI adresu, přes kterou se k nim přistupuje. Používá se v komunikaci client-server, kde dochází k oddělení zodpovědnosti. Například aplikace je rozdělena na více částí a mobilní aplikace komunikuje s webovým serverem, přes který přistupuje k datům. Pro práci se zdroji jsou definovány čtyři různé druhy operací označovány pojmem CRUD (Create, Read, Update, Delete). Tyto metody mohou pracovat nad jednotlivými daty či nad celými kolekcemi podle unikátních URI adresám. Definované metody, které se běžně používají jsou GET, POST, PUT a DELETE

Metoda GET (Retrieve)
Metoda GET je základní metodou pro získání dat. Může to být jednoduchý

GET požadavek na webvou stránku či na celý zdroj na určité URI adrese. Při vytváření požadavku je také možné definovat datový typ, ve kterém chceme data ze zdroje získat. Na výběr jsou typy XML, JSON, RSS a ATOM. Daný požadavek může vypadat například takto:

```
GET /user/3
Host: shooter.cz
```

Metoda POST
Metoda POST slouží pro vytváření nových dat. Pro webové vývojáře je tato metoda velmi známá, jelikož se používá ve většině webových formulářů. Tato metoda se volá na určitou adresu, která je vždy stejná, protože daný zdroj ještě neexistuje. V těle požadavku pak pošleme obsah, která se na dané adrese uloží. Například pokud bychom chtěli uložit nového uživatele pošleme **POST** požadavek na adresu **shooter.cz/user** s následujícím obsahem:

```
{
id: 13,
jmeno: "Jakub",
email: "jakub.baierl@gmail.com"
}
```

Po úspěšném vytvoření nového zdroje v tomto případě na adrese **shooter.cz/user/13** by se správně měl vrátit stavový kód **201 - Created**.

Metoda PUT
Metoda PUT je obdobná metody POST, ale používá se pro nahrazení či změnu stávajícího zdroje. Tudíž před zavoláním metody daný zdroj už na dané URI adrese existuje. V těle požadavku se pošle obsah, který daný zdroj zamění.

Metoda DELETE
Tato metoda je velice podobná metodě GET. Volá se ve stejném tvaru jen vyvolá smazání zdroje, který se na dané URI nachází.

4.2.3 Frontend

Slovem Frontend je označována většinou ta část aplikace, která je viditelná běžnému uživateli. Rozdělení aplikace na Frontend a Backend je velmi dobré pro SoC, což je oddělení odpovědností různých funkčních celků. Frontend můžeme chápat jako interface mezi uživatelem a serverem, kde se pracuje s jednotlivými daty. Je to vlastně obecnější použití MVC návrhového vzoru, který byl popisován v přechozí kapitole. Frontend této aplikace je konkrétně navržen jako SPA [30] s využitím dalších stránek pro zobrazení detailních informací

o hráčích a samotných hrách. Bude vytvořen v jazyce HTML5, CSS3 s využitím Javascriptu a AJAXu. Pro hlubší porozumění a lepší návrh UI doporučuji tento blog [31].

4.2.3.1 Single-Page aplikace

Single-page aplikace je většinou webová aplikace či webová stránka, kde se všechny nebo alespoň velká část všech informacích vyskytuje na jedné stránce, která bývá rozdělena do více částí. Často je tato technika spojována s responsivním designem popsaným dříve. Je zde velice důležité minifikovat všechny includované soubory a nahrávat obrázky v rozumné kvalitě, jelikož se stránka načítá celá najednou. Nejčastěji se také v tomto druhu aplikace používá AJAX pro dynamický obsah i případně Javascriptové frameworky, který AJAX využívají.

4.2.3.2 AJAX

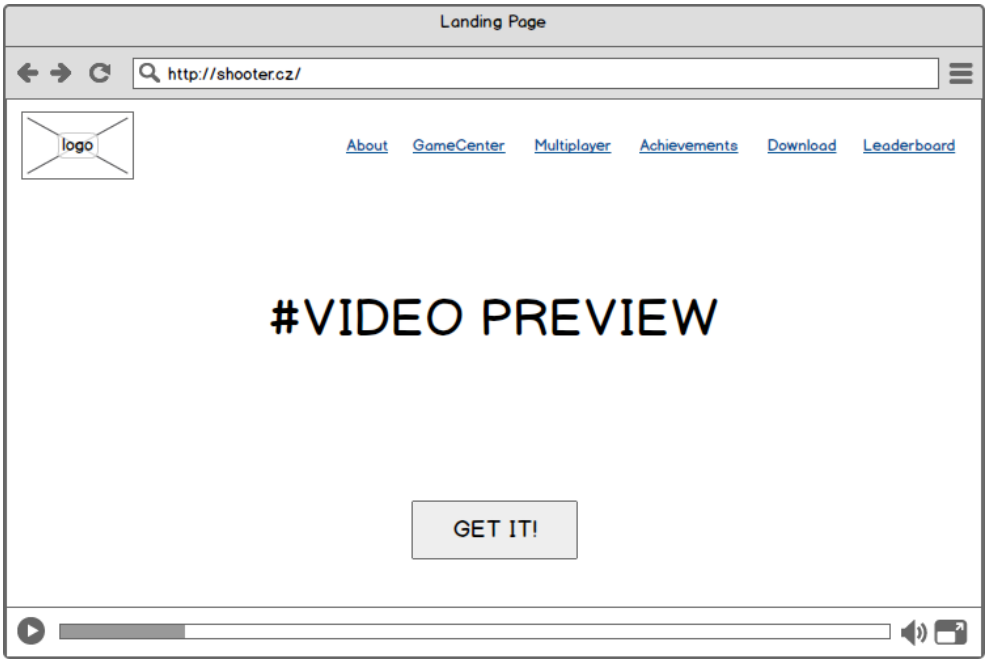
AJAX je webová technika pro dynamickou výměnu obsah dat na stránce bez nutnosti přenačtení celé stránky. Používá se právě pro vyvolání HTTP metod popisovaných dříve a pro uložení získaných dat ve zpětné odpovědi serveru. Při vytváření SPA s využitím AJAXu se často používá nějaký Javascriptový framework, například Ember.js či Angular.js, který velmi dobře umožňuje jednoduchou práci s AJAXovými požadavky i se samotnými daty, které jsou načítány buď po částech či všechny najednou.

4.2.3.3 jQuery

JQuery je velmi známá knihovna postavená nad jazykem JavaScript, která má za úkol spojit JavaScript a HTML a umožnit uživateli jednodušší práci s DOM elementy. Dále se často používá pro jednodušší práci s AJAXem, když není použit žádný JS framework. JQuery dále nabízí plno další funkčnosti například obsluhování událostí, změny CSS různých elementů, vyhledávání a přehazování elementů v DOM struktuře, efekty a různé originální animace, zpracování fotogalerií a mnoho dalších. Díky otevřenosti softwaru každý den vznikají nové a nové pluginy, které jQuery vylepšují a usnadňují vývojářům práci. O správě a vkládání těchto pluginů bude pojednávat další sekce **Balíčkovací systémy**

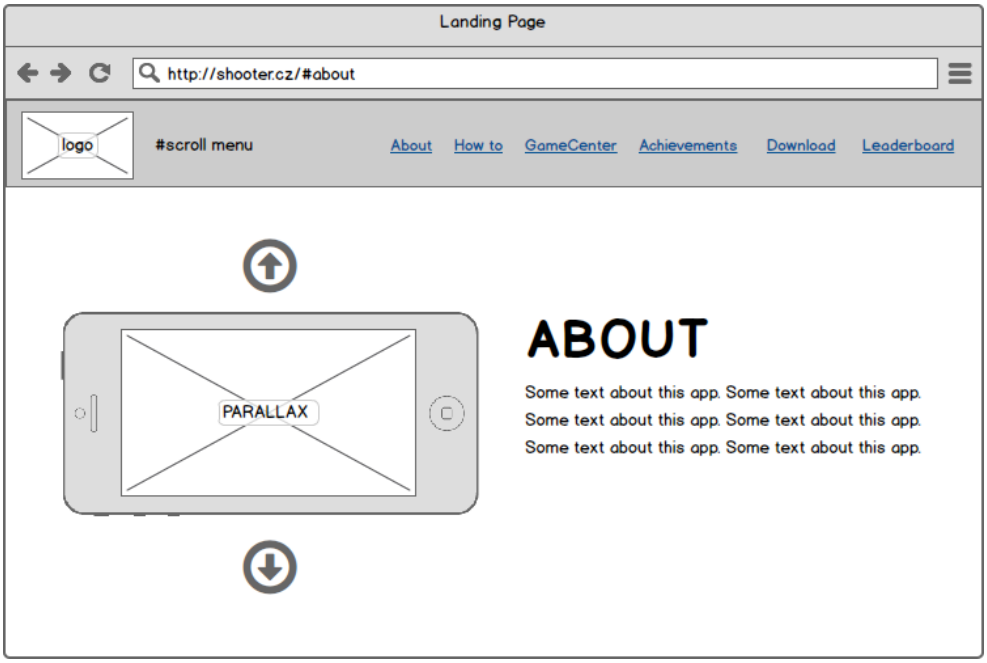
4.2.3.4 Wireframes

Samozřejmě i pro přední část webové aplikace jsem musel vytvořit wireframy, abych podle toho pak mohl navrhnout rozmístění prvků, funkčnost a grafické rozhraní pro uživatele co nejpřívětivější. Vybral jsem pár nejzajímavějších wireframů, které následně podrobněji popíši a ukážu funkčnost.



Obrázek 4.10: Návrh horní části hlavní singlepage stránky.

