UNIVERZITA PARDUBICE

FAKULTA ELEKTROTECHNIKY
A INFORMATIKY

BAKALÁŘSKÁ PRÁCE

2023 Jiří Hejduk

Univerzita Pardubice

Fakulta elektrotechniky a informatiky

Vývoj geolokační sběratelské mobilní hry

Bakalářská práce

2023 Jiří Hejduk

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: 2022/2023

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Jiří Hejduk**
Osobní číslo:	**I20092**
Studijní program:	**B0688A140009 Informační technologie**
Téma práce:	**Vývoj geolokační sběrtelské mobilní hry**
Zadávající katedra:	**Katedra informačních technologií**

Zásady pro vypracování

Cílem práce je návrh a implementace mobilní hry, ve které je cílem hráče sbírat jednotlivé tokeny, které se nacházejí poblíž turistických míst. Při nalezení a načtení tokenu v podobě QR kódu aplikací, uživatel získává odměnu a dané místo se mu přidá do virtualní sbírky. Hra umožňuje také porovnání hráčů na základě nasbíraných bodů, zobrazení jednotlivých turistických míst na mapě, virtualní sbírku navštívených míst a správu uživatelských profilů.

Rozsah pracovní zprávy: **min. 30 stran**
Rozsah grafických prací:
Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam doporučené literatury:

MURPHY, Mark L. *Android 2: průvodce programováním mobilních aplikací*. Přeložil Jakub MUŽÍK. Brno: Computer Press, 2011. ISBN 978-80-251-3194-7.

Ebel, N. *Mastering Kotlin: Learn advanced Kotlin programming techniques to build apps for Android, iOS, and the web*. Birmingham, UK, 2019. ISBN 978-1838555726.

STEPHENS, Ryan K. a Ronald R. PLEW. *Naučte se SQL za 21 dní: pochopte principy jazyka relačních databází : uplatněte získané dovednosti při tvorbě dotazů a databázových aplikací*. Brno: Computer Press, 2004. ISBN 80-722-6870-8.

Vedoucí bakalářské práce: **Ing. Jan Panuš, Ph.D.**
Katedra informačních technologií

Datum zadání bakalářské práce: **16. prosince 2022**
Termín odevzdání bakalářské práce: **12. května 2023**

L.S.

Ing. Zdeněk Němec, Ph.D. v.r.
děkan

Ing. Jan Panuš, Ph.D. v.r.
vedoucí katedry

V Pardubicích dne 28. února 2023

Prohlašuji:

Práci s názvem Vývoj geolokační sběratelské mobilní hry jsem vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Beru na vědomí, že v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů, a směrnicí Univerzity Pardubice č. 7/2019 Pravidla pro odevzdávání, zveřejňování a formální úpravu závěrečných prací, ve znění pozdějších dodatků, bude práce zveřejněna prostřednictvím Digitální knihovny Univerzity Pardubice.

V Pardubicích dne 12.05.2023

Jiří Hejduk

PODĚKOVÁNÍ

Tímto bych chtěl poděkovat Ing. Janu Panušovi, Ph.D. za odborné vedení a konzultace při vypracování této bakalářské práce. Dále také děkuji své rodině, partnerce a všem přátelům, kteří mě v průběhu studia podporovali a pomáhali mi.

ANOTACE

Bakalářská práce se zaměřuje na návrh a implementaci mobilní geolokační turistické sběratelské hry, ve které hráč hledá jednotlivé známky v podobě QR kódů umístěných v blízkosti turistických míst. Mobilní aplikace je vytvořena pro operační sytém Android. Serverová část aplikace je navržena podle návrhového vzoru mikroslužeb. Server komunikuje s mobilní aplikací pomocí REST API.

KLÍČOVÁ SLOVA

Hra, Android, Kotlin, mikroslužby, Go, REST, API, Docker, PostgreSQL

TITLE

Development of geolocation collectible mobile game

ANNOTATION

The bachelor thesis is focused on design and implementation of mobile geolocation tourist collecting game, which is used by player to collect QR code stamps, which are located near tourist destinations. Mobile application is implemented for Android operating system. The server side of the application is designed following microservices application pattern. The server communictes with the mobile app by REST API.

KEYWORDS

Game, Android, Kotlin, microservices, Go, REST, API, Docker, PostgreSQL

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM ZKRATEK

Zkratka	Význam
API	Application Programming Interface
AR	Augmentovaná realita
CCG	Collectible card game
CLI	Command-line Interface
F2P	Free-to-Play
GNU	GNU's Not Unix
GPS	Global Positioning System
GUI	Graphical User Interface
IDE	Integrated Development Environment
JSON	JavaScript Object Notation
JVM	Java Virtual Machine
OS	Operační systém
POI	Point of Interest
REST	Representational State Transfer
RPC	Remote Procedure Call
SQL	Structured Query Language
TCG	Trading card game
UUID	Universally Unique Identifier
XML	eXtensible Markup Language

ÚVOD

Cílem této bakalářské práce je navrhnutí a implementace mobilní aplikace ve formě virtuální sběratelské hry. Aplikace umožňuje správu uživatelského profilu, následně hráči sbírají odměny ve formě bodů, které získají po načtení speciální značky ve formě QR kódu v blízkosti zajímavého turistického místa. Navštívená lokace je přidána do virtuální sbírky a zobrazena na mapě. Hráči jsou porovnáváni v bodové tabulce na základě nasbíraných bodů.

Aplikace je řešena v několika úrovních. Serverová část aplikace je navržena ve formě jednotlivých mikroslužeb. K tvorbě byly využity programovací jazyky Go a Kotlin a databázový systém PostgreSQL. Služby jsou nasazovány jako jednotlivé kontejnery na platformě Docker.

Mobilní aplikace slouží hráči k vizualizaci svých výsledků a ke komunikaci se serverem pomocí REST API. Aplikace je vytvořena pro operační systém Android v programovacím jazyce Kotlin.

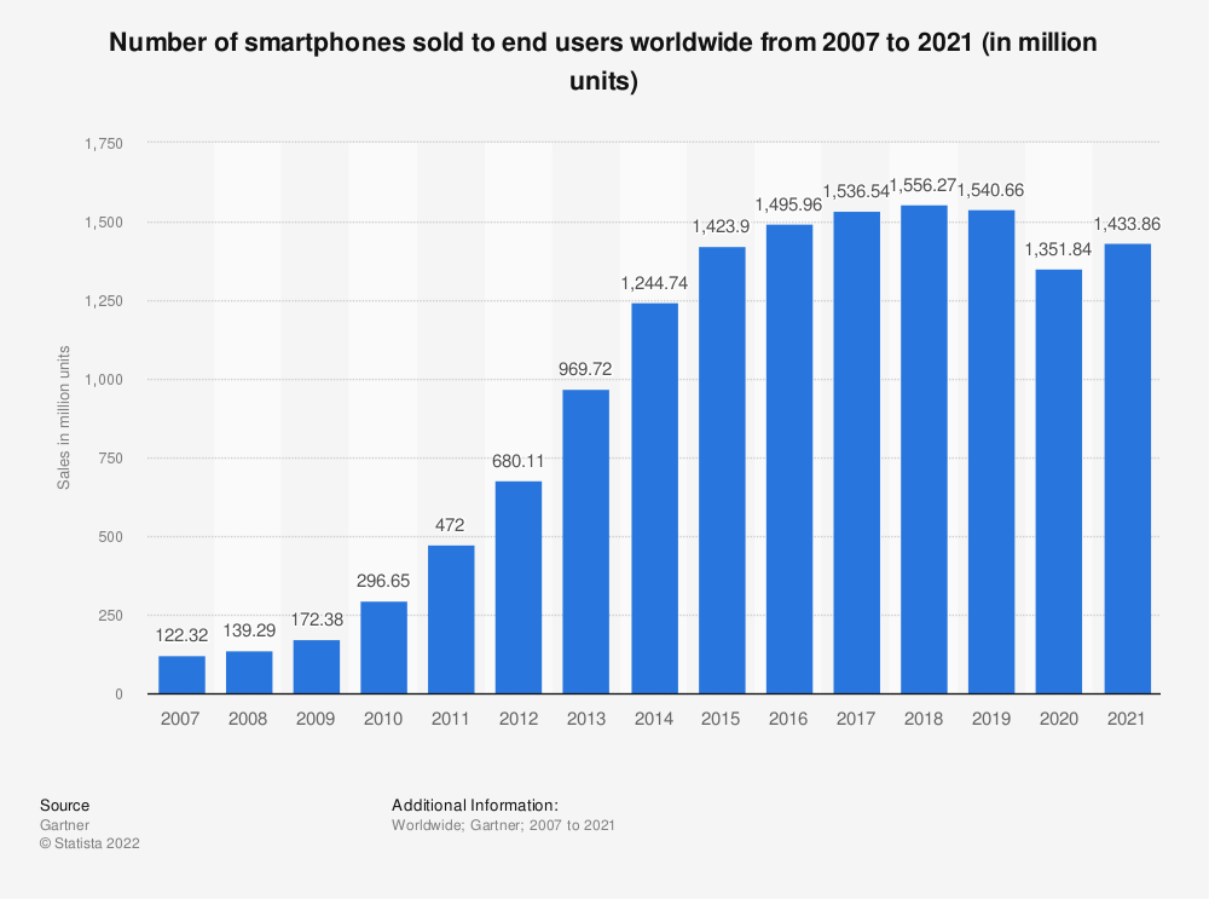
1 TEORETICKÁ ČÁST

1.1 Mobilní aplikace

Mobilní aplikací se obecně označuje taková aplikace, která je specificky navržena pro užívání na mobilních zařízeních jako je chytrý telefon či tablet. Tyto aplikace jsou velmi diverzifikované a poskytují uživateli různé funkcionality a služby. Nejčastěji se jedná o aplikace sociálních sítí, aplikace zaměřené na nakupování a kancelářskou činnost nebo aplikace poskytující různou formu zábavy.

Mobilní aplikace jsou velmi populární díky celosvětovému rozšíření mobilních zařízení a dostupnosti internetového připojení. Obrázek 1 uvádí vývoj prodeje chytrých telefonů od roku 2007 do roku 2021. Z grafu lze vyvodit, že od roku 2010 začal prodej výrazně stoupat. Dále k celkové popularitě mobilních aplikací také napomáhá jednoduchý návrh a vývoj samotných aplikací díky volné dostupnosti potřebných nástrojů [1].

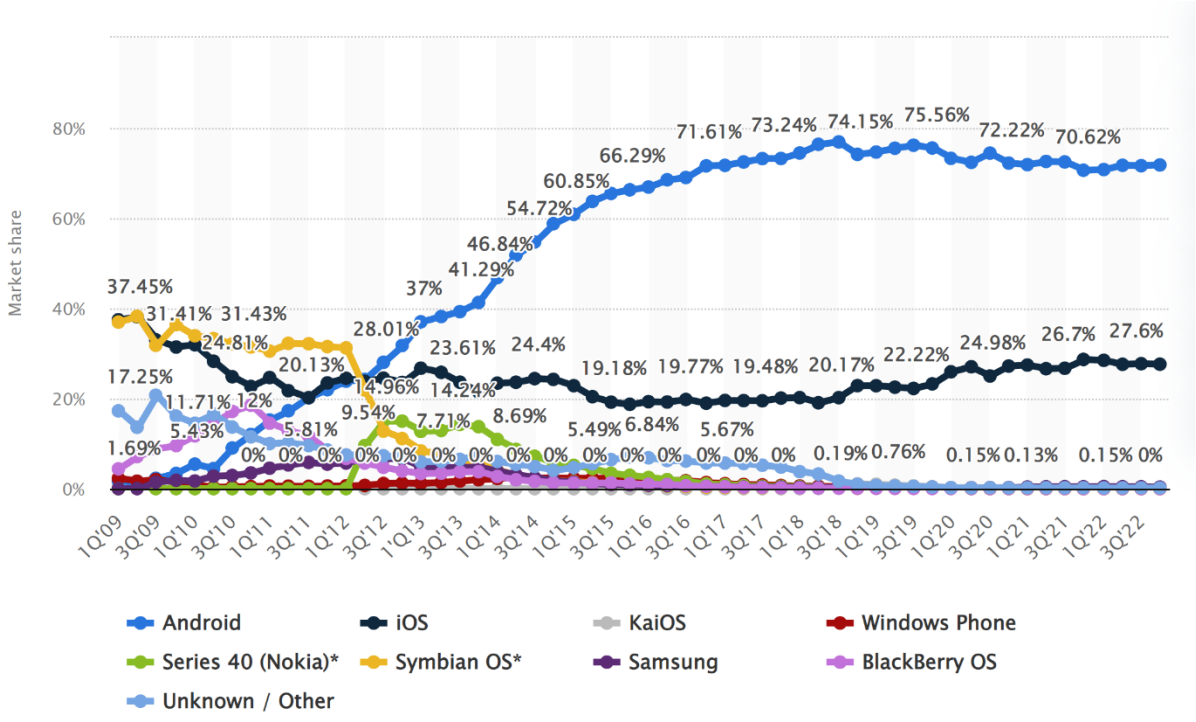
Mobilní aplikace jsou v současné době také integrovány do kritických systémů jako jsou systémy pro správu financí, kontrola infrastruktury nebo řízení logistiky.



