



**VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ**  
BRNO UNIVERSITY OF TECHNOLOGY

**FAKULTA INFORMAČNÍCH TECHNOLOGIÍ**  
FACULTY OF INFORMATION TECHNOLOGY

**ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ**  
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

**DOMÁCÍ KVÍZ S VYUŽITÍM MOBILNÍHO TELEFONU  
JAKO HLASOVACÍHO ZAŘÍZENÍ**

HOME QUIZ USING A MOBILE PHONE AS A VOTING DEVICE

**DIPLOMOVÁ PRÁCE**  
MASTER'S THESIS

**AUTOR PRÁCE**  
AUTHOR

**Bc. MARTIN BALÁŽ**

**VEDOUCÍ PRÁCE**  
SUPERVISOR

**Ing. MICHAL KAPINUS, Ph.D.**

**BRNO 2025**

## Zadání diplomové práce



Ústav: Ústav počítačové grafiky a multimédií (UPGM) 164010  
Student: **Baláž Martin, Bc.**  
Program: Informační technologie a umělá inteligence  
Specializace: Vývoj aplikací  
Název: **Domácí kvíz s využitím mobilního telefonu jako hlasovacího zařízení**  
Kategorie: Uživatelská rozhraní  
Akademický rok: 2024/25

Zadání:

1. Prostudujte postupy návrhu uživatelských rozhraní moderních mobilních a webových aplikací.
2. Seznamte se s existujícími kvízovými aplikacemi.
3. Vyberte vhodné metody a nástroje a navrhněte systém pro hraní domácího kvízu. V systému bude vystupovat jedno zařízení jako hlavní obrazovka zobrazující aktuální otázku a průběh kvízu (např. televize, počítač apod.) a dále každý hráč využívá svůj mobilní telefon jako zařízení pro odpovídání na otázky nebo pro přihlašování k odpovědi.
4. Navrhněte několik druhů kvízových her (např. odpovědi ABCD, otevřené odpovědi, slepá mapa, pantomima apod.). Dbejte na rozšířitelnost aplikace a snadnost editace databáze otázek a odpovědí.
5. Navrženou aplikaci implementujte. Zaměřte se na použitelnost a zábavnost výsledného řešení.
6. Proveďte uživatelské experimenty, demonstrujte a diskutujte vlastnosti vašeho řešení.
7. Vytvořte video prezentující vaši práci, její cíle a výsledky.

Literatura:

- Dieter Schmalstieg, Tobias Hollerer. *Augmented Reality: Principles and Practice*. Addison-Wesley, 2016. ISBN: 978-0321883575.
- Russ Unger, Carolyn Chandler. *A Project Guide to UX Design: For user experience designers in the field or in the making*. New Riders, 2012. ISBN: 0132931729.
- Dále dle pokynu vedoucího.

Při obhajobě semestrální části projektu je požadováno:

Body 1, 2, 3, 4 a značně rozpracovaný bod 5.

Podrobné závazné pokyny pro vypracování práce viz <https://www.fit.vut.cz/study/theses/>

Vedoucí práce: **Kapinus Michal, Ing., Ph.D.**

Vedoucí ústavu: Černocký Jan, prof. Dr. Ing.

Datum zadání: 1.11.2024

Termín pro odevzdání: 21.5.2025

Datum schválení: 12.11.2024

## Abstrakt

Cílem této diplomové práce je vytvoření kvízové aplikace pro operační systém Windows, která umožní hráčům soutěžit v jedné místnosti prostřednictvím lokální sítě. Otázky a průběh hry se zobrazují na hlavní obrazovce, zatímco hráči odpovídají pomocí svých mobilních zařízení. Řešení je realizováno jako webová aplikace s klientskou částí v Reactu a serverovou částí ve Flasku. Aplikace zahrnuje několik různorodých disciplín, které lze kombinovat do jedné hry, a funkce pro tvorbu a sdílení vlastních kvízů. Funkčnost herního režimu byla ověřena testováním se dvěma skupinami po šesti hráčích, přičemž výsledky byly nadprůměrné.

## Abstract

The aim of this master's thesis is to create a quiz application for the Windows operating system that allows players to compete in the same room over a local network. Questions and game progress are displayed on a main screen, while players submit their answers using their mobile devices. The solution is implemented as a web application with a client-side developed in React and a server-side in Flask. The application includes several diverse game disciplines that can be combined into a single game, along with features for creating and sharing custom quizzes. The functionality of the game mode was verified through testing with two groups of six players, yielding above-average results.

## Klíčová slova

kvízová aplikace, webová aplikace, lokální síť, mobilní zařízení, uživatelské rozhraní, uživatelská zkušenost, React, Flask, API, WebSocket, Socket.IO, ABCD kvíz, disciplíny, hlasování, vestavěný server, otázky, kvízy, MongoDB, Cloudinary

## Keywords

quiz application, web application, local network, mobile device, user interface, user experience, React, Flask, API, WebSocket, Socket.IO, ABCD quiz, disciplines, voting, embedded server, questions, quizzes, MongoDB, Cloudinary

## Citace

BALÁŽ, Martin. *Domácí kvíz s využitím mobilního telefonu jako hlasovacího zařízení*. Brno, 2025. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Michal Kapinus, Ph.D.

# Domácí kvíz s využitím mobilního telefonu jako hlasovacího zařízení

## Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením pana Ing. Michala Kapinuse, Ph.D. Při implementaci jsem využil asistenci nástroje GitHub Copilot (Claude model), který mi pomáhal s generováním dokumentačních komentářů, refaktORIZACÍ kódu a u typu Kreslení výrazně přispěl k implementaci kreslící funkcionality. Dále jsem využil nástroj Writefull pro kontrolu pravopisu v Overleafu a aplikaci ChatGPT pro jazykové úpravy textu. Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....  
Martin Baláž  
19. května 2025

## Poděkování

Rád bych poděkoval vedoucímu práce panu Ing. Michalu Kapinusovi, Ph.D. za odborné vedení, cenné rady a podporu, které mi byly velkou pomocí při psaní této práce a při vývoji aplikace. Dále děkuji všem, kteří se podíleli na testování aplikace, zejména svým přátelům a rodině, za jejich čas a cennou zpětnou vazbu.

# Obsah

<b>1</b>	<b>Úvod</b>	<b>2</b>
<b>2</b>	<b>Vývoj responzivní webové aplikace s využitím vestavěného serveru</b>	<b>3</b>
2.1	Back-end webové aplikace . . . . .	3
2.2	Uživatelské rozhraní . . . . .	5
2.3	Uživatelská zkušenosť . . . . .	9
2.4	Komunikace v reálném čase . . . . .	12
<b>3</b>	<b>Návrh řešení</b>	<b>15</b>
3.1	Průzkum podobných řešení . . . . .	15
3.2	Návrh kvízů . . . . .	22
3.3	Průzkum trhu . . . . .	33
3.4	Návrh uživatelského rozhraní . . . . .	35
3.5	Návrh NoSQL databáze . . . . .	42
3.6	Návrh API a WebSocketů . . . . .	44
<b>4</b>	<b>Architektura a implementace</b>	<b>49</b>
4.1	Architektura a technologie . . . . .	49
4.2	První spuštění aplikace a rychlá hra . . . . .	52
4.3	Systém správy kvízů . . . . .	56
4.4	Čekací místnost a spuštění hry . . . . .	60
<b>5</b>	<b>Implementace herních mechanik</b>	<b>64</b>
5.1	Společný základ všech typů otázek . . . . .	64
5.2	Základní typy otázek . . . . .	68
5.3	Pokročilé typy otázek . . . . .	72
5.4	Dynamické typy otázek . . . . .	77
5.5	Nedostatky implementace a možná vylepšení . . . . .	82
<b>6</b>	<b>Testování aplikace</b>	<b>83</b>
6.1	Průběh testování s uživateli . . . . .	84
6.2	Zpětná vazba uživatelů . . . . .	85
6.3	Závěrečné hodnocení aplikace . . . . .	86
<b>7</b>	<b>Závěr</b>	<b>88</b>
<b>Literatura</b>		<b>89</b>
<b>A Přehled odevzdaných souborů na NextCloudu</b>		<b>92</b>

# Kapitola 1

## Úvod

V dnešní době je hraní kvízů oblíbenou formou zábavy i vzdělávání. Lidé si často chtějí zahrát něco s rodinou, přáteli nebo kolegy, co je nejen pobaví, ale zároveň je donutí přemýšlet a soutěžit. Přestože existuje řada kvízových aplikací, většina z nich má své limity. Některé jsou složité na nastavení, většina není dostupná v českém jazyce a další postrádají dostatek rozmanitosti v nabízených disciplínách.