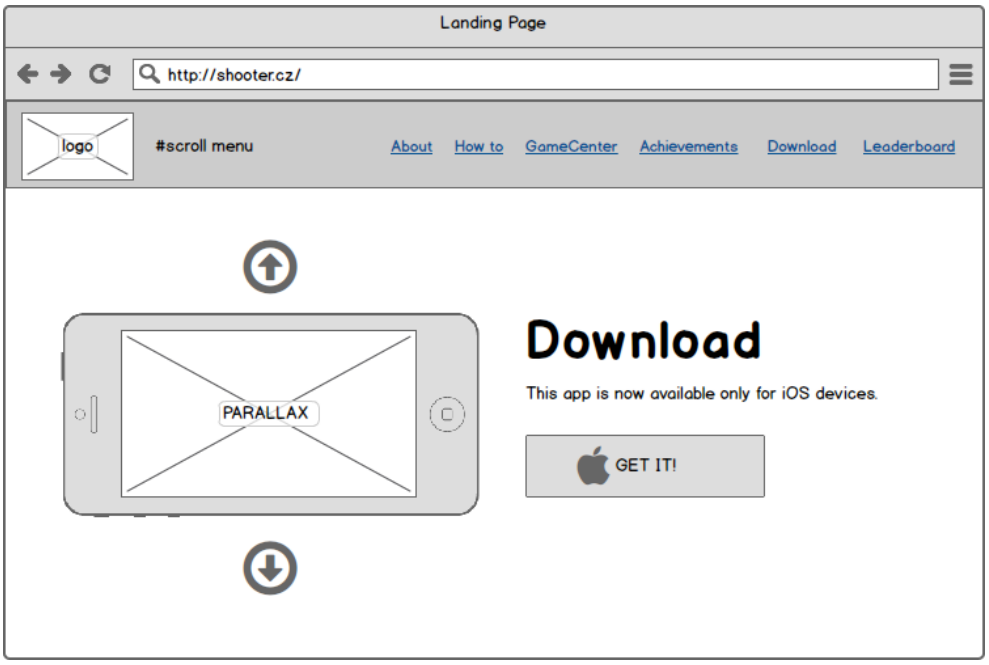
Landing page
Na obrázku 4.10 je navržena horní a vlastně úplně první část webové stránky, kterou uživatel uvidí. Proto jsem se snažil při návrhu zohlednit prvky, které by mohli uživatele ihned zaujmout. Proto na pozadí celé části je navrženo místo pro video, které bude popisovat, jak aplikace funguje a jak se používá v praxi. Dále je zde hlavní menu, umístěno hned nahoře v hlavičce, tam kde je většina uživatelů zvyklá menu hledat. Dále v levé části se nachází hlavní logo aplikace, aby bylo uživateli stále na očích a dole velké tlačítko které posune uživatele do další sekce.



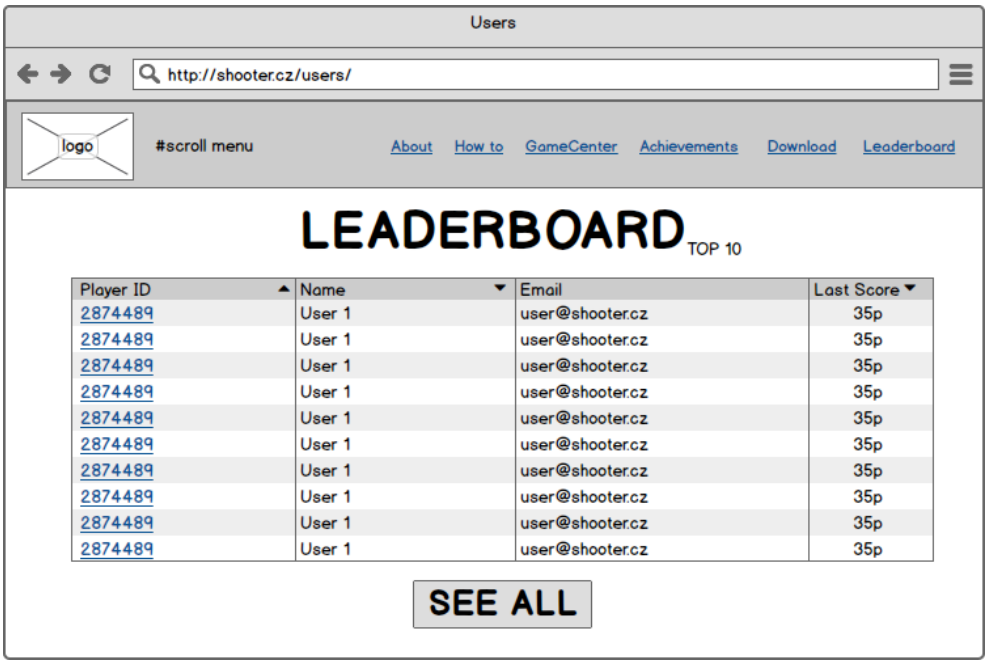
Obrázek 4.11: Návrh části About na hlavní stránce.

Parallax scrolling efekt
Jelikož je to singlepage webový návrh, je v dnešní době velice populární přidávat originální grafické či animované prvky. Zvolil jsem použití parallax scrolling efektu. Parallax je určitá technika používaná v počítačové grafice povětšinou při návrzích webové či mobilní aplikace a spočívá v neobyčejném chování pozadí obrazovky při scrolování obsahu. Je více technik, které se používají, ale nejpoužívanější jsou fixní pozadí či zpomalení posunu pozadí. Jak je tento parallax efekt vytvořen právě v této aplikaci bude podrobnější vysvětleno v příští kapitole. Efektů na této bázi je velké množství. Různé příklady si můžete prohlédnout zde [32]. Z různých částí hlavní stránky jsem vybral právě část About 4.11 a část Download 4.12

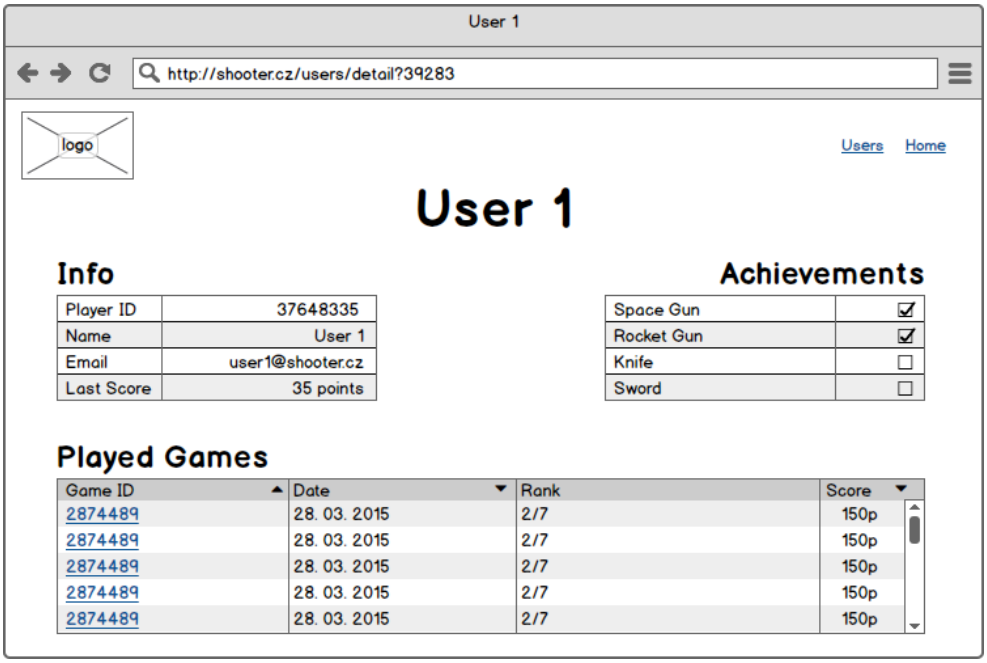
Leaderboard
Leaderboard byl navrhnut jako část hlavní stránky, jak je vidět na obrázku 4.13, ale všechny odkazy z něj na jednotlivé hráče vedou na novou stránku s dalšími přehledy. Je to tak navrženo kvůli ušetření stránek před větším množstvím všech dat.



Obrázek 4.12: Návrh části Download na hlavní stránce.



Obrázek 4.13: Návrh leaderboardu na hlavní stránce.



Obrázek 4.14: Návrh leaderboardu na hlavní stránce.

Přehled hráče

Na dalším wireframu 4.14 je vidět návrh obrazovky pro přehled konkrétního hráče. Je tu již jiné menu, které slouží pro vstup do přehledu všech hráčů nebo zpět na hlavní stranu. Dále jsou tu veškeré informace o hráči a o jeho získaných i nezískaných achievementech. V dolní části stránky se pak nachází sekce všech hráčových her, kde si může každý prohlédnout kolikátý byl v pořadí, kolik měl bodů či se může na danou hru kliknutím přesunout.

4.2.3.5 Balíčkovací systém

Balíčkovací systém se používá pro správu knihoven či dalšího potřebného softwaru, používaného při implementaci aplikace. Používá se již delší dobu pro správu softwaru v systémech na bázi UNIXu s pomocí nástrojů APT či YUM. V jazyce PHP je velice známý nástroj Composer, který se používá podobně jako správce závislostí na jiných knihovnách. Tyto nástroje se vlastně používají pro rychlejší a snadnější používání externích knihoven, které by se musely jinak obtížně hledat, stahovat a instalovat. Composer se o všechny tyto problémy postará sám. Stačí pouze definovat knihovny a verze, ve kterých tyto knihovny chceme naimportovat. V poslední době čím dál víc roste zájem o Frontend v oblasti webového inženýrství, tudíž bylo určitě potřeba vymyslet podobný systém i pro tuto část webových aplikací. V dnešní době mezi nejmodernější nástroje pro správu softwaru a závislostí pro Frontend jsou nástroje

NPM, Bower a GruntJS, které budou dále podrobněji popsány.