Obrázek 1. Počet prodaných chytrých telefonů ve světě od roku 2007 do roku 2021 (zdroj [1])

1.1.1 Operační systém mobilních zařízení

Na základě dat uvedených v grafu (Obrázek 2) jsou aktuálně nejběžnější dva operační systémy pro mobilní zařízení: operační systém Android a operační systém iOS.



Obrázek 2. Podíl mobilních operačních systémů od roku 2009 do roku 2022 (zdroj [2])

Android

Android je open source operační systém pro mobilní zařízení vlastněný společností Google Inc., která aktivně pracuje na jeho vývoji. Jedná se o nejrozšířenější operační systém, který představuje 70 % celého trhu s mobilními zařízeními [2].

První verze operačního systému byla veřejně dostupná v roce 2008 a právě aktuální verze, Android 13, je dostupná od roku 2022. Jádro operačního systému je modifikovanou verzí jádra Linux [3].

Otevřenost operačního systému Android napomáhá velkému rozšíření celé platformy, což vede k mnoha derivacím základního systému. Velké technologické společnosti se zaměřením na mobilní zařízení vyvíjí své vlastní nadstavby Androidu, aby dosáhli lepší integrace systému s dalšími službami dané společnosti. K takovým společnostem lze řadit například Samsung Group či Xiaomi Corporation [4].

Preferované programovací jazyky pro tvorbu aplikací pro tento operační systém je Java od společnosti Oracle Corporation nebo Kotlin od společnosti JetBrains, s. r. o. Programovací

jazyk Java byl podporovaný již od prvních verzí operačního systému Android a Kotlin na tuto tradici navazuje. Díky tomu existuje velké množství knihoven a již vytvořených řešení, které urychlují samotný vývoj aplikace [5].

iOS

Operační systém iOS je vlastněn a vyvíjen společností Apple Inc., je součástí všech mobilních zařízeních vyráběných touto společností. První verze operačního systému byla dostupná v roce 2007 v zařízení iPhone první generace. Aktuální verze, iOS 16, je dostupná od roku 2022. Operační systém iOS je postaven na operačním systému macOS, který je také vyvíjen společností Apple Inc. a je dostupný na zařízení iMac či MacBook [6] [7].

Na rozdíl od operačního systému Android, iOS je proprietární technologie s uzavřeným kódem. Tato skutečnost vede k relativně větší bezpečnosti celého systému, ale zároveň není možné implementovat případné specifické systémy, pokud je operační systém nepodporuje. Výjimkou jsou části softwaru, který je open source [6].

Preferovanými programovacími jazyky pro tuto platformu jsou Objective-C či Swift [6].

1.1.2 Distribuce aplikací

Distribuce aplikací od vývojářů k uživatelům platforem probíhá za pomoci integrovaných distribučních platforem, které jsou specifické pro každý operační systém.

Operační systém Android integruje distribuční platformu Google Play, která je vyvíjena a spravována společností Google Inc. Dalšími zdroji aplikací pro operační systém Android jsou distribuční platformy třetích stran či instalační balíčky aplikací. Tyto způsoby nejsou v základu podporovány a nejsou doporučeným způsobem získávání aplikací pro daný operační systém. Hlavním důvodem je důvěryhodnost a bezpečnost instalovaného softwaru, kdy u platforem třetích stran nemusí docházet ke kontrole důvěryhodnosti softwaru, což přináší bezpečnostní riziko.

Operační systém iOS integruje distribuční platformu App Store, která je vyvíjena a spravována společností Apple Inc. Jedná se o jedinou distribuční platformu pro operační systém iOS.

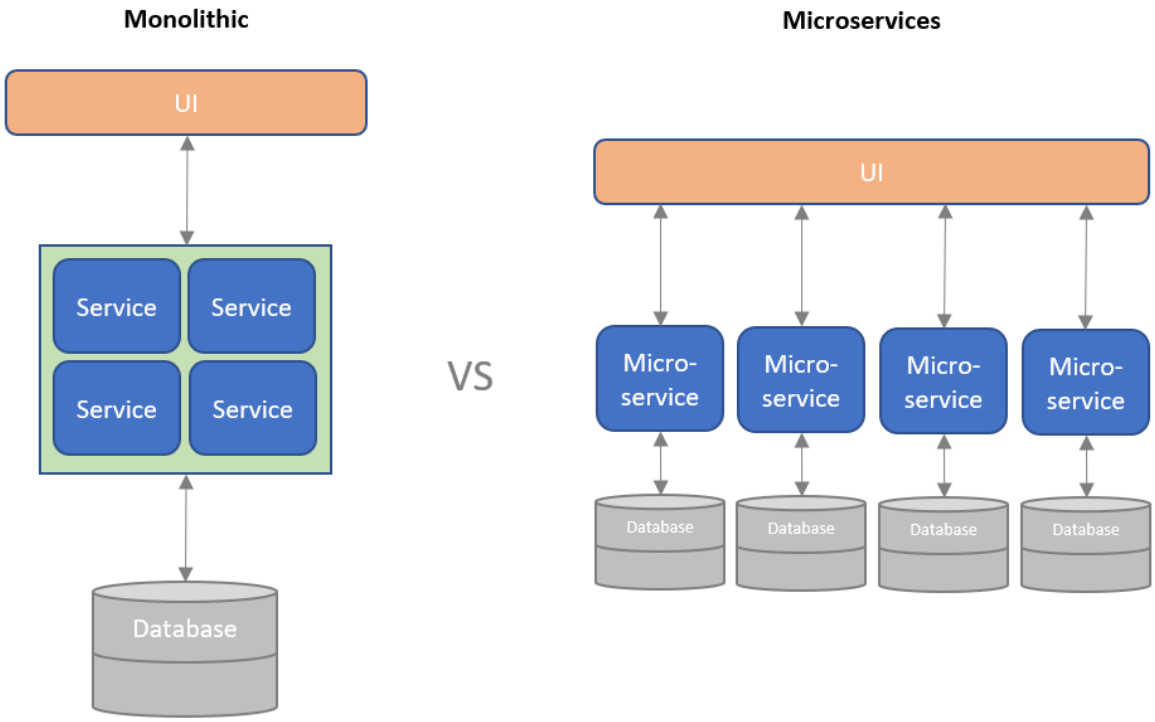
1.2 Mikroslužby

Architektura mikroslužeb je návrhový vzor, který rozděluje aplikaci na menší logické části. Takovéto části aplikace se nazývají služby. Každá služba poskytuje specifickou funkcionalitu, která je vlastní doméně, kterou reprezentuje. Služby mezi sebou komunikují pomocí síťových protokolů jako jsou například HTTP (Hypertext Transfer Protocol) nebo RCP (Remote

Procedure Call). Data jsou většinou přenášena ve formátu JSON (JavaScript Object Notation) a komunikace probíhá za využití REST (Representational State Transfer) API (Application Programming Interface) [8] [9].

Hlavním cílem tohoto návrhového vzoru aplikace je ulehčení jejího vývoje, při kterém lze v rámci jedné aplikace využít různé technologie k optimálnímu řešení dané problematiky. Další výhodou je lepší horizontální škálovatelnost jednotlivých služeb v čase a tím optimální využití přidělených zdrojů [8].

Nevýhodou tohoto návrhového vzoru je případná fragmentace celé aplikace a horší orientace v jednotlivých službách [10].



Obrázek 3. Porovnání monolitické architektury a architektury mikroslužeb (zdroj [11])

1.2.1 Vlastnosti služeb

Každá jednotlivá služba by měla být zodpovědná pouze za jednu problematiku dané aplikace. Tato vlastnost služby přináší velkou flexibilitu do samotného vývoje, jelikož lze pro každou problematiku využít různé technologie, které se pro daný případ nejlépe hodí [11] [12].

Služby by měly být na sobě nezávislé, tedy změny jedné služby by neměly mít vliv na funkce služeb ostatních. Tato vlastnost vede k řešení případných dílčích problémů, které nemusí ovlivňovat celou aplikaci [11] [12].

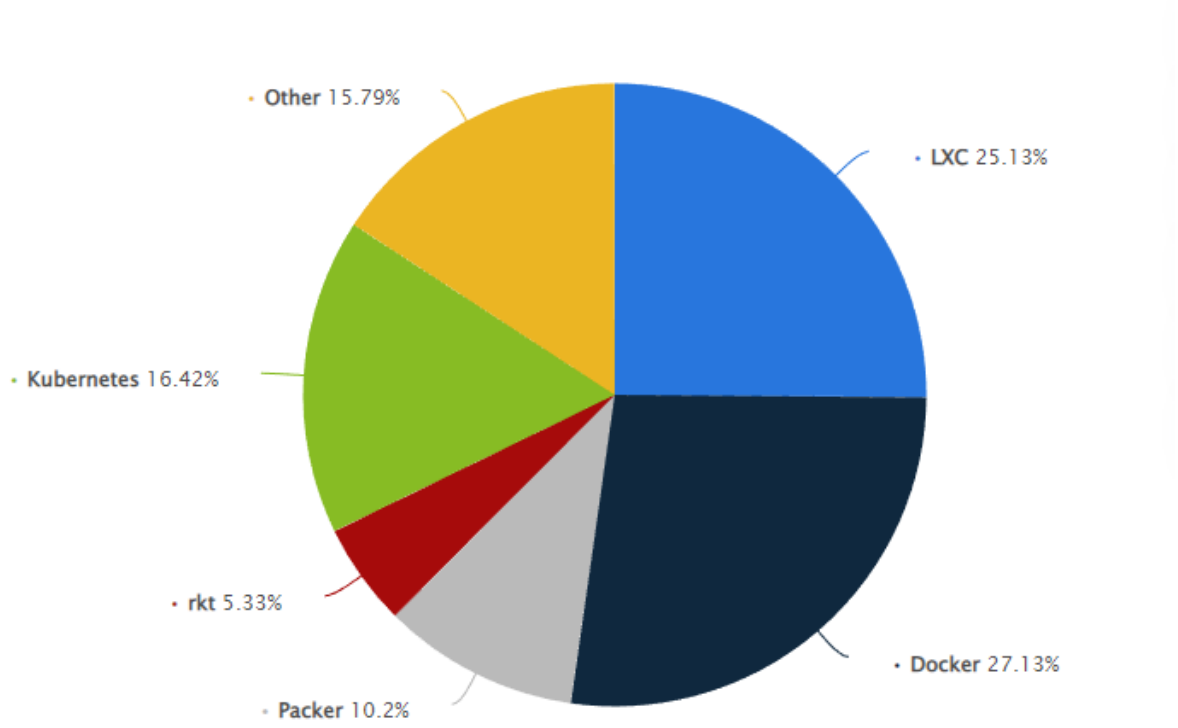
Služby jsou lépe škálovatelné v horizontálním směru než monolitické aplikace. Při škálování celé aplikace může mít každá služba přiděleno odlišné množství zdrojů. Přidělené zdroje se mohou měnit v čase, kdy dochází k výkyvům vytížení aplikace, případně může docházet k replikaci instancí jednotlivých služeb a následnému vyvažování vytížení (Load Balancing). Replikace jednotlivých služeb také napomáhá dostupnosti dané služby, kdy při výpadku jedné instance je služba stále dostupná [11] [12].

1.2.2 Nasazení mikroslužeb

Jednotlivé služby jsou převážně nasazovány do izolovaných virtuálních prostředí, které se nazývají kontejnery. Tato virtuální prostředí jsou virtualizována na úrovni jádra hostujícího operačního systému.

Virtualizace na úrovni operačního systému je pro architekturu mikroslužeb příhodná. Na rozdíl od virtualizace na úrovni hardwaru nedochází k vytváření plnohodnotných virtuálních strojů, což přináší menší dodatečnou zátěž na hardwarové zdroje. Při velkém množství služeb poskytuje tato technologie signifikantní výhodu v podobě úspory zdrojů. Dalším přínosem kontejnerů je jednodušší správa jednotlivých instancí služeb [12].

V dnešní době existuje několik platforem, které umožňují tvorbu a správu virtuálních prostředí v tzv. kontejnerech. Nejvýznamnější podíl na trhu zaujímají platformy Docker vyvíjená společností Docker, Inc. a platforma LXC vyvíjená společnostmi Canonical Ltd. Obě platformy operují s jádrem Linux. Podíly na trhu jednotlivých kontejnerizačních technologií uvádí následující Obrázek 4 [13].



Obrázek 4. Podíl kontejnerizačních technologií na trhu v roce 2022 (zdroj [13])

1.3 Sběratelské hry

Sběratelské hry jsou takové hry, ve kterých hráč shromažďuje sběratelské předměty, které představují určitou hodnotu. Sběratelské předměty mohou mít fyzickou, nebo virtuální podobu. Sběratelské předměty ve fyzické podobě mají nejčastěji formu karty, samolepky, figurky, štítku či známky.