Cílem této diplomové práce je vytvořit kvízovou aplikaci pro operační systém Windows, která překoná uvedené nedostatky, a která umožní hrát hru v jedné místnosti v rámci lokální sítě. Hra se spouští na počítači, kde se otázky a průběh hry zobrazují na připojené obrazovce. Alternativně lze využít i jiné zařízení v síti, například chytrou televizi s webovým prohlížečem. Všichni hráči se připojují pomocí svých mobilních telefonů, na kterých odpovídají. Klíčovým rysem aplikace je nabídka několika různorodých disciplín, které lze kombinovat do jedné hry. Tím je zajištěna rozmanitost a zábavnost, díky nimž hra jen tak neomrzí. Každou disciplínu je navíc možné hrát jak v týmovém režimu, tak v režimu všichni proti všem. Další přidanou hodnotou aplikace je možnost vytvářet vlastní kvízy a sdílet je s ostatními uživateli, stejně jako vytvářet kopie kvízů vytvořených jinými hráči. Aplikace je navržena primárně pro české uživatele a to tak, aby byla zábavná a jednoduchá na používání.

V teoretické kapitole 2 jsou uvedeny důležité informace týkající se uvažovaných programovacích jazyků, technologií pro komunikaci v reálném čase a dalších aspektů nezbytných pro vývoj webové aplikace. Kapitola 3 se zabývá podrobnějším popisem podobných aplikací a inspirací z populární televizní soutěže *Máme rádi Česko*, která výrazně ovlivnila výběr disciplín. Tato kapitola rovněž zahrnuje návrh uživatelského rozhraní aplikace, včetně průběhu hry, procesu vytváření kvízů a dalších funkcionalit. Návrh byl podpořen průzkumem trhu, jehož cílem bylo zjistit názory uživatelů na různé disciplíny. Architektura a implementace aplikace jsou popsány v kapitole 4, která pokrývá spuštění aplikace, tvorbu kvízů a organizaci herní místnosti. Implementace herních mechanik, včetně sdíleného základu i konkrétních typů otázek, je rozebrána v kapitole 5. Závěrečná kapitola 6 popisuje průběh testování aplikace s uživateli, získanou zpětnou vazbu a závěrečné zhodnocení aplikace. Tyto části pomohly ověřit její funkčnost, použitelnost i zábavnost. Zpětná vazba i zjištěné nedostatky ukázaly, jakým směrem by se mohl ubírat další vývoj aplikace.

## Kapitola 2

# Vývoj responzivní webové aplikace s využitím vestavěného serveru

Cílem bylo vytvořit desktopovou aplikaci určenou pro společné hraní domácího kvízu v jedné místnosti. Aplikace měla být snadno spustitelná, umožnit připojení více zařízení (hlavní obrazovky a mobilních zařízení hráčů) a intuitivní ovládání. K naplnění těchto požadavků jsem zvolil řešení postavené na webové aplikaci s vestavěným serverem pomocí frameworku Flask. Po spuštění desktopového .exe souboru se na hlavním zařízení otevře webový prohlížeč s uživatelským rozhraním pro řízení hry. Hráči se připojují prostřednictvím svých mobilních zařízení nebo notebooků přes lokální síť. Pro zajištění plynulé komunikace a synchronizace v reálném čase jsou využity WebSockets.

Následující podkapitoly se věnují klíčovým komponentům vývoje back-endu webové aplikace, včetně popisu webového (vestavěného) serveru a přehledu možných vývojových platforem, a také jejich výhod a nevýhod. Dále je zde rozebrána problematika návrhu uživatelského rozhraní s ohledem na UX (uživatelská zkušenost), včetně doporučených nástrojů pro vývoj front-endu. Nakonec je vysvětlena komunikace v reálném čase prostřednictvím websocketů, které jsou klíčové pro funkčnost této kvízové aplikace.

### 2.1 Back-end webové aplikace

Back-end webové aplikace tvoří serverová část, která zajišťuje komunikaci s klienty, správu dat a zpracování jejich požadavků. Tato podkapitola se zaměřuje na dva klíčové koncepty: webové servery a vestavěné webové servery. Následně budou podrobně představeny tři různé frameworky vhodné pro vývoj webového serveru v rámci této diplomové práce. Tyto frameworky umožňují implementaci websocketů a podporují využití vestavěného webového serveru, což je klíčové pro dosažení požadované funkcionality aplikace.

#### 2.1.1 Webový server a vestavěný webový server

Webový server je kombinace hardwaru a softwaru, která zpracovává a poskytuje obsah webových stránek uživatelům prostřednictvím protokolů, jako je HTTP. Ukládá a spravuje webové soubory, jako jsou texty, obrázky a videa, a odpovídá na požadavky uživatelů v prohlížečích. Webové servery využívají model klient-server a slouží nejen k hostování webových stránek, ale také pro přenos souborů nebo e-mailové služby. Jsou klíčovou součástí infrastruktury internetu [3].

Vestavěné servery jsou užitečné pro vytváření samostatných aplikací, které mohou poskytovat webové rozhraní bez potřeby externího serverového prostředí. To je zvláště výhodné pro desktopové aplikace nebo zařízení IoT<sup>1</sup>, kde je žádoucí mít integrovaný server pro konfiguraci, monitorování nebo interakci s uživateli.

Výhodou vestavěného webového serveru je jednodušší nasazení, neboť aplikace nevyžaduje instalaci a konfiguraci externího serveru. Dále umožňuje snadnější distribuci, protože aplikace může být nabízena jako jeden spustitelný balíček, což zjednoduší její šíření a instalaci. Nevýhodou tohoto přístupu může být nižší výkon ve srovnání s externím webovým serverem. Tento aspekt však v rámci této diplomové práce není problémem, protože server bude komunikovat současně maximálně s deseti hráči.

### 2.1.2 Vývojové platformy webového serveru

Existuje mnoho frameworků zaměřených na vývoj webových serverů. Pro svou práci jsem se rozhodl prozkoumat tři z nejpopulárnějších frameworků současnosti, přičemž každý je postaven na jiném programovacím jazyce. Tato sekce přináší jejich popis a závěrečné porovnání.

#### Flask

Flask je lehký webový framework založený na Pythonu, často označovaný jako mikroframework. Byl navržen tak, aby umožňoval rychlé prototypování a snadné rozšíření o další funkce prostřednictvím externích knihoven. Flask se zaměřuje na jednoduchost, modularitu a flexibilitu. Díky minimalistickému designu poskytuje vývojářům volnost v implementaci a zároveň podporuje rozšíření, jako je uživatelská autentizace, validace formulářů nebo práce s databázemi [5].

Flask je oblíbený zejména pro menší projekty a aplikace s nízkou složitostí. Pro větší projekty může být zapotřebí rozšířit funkcionalitu pomocí dalších knihoven nebo nástrojů. Jeho flexibilita z něj činí vhodnou volbu pro menší webové aplikace nebo prototypy, kde je potřeba rychlého vývoje a snadného přizpůsobení.

#### Node.js s Express.js

Node.js představuje platformu umožňující provozování JavaScriptu na straně serveru. Je založen na architektuře s událostmi (také známé jako event-driven architektura) a neblokujícím vstupem/výstupem, což z něj činí výkonnou volbu pro real-time aplikace a aplikace s vysokým zatížením [12].

Framework Express.js je minimalistický nástroj postavený na Node.js, který nabízí základní sadu funkcí pro vývoj webových a mobilních aplikací. Express.js přidává na jednoduchosti a flexibilitě, přičemž poskytuje intuitivní API pro směrování požadavků, middleware a různé HTTP utility. Díky svému modulárnímu návrhu umožňuje rychlou tvorbu aplikací a je často využíván při vývoji RESTful API. Node.js s Express.js je ideální volbou pro aplikace vyžadující nízkou latenci a škálovatelnost [23].

---

<sup>1</sup>IoT označované také jako Internet věcí (Internet of Things) je souhrnný název pro síť fyzických zařízení, která jsou vzájemně propojená, shromažďují a vyměňují si data přes internet nebo další komunikační síť.