4.2.3.6 NPM

NPM je nástroj postavený nad jazykem NodeJS, který slouží pro používání i sdílení knihoven mezi ostatními vývojáři. Tento nástroj funguje v podobné formě jako dříve zmiňovaný Composer. Použití spočívá v nadefinování externích knihoven, který vývojář chce použít a specifikaci jejich verzích. Toto se definuje v souboru **package.json**, ve kterém se dále definují metadata o autorovi i samotném projektu a dále se dají nastavit různé příkazy a skripty, které se po zadání předdefinovaných příkazů spouštějí, což se velmi hodí pro spolupráci s dalšími obdobnými nástroji. Pomocí NPM se také instalují dále popsané nástroje Bower a GruntJS.

4.2.3.7 Bower

Nástroj Bower funguje obdobně jako nástroj NPM jen s rozdílem, že Bower používá **bower.json**, pomocí kterého můžeme dále importovat specifičtější knihovny, které pro NPM nejsou vytvořené.

4.2.3.8 GruntJS

GruntJS se používá jako nástroj pro stavbu operací nad knihovnami, importované pomocí předchozích nástrojů. Konkrétně se používa například pro minifikaci souborů, vyhledávání a definování různých závislostí či "watchování"určitých souborů, které se před použitím musejí překládat jako například SaSS soubory, které se přes použitý preprocesor kompilují do podoby CSS. Dále se může používat pro vyhledání konkrétních souborů a vložení na správně místo pro správnou funkčnost. V kapitole Realizace bude použití těchto nástrojů demonstrováno na konkrétním přikladě.

4.2.3.9 Externí knihovny

Zde jsou podrobněji popsány externí knihovny použité ve Frontend části aplikace. Je důležité pořadí, v jakém jsou vkládány, kvůli svým závislostem.

jQuery
Nejpoužívanější knihovna v této aplikace je **jQuery**, která byla podrobně popsána dříve. Všechny další používané knihovny a komponenty tuto hlavní knihovnu používají, proto je nutné načítat jí hned jako první po načtení stránky.

Bootstrap
Druhá nejpoužívanější komponenta je **Bootstrap**, který nabízí celou řadu funkčních či jen vzhledových prvků. Pomocí bootstrapu se i práce s reponsivním designem stává jednodušší.

Waypoints
Další zde velmi používanou knihovnou je **waypoints.js**, což je knihovna postavená nad jazykem JavaScript, která umožňuje vývojáři obsluhovat události při výskytu na různých částech stránky, a to se s použitím SPA návrhu velice hodí.

Transitions
Dále se zde používá knihovna **transitions.js**, která využívá JavaScriptové funkce pro translace a rotace a nabízí snadnější syntax, a tudíž i jednodušší a přehlednější implementaci.

4.2.3.10 Sociální sítě

Sociální sítě jsou v současné době z hlediska PR velmi prospěšná služba, která při nejmenších pracovních nákladech dokáže vyrobit velkou reklamu. Proto jsem se rozhodl využít je i pro rozšíření této aplikace. Při prvních návrzích této aplikace se počítalo s využitím přihlašování do aplikace i pomocí účtů z těchto sociálních sítí, avšak v analýze bylo rozhodnuto pro využití služby GameCenteru, která již má své přihlašování implementované, tudíž by nebylo žádoucí, aby se uživatelé museli přihlašovat vícekrát. Proto v první verzi této aplikace budou sociální sítě sloužit pouze k propojení webové části aplikace se stránkami založenými právě na těchto sítích.

Facebook
Facebook je v současné době nejrozšířenější sociální síť na světě, kterou používá až 1,44 miliardy aktivních uživatelů měsíčně [33]. Proto jsem se rozhodl umístit na svou webovou část aplikace **Facebook Likebox**, což je interaktivní prvek, který slouží, jak pro informace o uživatelích a jejich komentářích na stránce, tak i jako rychlý způsob sdílení stránky či zanechání nějakého příspěvku rovnou z webu.

Twitter
Twitter je druhá nejrozšířenější sociální síť, velmi výrazně používaná uživateli hlavní z IT sektoru. Momentálně twitter používá 236 milionů aktivních uživatelů měsíčně [34]. Rozhodl jsem se proto, tuhle aplikaci uvěřejnit i na twitteru a na web umístit tlačítka pro sdílení.

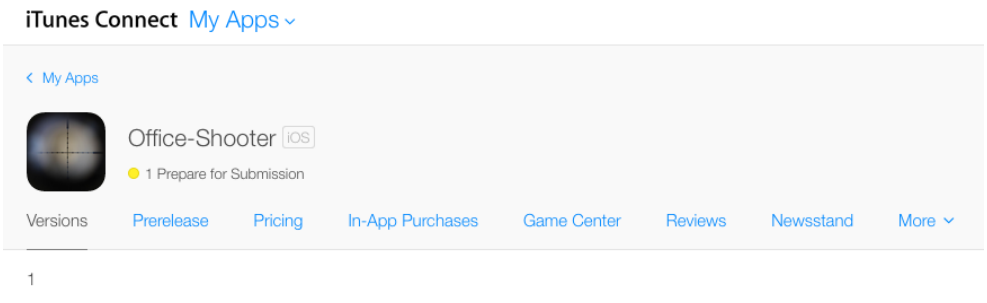
Kapitola 5

Realizace

Tato kapitola se bude podrobněji věnovat hlavně těm částem aplikace, které byly buď obtížně implementovatelné či při jejich tvorbě vznikaly nějaké problémy anebo jsou z hlediska implementace a samotné realizace nějak zajímavé. Další problémové části, které se projevily až v testovací části budou podrobněji popsány v kapitole Testování.

5.1 Mobilní aplikace

Realizace mobilní části byla velmi zábavná, jelikož se jedná o interaktivní hru, při jejíchž implementaci vývojář musí neustále testovat. Díky dobře provedené analýze a samotnému návrhu implementace byla tato část jednodušší, avšak velmi časově náročná. Dále se při realizaci vyskytlo nemálo obtížných částí, ze kterých bude pár vybráno a podrobněji popsáno dále a na ukázkách názorně ukázáno jejich vyřešení či zajimavá funkčnost.



Obrázek 5.1: Vytvořená aplikace v iTunes Connect.

5.1.1 Napojení služby GameCenter

Použití samotné služby GameCenter není zvlášť obtížné, ale příprava prostředí pro její použití je náročnější. Nejprve je nutné vytvořit si developerský účet na oficiálním webu pro iOS vývojáře - `https://itunesconnect.apple.com`. Dále si vývojář musí ve svém účtu vytvořit aplikaci 5.1, kterou musí provázat s konkrétním projektem pomocí XCode, kde se dále musí povolit použití služby GameCenter. Ve vytvořené aplikaci se dále musí vytvořit další nastavení, a to například nastavení služby GameCenteru, podpory hry více hráčů, povolení interních a externích testerů atp.

5.1.2 Registrování aplikace a hra na reálném zařízení

V počátcích implementace bylo možné aplikaci spouštět na **simulátoru**, který se dá nastavit jako jakýkoliv typ iOS zařízení. Nastal však problém při nasazení obrazu z kamery a přidávání různých prvků simulujících rozšířenou realitu. Simulátor nemá přístup k různým komponentám zařízení jako k GPS modulu, gyroskopu a právě ani ke kameře, tudíž bylo nutné testovat aplikaci na reálném zařízení, což ale u platformy iOS je trochu komplikované. Je důležité rozlišit čtyři skupiny nastavení.

5.1.2.1 Certifikát

Certifikáty se dělí podle využití do skupiny Development a Production. Při vyvíjení aplikace na soukromém zařízení je nutné vytvořit si svůj osobní certifikát, který se uloží do počítače a zaregistruje se na webu `http://developer.apple.com` pod svým developerským účtem. Pokud vývojář chce aplikaci už poskytnout do AppStoru musí mít produkční certifikát, ale to prozatím není případ této aplikace.

5.1.2.2 App ID

Každá aplikace má svůj App identifikátor, který se také dělí na Explicitní a Wildcard identifikátory. Wildcard identifikátory jsou jednodušší na používání a dají se používat pro více aplikací, což se při vývoji velice hodí. Avšak aplikace, které využívají další speciální služby jako například **HealthKit**, **HomeKit**, **In-app purchase** nebo konkrétně v této aplikaci **GameCenter**, musejí mít Explicitní identifikátor, který je pro každou aplikaci unikátní, a tedy se musí pro každou aplikaci vytvářet zvlášť.

5.1.2.3 Zařízení

Každé zařízení, na kterém chce vývojář spouštět aplikaci, musí být také registrováno v developerském účtu. Registruje se pouze pomocí unikátního UDID, které se dá vyčíst například v aplikaci iTunes v informacích o připojeném zařízení.

Obrázek 5.2: Ukázka obrazovky dosažených achievementů na reálném zařízení.

5.1.2.4 Provisioning profil

Nakonec se musí vytvořit nejdůležitější součást celého nastavení, a tou je Provisioning profil, který se také dělí na Development a Production. V případě této aplikace jsem vytvořil pouze Development profil, který se definuje skupinou certifikátů a zařízení, ve kterých jsem chtěl aplikaci spouštět.

5.1.3 Vytvoření samotného prostředí hry

Jelikož se při testování používá pouze **GameCenter SandBox**, je potřeba vytvořit testovací účty, pomocí kterých se uživatel může přihlásit do GameCenteru, a tak používat aplikaci. Dále se zde musí vytvořit **LeaderBoard**, který se svázán s aplikací a kam se ukládá momentální skóre, informace o uživatelích a dostupných achievementech 5.2. Nakonec je potřeba definovat různé achievementy a skóre, které je potřeba pro jejich dosažení. V samotné aplikaci se pak musí naincludovat GameKit framework, který nabízí interface pro práci se službou GameCenter.