Nejběžnější sběratelské hry jsou karetní sběratelské hry (také známé pod zkratkou TCG či CCG), ve kterých jsou sběratelské předměty reprezentovány kartami. Karty mohou mít fyzickou i virtuální podobu a obvykle jsou doprovázeny dodatečnými informacemi a ilustracemi. Některé sběratelské karty jsou součástí hracích předmětů deskových společenských her. Populárními karetními sběratelskými hrami jsou Magic: The Gathering a Pokémon TCG. Tyto hry mají i své virtuální varianty se shodnými herními mechanikami, které začaly u jejich fyzické obdoby.

1.3.1 Virtuální sběratelské hry

Některé virtuální sběratelské hry mohou mít předobraz ve fyzických sběratelských hrách, jiné jsou čistě virtuální. U některých sběratelských her dochází k propojení fyzicky vlastněných předmětů s virtuálními variantami. Takové propojení například můžeme vidět u figurek amiibo,

které jsou vydávány společností Nintendo. Fyzické figurky reprezentují fiktivní postavy a předměty, za které může uživatel následně hrát ve videohrách podporujících tuto technologii. Hraní za danou postavu je podmíněno vlastnictvím fyzické figurky dané fiktivní postavy.

Hearthstone

Hearthstone je počítačová F2P (Free-to-Play) karetní hra vyvíjená společností Blizzard Entertainment. Smyslem hry je sbírání hracích karet do sbírky každého hráče. Následně si hráč z nasbíraných karet staví vlastní balíčky, se kterými hraje proti ostatním hráčům. Nové karty lze pořídit za herní měnu, nebo za reálné peníze.

Hra je vyvíjena na herním enginu Unity od společnosti Unity Technologies a je dostupná na operačních systémech Windows, macOS, Android a iOS. První verze hry byla uveřejněna v roce 2014 a od té doby je v aktivním vývoji.

Obrázek 5. Hrací plocha z počítačové hry Hearthstone (zdroj [14])

Pokémon Go

Pokémon Go je mobilní sběratelská F2P hra vyvíjena společností Niantic v herním enginu Unity od společnosti Unity Technology, která využívá AR (augmentovanou realitu) pro sběr a interakci virtuálních zvířat. Hráči jsou rozděleni do frakcí, které mezi sebou soupeří. Interakce za využití AR probíhá za využití reálného prostředí, které je následně využito jako kulisa pro virtuální objekty samotné hry. Tato hra kromě AR využívá i geolokační údaje zařízení hráče

pro generování virtuálních zvířat v blízkosti hráče a pro jeho lepší orientaci se využívá mapových podkladů. Cílem hráče je sbírání virtuálních zvířat do kolekce, čehož dosáhne přiblížením se k jeho lokaci. Umístění zvířete a hráče je vizualizováno na mapě.

Hráči na mapě mohou také nalézt virtuální stavby, které jim pomáhají v postupu. Tyto virtuální stavby jsou navázány na pozice významných reálných budov a památek.

První verze hry byla dostupná na mobilních distribučních platformách v roce 2016 pro operační systémy iOS a Android. Hra je od té doby v aktivním vývoji.

Obrázek 6. Uživatelské rozhraní hry Pokémon Go (zdroj [15])

1.3.2 Turistické sběratelské hry

Turistické sběratelské hry jsou často spojeny s turisticky významnými místy, která jsou obvykle reprezentována fyzickým sběratelským předmětem. Často je takové místo reprezentováno samolepkou, štítkem či známkou. Takové předměty lze zakoupit například v informačních centrech.

Turistické známky

Turistické známky jsou turistické sběratelské suvenýry, které mají podobu dřevěného kolečka s graficky stylizovanou reprezentací daného turistického místa. Každé známkové místo reprezentované turistickou známkou má přidělené identifikační číslo. Pomocí těchto čísel si majitel turistické známky může vést evidenci již sebraných známek.

Při sesbírání 10 číselně po sobě jdoucích turistických známek má majitel nárok na prémiovou turistickou známku. Po nasbírání určitého počtu prémiových turistických známek obdrží majitel známek formální titul, například „Bronzový sběratel“ [16].

Obrázek 7. Turistické známky (zdroj [16])

Geocaching

Geocaching patří k celosvětově rozšířené hře honby za pokladem, která vznikla v roce 2000 v USA. Jedná se o geolokační hru, ve které hráči loví poklady v podobě tzv. kešek, které mají různé podoby. Pro zapojení se do hry je nezbytné vytvořit si vlastní hráčský účet. V současné době ke hře existuje mobilní aplikace, dříve hráči využívali pouze GPS (Global Positioning System) souřadnice a GPS navigace.

Obrázek 8. Geocaching – logo (zdroj [17])

Hra probíhá především v přírodě, ale také v zabydlených oblastech. Jednotliví hráči se snaží sesbírat, „ulovit“ co největší možný počet kešek, které jsou umístěny na rozličných místech. Kešky se liší jak svou podobou, tak i obtížností pro jejich objevení a dosažení. Kešku může představovat jakákoliv uzavíratelná nádoba od krabičky na film po velký plastový box s víkem. Obtížnost ulovení kešky může být udána terénem, ve kterém se poklad nachází, či sesbíráním potřebných indicií k dosažení finálního pokladu. Kešky mohou být umístěny na magnetu v zábradlí, na informační tabuli, nebo i v dutině stromu či v extrémních případech na dně

rybníka. Hráči mohou lovit kešky přímo – navigace vede k jedinému pokladu; nebo postupně – hráč má informace pouze k prvnímu bodu, kde dostane indicie pro další postup. Pro dosažení konečného cíle může tak projít několik mezikroků.

Keška vždy obsahuje logbook, do kterého hráč zapisuje datum, čas objevení pokladu a svou přezdívku. Některé kešky obsahují také určité poklady – drobné předměty, které hráči mohou vyměnit za jiné, případně mohou předmět pouze vložit, či vyjmout. Každý proces výměny je též zapsán do logbooku, což umožňuje přehled majiteli i ostatním hráčům. Nalezené kešky jsou následně vráceny na své původní místo a ulovení kešky je zapsáno do aplikace či na oficiální web.

Pravidla hry se mohou odlišovat v závislosti na zemi, ve které se hráč nachází. Je možné lovit kešky i v jiných státech, ovšem je doporučeno předem nastudovat pravidla pro místní úpravu [17].

1.4 QR kód

Quick Response kód, zkráceně QR kód, je dvourozměrný čárový kód, respektive čitelný grafický vzor, který může obsahovat nejrůznější formy dat. V kódu mohou být uložena data v numerické, alfanumerické či binární podobě. Pomocí QR kódu lze sdílet unikátní identifikátory, osobní údaje, telefonní čísla, adresy webových stránek, přihlašovací údaje k bezdrátovým sítím či obyčejný text. Maximální kapacita kódu je závislá na použité variantě kódu a na požadované replikaci dat v samotném kódu. Replikace dat se využívá k získání dat z kódu i při částečném fyzickém poškození části kódu [18].

QR kód byl vynalezen japonskou firmou Denso Wave jako technologie k označení součástek dodávaných do automotivů [18].

Obrázek 9. QR kód odkazující na webové stránky www.google.com (zdroj vlastní)

2 EXPERIMENTÁLNÍ ČÁST

2.1 Použité technologie

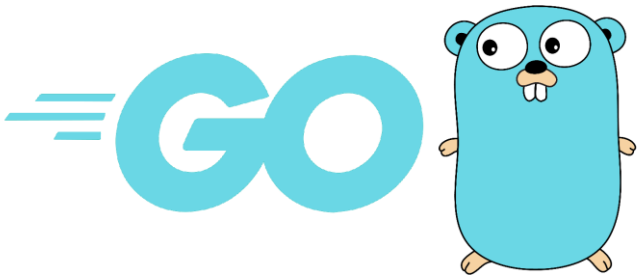
2.1.1 Programovací jazyky

Programovací jazyk Go

Programovací jazyk Go je open source projekt společnosti Google, Inc. Jedná se o staticky typový programovací jazyk, který se ve velké míře inspiruje ostatními programovacími jazyky jako jsou C či Python. Jazyk Go je kompilovaný jazyk s integrovaným garbage collectorem, který se stará o správu alokované paměti [19].

Vývoj programovacího jazyka začal v roce 2007 jako interní projekt společnosti Google, Inc. V roce 2012 byla uvedena první veřejně dostupná verze. S verzí 1.18 přichází podpora generických typů, která byla dlouho komunitou žádána. Programovací jazyk je stále v aktivním vývoji a aktuální verze nese označení 1.20 [19].

Hlavní motivace pro tvorbu tohoto programovacího jazyka, bylo usnadnění a urychlení serverových aplikací a náhrada stávajících standardních programovacích jazyků C++ a Java. Nejčastější uplatnění jazyka Go nalezneme u serverových aplikací, především u tvorby aplikací jako mikroslužeb [19].



Obrázek 10. Oficiální logo a maskot programovacího jazyka Go (zdroj [19])

Programovací jazyk Kotlin

Programovací jazyk Kotlin je open source projekt společnosti JetBrains, s. r. o. Jedná se hybridní staticky typový programovací jazyk, který je kompatibilní s programovacím jazykem Java, jeho standardními knihovnami a runtime prostředím JVM (Java Virtual Machine) [20].

Nejčastější využití jazyka Kotlin nalezneme u velkých serverových aplikací, u mikroslužeb či mobilních aplikací [20].

Programovací jazyk Kotlin je plně podporovaný operačním systémem Android a je doporučeným jazykem pro tvorbu nových mobilních aplikací. Díky kompatibilitě s programovacím jazykem Java je většina dostupných knihoven kompatibilní s prostředím Kotlin a lze je v nových projektech využít [20].

Vývoj jazyka Kotlin započal v roce 2010 a první stabilní verze byla uveřejněna v roce 2016. Aktuální stabilní verze nese označení 1.8.21, která byla uveřejněna v roce 2023. Vývoj jazyka je stále aktivní [20].

Obrázek 11. Oficiální logo programovacího jazyka Kotlin (zdroj [21])

2.1.2 Knihovny

Gin

Gin je open source webový framework, který je naprogramovaný v jazyce Go. Tento framework slibuje rychlé zpracování dotazů, podporu validace dat ve formátu JSON, ochranu proti pádu a správu chyb vzniklých za běhu aplikace. Také lze tento framework rozšířit o další funkcionality díky externím knihovnám [22].

Ktor

Ktor je open source webový framework naprogramovaný v jazyce Kotlin. Vývoj financuje a vede společnost JetBrains, s. r. o. Ktor je určen pro tvorbu webových aplikací a mikroslužeb. Základní funkcionalitu frameworku lze rozšiřovat dodatečnými moduly, které jsou vytvářeny přímo společností JetBrains, s. r. o. či komunitou vývojářů. Mezi takové moduly je například zařazena serializace dat do textových formátů jako JSON či XML (eXtensible Markup Language) nebo podpora autentizačních protokolů jako OAuth [23].

ZXing

ZXing je open source knihovna pro zpracování 1D a 2D čárových kódů různých typů (EAN, QR, …). Tato knihovna je vhodná i pro využití v mobilních aplikacích pro operační systém Android. Knihovna již není v aktivním vývoji, pouze v režimu udržování [24].

2.1.3 Databázové servery

PostgreSQL

PostgreSQL, nebo také Postgres, je open source objektově-relační databázový systém, který byl původně vyvíjen Kalifornskou univerzitou v 80. letech 19. století. Jedná se o efektivní databázový systém, který poskytuje podobné funkcionality jako konkurenční komerční systémy [25].

V dnešní době je stále v aktivním vývoji vedený skupinou PostgreSQL Global Development Group, která je složena z vývojářů dobrovolníků z celého světa. Aktuální verze nese označení 15.2 [26].