## Spring Boot

Spring Boot je framework postavený na Javě, který výrazně zjednodušuje vývoj aplikací díky svému přístupu založenému na principech „konvence místo konfigurace“, což znamená, že framework využívá výchozí nastavení pro běžné scénáře, aniž by vývojář musel provádět složitou ruční konfiguraci. Navíc poskytuje vestavěné servery (např. Tomcat nebo Jetty) a automatizaci konfigurace, která dále usnadňuje vývoj a eliminuje nutnost psaní nadbytečného kódu [18].

Spring Boot je známý svou robustností a podporou při vývoji mikroslužeb. Jeho součástí je modul Actuator, který poskytuje nástroje pro sledování aplikací a jejich stavu. Tento framework je preferovanou volbou pro podnikové aplikace díky své stabilitě, integraci s cloudovými službami a podpoře moderních vývojových praktik [23].

### 2.1.3 Porovnání frameworků

Výše zmíněné frameworky jsou v této podsekci blíže porovnány mezi sebou. Informace v této podsekci jsou převzaté z [23].

- **Výkon** – Node.js s Express.js vyniká výkonem díky neblokující architektuře, což jej činí ideálním pro aplikace s vysokým zatížením a real-time komunikací. Flask je méně výkonný, ale pro menší aplikace je jeho jednoduchost dostatečná. Spring Boot poskytuje robustní výkon, ale jeho režijní náklady mohou být vyšší v porovnání s minimalistickými frameworky.
- **Škálovatelnost** – Node.js s Express.js a Spring Boot jsou vysoce škálovatelné a často se používají pro rozsáhlé aplikace a mikroservisní architektury. Flask je vhodný pro menší a středně velké aplikace, ale při větších nasazeních může vyžadovat další optimalizace.
- **Snadnost použití** – Flask nabízí jednoduchost a intuitivní použití, což z něj činí ideální volbu pro začátečníky a menší projekty. Node.js s Express.js vyžaduje znalost JavaScriptu a práce s asynchronními operacemi. Spring Boot je komplexnější, což může znamenat strmější křivku učení, ale nabízí bohaté možnosti konfigurace a rozšíření.

Pro svou diplomovou práci jsem zvolil framework Flask především kvůli jeho jednoduchosti, přehlednosti a snadnému použití. Flask umožňuje rychlý vývoj bez nutnosti studovat složitou strukturu větších frameworků, jako je například Django. Flask navíc obsahuje vestavěný webový server, díky čemuž lze aplikaci snadno spouštět lokálně bez složité konfigurace. Vzhledem k tomu, že cílem bylo vytvořit menší, lokálně běžící aplikaci bez nároků na škálovatelnost, je Flask zcela dostačujícím řešením. Volba Pythonu zároveň představovala příležitost rozšířit své znalosti o nový programovací jazyk.

## 2.2 Uživatelské rozhraní

Uživatelské rozhraní (známé pod anglickou zkratkou UI – User Interface) zahrnuje různé způsoby interakce mezi uživatelem a systémem, přičemž existují tři hlavní typy: grafické uživatelské rozhraní (GUI), uživatelské rozhraní založené na gestech a hlasové rozhraní (známé pod anglickou zkratkou VUIs – Voice-controlled interfaces). Nejrozšířenějším typem je GUI,

které se zaměřuje na vizuální prvky, jako jsou tlačítka, ikony nebo menu, a umožňuje uživatelům intuitivní interakci prostřednictvím obrazovek. Tento přístup klade důraz na estetiku, jednoduchost a konzistenci. GUI se nachází v mnoha typech zařízení, včetně mobilních telefonů, tabletů, počítačů, chytrých televizí a dalších, a je standardem pro většinu moderních aplikací. Uživatelské rozhraní založené na gestech, které zahrnuje interakci prostřednictvím pohybů nebo gest, se stává stále populárnějším, zejména v oblasti mobilních zařízení a virtuální reality, kde se klade důraz na přirozený a interaktivní zážitek. Hlasové rozhraní, běžně využívané ve virtuálních asistentech, poskytuje další alternativu pro ovládání zařízení bez nutnosti fyzického kontaktu [11].

Vzhledem k rozšírenosti grafických rozhraní a důrazu na jejich vizuální kvalitu se v praxi využívají návrhové jazyky, které pomáhají zajistit jednotný a přívětivý vzhled aplikací. Material Design<sup>2</sup> byl vytvořen s cílem usnadnit grafickým návrhářům tvorbu uživatelských rozhraní a zajistit vizuální konzistenci mezi různými vývojáři a designéry. Tento návrhový jazyk, který byl představen společností Google v roce 2014, se zaměřuje na poskytování intuitivního a jednotného uživatelského zážitku napříč různými platformami a zařízeními [7]. Kromě zjednodušení návrhu rozhraní, Material Design usiluje o to, aby výsledný vzhled aplikací byl vizuálně soudržný a uživatelsky přívětivý ve všech typech prostředí.

Vývoj moderních front-endových aplikací dnes běžně staví na pokročilých frameworcích a knihovnách, které usnadňují správu komponent, stavů a komunikaci se serverem. Mezi nejrozšířenější patří React, Vue.js a Angular, které rozšiřují možnosti práce s JavaScriptem či TypeScriptem a přinášejí lepší strukturu a škálovatelnost. V případě této diplomové práce bylo zapotřebí řešení, které umožní snadnou údržbu i rozšiřování aplikace. Tato podkapitola se zaměřuje na přiblížení zmíněných platforem a na posouzení jejich výhod a nevýhod s ohledem na požadavky kvízové aplikace.

### 2.2.1 Vývojové platformy uživatelského rozhraní webové aplikace

Existuje mnoho frameworků určených pro vývoj uživatelského rozhraní webových aplikací. Pro svou práci jsem se rozhodl prozkoumat tři velice známé frameworky dnešní doby, přičemž každý z nich nabízí odlišný přístup k tvorbě moderních webových aplikací. Při výběru jsem kladl důraz na schopnost frameworku efektivně komunikovat v reálném čase a na dostupnost knihoven pro Material Design, které zajišťují konzistentní a moderní vzhled aplikace. Tato sekce přináší jejich popis a závěrečné porovnání.

#### Angular

Angular je robustní front-end framework vyvinutý společností Google, zaměřený na vývoj škálovatelných webových aplikací. Využívá TypeScript, což přináší výhody statického typování a lepší organizaci kódu. Angular je známý svou modulární architekturou, která umožňuje strukturovat aplikace do přehledných částí a podporuje opětovné využití kódu. Klíčovou vlastností Angularu je dvoucestná datová vazba, která zajišťuje automatickou synchronizaci mezi modelem a uživatelským rozhraním, což usnadňuje správu stavu aplikace. Díky vestavěným nástrojům, jako je Angular CLI, směrování, formuláře a správa závislostí, poskytuje Angular kompletní řešení pro vývoj komplexních aplikací bez nutnosti integrace externích knihoven. I když se jeho osvojení může zdát náročnější a vývoj v něm může být složitější než u jiných frameworků, jeho strukturovaný přístup a dlouhodobá podpora od Google z něj činí atraktivní volbu pro rozsáhlé podnikové aplikace [17].