5.1.4 Orientace displeje a obrazu z kamery

Mezi prvotní problémy při implementaci samotné detekce obličeje bylo sjednocení orientace obrazu z kamery a displeje, kde se obraz promítá. V první verzi toto přesně definováno nebylo, a tudíž docházelo k naprosto nesmyslným výsledkům detekce. Bylo tedy zapotřebí nastavit orientaci video vrstvy, která obsahuje obraz nastavit použití zadní čočky a pravou stranu na šířku pomocí **AVCaptureVideoOrientationLandscapeRight** a v samotném nastavení projektu v XCode nastavit to samé celé aplikaci.

```
CGAffineTransform transform = CGAffineTransformMakeScale(1, -1);
transform = CGAffineTransformTranslate(transform, 0, -[self.img_height floatValue]);
```

Obrázek 5.3: Vytvoření affinní transformace v jazyce obj-c.

5.1.5 Přesnost detekce obličeje

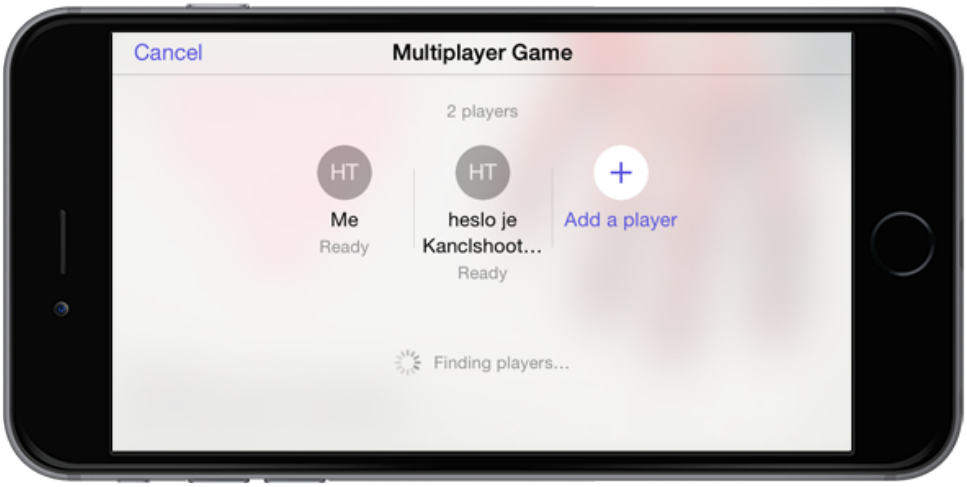
S touto částí implementace jsem strávil asi nejvíce času, protože sjednotit rozlišení displeje s rozlišením pořízeného snímku a s detekovanými souřadnicemi částí obličeje bylo velmi náročné. Navíc vše běží v reálném čase, a proto bylo obtížné uhlídat, aby se hodnoty nepřepisovaly nebo neztrácely. Dále byly ze začátku problémem i různé verze používané knihovny a velmi střídmá dokumentace, kde nejnovejší verze otáčela y-ové souřadnice, nad kterými se musela tedy provést lehká afinní transformace 5.3, která se později aplikovala na výsledné souřadnice po detekci.

5.1.6 Špatná viditelnost

Dalším menším problémem bylo dosahování horších detekčních výsledků při horších světelných podmínkách. Proto jsem se rozhodl dát uživateli možnost použít svítilnu, která obyčejně slouží jako blesk. Bohužel to nemělo na výsledky velký vliv z větší vzdálenosti, ale s tím se u tak malé svítilny ani nedalo počítat. Pro jazyk obj-c se k ovládání této svítilny používá **AVFoundation framework**, který také v této aplikaci nabízí interface pro práci z kamerou. Nová verze tohoto frameworku nabízí dokonce ovládání úrovně svícení. Byly vyzkoušeny tři z těchto úrovní, přičemž nejvyšší úroveň sloužila nejlépe.

5.1.7 Zvuky v aplikaci

Neodmyslitelnou součástí aplikace takového rázu patří samozřejmě i zvuky. O přístup k reproduktorům se také postará výše zmíněný **AVFoundation** framework, který nabízí třídu **AVAudioPlayer**, jejíž instancemi se mohou stát různé zvuky či hudba na pozadí. Umožňuje také nastavení hlasitosti přehrávaného zvuku či stopnutí kdykoliv i v průběhu přehrávání. V této aplikaci tuto třídu využívám pro hudbu, která je zapnutá po celou dobu zapnutí aplikace a také pro různé výstřely ze zbraní, aby se uživatel cítil více jako v opravdové hře. Zvuky i hudbu na pozadí má uživatel možnost zapnout či vypnout v nastavení. Prozatím jsou zvuky stažené z webu `http://www.audiomicro.com/free-sound-effects/free-guns-and-weapons`, kde byly vybrány jednoduché zvuky, které byly zdarma. Do budoucna by bylo dobré zakoupit nějaký vhodný balíček zvuků, které by byly profesionálnější.



Obrázek 5.4: Ukázka obrazovky pro spojení více hráčů na reálném zařízení.

5.1.8 Hra na více zařízeních

Bez zkoušení na více různých zařízeních, by tato aplikace nikdy nemohla vznikout, proto bylo nutné sehnat ještě nějaké další iOS zařízení (za což vděčím firmě **Ackee s. r. o.**, která mi byla ochotna půjčit všemožné zařízení od nejstarších iPhonů až po nejnovější iPady), na kterém se mohly zkoušet části aplikace, které se na simulátoru testovat nedaly. Zkoušelo se hlavně spojení telefonů do stejné hry 5.4, posílání a přepočítávání skóre a odlazení všech i nepatrných vad, které pak způsobovaly zpomalení komunikace. V současné době služba GameCenter SandBox nefunguje zcela 100 % celé dny nonstop, a tudíž se často stávalo, že se telefony nedokázaly spojit či se uživatel nemohl do testovacího účtu přihlásit. Tyto občasně výpadky nejsou hlášeny ani není šance poznat zda se jedná o chybu implementace či výpadek služby, a proto dřív tyto případy vyvolaly mnoho zbytečných problémů.

5.1.9 Real-time přepočítávání skóre

U většiny her více hráčů jsou uživatelé zvyklí na příznak pořadí, kde se nachází vzhledem k ostatním hráčům. Proto jsem se rozhodl implementovat tuto komponentu i do této aplikace, jak již bylo popsáno v kapitole Návrh. Zde bych rád zmínil, že služba GameCenter poskytuje kanál pro spojení se s hráči, kteří jsou momentálně v jedné hře. Bylo potřeba navrhnout datovou strukturu pro posílání dat a v logice posílání, aby to bylo co nejjednodušší na výpočet a na paměť. Nakonec nejlepším způsobem byla struktura **MessagePacket**, která obsahuje pole **charů** pro ID daného hráče a **int** pro uložení momentálního skóre. Po přijmutí těchto dat každé zařízení aktualizuje své skóre vůči ostatním tak, že seřadí pole (playerID, skóre) pomocí třídy **NSSortDescriptor**

Obrázek 5.5: Ukázka obrazovky konkrétní hry, skóre a pořadí konkrétního hráče na reálném zařízení.

podle nejvyššího skóre a najde svou pozici, kterou zobrazí na obrazovku 5.5.

5.1.10 Real-time komunikace

Dalším vylepšením této aplikace je i jednoduchá komunikace mezi uživateli přímo ve hře. Uživatel má možnost napsat krátkou zprávu, která se ihned objeví na ostatních spárovaných zařízeních v horní části obrazovky spolu se jménem odesílatele. Komunikace funguje na stejném principu jako přepočítávání skóre jen s rozdílem v použití struktuy **CommunicationPacket**.

5.2 Webová aplikace

Realizace webové aplikace se dělila do dvou hlavních částí, a to Backend a Frontend aplikace. Hlubší proniknutí do vytvoření webového serveru s podporou REST API a komunikace s MySQL bylo pro mě velkým přínosem, jelikož jsem s tím měl dříve pouze menší zkušenosti. Za to realizace Frontendové části byla velmi zábavná. Snažil jsem se najít a použít moderní nástroje, což se uskutečnilo a o to rychlejší a čistší celý Frontend je.

5.2.1 Správa závislostí

V analýze a návrhu této aplikace bylo již podrobně popsáno využití nástrojů pro správu závislostí v přední části aplikace. Zde bych rád konkrétně popsal použití a výhody vznikající ve společném použití těchto vybraných nástrojů. Nejprve byl vytvořen soubor **package.json**, ve kterém se definují verze dalších nástrojů, které NPM později stáhne a nainstaluje. Tímto se nainstalují

nástroje Bower a Grunt a po dokončení instalace se spustí postinstall skript, který tyto nástroje spustí. Spuštěním Boweru se rozumí nainstalování dalších závislostí definovaných v souboru **bower.json**, kde už jsou konkrétní JS knihovny, které budou využívány. Po instalaci těchto knihoven se spustí nástroj Grunt pomocí souboru **Gruntfile.js**, který definuje všechny použité soubory a jejich přípony, pokud se tyto soubory musejí dále překládat. Toto se nejvíce používá pro překládání SaSS souborů do obvyklého CSS, který se po přeložení includuje do stránky. Grunt se dále umí postarat o vyhledání správným includovaných souborů a jejich automatické umístění na předem definované místo v projektu. Všechny tyto operace se spustí příkazem **npm install**, který se už dále o vše postará, právě kvůli přesně definovaným operacím v těchto souborech. Toto se velmi hodí pro práci v týmu, kde jeden člověk vymyslí strukturu a jednotlivé operace v těchto souborech a další vývojáři už dají pouze jeden příkaz, který se o vše další postará. Při návrhu jsem čerpal také z těchto dokumentací [35] a [36].