Obrázek 12. Oficiální logo databázového systému PostgreSQL (zdroj [26])

2.1.4 Integrovaná vývojová prostředí

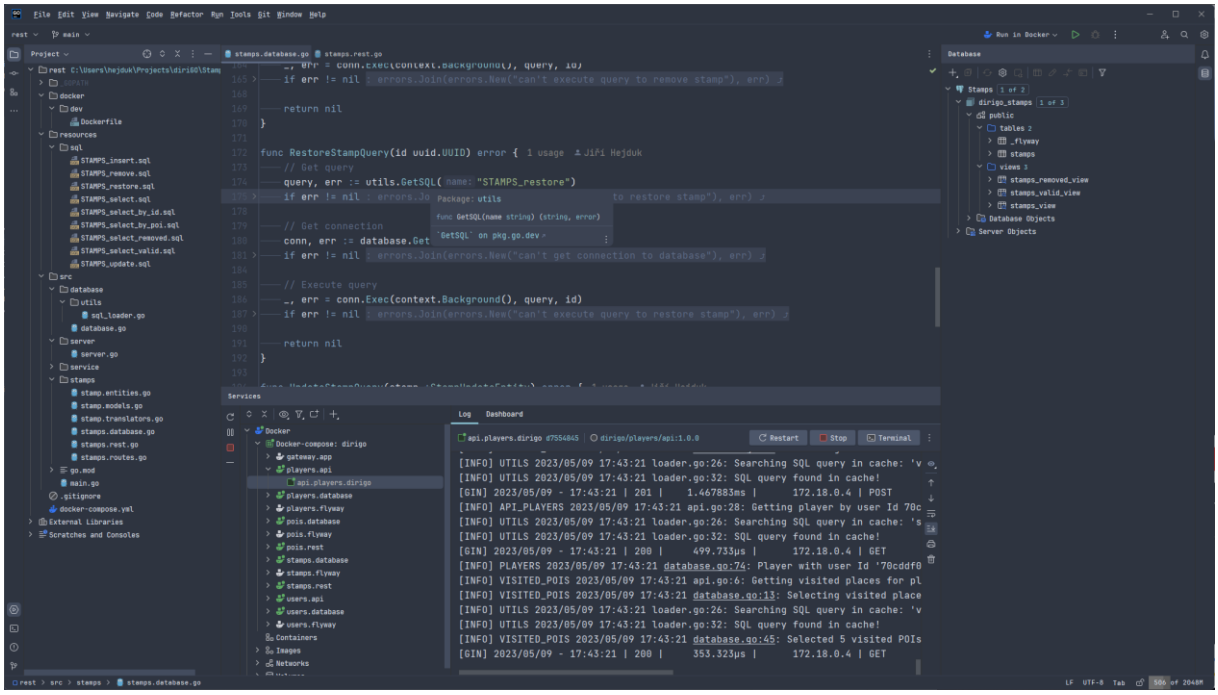
Integrované vývojové prostředí, zkráceně IDE (Integrated Development Environment), je nástroj, který slouží k psaní zdrojového kódu v daném programovacím jazyce a poskytuje další dodatečné nástroje, které pomáhají vývoji případné aplikace a zrychlují ho. Mezi takové dodatečné nástroje patří syntaktická kontrola zdrojového kódu, našeptávač, debugger či profiler pro monitoring využitých zdrojů spuštěné aplikace. Většina prostředí podporuje hned několik programovacích jazyků. Většinou se jedná o populární programovací a skriptovací jazyky jako například C#, Java, Kotlin, Python nebo JavaScript [27].

IDE mohou mít podobu desktopových nebo webových aplikací. Populární desktopové IDE jsou Visual Studio od společnosti Microsoft, Eclipse IDE vyvíjené komunitou Eclipse Foundation a IntelliJ IDEA od společnosti JetBrains, s. r. o [27].

JetBrains GoLand

GoLand je integrované vývojové prostředí od společnosti JetBrains, s. r. o. Je zaměřeno na programovací jazyk Go od společnosti Google, Inc. Kromě podpory jazyka Go nástroj integruje

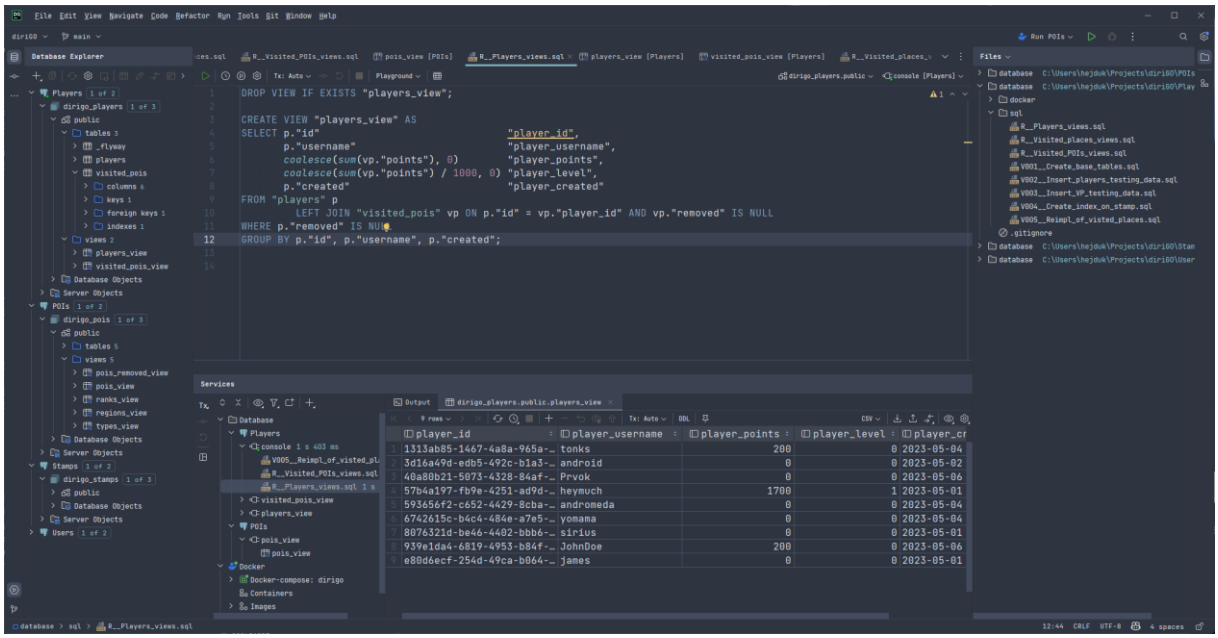
například rozhraní pro kontejnerizační nástroj Docker či klienta pro připojení k databázovým serverům různého typu [28].



Obrázek 13. Integrované vývojové prostředí JetBrains GoLand (zdroj vlastní)

JetBrains DataGrip

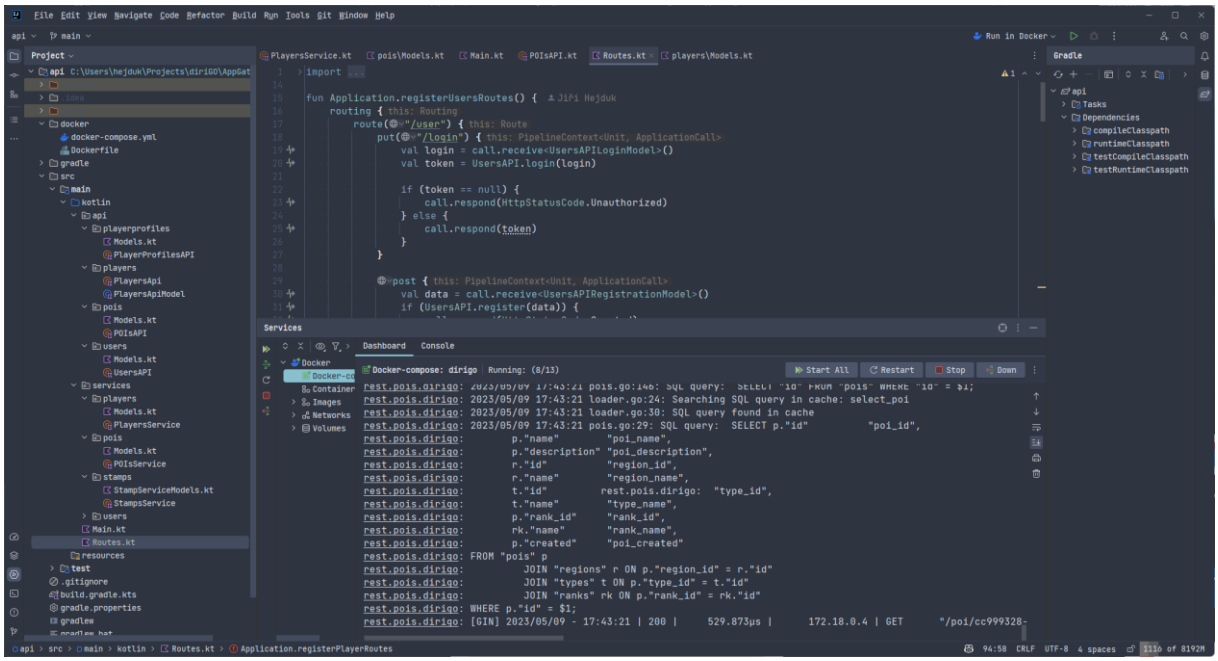
DataGrip je integrované vývojové prostředí od společnosti JetBrains, s. r. o. Je zaměřeno na dotazovací jazyk SQL (Structured Query Language) a na správu databázových serverů. Podporuje velké spektrum databází jako například Microsoft SQL, MySQL, PostgreSQL a další. Kromě relačních databází podporuje například i dokumentovou databázi MongoDB nebo key-value databázi Redis [29].



Obrázek 14. Integrované vývojové prostředí JetBrains DataGrip (zdroj vlastní)

JetBrains IntelliJ IDEA

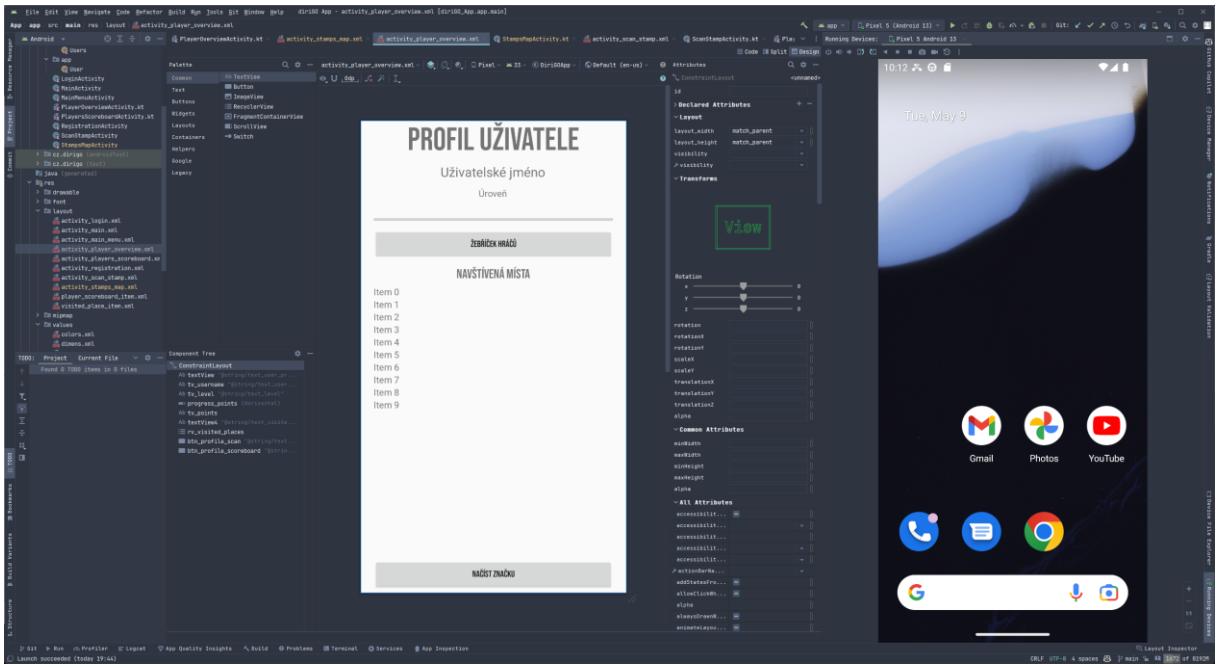
IntelliJ IDEA je integrované vývojové prostředí od společnosti JetBrains, s. r. o. Je zaměřeno především na programovací jazyk Java a jazyky využívající JVM. Kromě těchto podporuje další programovací jazyky prostřednictvím dodatečných pluginů [30].



Obrázek 15. Integrované vývojové prostředí JetBrains IntelliJ IDEA (zdroj vlastní)

Android Studio

Android Studio je IDE od společností Google, Inc. a JetBrains, s. r. o. Jedná se o oficiální vývojové prostředí pro aplikace určených pro operační systém Android, konkrétně o modifikovanou verzi vývojového prostředí JetBrains, s. r. o. Kromě podpory programovacích jazyků Java a Kotlin poskytuje integrovaný emulátor pro operační systém Android a vizuální editor uživatelských prostředí Android aplikací [5].



Obrázek 16. Integrované vývojové prostředí Android Studio (zdroj vlastní)

2.1.5 Ostatní nástroje

Gradle

Gradle je open source nástroj pro automatizaci sestavování aplikací ze zdrojového kódu, dále také obstarává správu případných závislostí, které jsou nezbytné k sestavení softwaru. Podporuje mnoho programovacích jazyků (C, C++, Java, JavaScript), ale nejčastěji je spojovaný s programovacími jazyky založenými na JVM (například Java, Kotlin, Clojure). Integrace tohoto nástroje je zahrnuta do mnoha integrovaných vývojových prostředí, které podporují jazyk Java či Kotlin [31].