---

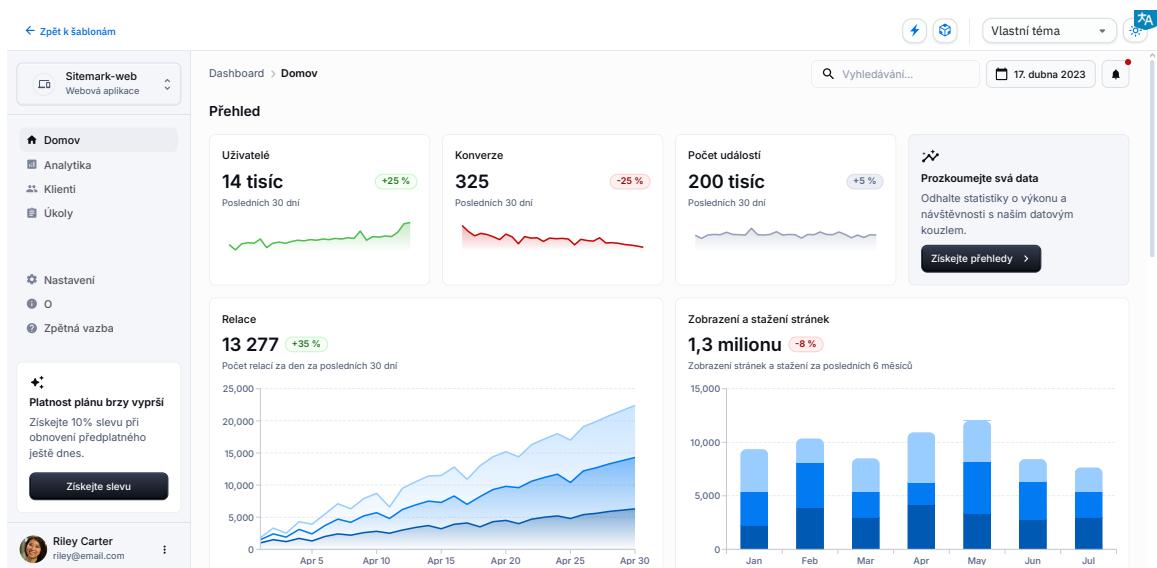
<sup>2</sup><https://m3.material.io/>

V Angularu se komponenty definují pomocí dekorátoru `@Component`, kde se specifikuje selektor, šablona a styly komponenty. Každá komponenta má svůj životní cyklus řízený frameworkem a implementuje metody jako `ngOnInit()` pro inicializaci dat a `ngOnDestroy()` pro uvolnění zdrojů. Pro práci s daty Angular nabízí několik typů datové vazby, například interpolaci `property_name`, která umožňuje přístup k hodnotám proměnných v šabloně. K propojení vlastností komponenty s HTML atributy se používá vlastnostní vazba `[property]`. Pokud je třeba zajistit obousměrnou synchronizaci vstupního pole s datovým modelem, využívá se obousměrná vazba `[(ngModel)]`. Angular dále podporuje **Dependency Injection**, což umožňuje efektivní správu závislostí v aplikaci a usnadňuje testování. Důležitou součástí frameworku jsou také **NgModules**, které umožňují rozdělit aplikaci do logických celků, čímž se zlepšuje její organizace a udržovatelnost [4].

## React

React, vyvinutý společností Meta (dříve Facebook), je knihovna pro vytváření uživatelských rozhraní, která se zaměřuje na komponentově orientovaný vývoj. Jednou z jeho hlavních vlastností je využití virtuálního DOM (Document Object Model), který optimalizuje vykreslování a zvyšuje výkon aplikací. React používá syntaxi JSX, což je kombinace JavaScriptu a HTML, umožňující intuitivní zápis komponent. Díky své flexibilitě může být React použit nejen pro webové aplikace, ale i pro vývoj mobilních aplikací prostřednictvím React Native. Na rozdíl od Angularu neposkytuje kompletní řešení pro řízení stavu aplikace nebo směrování, takže pro tyto funkcionality je často nutné využít externí knihovny, jako jsou Redux nebo React Router. Přestože React nabízí rychlý vývoj a širokou komunitní podporu, jeho flexibilita může vést k vyšší složitosti při rozhodování o architektuře aplikace [17].

Na React existuje také populární knihovna založená na zásadách Material designu, která se nazývá Material UI, nebo zkráceně MUI. Tato knihovna poskytuje sadu předpřipravených komponent, které umožňují rychle vytvářet responzivní a vizuálně konzistentní uživatelská rozhraní. Na obrázku 2.1 je ukázka šablony z oficiální dokumentace, která demonstrouje použití základních komponent, jako jsou tlačítka, karty nebo navigační lišty.



Obrázek 2.1: Na obrázku lze vidět ukázku šablony z Material UI dokumentace [15], na které lze vidět různé komponenty z knihovny MUI pro React.

React komponenty mohou být funkční nebo třídové. Funkční komponenty využívají Hooks, jako `useState()` pro správu stavu a `useEffect()` pro provádění vedlejších efektů. Třídové komponenty implementují metody životního cyklu jako `componentDidMount()` pro inicializaci dat a `componentWillUnmount()` pro úklid před odstraněním komponenty. React neumožňuje komponentám měnit své vlastní vlastnosti (props), proto se ke správě změn stavu využívá metoda `setState()` v třídových komponentách nebo `useState()` v těch funkčních [4].

## Vue.js

Vue.js je progresivní framework, který kombinuje prvky Angularu a Reactu a klade důraz na jednoduchost a snadnou integraci. Jeho základní koncepty jsou snadno pochopitelné i pro začátečníky, což z něj činí atraktivní volbu pro nové vývojáře i pro týmy, které potřebují rychlý vývoj aplikací. Vue.js využívá reaktivní datovou vazbu podobně jako Angular a pracuje s virtuálním DOM jako React, což mu umožňuje dosahovat vysokého výkonu. Jedním z jeho hlavních benefitů je to, že vývojáři mohou Vue.js postupně integrovat do stávajících projektů nebo jej použít k vytvoření plně škálovatelné aplikace. Framework nabízí oficiální knihovny pro správu stavu (Vuex) a směrování (Vue Router), což usnadňuje vývoj a údržbu aplikací. Přestože má menší komunitu a ekosystém než React a Angular, jeho dobře strukturovaná dokumentace a aktivní vývojářská komunita z něj činí silného konkurenta na poli moderních JavaScriptových frameworků [17].

Vue.js aplikace začíná vytvořením instance Vue, která obsahuje objekt možností. Tento objekt zahrnuje vlastnost `el`, která definuje kořenový HTML element aplikace, a vlastnost `data`, která obsahuje stav aplikace. Vue.js podporuje datovou vazbu přes interpolaci `property_name` a využívá direktivy jako `v-if` pro podmíněné vykreslování a `v-for` pro iteraci přes pole. Komponenty ve Vue.js jsou definovány jako objekty s vlastností `template` pro HTML strukturu a metodou `methods` pro definici funkcí [4].

### 2.2.2 Porovnání frameworků React, Vue.js a Angular

Porovnání zmíněných front-endových frameworků se zaměřuje na jejich klíčové vlastnosti, způsob použití a vhodnost pro různé typy aplikací. Každý z těchto frameworků nabízí odlišný přístup k vývoji a má své silné a slabé stránky. Tato sekce byla inspirována z [19].

- **Typ a vývojové prostředí** – Angular je plnohodnotný framework, který poskytuje ucelené řešení pro vývoj webových aplikací a je založen na jazyce TypeScript. Oproti tomu React je knihovna zaměřená na tvorbu uživatelských rozhraní, která vyžaduje doplňkové knihovny pro směrování a správu stavu aplikace. Vue.js je progresivní framework, který lze snadno integrovat do existujících projektů a jeho syntaxe kombinuje prvky Angularu i Reactu.
- **Osvojení a přístupnost** – Vue.js je považován za nejjednodušší framework na naučení díky své intuitivní syntaxi a jasně strukturované dokumentaci. React je složitější, ale jeho flexibilita umožňuje snadnou adaptaci. Angular je nejsložitější na naučení, protože využívá TypeScript a komplexní architekturu s mnoha vestavěnými funkcemi, což však přináší výhody v podobě škálovatelnosti.
- **Výkon a efektivita** – React a Vue.js využívají virtuální DOM, což zajišťuje vyšší efektivitu při změnách v uživatelském rozhraní. Angular naopak používá vlastní mechanismus změnové detekce, který může být v některých případech méně efektivní než

virtuální DOM. Vue.js je díky své lehké architektuře obecně rychlejší než Angular, což jej činí vhodným pro vývoj rychlých a interaktivních webových aplikací.

- **Použití v praxi** – Angular je preferovanou volbou pro velké podnikové aplikace, které vyžadují škálovatelnost a dlouhodobou podporu. React je často využíván ve vývoji dynamických webových stránek a mobilních aplikací, přičemž React Native umožňuje snadné přizpůsobení kódu pro mobilní platformy. Vue.js se používá pro tvorbu single-page aplikací a jeho flexibilita jej činí vhodným pro menší projekty a také prototypy.

Na základě uvedených faktorů lze shrnout, že React je často preferovanou volbou pro flexibilní projekty s širokou komunitní podporou. Vue.js je vhodný pro menší a středně velké projekty, kde je klíčová jednoduchost a rychlá implementace. Angular je robustní framework, který nabízí rozsáhlé funkcionality vhodné pro podnikové aplikace s důrazem na škálovatelnost a stabilitu.