5.2.2 Grid system

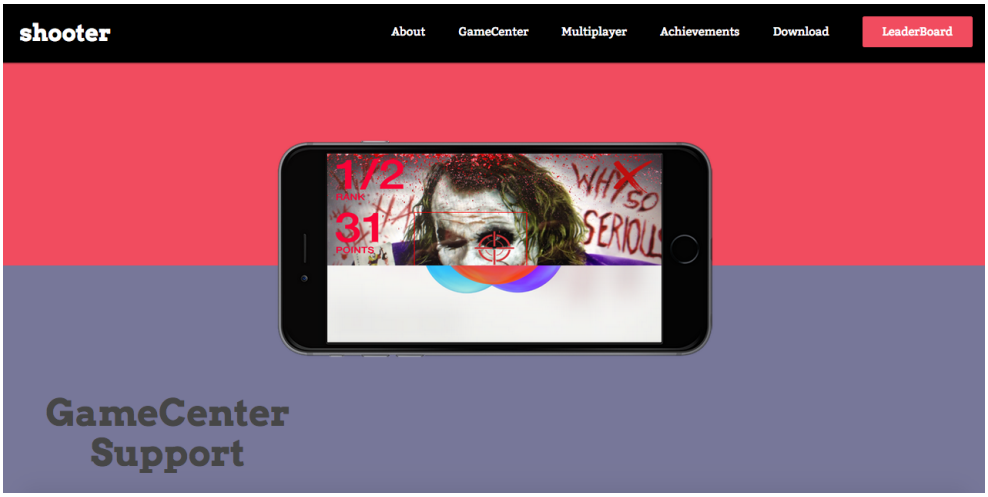
Veškeré informace o hrách, uživatelích atp. jsou zobrazovány v gridu s použitím JS knihovny **grido.js**. Tento systém umožňuje jednoduché a přehledné zobrazení dat rovnou z databáze a navíc umožňuje k samotným datům přiřazovat akční tlačítka, které můžou data mazat, upravovat či zobrazovat. Bohužel knihovna nepočítá s responsivním designem, a tudíž se data nedají přehledně zobrazovat na menších rozlišeních. Tento problém jsem prozatím vyřešil povolením scrollování obsahu tabulky s daty. Bohužel to není dostatečně intuitivní, tak se bude do budoucna muset najít nějaký elegantnější způsob.

5.2.3 Uživatelské rozhraní a design

V současné době u webových aplikací je velice důležité uživatelské rozhraní a samotný design. Proto jsem se rozhodl přečíst pár dokumentů o UI a UX (např. [31]), které mi pomohly navrhnout přívětivé uživatelské rozhraní a poskytly základy pro návrh designu. Hlavní design se skládá z vybrané palety barev pro web tak, aby barvy spolu tvořily příjemné prostředí. Dále jsem navrhl hlavní menu v záhlaví stránky, které se při scrollování ukazuje, aby bylo viditelné vždy a uživatel měl možnost se kamkoliv přesunout. Stránky obsahující informace z aplikace jsou tvořeny pomocí responsivních tabulek či výše zmíněného gridu. Hlavním grafickým prvkem na hlavní stránce je obrázek telefonu, který je ověšen efektem **parallaxu**.

5.2.4 Parallax efekt

Parallax efekt nebo podobné efekty se poslední dobou objevují na většině moderních SPA webů, a proto jsem chtěl i v této aplikaci něco podobného použít.



Obrázek 5.6: Ukázka překrývání sekcí na způsob parallax efektu.

Prošel jsem více různých knihoven pro tvorbu parallaxů, avšak žádný nepasoval pro tento případ, proto jsem se rozhodl napsat si svůj plugin sám. Velká část celé komponenty zabírá grafická část, jelikož tento konkrétní efekt je postaven na grafickém prvku (v tomto případě telefonu), který je v každé sekci jako obrázek na pozadí na přesně stejném místě. Dále se kontroluje pozice na stránce. Pokud se uživatel dostane přesně na začátek první sekce s parallaxem, přepne se pozadí těchto sekcí v CSS pomocí deklarace **background-attachment: fixed;**, která způsobí zafixování pozadí při scrollu, tudíž se pohybuje jen obsah sekce. Takto se to nastaví u každé sekce s tímto grafickým prvkem, což způsobí překrývání jednotlivých pozadí .

5.2.5 Video

V horní sekci webové aplikace se nachází video, které slouží jak pro informaci, tak i jako modernější prvek webu. Video má uživateli přiblížit funkčnost aplikace v reálné hře. Syrové video bylo natočeno a naimportováno do aplikace iMovie, která je defaultní aplikací všech Mac OS X zařízení. Následně byl odebrán zvuk a byly přidány efekty, aby video nepřitahovalo veškerou uživatelovu pozornost, která by se měla spíše zaměřit na obsahovou část. Video bylo zkomprimováno na menší velikost a následně převedeno do podporovaných formátů **mp4** a **webm**.

Kapitola 6

Testování

V této kapitole bych chtěl podrobně popsat veškeré testování, které probíhalo po i během implementace všech částí aplikace. Testování bylo zaměřeno spíše na mobilní aplikaci, jelikož na webové aplikaci se vyskytuje minimum funkčních prvků. Během usability testování byly vytvořeny testovací scénaře (součást přílohy), které testerům pomáhaly s otestováním problémových částí aplikace. Aplikaci jsem testoval i v rámci heuristické analýzy, která mi pomohla upravit zejména UI aplikace.

6.1 Heuristická analýza

Heuristická analýza patří mezi nejznámější metody usability testování. Spočívá ve srovnání aplikace s danými pravidly této analýzy, které vytvořil Jakob Nielsen [37]. Desatero definovaných pravidel zní přibližně takto:

1. **Viditelnost stavu systému**
   Uživatel by měl vždy vědět, co a kde se právě v aplikace děje.

2. **Propojení systému a reálného světa**
   Jazyk, kterým aplikace komunikuje s uživatelem by měl být uživateli známý a měl by působit reálně a logicky navazovat.

3. **Uživatelská kontrola a svoboda**
   Uživatel by měl mít vždy možnost vrátit se do stavu, ze kterého omylem vystoupil špatným směrem.

4. **Standardizace a konzistence**
   Je důležité dodržovat stejná slova či akce pro stejné prvky na různých místech. Tatáž věc by neměla být označována více různými pojmy.

5. **Prevence chyb**
   Uživatel by v nejlepším případě neměl mít možnost vyvolat v aplikaci

error. Pokud se tomu nedá vyhnout, měly by být vytvořeny minimálně potvrzovací okna před vyvoláním konkrétní akce.

6. **Rozpoznání namísto vzpomínání**
   Vždy by mělo být vidět, kde se uživatel nachází. Aplikace by neměla počítat s tím, že si uživatel musí pamatovat kroky, jakými se do současného stavu dostal.

7. **Flexibilní a efektivní použití**
   Aplikace by měla počítat jak se zkušeným uživatelem tak i s nezkušeným. Měla by i zkušenějšímu uživateli nabízet možnosti zrychlených akcí, které častěji využívá.

8. **Estetický a minimalistický design**
   Aplikace by neměla obsahovat zbytečné prvky, či prvky které jsou špatně rozložené, které pak vedou ke zmatení uživatele.

9. **Pomoc uživatelů pochopit, poznat a vzpamatovat se z chyb**
   Pokud nastane problém, měl by být jasně specifikovaný normálním jazykem bez počítačových kódů. Dále by mělo být navrhnuto nějaké řešení.

10. **Nápověda a návody**
    Dobrá aplikace by měla být použitelná i bez návodů. Pokud se bez návodu neobejde, měl by návod být přesný a ne zbytečně dlouhý.

6.2 Usability testování

Usability testování neboli testování použitelnosti odhaluje chyby sledováním testera při používání aplikace. Často se používá již během vývoje, aby se vývojář vyvaroval podobným chybám v další implementaci. Celé testování se skládá z následujících částí:

1. **Vymezení potenciálních uživatelů**
   Je důležité vymezit různé typy uživatelů, kteří budou aplikaci používat. Testovat pak budou tito cílení uživatelé či testeři, kteří budou jejich chování simulovat.

2. **Scénáře testování**
   Nutnou součástí je i scénář, ve kterém se definují konkrétní úkony, které testeři pak vypracovávají.

3. **Testeři**
   Kvalita testování neznamená vždy velká kvantita testerů. Při výběru se musí zohlednit různé typy testerů s různými zkušenostmi.

4. **Testování**
   Testování probíhá tak, že testeři konají postupně všechny kroky ze scénáře. Je lepší, když samotný vývojář nebo nějaký specialista testera pozoruje po celou dobu testování.

5. **Analýza testování**
   Výstupy z testování se zhodnotí a zkusí se najít nejlepší řešení.

6. **Výsledky**
   Na závěr se musí všechny poznatky sesumarizovat a přetvořit do přehledné formy vhodné pro prezentaci.

6.3 Mobilní aplikace

Testování mobilní aplikace probíhalo několik měsíců, jelikož probíhalo i během implementace. Na webu `https://itunesconnect.apple.com` pod svým účtem jsem si povolil službu TestFlight, která slouží k beta testování interními i externími testery.