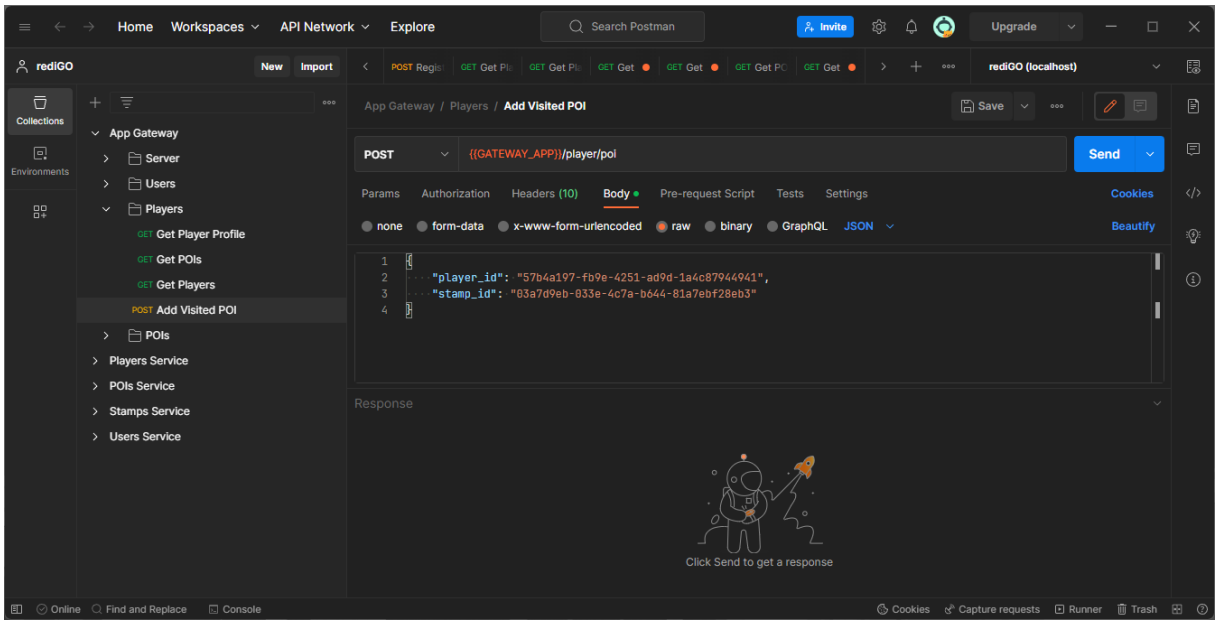
Flyway

Flyway je open source nástroj pro migraci databázových systémů vyvíjený společností Red Gate Software Ltd. Hlavní předností tohoto nástroje je možnost verzování modelu a dat v daném databázovém systému. Podporuje velké množství databázových systémů jako například MySQL, PostrgreSQL, MSSQL či Oracle. Nástroj je vytvořen v programovacím

jazyce Java. Je distribuován jako CLI (Command-line Interface) nástroj nebo jako knihovna pro programovací jazyk Java či Kotlin [32].

Postman

Postman je open source nástroj pro návrh, tvorbu a testování API rozhraní vyvíjený společností Postman, Inc. Podporuje populární protokoly síťové komunikace jako REST, gRPC nebo WebSocket. Platforma Postman podporuje monitorování běhu a testování jednotlivých služeb, které probíhá za využití skriptovacího jazyka JavaScript [33].



Obrázek 17. GUI (Graphical User Interface) nástroje Postman (zdroj vlastní)

Docker

Kontejnerizační platforma Docker je open source projekt společnosti Docker, Inc. Tato společnost financuje a vede vývoj samotné platformy a poskytuje další dodatkové služby pro firmy i jednotlivce. Docker je populární platformou pro kontejnerizaci a využívají ji další technologie jako Kubernetes či Docker Compose.

Docker, přesněji Docker Engine, je vytvořen v programovacím jazyce Go. Zajišťuje kompletní management kontejnerů a správu přiřazených zdrojů, které kontejnery využívají. Pro své fungování vyžaduje funkcionalitu jádra systému Linux, který poskytuje nástroje pro vytvoření izolovaných prostředí.

První verze platformy byla dostupná v roce 2013 pro operační systém GNU/Linux. Aktuální verze 23.0.5 je dostupná i pro operační systémy Windows a macOS za využití hardwarové virtualizace linuxového operačního systému.

Docker Compose

Docker Compose je doplňující nástroj pro kontejnerizační nástroj Docker. Nástroj Docker Compose umožňuje definování a spuštění vícekontejnerových aplikací na platformě Docker. Konfigurace jednotlivých kontejnerů je definována v souboru docker-compose.yml, která je psána ve standardním formátu YAML. První verze nástroje byla vytvořena ve skriptovacím jazyce Python. Aktuální verze 2 je vytvořena v programovacím jazyce Go stejně jako platforma Docker. Současný vývoj vede společnost Docker, Inc [34].

2.2 Návrh aplikace

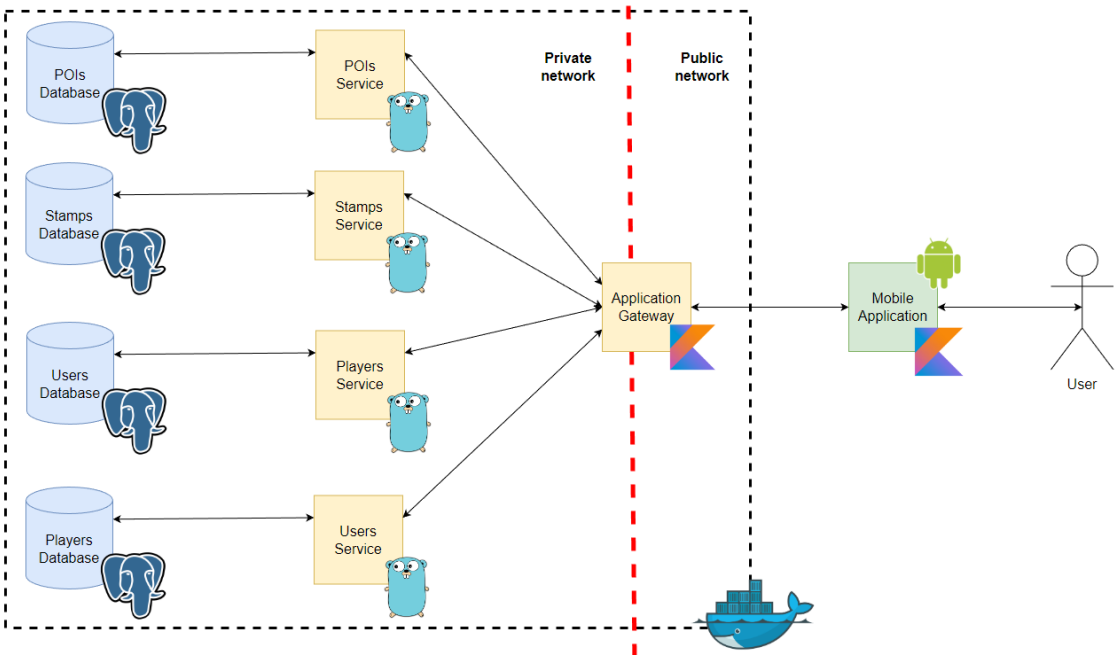
Hlavním cílem projektu je vytvořit geolokační turistickou sběratelskou hru. Nejdůležitějšími vlastnostmi této hry je sběr zajímavých turistických míst na území České republiky. Jednotlivá místa jsou reprezentována unikátní známkou, která je přístupná na veřejně dostupném místě v podobě QR kódu. Ten lze na této pozici načíst.

Hra by měla být dostupná největšímu možnému počtu hráčů, proto by měla být implementována jako mobilní aplikace.

Aplikace by měla umožňovat registraci nových hráčů a jejich následné přihlášení, načítání známek, zobrazení přehledu profilu hráče a žebříčku hráčů. Zmíněné známky reprezentují turistická místa, která jsou pak k nalezení v přehledu profilu hráče jako jeho sbírka. Hráči ve svém profilu dále uvidí svůj celkový postup ve hře a mohou nahlédnout do žebříčku, který je řadí na základě bodů získaných za jednotlivé nasbírané známky. Body se hráčům přičítají na účet po načtení známky jako QR kódu.

2.3 Implementace aplikace

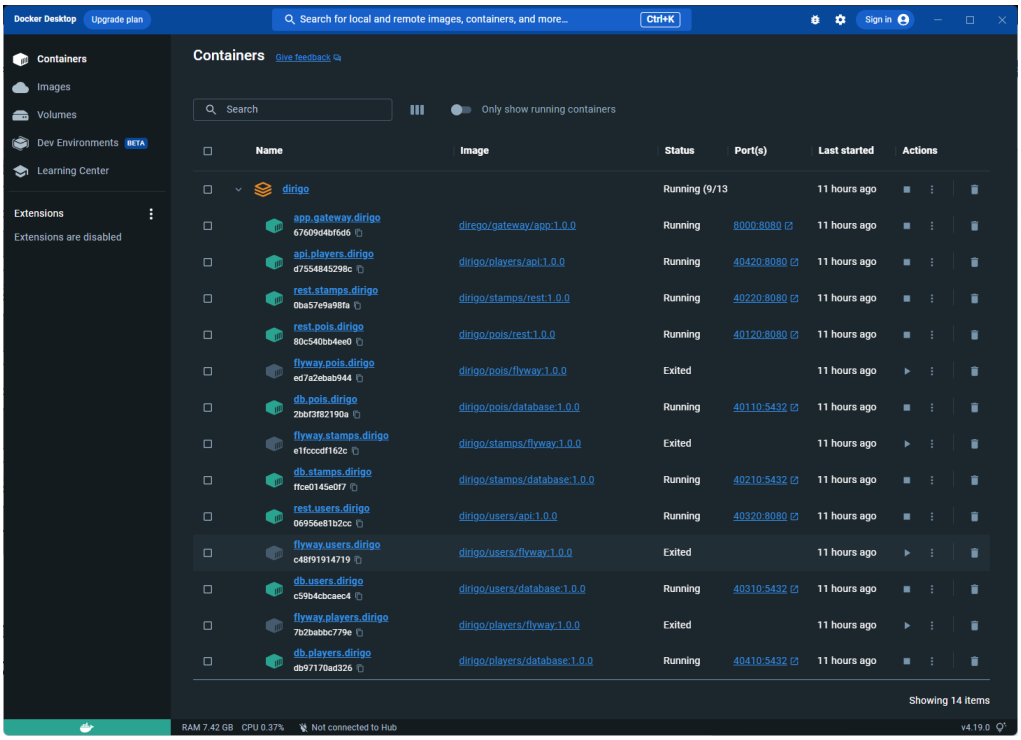
Tento projekt je implementován jako mobilní aplikace pro operační systém Android, která komunikuje se serverem aplikace pomocí protokolu HTTP. Data mezi mobilní aplikací a serverem jsou přenášena ve formátu JSON. Serverová část aplikace je navržena podle návrhové vzoru mikroslužeb, ve kterém je server rozdělen na jednotlivé služby. Serverové služby jsou spouštěny jako kontejnery v kontejnerizační platformě Docker.



Obrázek 18. Schéma sestavení aplikace (zdroj vlastní)

2.3.1 Serverová část aplikace

Serverová část aplikace je navržena dle návrhového vzoru mikroslužeb, jednotlivé služby tedy obsluhují dílčí funkcionality celé aplikace. Služby jsou vytvořené za využití programovacího jazyka Go a webového frameworku Gin. Pro vývoj těchto služeb bylo použito IDE GoLand. Výjimku představuje služba zajišťující komunikaci hlavních služeb aplikace s mobilní aplikací. Tato služba je vytvořena za využití programovacího jazyka Kotlin a webového frameworku Ktor, pro její vývoj bylo využito IDE IntelliJ IDEA. K ukládání dat jednotlivých služeb je využit databázový systém PostgreSQL, pro jehož správu bylo využito IDE DataGrip. Verzování a migraci databázových systémů zajišťuje nástroj Flyway, který se stará o integritu modelů a dat v jednotlivých databázích. Služby, databázové systémy a nástroje jsou spouštěny jako jednotlivé kontejnery na kontejnerizační platformě Docker. Pro zjednodušení spouštění celé aplikace jsou vytvořeny konfigurační soubory pro spuštění pomocí Docker Compose.