Pro svou diplomovou práci jsem se rozhodl použít React, neboť Angular je pro mou práci příliš komplexní a jeho použití by vedlo k nadměrné složitosti. React oproti tomu poskytuje dostatečnou flexibilitu a širokou komunitní podporu, což usnadňuje vývoj a řešení případných problémů. Dalším faktorem je jeho popularita na pracovním trhu, díky které je výhodné se jej naučit. Navíc React umožňuje snadné rozšíření o další knihovny a frameworky, například Vue.js, pokud by bylo potřeba v budoucnu upravit nebo rozšířit funkcionalitu aplikace.

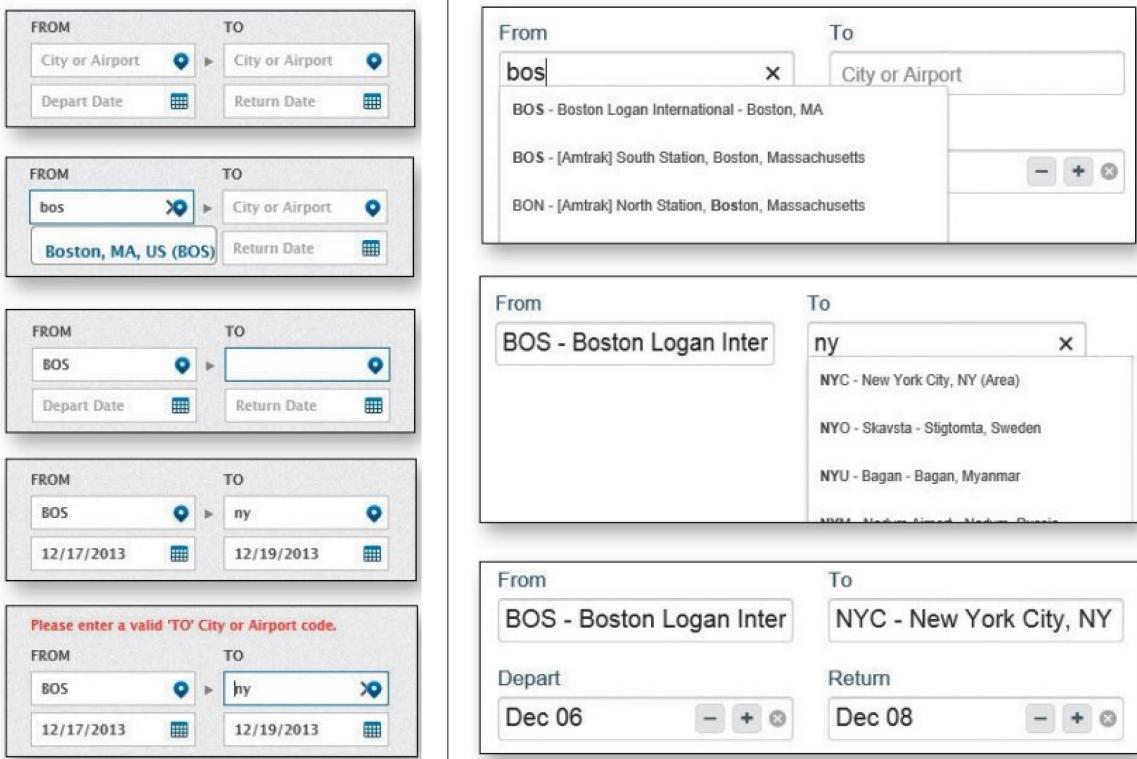
## 2.3 Uživatelská zkušenost

Není-li dále uvedeno jinak, tato podkapitola byla inspirována z [20]. Uživatelská zkušenost, neboli také User experience známé pod zkratkou UX, se zaměřuje na interakci uživatele s produktem nebo službou a zahrnuje jeho pocity, vnímání a spokojenosť během používání. Cílem UX designu je vytvořit intuitivní, efektivní a příjemné prostředí, které odpovídá potřebám uživatelů. Dále jsou popsány základní informace o UX, včetně jeho hlavních principů, metod výzkumu a testování, významu vizuálního designu a praktických doporučení pro zlepšení UX na webu. Principy v této podkapitole lze dobře ilustrovat na srovnání příkladů dobrého a špatného UX designu, jak ilustruje obrázek 2.2.

### Zásady UX designu

UX design se řídí několika klíčovými principy, které pomáhají vytvářet uživatelsky přívětivé rozhraní:

- **Viditelnost systémového stavu** – Uživatel by měl být vždy informován o tom, co se v systému děje.
- **Shoda mezi systémem a reálným světem** – Rozhraní by mělo používat jazyk a koncepty, které uživatelé znají.
- **Kontrola a svoboda uživatele** – Možnost snadného vrácení akce nebo úniku ze situace.
- **Minimalistický a estetický design** – Odstranění nepotřebných prvků pro usnadnění navigace.



Obrázek 2.2: Na obrázku jsou dva příklady UX při rezervaci letenek. Vlevo je špatný příklad, kde po výběru odletu z Bostonu zůstane zobrazen pouze kód „BOS“ a dále systém nerozpozná zadání „ny“ jako New York. Vpravo je správný příklad, kde jsou tyto problémy odstraněny. Obrázky byly převzaty z knihy [13].

## Výzkum uživatelů v UX designu

Výzkum uživatelů je klíčovou součástí UX designu a pomáhá identifikovat potřeby, cíle a chování uživatelů. Existuje několik metod výzkumu:

- **Uživatelské rozhovory** – Přímé rozhovory s uživateli k pochopení jejich potřeb.
- **Kontextuální analýza** – Pozorování uživatelů při práci s produktem.
- **Uživatelské testování** – Testování rozhraní s reálnými uživateli a sběr zpětné vazby.

## Testování použitelnosti

Testování použitelnosti (známé také jako usability testing) je klíčovou metodou UX designu. Zaměřuje se na to, jak snadno a efektivně uživatelé dokončují úkoly v aplikaci nebo na webu. Existují dva hlavní přístupy, které se liší nejen způsobem sběru dat, ale také vhodností použití v různých fázích vývoje:

- **Kvantitativní testování** – Měří přesně definované metriky, jako je doba potřebná k dokončení úkolu, počet chyb nebo úspěšnost dokončení. Tento přístup je často využíván při A/B testování, kdy se porovnávají různé varianty rozhraní a rozhoduje se na základě statisticky podložených výsledků [21].

- **Kvalitativní testování** – Zaměřuje se na hlubší porozumění chování a motivaci uživatelů. Pomocí pozorování, rozhovorů nebo metody hlasitého uvažování odhaluje, proč uživatelé chybují nebo co jim připadá matoucí. Hodí se zejména v počátečních fázích vývoje nebo při testování zcela nové aplikace, kdy je důležité pochopit celkové vnímání a intuitivnost uživatelského rozhraní.

## Význam vizuálního designu v UX

Vizuální design hraje klíčovou roli v UX, protože ovlivňuje první dojem uživatele a jeho důvěru v produkt. Mezi hlavní zásady vizuálního designu patří:

- **Jednotnost a rozmanitost** – Zajištění vizuální konzistence s dostatečnou variabilitou pro udržení pozornosti.
- **Hierarchie a dominance** – Zvýraznění nejdůležitějších prvků na stránce.
- **Proporce a rovnováha** – Harmonické rozložení prvků pro usnadnění navigace.

## Praktické aspekty zlepšení UX na webu

Zlepšení UX lze dosáhnout dodržováním následujících zásad [10]:

- **Přehledný a čistý design** – Zajištění jednoduchého a intuitivního designu, který umožňuje snadnou orientaci.
- **Optimalizace pro mobilní zařízení** – Web by měl být responzivní a dobře použitelný na různých platformách.
- **Rychlosť načítání stránek** – Minimalizace doby načítání obsahu prostřednictvím optimalizace obrázků a skriptů.
- **Kvalitní a relevantní obsah** – Poskytování hodnotného obsahu, který odpovídá potřebám uživatelů.
- **Jednoduchá navigace** – Logická struktura webu a srozumitelné navigační prvky.
- **Konzistentní vizuální styl** – Dodržování jednotného vizuálního stylu pro zvýšení důvěryhodnosti stránek.
- **Zpětná vazba od uživatelů** – Analýza zpětné vazby uživatelů a její využití pro úpravy designu.
- **Testování použitelnosti** – Pravidelné testování s reálnými uživateli pro identifikaci a odstranění problémů.
- **Optimalizace pro vyhledávače (SEO<sup>3</sup>)** – Použití osvědčených metod SEO pro lepší dohledatelnost webu.
- **Bezpečnost a důvěryhodnost** – Ochrana dat uživatelů a zajištění bezpečnosti webových stránek.