6.3.1 TestFlight

Tato poměrně nová služba dříve poskytovala prostředí pro testování jak Android tak i iOS aplikací, avšak v březnu 2014 Apple tuto službu odkoupil a od té doby slouží k testování pouze iOS aplikací, jelikož je použitelná pouze prostřednictvím XCode. Od té doby je také součástí iTunesConnectu, což velmi usnadňuje práci. Tuto službu jsem začal používat od prvního funkčního hi-fi prototypu, abych se díky zpětné vazbě testerů více zaměřil na problémové úseky. Informace jsem čerpal hlavně z oficiálního zdroje [38]. Služba funguje tak, že prostřednictvím aplikace XCode nahrajete funkční build na oficiální server Apple, a dále je tento build viditelný v iTunesConnect, kde se definuje seznam testerů 6.1 a veškeré pokyny k testování, což je dále posláno testerům přímo do jejich zařízení prostřednictvím emailu. Testeři poté mohou posílat zpětné vazby, kde popisují problémy vzniklé při testování aplikace.

6.3.2 Výsledky testování

Mobilní aplikace je největší součástí celé práce, tudíž se v rámci testování objevilo více problémů, z nichž vyberu jen ty nejzávažnější, které detailněji přiblížím. Všechny dále zmíněné problémy byly úspěšně vyřešeny, jen v případě problému špatných světelných podmínek dosažené řešení nedosahuje nejoptimálnějšího výsledku.

Obrázek 6.1: Seznam externích testerů povolených pro testování verze 1.9.

6.3.2.1 Detekce na různých zařízeních

Největším problémem zjištěným při testování je špatné detekování obličeje na různých zařízeních způsobené různým rozlišením. Tento problém bylo velice obtížné řešit, jelikož jsem měl pouze své zařízení iPhone 4S. Velice mi pomohla podpora firmy Ackee se zapůjčením všech typů iOS zařízení. Na obrázku 6.2 můžete vidět výsledek detekce na zařízení iPad Mini. Pro výraznější objasnění problému jsem přidal zaměřovací body, a to červený rám simulující hrany obličeje a zelené čtverečky pro simulaci očí. Je zřejmé, že x-ové souřadnice červeného čtverce odpovídají, avšak y-ové nikoliv. Dále je vidět, že zelené čtverce jsou vzhledem k červenému v dobré poloze. Problém byl v nekonzistenci velikosti obrazu v zařízení vzhledem k rozlišení pořízeného snímku, na kterém běžela detekce. Nejrychlejším řešením by bylo definovat přesné rozlišení snímku, avšak muselo by se počítat pouze s nejvyšším rozlišením, které umí nejstarší iOS zařízení. Toto řešení je ovšem nedostačující, jelikož pak uživatel s novějším zařízením nemá žádnou výhodu, která se s lepším fotoaparátem dá dosáhnout. Proto jsem vytvořil pomocnou funkci, která upravuje x-ovou souřadnici pomocí podílu šířky displeje, avšak to také nebylo dostačující, proto se

Obrázek 6.2: Ukázka špatné detekce na zařízení iPad Mini.

museloo u pár konkrétních typů upravovat pomocnými konstantami. Aplikace již funguje přesně na všech zařízeních, bohužel ale nevýhodou je nutná editace v případě oznámení nového zařízení společností Apple.

6.3.2.2 Nefunkční přihlašování

Tento problém nastával hned u první testovací verze, a byl zapříčiněn nedostatečnou informovaností testerů. Jelikož je aplikace v procesu beta testování, služba GameCenter dovoluje použití pouze Sandbox testovacího prostředí. Tudíž testeři měli sice k dispozici testovací účty, avšak nedostaly informaci o přepnutí se do Sandbox prostředí, které se dělá v nastavení konkrétního zařízení. Dále pak chyběla informace, pokud tester neměl přístup k síti, a tudíž se nemohl do svého účtu přihlásit.

6.3.2.3 Neúspěšné spojení s ostatními hráči

Podobně jako u předchozího problému zde byl největší problém v nedostatečně zpracovaném manuálu. Hráči dostali přibližně tři různé testovací účty, avšak pokud se více hráčů přihlásilo na tentýž účet, a pak se pokoušeli spojit, systém jim to neumožňoval. Dalším problémem byly časté výpadky GameCenter Sandboxu, které se nedají předpovídat ani jednoduše zjistit. Systém pak neumožní nalézt jiné hráče bez udání důvodu.

6.3.2.4 Občasné nevykreslení hlavní zbraně ve hře

Nevykreslení zbraní na nějakých zařízeních byl problém implementační, jelikož docházelo k nesprávnému uvolňování paměti. Proto se na nějakých zařízeních tlačítko zbraně uvolňovalo ihned po vykreslení a vývojář to ani nestačil postřehnout. Postupným opravováním se zjistilo, že chyba nastává jen u zařízení s retina displejem, což je mi doteď záhadou i po konzultaci s profesionály z této branže. Ovšem i tento problém se podařilo opravit.

6.3.2.5 Nemožnost změny nastavení v průběhu hry

Testeři narazili na zcela nelogický problém, týkající se špatného návrhu UI aplikace. Hráč, který je již v započaté hře, nemohl hru přerušit a změnit různá nastavení bez toho, aby hru zcela ukončil. Tento problém byl vyřešen ponecháním běhu hry na pozadí do doby, kdy uživatel sám hru po dodatečném upozornění neukončí.

6.3.2.6 Detekce za špatných světelných podmínek

Již dříve byl zmíněn problém detekce za horších světelných podmínek. Do další testovací verze byla přidána možnost rozsvícení svítilny zařízení, avšak na opravdu špatné podmínky ani tato funkčnost nestačí.

6.3.2.7 Přidání modálního okna při výběru zbraně

Při výběru jiné zbraně je důležitý počet bodů, které uživatel za hru nasbíral. Systém následně vyhodnotí, jestli má uživatel pro danou zbraň dostatek bodů. V chybné verzi mohl nastat stav, kdy uživatel chtěl vybrat zbraň s nedostatkem bodů, systém následně zbraň nedovolil, avšak uživateli o tom neposkytl žádnou informaci.

6.3.2.8 Zobrazování pořadí i v single hře

Chybné zobrazování pořadí bylo další chybou UI, kdy se před vykreslením pořadí systém díval, jestli hraje jeden čí více hráčů. V prvním spuštěním single hry se tato chyba stát nemohla, avšak pokud uživatel hrál hru více hráčů a následně si spustil hru pro jednoho, systém byl zmaten, jelikož se špatně mazala paměť.

6.4 Webová aplikace

Testování webové aplikace probíhalo pouze na Frontendu, jelikož samotná funkčnost backendu by se musela podrobit jednotkovým testům, které by pro

tento systém neměly žádoucí účinky. Na Frontendu se testovalo hlavně z hlediska uživatelského prostředí, pochopení struktury webu a rozložení funkčních prvků.

6.4.1 Výsledky testování

Jelikož se jedná o webovou aplikaci s minimálním množstvím funkčních prvků, výsledků testování nebylo mnoho a jednalo se spíše o změnu rozložení prvků či jejich úpravu. Všechny dále zmíněné problémy byly úspěšně vyřešeny, jen v případě problému tabulek není momentálně řešení zcela dostačující.

6.4.1.1 Tabulky s informacemi v mobilních zařízeních

Důležitou součástí webové aplikace je velké množství všech informací o hráčích a hrách, které se zobrazuje pomocí tabulek a gridů. Ovšem navrhnutá struktura neumožňuje přehledné zobrazení všech informací na malém rozlišení např. u mobilních zařízení. Nejlepší řešením tohoto problému by bylo úplné předělání struktury tabulek v případě menších rozlišení, avšak při velkém množství dat by se web stal naprosto nepřehledným. Problém byl zatím vyřešen v podobě posuvníků, které umožňují posouvat obsahy tabulek stále uvnitř obsahu webu. Dále byla přidána možnost vyhledávání a filtrování hráčů pro větší přehlednost.

6.4.1.2 Různé rozlišení

Různá rozlišení způsobila řadu problémů i při testování webové aplikace. Parallax efekt, jehož funkčnost byla popisována již dříve, měl nastavené statické hodnoty, při kterých se fixuje pozadí sekcí s telefonem. Tyto hodnoty odpovídají vzdálenosti začátku této sekce od vrcholu celé stránky. Samozřejmě na jiných rozlišeních tyto hodnoty neodpovídaly, proto se musela navrhnout funkce, která dynamicky počítá součet výšek předchozích sekcí, čímž se dostane hodnota vrcholu dané sekce. U mobilních zařízení se bohužel nepodařilo tento efekt docílit, tudíž se telefony scrollují spolu s obsahem po pravé straně.

6.4.1.3 Video

Video způsobovalo při testování také řadu problémů. Zaprvé se nezobrazovalo na stejné pozici v různých prohlížečích, což bylo opraveno funkcí detekce prohlížeče a následným ošetřením a zadruhé ve větším množství mobilních zařízení video nefungovalo v žádném vyzkoušeném formátu. Proto bylo rozhodnuto nahrazení videa statickým obrázkem v mobilních zařízeních.

6.4.1.4 Absence návratového odkazu

Dále byl zjištěn problém absence odkazu na hlavní stránku ze sekcí s tabulkami informací o hráčích. Na hlavní stránce se ukazuje pouze jedna tabulka

s informacemi deseti nejlepších hráčů spolu s odkazem na další data. Avšak po přesunutí prostřednictvím tohoto odkazu nebyla žádná možnost návratu kromě tlačítka zpět v prohlížeči.

6.5 Google Analytics

Google Analytics je oficiální nástroj vytvořený společností Google, který slouží k zachycení veškerých příchodů a předdefinovaných událostí způsobených uživatelem webové aplikace. Dá se kontrolovat návštěvnost jednotlivých stránek, země či zařízení, ze kterých je k webu přistupováno, i doba, kterou uživatel na webu strávil. Nástroj také umožňuje sledování chování uživatelů, prokliky mezi stránkami či z jaké stránky nejčastěji uživatelé web opouští. Všechna tato kritéria mohou vývojáři pomoci s následným zlepšením problémových pasáží, proto i na tento web GA nasazeny a používány. Zatím není aplikace zveřejněna, tudíž mohu vyvozovat závěry pouze z návštěv testerů, pomocí čehož byl například potvrzen výše zmíněný problém absence návratového odkazu, jelikož nejčastější opouštění aplikace bylo právě z těch sekcí, odkud nebylo návratu.