Obrázek 19. Přehled kontejnerů aplikace v Docker Desktop (zdroj vlastní)

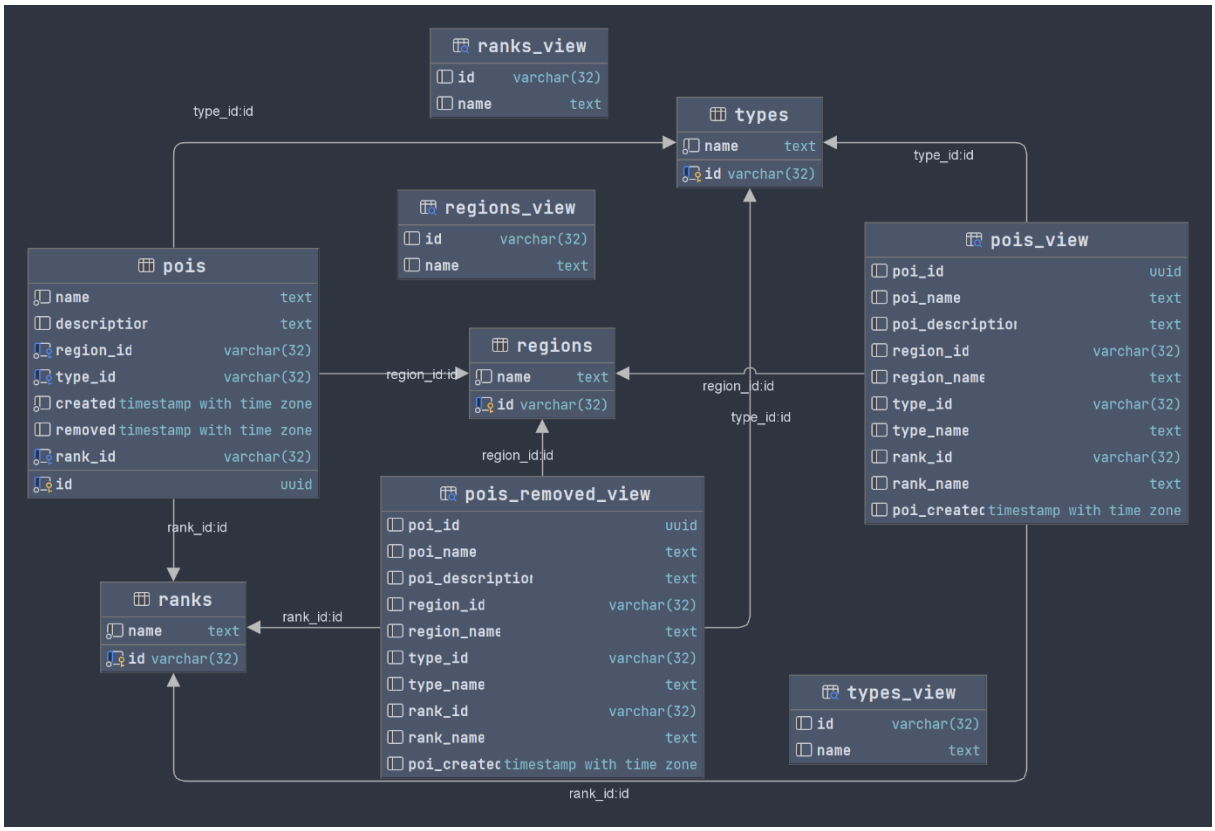
Služba POIs

Služba POIs (Point of Interests) poskytuje informace a správu jednotlivých turistických míst. Pomocí této služby lze zakládat a upravovat jednotlivá místa, ta lze zařazovat do regionů, ve kterých se nachází (například Středočeský kraj). Dále je možné jednotlivým lokacím přiřazovat hodnocení, na jehož základě hráč získává určitý počet bodů. Také je možné turistickému místu přiřadit typ, který představuje jeho druh (například rozhledna).

Tabulka 1. Výběr koncových bodů služby POIs

URI	HTTP metoda	Popis
/pois	GET	Tento bod slouží k získání přehledu vytvořených míst, které lze navštívit.
/poi	POST	Tento bod slouží k vytvoření nového místa.
/poi/:id	PUT	Tento bod slouží k úpravě záznamu existujícího místa.
/poi/:id/remove	PUT	Tento bod slouží k odebrání existujícího místa.
/poi/:id/restore	PUT	Tento bod slouží k obnovení odebraného místa.
/regions	GET	Tento bod slouží k získání přehledu všech použitelných regionů, do kterých lze místa zařadit.

URI	HTTP metoda	Popis
/types	GET	Tento bod slouží k získání přehledu o jednotlivých typech míst.
/ranks	GET	Tento bod slouží k získání přehledu o jednotlivých ohodnoceních, které lze místům přiřadit.



Obrázek 20. Fyzický model databáze, kterou využívá služba POIs (zdroj vlastní)

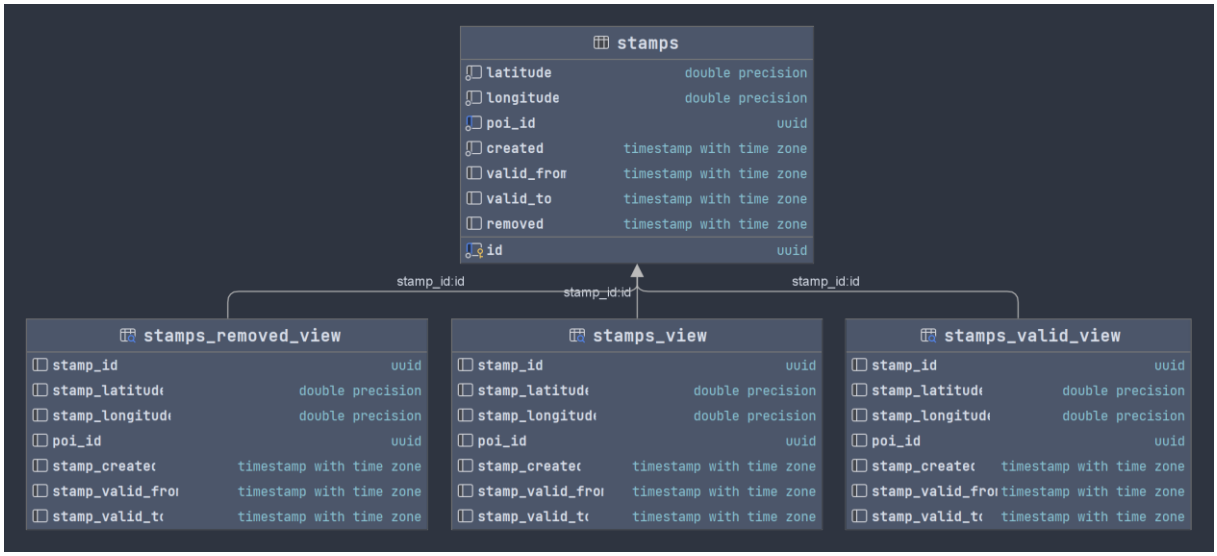
Služba Stamps

Služba Stamps přiřazuje k jednotlivým turistickým místům konkrétní známku. Jedno turistické místo může mít přiřazeno více známek, které mohou mít různou platnost. Známka zahrnuje referenci na turistickou lokaci, interval své vlastní platnosti v systému a GPS.

Unikátní identifikátor dané známky je následně uložen ve fyzické podobě známky ve formě QR kódu.

Tabulka 2. Výběr koncových bodů služby Stamps

URI	HTTP metoda	Popis
/stamps	GET	Tento bod slouží k získání přehledu všech známek, které jsou registrované v systému.
/stamp/poi/:id	GET	Tento bod slouží k získání známek k danému turistickému místu.
/stamp/:id	GET	Tento bod slouží k získání informací o dané jedné známce.
/stamp	POST	Tento bod slouží k vytvoření nové známky pro dané turistické místo. Pro vytvoření známky je nutné dodat GPS souřadnice dané známky, identifikátor turistického místa a informace o validitě známky.
/stamp/:id	DELETE	Tento bod slouží k odstranění existující známky v systému.



Obrázek 21. Fyzický model databáze, kterou využívá služba Stamps (zdroj vlastní)

Služba Users

Služba Users je využita k uchování přihlašovacích údajů registrovaných uživatelů. Přihlašovací údaje uživatelů jsou reprezentovány e-mailovou adresou a heslem. Oba tyto údaje jsou uloženy v podobě hashů.

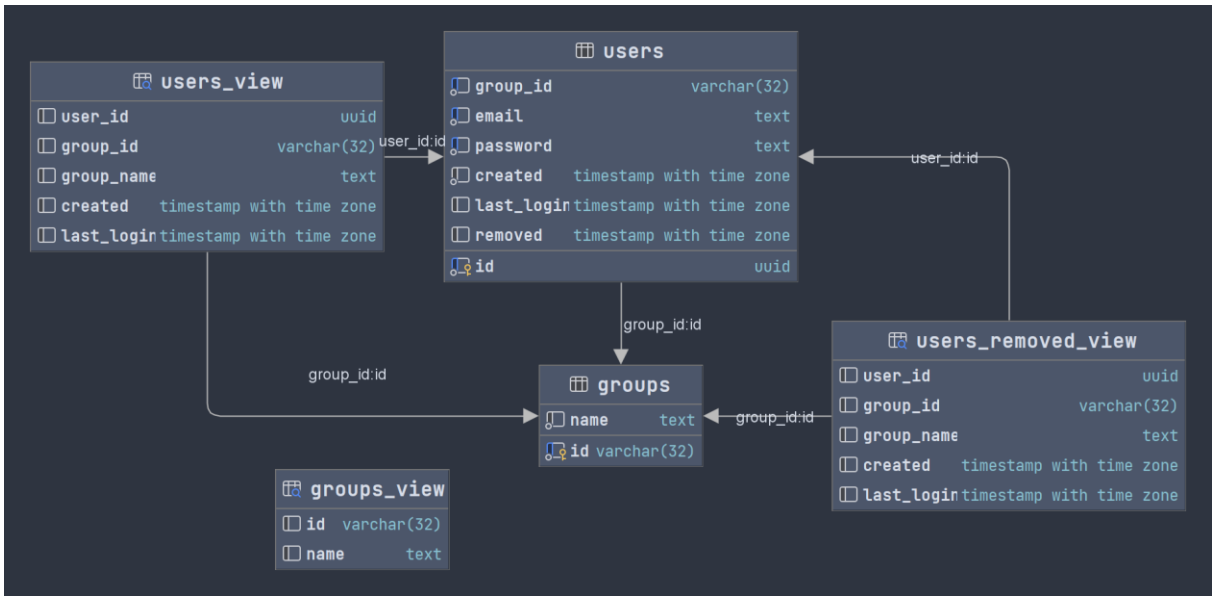
```
INSERT INTO "users"("email", "password", "group_id")
VALUES (encode(digest('example@example.com', 'sha256'), 'hex'),
        crypt('password', gen_salt('bf')), 'USER');
```

Obrázek 22. Příklad SQL dotazu k vytvoření nového uživatele (zdroj vlastní)

Každý uživatel je zařazen do uživatelské skupiny, která je v aktuální implementaci nevyužita, ale v dalším vývoji jí lze využít k přiřazení oprávnění.

Tabulka 3. Výběr koncových bodů služby Users

URI	**HTTP metoda**	**Popis**
/users	GET	Tento bod slouží k získání přehledu registrovaných uživatelů v systému.
/user	POST	Tento bod slouží k vytvoření nového uživatele v systému. Pro vytvoření je nutné poskytnout e-mailovou adresu, heslo a skupinu, do které bude uživatel zařazen.
/user/login	PUT	Tento bod slouží ke kontrole vstupních přihlašovacích údajů a údajů uložených v databázi.
/user/change-login	PUT	Tento bod slouží ke změně přihlašovacích údajů existujícího uživatele.
/user/:id	DELETE	Tento bod slouží k deaktivaci uživatele v systému.



Obrázek 23. Fyzický model databáze, kterou využívá služba Users (zdroj vlastní)

Služba Players

Služba Players slouží ke správě jednotlivých hráčů, kteří jsou registrováni v systému. Služba poskytuje základní informace o hráči – jeho uživatelské jméno či počet navštívených míst včetně bodového ohodnocení.

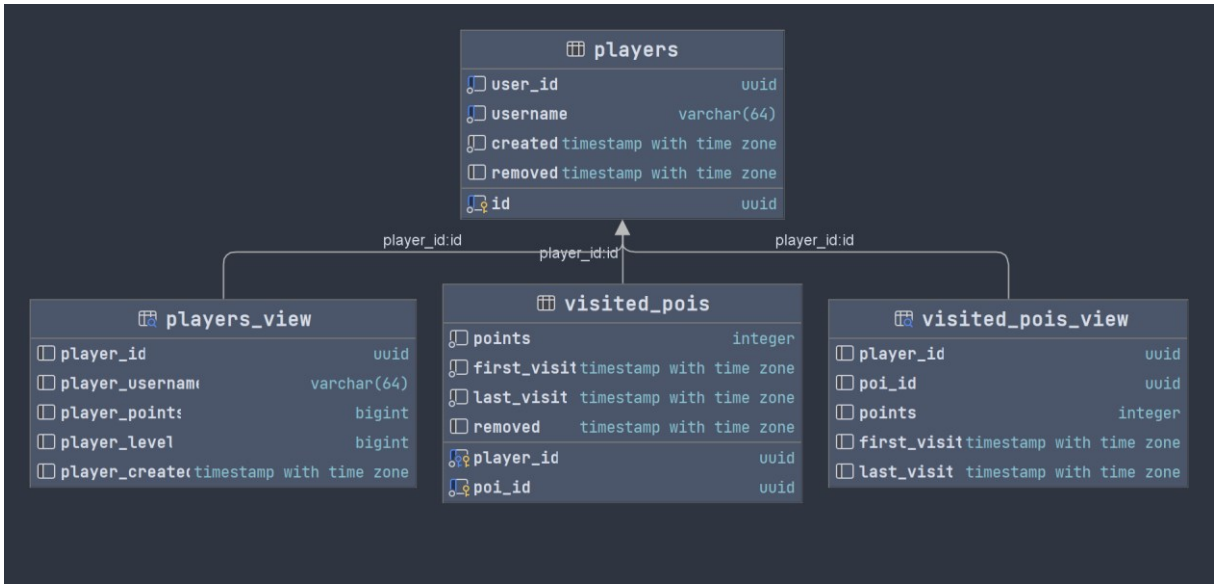
Jednotliví hráči jsou navázáni na službu uživatelů. Navštívená místa jsou navázána na službu turistických míst. Každé navštívené místo uchovává počet obdržených bodů za návštěvu a datum první a poslední návštěvy daného místa.