Dodržování těchto principů přispívá ke zlepšení UX, což vede k vyšší spokojenosti návštěvníků a efektivnějšímu využívání webových služeb.

---

<sup>3</sup>Search Engine Optimization, neboli optimalizace pro vyhledávače je soubor technik zaměřených na zlepšení viditelnosti webových stránek ve výsledcích vyhledávání.

## 2.4 Komunikace v reálném čase

Komunikace v reálném čase je klíčovou součástí moderních webových aplikací, které vyžadují okamžitou výměnu dat mezi serverem a klientem. Typickými příklady jsou chatovací aplikace, online hry, finanční trhy, kolaborativní nástroje nebo právě kvízové aplikace. V této podkapitole jsou popsány různé techniky pro dosažení efektivní real-time komunikace mezi serverem a klientem, včetně jejich výhod, nevýhod a vhodnosti pro různé typy aplikací.

### 2.4.1 Metody komunikace v reálném čase na webu

Existuje několik metod, jak implementovat komunikaci v reálném čase mezi serverem a klientem. Mezi nejběžnější techniky patří short polling, long polling, Server-Sent Events (dále označované zkratkou SSE) a WebSockets. Každá z těchto metod má své specifické využití v závislosti na požadavcích aplikace, například na latenci, škálovatelnosti nebo podpoře různých protokolů. Tato podsekce obsahuje jednotlivé techniky, jejich principy fungování a porovná jejich výhody a nevýhody. Obrázek 2.3 přehledně znázorňuje a porovnává jednotlivé techniky komunikace v reálném čase.

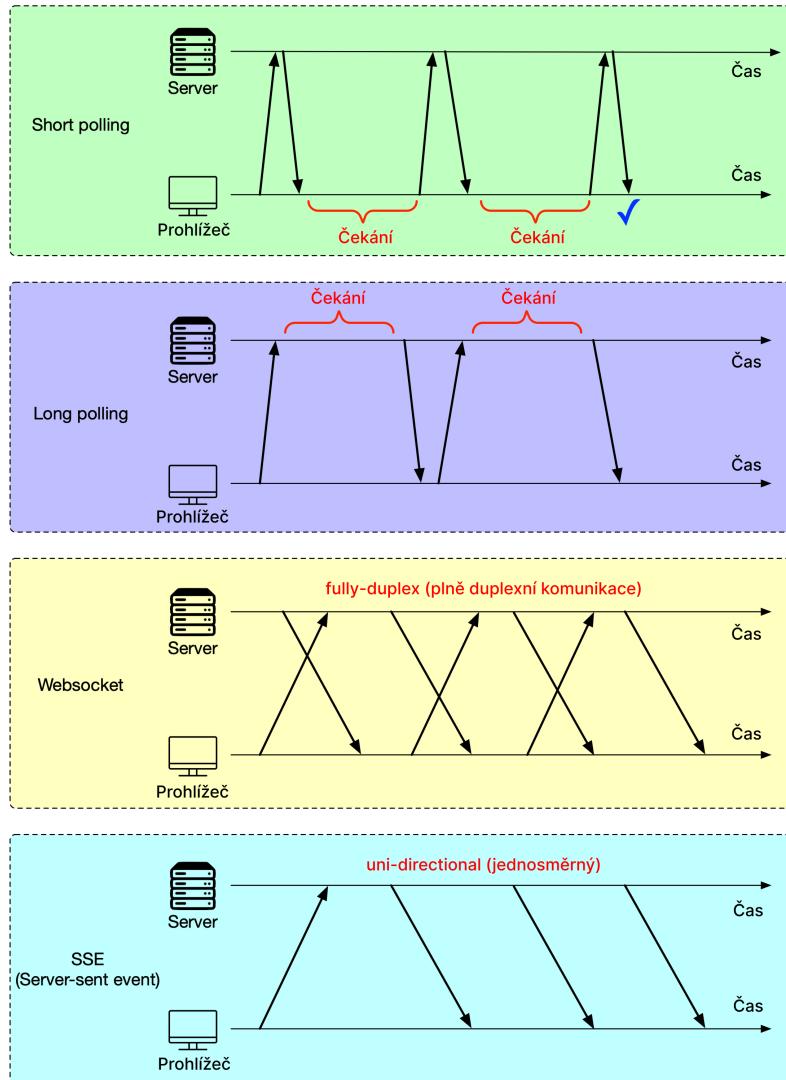
#### Short/Long polling

Short polling představuje jednoduchou techniku, kdy klient v pravidelných intervalech odešlá HTTP požadavky na server, aby zjistil, zda jsou k dispozici nová data. Tento přístup je snadno implementovatelný a nevyžaduje žádné speciální konfigurace serveru, což jej činí široce kompatibilním. Hlavní nevýhodou je však vysoký počet opakovaných požadavků, které mohou vést k nadmerné zátěži serveru a zbytečnému síťovému provozu. Navíc jsou aktualizace dostupné pouze v okamžiku požadavku, což může způsobit nežádoucí latenci v doručování informací. Příkladem použití je aplikace zobrazující hodnoty akcií, kde se ceny aktualizují každých několik sekund, ale mezi jednotlivými dotazy může docházet ke zpoždění [16].

Long polling se od krátkého liší tím, že server po přijetí požadavku připojení neukončí okamžitě. Místo toho ponechá spojení otevřené, dokud není dostupná nová informace, kterou následně odešle klientovi. Po doručení odpovědi se spojení uzavře a klient ihned odešle nový požadavek, čímž vzniká téměř nepřetržité spojení. Tento přístup omezuje zbytečný síťový provoz a zajistuje aktualizace téměř v reálném čase. Vyžaduje však vyšší nároky na serverové zdroje, protože udržuje mnoho otevřených spojení současně. Long polling se často používá v chatovacích aplikacích, kde je klíčové, aby zprávy byly doručeny okamžitě po jejich odeslání [16].

#### Server-sent events

SSE umožňují serveru jednosměrně odesílat data klientovi prostřednictvím udržovaného HTTP připojení. Klient iniciuje požadavek, server drží spojení otevřené a posílá data podle potřeby. SSE jsou efektivnější než polling, protože eliminují zbytečné požadavky, ale jsou jednodušší než WebSockets, které vyžadují plně duplexní komunikaci. Jsou vhodné pro aplikace jako živé zpravodajství, burzovní kurzy nebo monitoring, kde je potřeba pravidelných aktualizací, ale klient nepotřebuje zasílat odpovědi serveru. Mezi hlavní výhody patří jednoduchá implementace, automatické obnovení spojení při výpadku a nižší zatížení serveru ve srovnání s pollingem. Nevýhodou je omezení na jednosměrný tok dat a závislost na HTTP protokolu [9].



Obrázek 2.3: Na obrázku lze vidět srovnání jednotlivých metod komunikace v reálném čase. Při short pollingu prohlížeč pravidelně opakuje požadavky na server. Long polling umožňuje serveru odpovědět až ve chvíli, kdy jsou dostupná nová data. WebSocket a SSE udržují otevřené spojení, přičemž SSE podporuje pouze jednosměrnou komunikaci, zatímco WebSocket umožňuje obousměrnou výměnu dat. Obrázek byl převzat a přeložen z [22].

## WebSocket

V této podsekci se vychází z informací získaných z [6]. WebSocket je komunikační protokol, který umožňuje plně duplexní, obousměrnou komunikaci mezi klientem a serverem prostřednictvím jednoho dlouhodobě otevřeného TCP spojení. Spojení začíná s HTTP handshake, při kterém klient odešle požadavek s hlavičkou **Upgrade: websocket** a server odpoví stavovým kódem **101 Switching Protocols**, čímž přepne komunikaci na WebSocket protokol. WebSockets podporují textová i binární data a nacházejí uplatnění v aplikacích, jako jsou chatovací systémy, online hry nebo IoT zařízení.