Kapitola 7

Rozšíření

V průběhu návrhu a implementace všech částí aplikace vznikalo plno dalších nápadů na vylepšení či doplnění aplikace, avšak na jejich realizaci by nestačil čas, který je pro Diplomovou práci vyhrazen. I nadále bych se této aplikaci chtěl věnovat a dále jí rozvíjet, aby se časem mohla rozšířit mezi další uživatele. Do té doby bych se rád zaměřil na následující další komponenty, které by celé aplikaci přidaly na funkčnosti i zajímavosti.

7.1 Další platformy

Jelikož jsem již v první kapitole procházel různé možnosti návrhu i na ostatních platformách, chtěl bych se postupně posunout i tímto směrem a aplikaci zkusit rozšířit i tímto směrem. Nevýhodou je nemožnost použití již naprogramovaných komponent v obj-c. Avšak díky nezávislosti webové aplikace webový server, Frontend i celá databáze může zůstat stejná.

7.2 Design mobilní aplikace

Hlavní změnou aplikace před případným zveřejněním je změna designu mobilní aplikace. Momentálně se v ní objevuje spoustu primitivní grafiky, kterou by měl nahradit nový design, avšak s použitím současného návrhu podle wireframů. Design by měl být více provázán s grafickou částí webové aplikace a neměl by být závislý na použité platformě. Jelikož se jedná o herní aplikaci, z hlediska UI nemusí být tak striktně dodržovány platformní prvky.

7.3 Push notifikace pro zastřeleného

V původním nápadu se již vyskytovala myšlenka notifikací a odebírání bodů zasaženého uživatele. Tato komponenta měla využívat ke zjištění totožnosti uživatele GPS modul a gyroskop, jejíchž kombinací by se teoreticky mohl

najít nejbližší protihráč ve směru střelby. Bohužel v současné době GPS modul je pro takovouto práci naprosto nedostačující, hlavně pokud hráči hrají v nějakém uzavřeném prostoru. Další technologie, které jsem analyzoval pro zjišťování totožnosti zasaženého uživatele jsou Face Recognition a technologie iBeacon.

7.4 Face recognition

Face recognition je technika na rozpoznání obličeje na fotografii a následným vyhledáním uživatele v databázi. V současné době tuto funkcionalitu nabízejí pouze dvě knihovny, a to **OpenCV** a **Kairos SDK**. Implementace bude velmi náročná jak na čas tak na pamět, ale v případě úspěchu by tato komponenta mohla aplikaci posunout daleko vpřed.

7.5 Použití technologie iBeacon

iBeacon je nová technologie, která zlepšuje použití polohových služeb. Jsou to předměty se zakomponovaným vnitřním systémem s podporou Bluetooth 4, neustále vysílající signál, který ostatní zařízení dokáží zachytit. Podle síly signálu pak zařízení dokáže vyhodnotit vzdálenost od předmětu. S použitím této technologie by bylo daleko snažší rozeznat trefeného hráče, jelikož předměty mají své unikátní identifikátory, které by se předem uložily do databáze. Avšak problémem je cena těchto předmětů, která se pohybuje okolo 700 Kč za kus. Koncoví uživatelé by byli aplikací tlačeni ke koupi těchto předmětů, což by určitě nevyvolalo kladný ohlas. Jako nejlepší řešení se zdá použití těchto předmětů při různých prezentacích této aplikace.

Závěr

Po několika měsících vznikla funkční mobilní aplikace, která umožňuje uživatelům hrát samostatnou hru či se spojit s více hráči. Byl také vytvořen funkční Backend aplikace, který se stará o komunikaci s databází, kam ukládá veškeré informace o hráčích a hrách. Server také nabízí REST API s předdefinovanými endopointy, na které se napojuje mobilní aplikace a přenáší veškerá data. Pro zobrazení těchto dat a dalších informací o aplikaci slouží webová aplikace, která je dosažitelná ze všech stolních i mobilních zařízení a díky své responsivitě i na všech různých rozlišeních. Uživatel má zde možnost dozvědět se veškeré informace o odehraných hrách a hráčích nebo i více o samotné aplikaci či na konkrétní ukázce shlédnout jak aplikace funguje a následně si ji stáhnout do svého zařízení. Snažil jsem se práci zpracovat tak, aby byla do budoucna použitelná pro další potenciální studenty, kteří se tomuto nebo podobnému tématu budou věnovat. Obzvláště služba GameCenter je v současné době pro real-time hry více hráčů probádaná míň než se ze začátku zdálo, narozdíl od jejího použití pro obyčejné single hry jednoho hráče. I přes prvotní nezkušenost s těmito technologiemu a větším množstvím objevených problémů bych chtěl poděkovat za možnost práce na takovém projektu, jenž byl zároveň velmi poučný i zábavný a upřímně doufám, že aplikace dokáže zapůsobit na co nejvíc lidí z celého světa.

Literatura

[1] Médiář.cz: *Z aplikací studentů ČVUT vybrán i KanclShooter*. [cit. 2015-03-06]. Dostupné z: `http://www.mediar.cz/z-aplikaci-studentu-cvut-vybran-i-kancl-shooter/`

[2] Bonsor, K.: *Augumented reality*. [cit. 2015-03-06]. Dostupné z: `http://computer.howstuffworks.com/augmented-reality.htm`

[3] PSOFT MOBILE: *AR Missile - Automatic Target Tracking*. [cit. 2015-03-06]. Dostupné z: `http://psoftmobile.net/en/armissile.html`

[4] Soulbit7: *AR Invaders*. [cit. 2015-03-06]. Dostupné z: `http://invaders.soulbit7.com/`

[5] Yii Studio: *Real Strike - The Original 3D Augmented Reality FPS Gun App*. [cit. 2015-03-06]. Dostupné z: `https://itunes.apple.com/us/app/real-strike-original-3d-augmented/id507884100?mt=8`

[6] Bendy Tree, LLC: *Lazer Tag!!!* [cit. 2015-03-06]. Dostupné z: `https://itunes.apple.com/CZ/app/id420570108?mt=8`

[7] Google Inc.: *Play Gmes Services for Android*. [cit. 2015-02-10]. Dostupné z: `https://developers.google.com/games/services/android/realtimeMultiplayer`

[8] Apple Inc.: *GameKit Framework*. [cit. 2015-02-11]. Dostupné z: `https://developer.apple.com/library//ios/documentation/GameKit/Reference/GameKit_Collection/index.html`

[9] Apple Inc.: *Game Center*. [cit. 2015-02-11]. Dostupné z: `https://developer.apple.com/game-center/`

[10] Gite, V.: *Difference between TCP and UDP*. [cit. 2015-03-15]. Dostupné z: `http://www.cyberciti.biz/faq/key-differences-between-tcp-and-udp-protocols/`

[11] Bricklin, D.: *Network topologies*. [cit. 2015-03-15]. Dostupné z: http://www.bricklin.com/p2ptaxonomy.htm

[12] Carrisson, A.: *Using websockets in native mobile apps*. [cit. 2015-01-20]. Dostupné z: http://www.elabs.se/blog/66-using-websockets-in-native-ios-and-android-apps

[13] Thompson, M.: *Multipeer connectivity*. [cit. 2015-01-20]. Dostupné z: http://nshipster.com/multipeer-connectivity/

[14] Jahnen, M.: *Different ways to make multiplayer game with iOS*. [cit. 2015-01-20]. Dostupné z: http://www1.in.tum.de/lehrstuhl_1/people/98-teaching/tutorials/508-sgd-ws13-tutorial-multiplayer-games

[15] W3C: *AJAX Introduction*. [cit. 2015-04-11]. Dostupné z: http://www.w3schools.com/ajax/ajax_intro.asp

[16] Nette: *Routování v Nette frameworku*. [cit. 2015-04-28]. Dostupné z: http://doc.nette.org/cs/2.3/routing

[17] MySQL.com: *Top Reasons to Use MySQL*. [cit. 2015-04-28]. Dostupné z: https://www.mysql.com/why-mysql/topreasons.html

[18] Long, J.: *Frontend vs. Backend*. [cit. 2015-04-28]. Dostupné z: http://blog.teamtreehouse.com/i-dont-speak-your-language-frontend-vs-backend

[19] Cox, P.: *10 reasons to use HTML5*. [cit. 2015-04-28]. Dostupné z: http://tympanus.net/codrops/2011/11/24/top-10-reasons-to-use-html5-right-now/

[20] 1stwebdesigner: *New features in CSS3*. [cit. 2015-04-28]. Dostupné z: http://www.1stwebdesigner.com/css3-introduction/

[21] Michálek, M.: *CSS3 Media Queries – podmíněné zobrazení pro média*. [cit. 2015-04-28]. Dostupné z: http://www.vzhurudolu.cz/prirucka/css3-media-queries

[22] 1stwebdesigner: *How Fluid Grids Work in Responsive Web Design*. [cit. 2015-04-28]. Dostupné z: http://www.1stwebdesigner.com/fluid-grids-in-responsive-design/

[23] Hernandez, G.: *How to Choose the Right CSS Preprocessor*. [cit. 2015-04-28]. Dostupné z: http://blog.teamtreehouse.com/how-to-choose-the-right-css-preprocessor

[24] The Front-end Tooling Book: *Book of Modern frontend tooling*. [cit. 2015-04-28]. Dostupné z: http://tooling.github.io/book-of-modern-frontend-tooling/index.html

[25] Pearce, S.: *Git and SVN comparison.* [cit. 2015-02-18]. Dostupné z: https://git.wiki.kernel.org/index.php/GitSvnComparison