Služba také poskytuje přehled všech hráčů a jejich žebříček založený na porovnání počtu nasbíraných bodů.

Tabulka 4. Výběr koncových bodů služby Players

URI	**HTTP metoda**	**Popis**
/players	GET	Tento bod poskytuje přehled o všech registrovaných hráčích
/player/user/:id	GET	Tento bod zajišťuje informace o hráči na základě uživatelského identifikátoru.
/player	POST	Tento bod poskytuje registraci nového hráče. Vyžaduje poskytnutí identifikátoru uživatele a uživatelské jméno nového hráče.

URI	HTTP metoda	Popis
/player/:id/visited_pois	GET	Tento bod poskytuje přehled navštívených turistických míst daným hráčem včetně přehledu nasbíraných bodů.



Obrázek 24. Fyzický model databáze, kterou využívá služba Players (zdroj vlastní)

Služba Application Gateway

Služba Application Gateway slouží jako brána mezi mikroslužbami aplikace na straně serveru a mobilní aplikací. Pomocí této služby jsou mikroslužby aplikace izolovány od veřejné sítě a přístupové body mikroslužeb jsou dostupné právě skrze tuto službu. Služba poskytuje přístupové body, které jsou určeny pouze pro mobilní aplikaci.

Tato služba se také stará o autorizaci jednotlivých dokumentů. Oprávnění je řešeno na úrovni brány, ostatní izolované služby tuto problematiku pomíjí. Hlavní důvodem je zjednodušení případných úprav oprávnění na jednotlivé koncové body.

Tabulka 5. Vybrané koncové body služby Application Gateway

URI	HTTP metoda	Popis
/user/login	PUT	Tento bod zajišťuje kontrolu validity zadaných přihlašovacích údajů. Požaduje e-mailovou adresu a heslo. V případě shody vrací identifikační token.
/user/change-email	PUT	Tento bod umožňuje změnu e-mailové adresy, která je nutná pro přihlášení uživatele.
/user/change-password	PUT	Tento bod umožňuje změnu uživatelského hesla.
/user	POST	Tento bod zajišťuje registraci nového uživatele. Vyžaduje e-mailovou adresu, heslo a uživatelské jméno. E-mailová adresa a uživatelské jméno musí být unikátní v celém systému.
/player/profile	GET	Tento bod vrací profil hráče. Vyžaduje hodnotu pro pole Authorization v hlavičce dotazu. Na hodnoty tohoto pole je vrácena struktura dat představující profil hráče.
/player/:id/pois	GET	Tento bod vrací seznam navštívených míst daným hráčem. Vyžaduje hodnotu pro pole Authorization v hlavičce dotazu.
/players	GET	Tento bod vrací přehled všech registrovaných hráčů.
/player/poi	POST	Tento bod umožňuje přidání navštíveného místa do sbírky hráče. Vyžaduje identifikátor načtené známky a identifikátor uživatele. Také vyžaduje hodnotu pro pole Authorization.
/pois	GET	Tento bod vrací přehled všech vytvořených turistických míst. Jedná se o kombinovaný výstup služeb Stamps a POIs.

2.3.2 Fyzická podoba známky

Známka je reprezentována standardizovaným QR kódem, který nese identifikátor dané známky. Tento QR kód je naskenován mobilní aplikací a získaný identifikátor se využívá k vložení

navštíveného místa do kolekce daného hráče. Identifikátor je reprezentován ve formátu UUID (Universally Unique Identifier).

QR kódy umožňují replikaci dat, kdy při částečném poškození jsou data v kódu stále čitelná. Pro toto použití byla zvolena úroveň replikace Q, čímž je zajištěna čitelnost dat při poškození až 25 % kódu a komplexita kódu není příliš vysoká. Vyšší úroveň H nabízí čitelnost při poškození až 30 % kódu, ale výsledný kód je komplexnější. Menší úrovně replikace se již jeví nedostatečné po uvážení povětrnostních podmínek, které by mohly kód poškodit. Porovnání jednotlivých úrovní replikací je možné vidět na obrázku (Obrázek 25).

Výslednou známku v podobě QR kódu je možné umístit na veřejně dostupné místo jako jsou například informační tabule u dané památky. Známka může nabývat tištěné podoby nebo být vyhotovena na více odolný a trvanlivý materiál (např. dřevo, kov) pomocí laserového gravírování.

Obrázek 25. Porovnání komplexnosti QR kódu (zdroj vlastní)

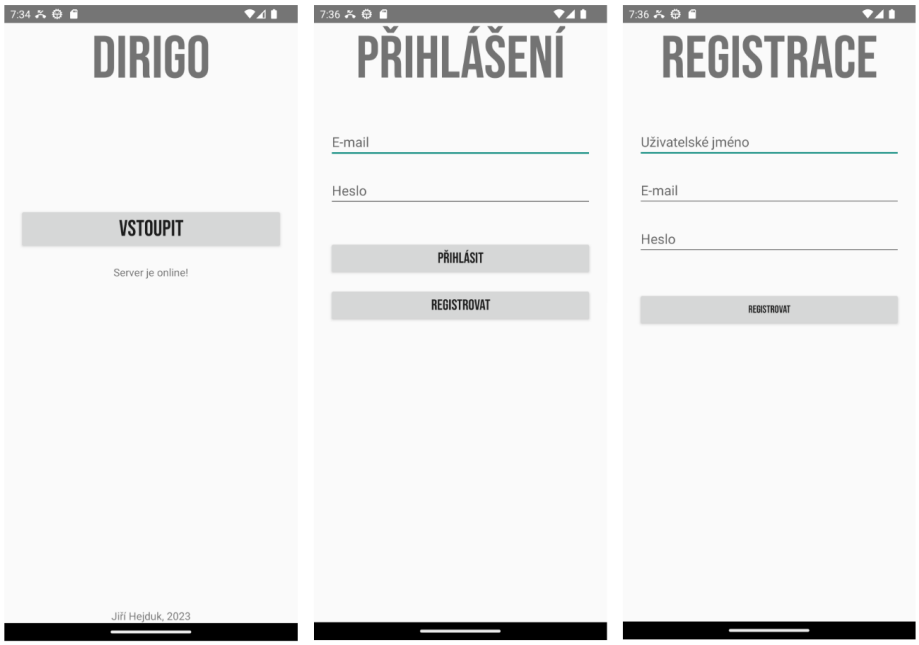
2.3.3 Mobilní aplikace

Mobilní aplikace je navržena pro použití hráči ke komunikaci se serverem a k prezentaci dat získaných ze serveru. Aplikace je vytvořena pro operační systém Android 13 a naprogramována v programovacím jazyce Kotlin. V aplikaci je použita knihovna OkHttp3, která obstarává http komunikaci se serverem, a knihovna ZXing, která slouží ke zpracování QR kódu. K vývoji bylo použito oficiální vývojové prostředí Android Studio.

Vstup do aplikace

Vstupní obrazovka aplikace slouží ke kontrole dosažitelnosti serveru, ke kterému se aplikace připojuje. V případě, že je server nedosažitelný, je o tom uživatel informován textovým sdělením a není vpuštěn do aplikace dále – tlačítko „Vstoupit“ deaktivované a uživatel nemůže pokračovat.

Po vstupu do aplikace, je uživatel vyzván k zadání přihlašovacích údajů. Pro přihlášení je požadována e-mailová adresa a heslo. V případě, že je uživatel úspěšně přihlášen, je přesměrován do hlavního menu. Pokud uživatel ještě není registrován, může přejít na registrační obrazovku kde vyplní požadované údaje a následně mu je vytvořen nový účet.



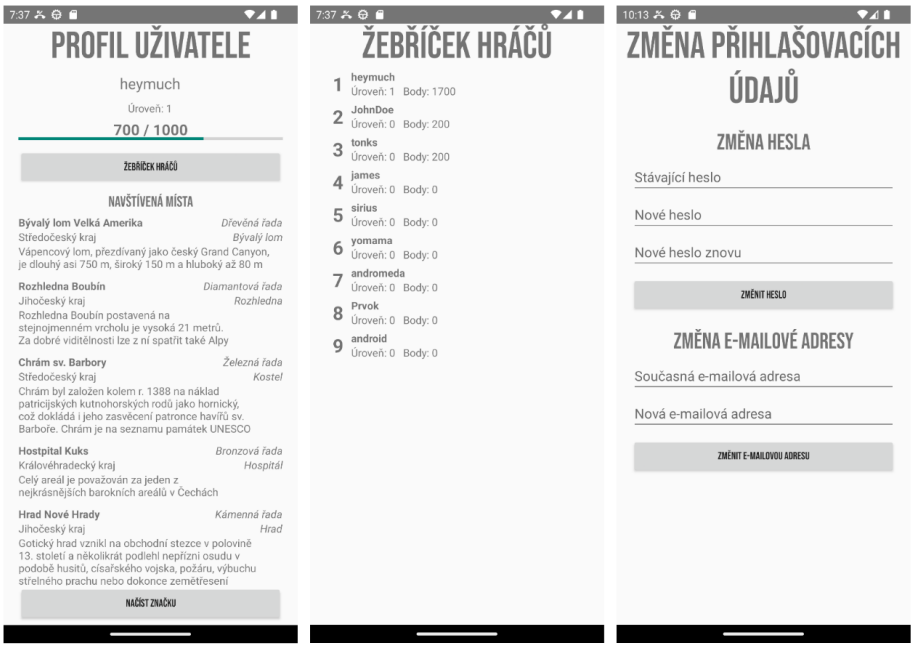
Obrázek 26. Úvodní obrazovka; Přihlášení uživatele; Registrace uživatele

Profil a správa uživatele

Profil hráče je tvořen dvěma sekcemi. V první sekci jsou základní informace o hráči, které jsou reprezentovány uživatelským jménem hráče, počtem nasbíraných bodů a postupem k získání další úrovně. Ve druhé sekci se nachází přehled již navštívených turistických míst. U každého místa je uveden název, oblast, krátký popis, typ památky a ohodnocení. Oblastmi jsou kraje České republiky. Typ památek napovídá, o jakou památku se jedná (skanzen, rozhledna, …). Ohodnocení udává, kolik bodů za navštívení dané památky hráč obdržel („Diamantová řada“ připisuje 700 bodů, „Dřevěná řada“ připisuje 100 bodů).

Žebříček hráčů poskytuje přehled všech registrovaných hráčů. Hráči jsou ve výčtu seřazeni sestupně dle získaných bodů.

Uživatelé si mohou v rámci správy svého účtu změnit přihlašovací údaje, které jsou reprezentovány heslem a e-mailovou adresou. Tato změna probíhá na separátní obrazovce „Změna přihlašovacích údajů“.