Hlavní výhodou WebSocketů je jejich nízká latence a vysoká efektivita, protože eliminují opakování HTTP požadavky typické pro polling. Díky tomu jsou vhodné pro aplikace, kde

je nutná okamžitá odezva a vysoká frekvence přenosu dat. Nevýhodou může být vyšší náročnost na správu připojení a škálovatelnost, protože server musí udržovat stav každého klienta. WebSockets mohou také čelit problémům s firewally a proxy servery, které nemusí tento protokol správně podporovat. Pro jednodušší implementaci a správu spojení lze využít knihovny, jako je `Socket.IO`, která přidává vrstvu nad WebSockets a zjednodušuje práci s nimi. Více o této knihovně a její integraci bude popsáno v následující kapitole.

Každá z popsaných technologií má své výhody a omezení v závislosti na požadavcích aplikace. Polling může být jednoduchý na implementaci, ale zatežuje server. SSE jsou efektivnější, avšak omezené na jednosměrný tok dat. WebSocket umožňují obousměrnou komunikaci s nízkou latencí, což je klíčové pro aplikace s okamžitou výměnou dat. Pro potřeby mé kvízové aplikace, kde je nutná rychlá synchronizace mezi hráči a serverem, jsem zvolil WebSockets. Umožňují efektivní přenos odpovědí a výsledků v reálném čase, což zajišťuje plynulý průběh kvízu a lepší uživatelský zážitek.

#### 2.4.2 Flask-SocketIO

Tato podsekce je převzata z oficiálního Flask-SocketIO manuálu [8]. Flask-SocketIO je rozšíření pro framework Flask, které umožňuje snadnou implementaci WebSocket komunikace v Python aplikacích. Poskytuje podporu pro obousměrnou komunikaci mezi serverem a klientem, přičemž umožňuje práci s událostmi a efektivní správu připojení. Kromě WebSocketu podporuje také fallback mechanismy, jako je polling, čímž zajišťuje kompatibilitu i v prostředích, kde WebSockets nejsou dostupné. Flask-SocketIO je vhodnou volbou pro aplikace, které vyžadují real-time interakci, například chaty nebo kvízové aplikace, jako je ta popisovaná v této práci. Následující seznam shrnuje nejdůležitější funkce této knihovny:

- **emit()** – Slouží k odesílání událostí s daty klientům. Lze specifikovat konkrétní klienty nebo vysílat zprávy globálně všem připojeným.
- **send()** – Používá se k odesílání jednoduchých textových zpráv, aniž by bylo nutné definovat vlastní typ události.
- **on()** – Registruje funkci jako obsluhu konkrétní události, která se spustí po přijetí odpovídající zprávy od klienta.
- **event()** – Alternativní způsob registrace obslužných funkcí, který umožňuje jednodušší a přehlednější deklaraci handlerů.
- **join\_room() / leave\_room()** – Umožňuje klientům připojit se do místnosti nebo ji opustit. Místnosti jsou užitečné například pro skupinovou komunikaci, kde je potřeba zasílat zprávy jen vybrané skupině uživatelů.
- **rooms()** – Vrací seznam místností, ke kterým je klient aktuálně připojen. To umožňuje přehlednou správu skupinového chatu nebo jiných kolaborativních funkcí.
- **connect() / disconnect()** – Slouží k detekci připojení a odpojení klientů. Mohou být využity například pro logování uživatelů nebo uvolnění serverových zdrojů po jejich odpojení.

Pro mou kvízovou aplikaci jsem se rozhodl využít Flask-SocketIO pro implementaci real-time komunikace, jelikož poskytuje stabilní a efektivní způsob přenosu otázek, odpovědí a výsledků mezi hráči a serverem. Tato technologie zajišťuje rychlou synchronizaci a plynulý průběh kvízu, čímž přispívá ke zlepšení uživatelského zážitku.

# Kapitola 3

## Návrh řešení

V této kapitole se zabývám průzkumem podobných řešení, návrhem grafického uživatelského rozhraní (dále jen GUI – Graphics User Interface) a návrhem jednotlivých kvízových typů. Má diplomová práce je zaměřena na kvízové hry, a proto jsem analyzoval různé kvízové aplikace, ale také televizní pořady, jako například *Máme rádi Česko*, který byl hlavní inspirací pro návrh kvízových disciplín. Následně jsem vytvořil návrh GUI v programu Figma a připravil dotazník k ověření návrhu pro získání zpětné vazby od uživatelů na kvízové typy. Nakonec jsem navrhl strukturu databáze, API a základní WebSockets.

### 3.1 Průzkum podobných řešení

Existuje několik řešení, která mě inspirovala při návrhu mé diplomové práce. Jedním z klíčových zdrojů inspirace byla platforma Kahoot<sup>1</sup>, která svou funkcionalitou nabízí podobné možnosti, jaké jsou plánovány v mé práci. Další inspirací byla kreslící platforma skribl.io<sup>2</sup>, která vyniká snadnou dostupností a možností okamžité interakce mezi hráči, přestože její design působí zastarale. Zajímavé herní režimy, jako je „Přežítí“, byly převzaty z mobilní aplikace *QuizzLand*, kde hráč po špatné odpovědi končí, což poskytuje zajímavou alternativu ke klasickému bodovému systému.

Dále mě ovlivnila televizní soutěž *Máme rádi Česko*, která mi poskytla základ pro návrh konkrétních kvízových typů, jako jsou disciplíny „Krabice“ nebo „Přesmyčka a slepá mapa“, které byly upraveny pro použití v prostředí mobilních zařízení. V rámci průzkumu byly analyzovány i bakalářské práce zabývající se podobným tématem, konkrétně [1], [2] a [14]. Tyto práce byly zaměřeny na jednodušší kvízy typu ABCD, které mi poskytly základní představu o požadavcích na implementaci a fungování aplikace.

#### 3.1.1 Máme rádi Česko

Pořad *Máme rádi Česko* je vědomostní televizní show vysílaná na TV Prima, ve které se dva týmy celebrit pod vedením stálých kapitánů utkávají v disciplínách zaměřených na znalosti o Česku. Česká verze měla premiéru současně se slovenskou variantou *Milujem Slovensko* v roce 2013. Soutěž pokrývá široké spektrum témat, jako jsou historie, kultura, sport, geografie, hudba, film i všeobecný přehled. Složení týmů se s každým dílem mění, ale kapitáni zůstávají neměnní během celé řady. Soutěž má dynamický a zábavný formát, kdy i díky „kolu štěstí“ může dojít k nečekaným zvratům v průběhu hry.

---

<sup>1</sup><https://kahoot.com/>

<sup>2</sup><https://skribbl.io/>

Cílem hry je nasbírat co nejvíce bodů prostřednictvím jednotlivých disciplín, které kombinují znalosti, dovednosti a rychlé reakce. Na konci soutěže vítězný tým získává hlavní cenu, obvykle typicky český pokrm či předmět, později také deskovou hru inspirovanou pořadem.