[26] apiary: *API Blueprint tutorial.* [cit. 2015-04-30]. Dostupné z: https://apiary.io/blueprint

[27] Cocoapods.org: *Using CocoaPods.* [cit. 2015-04-30]. Dostupné z: https://guides.cocoapods.org/using/using-cocoapods.html

[28] Thompson, M.: *AFNetworking 2.0.* [cit. 2015-04-30]. Dostupné z: http://nshipster.com/afnetworking-2/

[29] Skřivan, J.: *Databáze a jazyk SQL.* [cit. 2015-04-30]. Dostupné z: https://www.interval.cz/clanky/databaze-a-jazyk-sql/

[30] Valdarrama, S. L.: *Is a Single-Page Application what you really need?* [cit. 2015-04-30]. Dostupné z: https://blog.svpino.com/2014/10/15/is-a-single-page-application-what-you-really-need

[31] Toxboe, A.: *How to get better at UI and UX design.* [cit. 2015-04-30]. Dostupné z: http://ui-patterns.com/blog/How-to-get-better-at-UI-design

[32] Jan Paepke: *Parallax Scrolling.* [cit. 2015-04-30]. Dostupné z: http://janpaepke.github.io/ScrollMagic/examples/advanced/parallax_scrolling.html

[33] Statista Inc.: *Number of monthly active Facebook users worldwide.* [cit. 2015-04-30]. Dostupné z: http://www.statista.com/statistics/264810/number-of-monthly-active-facebook-users-worldwide/

[34] Statista Inc.: *Number of monthly active Twitter users worldwide.* [cit. 2015-04-30]. Dostupné z: http://www.statista.com/statistics/282087/number-of-monthly-active-twitter-users/

[35] Adams, C.: *How I Use Grunt And Bower for Frontend Packages.* [cit. 2015-04-30]. Dostupné z: http://www.realchaseadams.com/2013/11/grunt-build-and-bower-package-management/

[36] Hayaran, N.: *Getting Started With Grunt & Bower.* [cit. 2015-04-30]. Dostupné z: http://www.nitinh.com/2013/05/getting-started-with-grunt-bower/

[37] Nielsen, J.: *Heuristic Evaluation and expert reviews.* [cit. 2015-05-10]. Dostupné z: http://www.nngroup.com/articles/ten-usability-heuristics/

[38] Apple Inc.: *TestFlight Beta Testing.* [cit. 2015-05-10]. Dostupné z: https://developer.apple.com/testflight/

Příloha A

Seznam použitých zkratek

REST Representational state transfer

API Application Programming Interface

iOS Operační systém vytvořený společností Apple Inc

NPM Node Package Manager

AR Augumented reality

GPS Global Positioning System

RPG Role-playing game

UFO Unidentified Flying Object

TCP Transmission control protocol

UDP User datagram protocol

TLS Transport Layer Security

PHP PHP: Hypertext Preprocessor

SQL Structured Query Language

AJAX Asynchronous JavaScript and XML

MVC Model-view-controller

URL Uniform Resource Locator

SEO Search Engine Optimization

HTML Hypertext markup language

CSS Cascading Style Sheets

SVN Subversion

HTTP Hypertext Transfer Protocol

JSON JavaScript Object Notation

DSL Domain specific language

UI User interface

EER Enhanced entity–relationship model

ER Entity–relationship model

URI Uniform Resource Identifier

XML Extensible Markup Language

RSS Rich Site Summary

ATOM Atom Syndication Format

SoC Separation of concerns

SPA Single-page application

JS Javascript

DOM Document Object Model

YUM The Yellowdog Updated, Modified

APT Advanced Packaging Tool

PR Public relations

IT Information technology

UX User experience

GA Google analytics

SDK Software development kit

Příloha B

Testovací scénáře

V prvních týdnech implementace byly tvořeny testovací verze, které vždy obsahovaly popis, co a jak mají testeři zkoušet. Tento postup nebyl však dostatečný. V kapitole Testování byla zmíněna tvorba testovacích scénářů, které sloužily testerům jako pomocný materiál. S poslední testovací verzí aplikace byly vytvořeny scénáře pro testování jak mobilní tak webové aplikace, ale i pro testování celkového chování aplikace jako celku.

B.1 Mobilní aplikace

1. **Instalace a spuštění**
   - Nejprve otestujte instalaci a spuštění testovací verze pomocí aplikace TestFlight, kterou si nainstalujete do svého zařízení a následně se přihlásíte stejným emailem, který je zaregistrován mezi interními či externími testery.

2. **Přihlášení do aplikace**
   - Přihlašte se do aplikace pomocí svého Game Center účtu. Nezapomeňte přepnout v nastavení zařízení použití Game Center Sandbox (testovací prostředí služby Game Center). Uživatelské údaje jméno/heslo jsou **tester2@shooter.cz/Kanclshooter1**.

3. **Změna nastavení**
   - Zkuste změnit různé parametry v nastavení. Zapnutí/vypnutí zvuků či hudby v aplikaci. Přihlášení či odhlášení z Game Centeru.

4. **Hra pro jednoho**
   - Vyzkoušejte hru jednoho hráče. Zejména přesnost zaměřování a správné přičítání bodů.

5. **Svítilna**
   - Otestujte funkčnost zapínání svítilny, zejména v horších světelných podmínkách. Pokud možno otestujte i ve tmě.

6. **Prohlídka leaderboardu**
   - Zkuste prohlížet dosažené achievementy, statistiky vašich her či přátel.

7. **Změna zbraní**
   - Změňte hlavní zbraň za jinou s různým dosaženým skóre. Každá nová zbraň by měla jít změnit po dosažení padesáti bodů. Každá zbraň přidává jiný počet bodů.

8. **Hra více hráčů**
   - Hrajte hru více hráčů. Spojte se s ostatními hráči v okolí pomocí automatické hry.

9. **Pořadí**
   - Otestujte správnou změnu pořadí ve více hráčích.

10. **Komunikace**
    - Posílejte a čtěte zprávy s ostatními hráči během hry pomocí kliku na text v horní části herní obrazovky.

B.2 Webové aplikace

1. **Prohlídka informací**
   - Prohlížejte informace zobrazované na hlavní stránce na různých rozlišeních.

2. **Video**
   - Otestujte zobrazení videa na různých rozlišeních či zařízeních.

3. **Menu**
   - Přesunujte se pomocí normálního či vyjíždějícího menu do různých sekcí i na různých rozlišeních.

4. **Parallax**
   - Pozorujte chování parallax efektu při přejíždění mezi různými sekcemi.

5. **Statistiky**
   - Prohlížejte statistiky hráčů, her a dalších informací.

6. **Mobilní verze**
   - Testujte na různých mobilních i stolních zařízeních.

B.3 Celkové flow aplikace

1. **Přičítání skóre**
   - V mobilní aplikaci spusťte hru pro jednoho čí více hráčů a po přičtení skóre otestujte, jestli se na webu daný bodový zisk aktualizoval.

2. **Aktualizace achievementů**
   - Získejte další achievementy a otestujte na webu na svém profilu, jestli se správně aktualizují.

3. **Ukládání her**
   - Hrajte hru jednoho čí více hráčů a po ukončení hry v mobilní aplikaci zkontrolujte správné ukládání informací na webu.

Příloha C

Instalační příručka

C.1 Mobilní aplikace

Bez developerského účtu může být docíleno pouze zkompilování aplikace a spuštění na interním simulátoru v aplikaci XCode.

C.1.1 Instalace podů

V práci byly již zmíněny externí knihovny (pody), které se do aplikace instalují službou CocoaPods. Pro úspěšné importování knihoven je nutné v hlavní složce mobilní aplikace spustit příkaz **pod install**, k čemuž je potřeba mít dříve nainstalován ruby gem CocoaPods.

C.1.2 Spuštění aplikace

Při otevírání projektu je důležité otevřít v programu XCode soubor **Shooter.xcworkspace** nikoliv soubor Shooter.xcodeproj. Vše ostatní by již mělo být připraveno, tudíž pro úspěšný build aplikace stačí spustit projekt.

C.1.3 Spojení přes API

Aplikace je již spojena s webový rozhraním, které je dosažitelné na `http://shooter.webcrafting.cz`. Pokud chcete testovat funkčnost posílání a ukládání dat, je nutné testovat přímo na tomto serveru.

C.2 Webová aplikace

C.2.1 Server

Pro běh aplikace je třeba mít v provozu Apache server s podporou PHP 5.3 a MySQL 5.5. Apache server musí mít povoleny .htaccess příkazy. Vyzkoušen je například program XAMPP pro Windows – program stačí nainstalovat a

spustit potřebné služby. Pro Mac má obdobnou funkčnost program MAMP. V Linuxu lze vše potřebné nainstalovat přímo z konzole pomocí instalačních balíčků.

C.2.2 Databáze

Je nutné mít vytvořenou databázi v InnoDB režimu (podpora cizích klíčů). Vytvoření tabulek lze provést importem souboru
src/application/web/db/0/new_structure.sql na přiloženém CD. Součástí scriptu je i insert základních testovacích dat.

C.2.3 Instalace aplikace

Pro instalaci aplikace je potřeba nakopírovat celý adresář src/application/web z přiloženého CD na server. Po nakopírování se nastaví přístup k databázi v souboru /app/config/config.neon. Je nutné ještě nastavit práva na zápis složkám: /temp a /logs.
Aplikace je nyní dostupná z webového rozhraní v adresáři root/www.

Příloha D

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `src`
  - `application`
    - `mobile` ..... zdrojové kódy implementace mobilní aplikace
    - `web` ..... zdrojové kódy implementace webové aplikace
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... text práce
  - `baierjak.pdf` ..... text práce ve formátu PDF
- `appendix` ..... přílohy