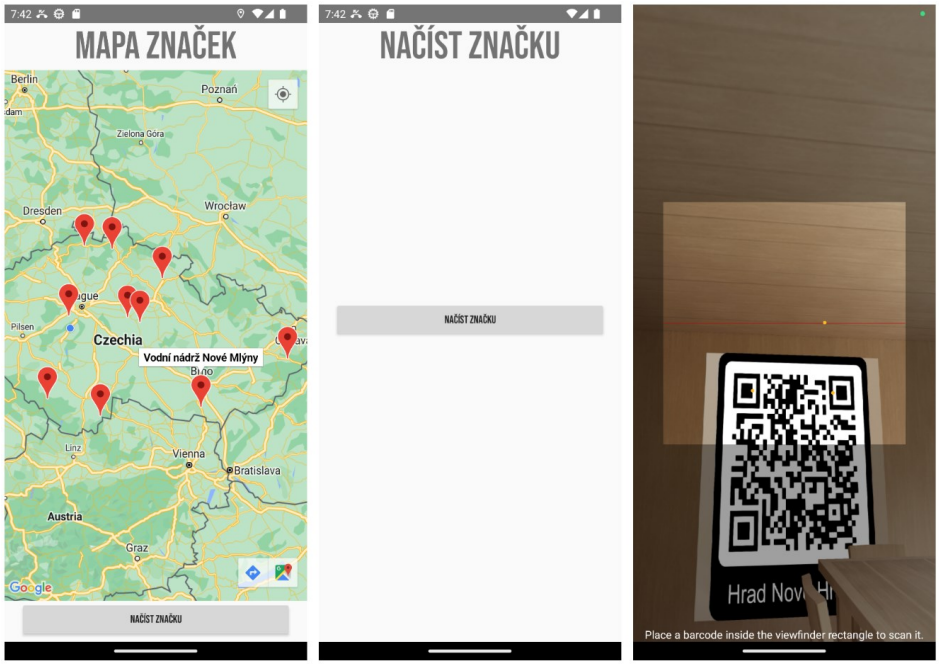


Obrázek 27. Profil hráče; Žebříček hráčů; Správa přihlašovacích údajů

Přehled a načítání známek

Přehled všech známek je řešen jako vizualizace jednotlivých známek v podobě červených bodů na mapovém podkladu. Pro mapový podklad byly využity mapové podklady od společnosti Google, Inc., které mají připravenou implementaci pro Android aplikaci. Při klepnutí na daný bod v mapě, se zobrazí název památky, která se zde nachází. Modrým bodem je reprezentována aktuální poloha hráče. K zjištění aktuální polohy hráče aplikací je zapotřebí udělení oprávnění uživatelem. Poskytnutí tohoto oprávnění je požadováno při otevření přehledu jednotlivých turistických míst.

Načtení známky probíhá na separátní obrazovce, přičemž je aktivován fotoaparát na mobilním zařízení a je průběžně vyhledáván QR kód, který lze přečíst. Pro načítání a zpracování QR kódů je použita knihovna ZXing. Pří úspěšném načtení známky je dané turistické místo přiřazeno do uživatelovy sbírky a uživatel je přesměrován zpět na svůj profil.



Obrázek 28. Přehled známek; Načítání značek

2.4 Možná rozšíření aplikace

2.4.1 Administrace

V rámci této bakalářské práce byla implementována pouze API brána určená pro použití s mobilní aplikací, která je využívána výhradně hráči. Pro další administraci celého systému je nutné implementovat nová rozhraní pro administraci společně s další aplikací. Je důležité, aby rozhraní a aplikace pro administraci byla oddělená od rozhraní a aplikace pro hráče kvůli zachování použití mikroslužeby, kde každá služba vyřizuje jednu problematiku. V tomto případě je doporučena implementace webové aplikace z důvodu dostupnosti jak na PC, tak na mobilních zařízeních.

2.4.2 Škálování, monitorování, zabezpečení

V produkčním prostředí by bylo vhodné implementovat způsob monitoringu jednotlivých částí celého projektu. Zde se nabízí řešení monitoringu pomocí open source technologií Logstash a Grafana, které umožňují přehled stavů a logů jednotlivých služeb.

Mikroslužby jsou navrhovány tak, aby bylo možné jejich jednoduché škálování. I přes tuto skutečnost existuje nutnost některé části aplikace na škálování připravit. Zásadní je zde synchronizace dat mezi jednotlivými instancemi databázových systémů. PostgreSQL umožňuje několik variant synchronizace dat mezi jednotlivými instancemi, ovšem je nutná jejich konfigurace. Dále je nutné vyřešit směrování dotazů mezi jednotlivé instance téže služby.

K tomu lze využít load balancer HAProxy, který zajistí rozložení HTTP dotazů dle předem definovaných pravidel.

V produkčním prostředí je také nutné zajistit bezpečný přenos dat mezi serverem a aplikací. K tomu lze použít protokol HTTPS, který zajišťuje šifrování obsahu jednotlivých HTTP dotazů. Pro vyřešení tohoto požadavku by bylo vhodné použít open source nástroj Nginx Proxy Manager, který poskytuje mnohé funkcionality včetně vystavování a obnovování SSL certifikátů, které jsou pro komunikaci pomocí HTTPS nutností.

ZÁVĚR

Tato bakalářská práce se zaměřuje na návrh a implementaci mobilní aplikace ve formě virtuální sběratelské hry. Aplikace umožňuje správu uživatelského profilu, následně hráči sbírají odměny ve formě bodů, které získají po načtení speciální značky ve formě QR kódu v blízkosti zajímavého turistického místa. Navštívená lokace je přidána do virtuální sbírky a zobrazena na mapě. Hráči jsou porovnáváni v bodové tabulce na základě nasbíraných bodů.

V teoretické části byly popsány technologie, které se vztahují k tvorbě mobilních aplikací a k návrhu aplikace jako mikroslužeb. Dále byly popsány projekty, které v návrhu sdílí některé funkcionality s navrhovanou aplikací.

Samotná implementace proběhla ve dvou oddělených postupech, přičemž nejprve byla implementována serverová část aplikace. Tato část byla navržena dle vzoru mikroslužeb a k vytvoření byly využity programovací jazyky Go a Kotlin. Také byly využity webové frameworky Gin a Ktor. Jednotlivé služby byly následně nasazeny jako kontejnery na platformě Docker za využití Docker Compose.

V další fázi byla implementována mobilní aplikace pro operační systém Android. Jejím hlavním úkolem je vizualizace dat pro uživatele a interakce se serverovou částí pomocí rozhraní REST API. Tato část aplikace je vytvořena v programovacím jazyce Kotlin a využívá externí knihovny OkHttp3 jako HTTP klienta a ZXing pro zpracování načtených QR kódů.

Mezi další budoucí rozšíření aplikace lze zahrnout implementaci administrační části, která bude sloužit pro správu jednotlivých uživatelů a turistických míst. Pro produkční nasazení by bylo vhodné zabezpečit komunikaci mezi aplikací a serverem pomocí protokolu HTTPS, který zajišťuje šifrovanou komunikaci. Dalším případným rozšířením může být optimalizace služeb pro škálovatelnost, aby se navýšila dostupnost aplikace i při vytížení větším počtem hráčů.

POUŽITÁ LITERATURA

[1] LARICCHIA, Federica. Number of smartphones sold to end users worldwide from 2007 to 2021. In: *Statista* [online]. 2022 [cit. 2023-05-08]. Dostupné z: https://www.statista.com/statistics/263437/global-smartphone-sales-to-end-users-since-2007/

[2] Mobile operating systems' market share worldwide from 1st quarter 2009 to 4th quarter 2022. In: *Statista* [online]. 2023 [cit. 2023-05-07]. Dostupné z: https://www.statista.com/statistics/272698/global-market-share-held-by-mobile-operating-systems-since-2009/

[3] What is Android?. In: *Android* [online]. Mountain View, California, USA: Google Inc., 2022 [cit. 2023-05-07]. Dostupné z: https://www.android.com/what-is-android/

[4] What is Android: Here's everything you need to know. In: *Android Authority* [online]. Newark, Delaware, United States: Authority Media, 2022 [cit. 2023-05-07]. Dostupné z: https://www.androidauthority.com/what-is-android-328076/

[5] Android Studio. In: *Google Developers* [online]. California: Google, 2023 [cit. 2023-05-07]. Dostupné z: https://developer.android.com/studio

[6] VOLLE, Adam. IOS: Operating system. In: *Encyclopedia Britannica* [online]. United States: Encyclopedia Britannica, 2022 [cit. 2023-05-07]. Dostupné z: https://www.britannica.com/topic/iOS

[7] IOS 16. In: *MacRumors* [online]. Virginia: MacRumors, 2023 [cit. 2023-05-07]. Dostupné z: https://www.macrumors.com/roundup/ios-16/

[8] JOSEPH, Christina Terese a K. CHANDRASEKARAN. Straddling the crevasse: A review of microservice software architecture foundations and recent advancements. *Softw: Pract Exper*. 2019, **49**(10), 1448–1484. Dostupné z: doi:https://doi.org/10.1002/spe.2729

[9] CORREIA, José a António RITO SILVA. Identification of monolith functionality refactorings for microservices migration. *Software: Practice and Experience* [online].

2022, **52**(12), 2664-2683 [cit. 2023-05-07]. ISSN 0038-0644. Dostupné z: doi:10.1002/spe.3141

[10] BALALAIE, Armin, Abbas HEYDARNOORI, Pooyan JAMSHIDI, Damian A. TAMBURRI a Theo LYNN. Microservices migration patterns. *Software: Practice and Experience* [online]. 2018, **48**(11), 2019–2042 [cit. 2023-05-07]. ISSN 00380644. Dostupné z: doi:10.1002/spe.2608

[11] PHILLIPS, Mattie. What is Microservices Architecture?. In: *PDF Tables* [online]. 2020 [cit. 2023-05-08]. Dostupné z: https://pdftables.com/blog/what-is-microservices-architecture

[12] Microservices. In: *MartinFowler.com* [online]. 2014 [cit. 2023-05-08]. Dostupné z: https://martinfowler.com/articles/microservices.html

[13] VAILSHERY, Lionel Sujay. Leading containerization technologies market share worldwide in 2022. In: *Statista* [online]. 2022 [cit. 2023-05-08]. Dostupné z: https://www.statista.com/statistics/1256245/containerization-technologies-software-market-share/

[14] Hearthstone. In: *Google Play* [online]. Blizzard Entertainment, 2014 [cit. 2023-05-08]. Dostupné z: https://play.google.com/store/apps/details?id=com.blizzard.wtcg.hearthstone&pli=1

[15] Pokémon Go. In: *Polygon* [online]. 2016 [cit. 2023-05-08]. Dostupné z: https://www.polygon.com/2016/7/14/12183956/pokemon-go-review-ios-android-nintendo-niantic-company-mobile-game

[16] Turistické známky. In: *Turistické známky* [online]. 1999 [cit. 2023-05-08]. Dostupné z: https://www.turisticke-znamky.cz

[17] Geocaching. In: *Geocaching* [online]. [cit. 2023-05-09]. Dostupné z: https://www.geocaching.com/play

[18] What is a QR code?. In: *Insider* [online]. 2021 [cit. 2023-05-10]. Dostupné z: https://www.businessinsider.com/guides/tech/what-is-a-qr-code?op=1

[19] The Go Programming Language Specification. In: *The Go Blog* [online]. California: Google, 2022 [cit. 2023-05-07]. Dostupné z: https://go.dev/ref/spec

[20] Kotlin. In: *Kotlin* [online]. Praha: JetBrains, 2023 [cit. 2023-05-07]. Dostupné z: https://kotlinlang.org/docs/faq.html

[21] Kotlin brand assets. In: *Kotlin* [online]. 2023 [cit. 2023-05-09]. Dostupné z: https://kotlinlang.org/docs/kotlin-brand-assets.html

[22] Gin Web Framework. In: *Gin Web Framework* [online]. 2022 [cit. 2023-05-09]. Dostupné z: https://gin-gonic.com/docs/introduction/

[23] Ktorio / ktor. In: *GitHub* [online]. 2022 [cit. 2023-05-09]. Dostupné z: https://github.com/ktorio/ktor

[24] Zxing / zxing. In: *GitHub* [online]. 2023 [cit. 2023-05-09]. Dostupné z: https://github.com/zxing/zxing

[25] *What is PostgreSQL?* [online]. In: . 1986 [cit. 2023-05-09]. Dostupné z: https://www.ibm.com/topics/postgresql

[26] PostgreSQL. In: *PostgreSQL* [online]. 2023 [cit. 2023-05-09]. Dostupné z: https://www.postgresql.org/about/

[27] TIŠNOVSKÝ, Pavel. Webová pískoviště a integrovaná vývojová prostředí. In: *Root.cz* [online]. 2023 [cit. 2023-05-09]. Dostupné z: https://www.root.cz/clanky/webova-piskoviste-a-integrovana-vyvojova-prostredi/

[28] GoLand. In: *JetBrains* [online]. 2023 [cit. 2023-05-09]. Dostupné z: https://www.jetbrains.com/go/

[29] DataGrip. In: *JetBrains* [online]. 2023 [cit. 2023-05-09]. Dostupné z: https://www.jetbrains.com/datagrip/

[30] IntelliJ IDEA. In: *JetBrains* [online]. 2023 [cit. 2023-05-09]. Dostupné z: https://www.jetbrains.com/idea/

[31] What is Gradle?. In: *Gradle* [online]. 2023 [cit. 2023-05-09]. Dostupné z: https://docs.gradle.org/current/userguide/what_is_gradle.html

[32] Flyway Documentation. In: *Redagate Documentation* [online]. 2023 [cit. 2023-05-10]. Dostupné z: https://documentation.red-gate.com/fd

[33] *Postman* [online]. San Francisco: Postman, 2023 [cit. 2023-05-07]. Dostupné z: https://www.postman.com/product/what-is-postman/

[34] Docker / compose. In: *GitHub* [online]. 2022 [cit. 2023-05-09]. Dostupné z: https://github.com/docker/compose