## Disciplíny

- **Filmový archiv:** Soutěžící odpovídají na otázku k filmové ukázce. Každý tým vysílá jednoho zástupce, který se snaží být rychlejší než soupeř a správně odpovědět na otázku k dané ukázce. Disciplína prověruje znalosti české kinematografie i pohotovost.
- **Krabice:** Soutěžící sedí u stolu a na střídačku odpovídají na otázky. Správná odpověď umožňuje předat krabici členovi protějšího týmu. Body získává pouze tým, u nějž krabice nevybuchla. Disciplína prověruje kombinaci pohotovosti, vědomostí a zvládání časového tlaku.
- **Kartičky:** Kapitán má za úkol během dvou minut popsat co nejvíce pojmu (osobnosti, místa, pokrmy, apod.), aniž by použil samotné slovo. Tým musí uhodnout co nejvíce pojmu. Tato disciplína kombinuje schopnost popisu a týmovou spolupráci.
- **Kreslení:** Jeden člen týmu kreslí na průhlednou tabuli pojmy, které má na kartičce, zatímco ostatní hádají. Za správně uhodnuté pojmy získává tým body. Rychlosť kreslení a schopnost pochopení obrázků jsou klíčové.
- **Přesmyčka a slepá mapa:** Soutěžící musí rozluštit slovo skryté v přesmyčce. Tato disciplína prověruje slovní zásobu a rychlé myšlení. Vítězný tým dostane přednost v disciplíně **Slepá mapa**, kde týmy umisťují zeměpisné pojmy na mapu Česka a označují města. Správné odpovědi přinášejí body, přičemž za poslední úkol je nejvíce bodů.
- **Uhodni písničku:** Soutěžící musí poznat skladbu podle hudební ukázky, obrázku nebo textu a případně uvést i interpreta. Tato disciplína testuje znalosti populární i lidové hudby.
- **Pomíchané osobnosti:** Na fotografiu jsou kombinované obličeje dvou českých osobností. Tým musí identifikovat obě osobnosti, což vyžaduje znalost české populární kultury a pozorovací schopnosti.
- **Tichá pošta:** Soutěžící si mezi sebou šeptem předávají text, který poslední člen týmu (kapitán) převypráví. Správnost vyprávění a počet zachovaných klíčových slov rozhodují o úspěchu.
- **Uhodni číslo s kolem štěstí:** Soutěžící v jednom týmu se snaží zadat odpověď co nejblíže ke správnému číslu. Druhý tým následně odhadne, zda je správná odpověď větší, nebo menší. Vítězný tým kola zatočí kolem štěstí, aby získal body pro svůj tým.
- **Kasa:** Soutěžící si musí zapamatovat co nejvíce předmětů, které moderátor pokládá na pult. Kdo udělá chybu nebo si nevpomene, vypadává. Tým, který zůstane poslední, získává body.

Při průzkumu kvízových soutěží se pořad *Máme rádi Česko* ukázal jako velmi inspirativní pro mou práci. Soutěž obsahuje disciplíny jako „Krabice“, „Kreslení“, „Přesmyčka a slepá mapa“, „Uhodni číslo s kolem štěstí“ či „Kasa“, které mě inspirovaly při návrhu kvízových typů do mé aplikace. Tyto disciplíny budou přizpůsobeny pro využití na mobilních

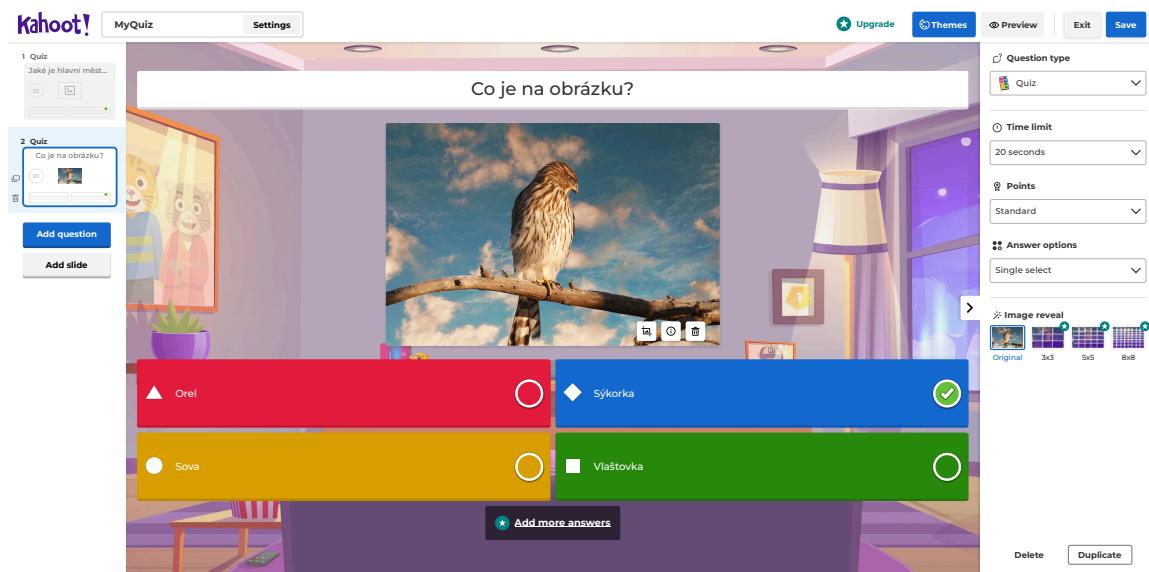
zařízeních. Zajímavým prvkem soutěže je také „Kolo štěstí“, které umožňuje dramatické zvraty ve skóre a přispívá k napínávánímu průběhu hry. Cílem mé práce je implementovat podobné prvky do aplikace, kde hráči budou soutěžit prostřednictvím mobilních zařízení, zatímco hlavní obrazovka zobrazí průběh hry a zachová její interaktivitu a soutěživost, která je charakteristická i pro tento pořad.

### 3.1.2 Kahoot

Kahoot je multiplatformní interaktivní platforma určená primárně pro školy, která umožňuje vytváření a hraní vzdělávacích kvízů. Je dostupná na široké škále zařízení, funguje na moderních webových prohlížečích podporujících HTML5 a JavaScript. Kromě toho je k dispozici i mobilní aplikace, která je dostupná pro uživatele operačních systémů Android a iOS. Princip hry spočívá v tom, že otázky a odpovědi jsou zobrazeny na hlavní obrazovce, například na projektoru, zatímco hráči odpovídají pomocí svých zařízení, například mobilních telefonů, tabletů nebo notebooků.

Registrace na platformě Kahoot není nutná pro samotné hraní kvízů, což umožňuje rychlé zapojení hráčů bez zbytečných administrativních kroků. Pokud však uživatel plánuje vytvářet vlastní kvízy, je potřeba se zaregistrovat. Registrace je bezplatná a vyžaduje vytvoření účtu typu učitel nebo student. Pro uživatele mladší 16 let jsou u studentských účtů stanovena určitá omezení, například nemají přístup k veřejné databázi kvízů a jejich vlastní kvízy nemohou být označeny jako veřejné.

Kahoot nabízí široké spektrum funkcí, z nichž mnohé jsou dostupné pouze v placené verzi. Při tvorbě kvízu lze v bezplatné verzi vytvářet pouze otázky typu ABCD a Pravda/Lež. Další typy otázek, jako například otevřené odpovědi, odhady čísel pomocí posuvníku, výběr správné odpovědi na obrázku, puzzle (seřazování odpovědí ve správném pořadí) nebo otázky s audiem, jsou dostupné pouze uživatelům s prémiovým účtem. Kvízy se tvoří ve stylu prezentací, podobně jako v PowerPointu. Vytváření kvízů lze vidět na obrázku 3.1.



Obrázek 3.1: Na obrázku je zobrazen proces vytváření kvízu na platformě Kahoot, kde uživatel může přidávat otázky a odpovědi, obrázky, nastavovat parametry, jako je časový limit na odpověď, bodování nebo dostupnost kvízu, a volit další detailly prezentace.

U každé otázky lze přidat obrázky, které mohou být odkrývány postupně, a také připojit obrázky k jednotlivým odpovědím. Uživatelé mohou nastavovat různé parametry, jako například časový limit na odpověď, bodování (standardní, dvojnásobné nebo bez bodů) a zda otázka umožňuje jednu správnou odpověď, nebo více. Dále je možné nastavit název a popis kvízu a zvolit jeho veřejnou či soukromou dostupnost. Na jednotlivé slidy lze také přidat pozadí.

Jelikož Kahoot neslouží pouze jako kvízová aplikace, je možné zde přidávat také slidy a upravovat je podobně jako v PowerPointu. Tato funkce je však placená. Pro uživatele s prémiovým účtem je navíc k dispozici generování otázek pomocí umělé inteligence. K dispozici je také databáze otázek vytvořených ostatními uživateli, které jsou seřazeny podle počtu zahrání.

Po výběru kvízu ke hraní je možné mimo jiné zvolit mezi klasickou hrou, kde každý hraje sám za sebe, nebo týmovou hrou. V týmové hře je možné hrát buď s jedním zařízením na tým, nebo s vlastním zařízením pro každého hráče. V bezplatné verzi lze hrát maximálně ve dvou týmech.

Při spuštění kvízu se zobrazí QR kód a webová adresa, na které lze zadat zobrazený PIN. Na hlavní obrazovce se poté zobrazují připojení hráči. Ti se připojují zadáním PINu, případně naskenováním QR kódu, a zvolí si přezdívku. Po připojení se jejich jména objeví na hlavní obrazovce a je jim přiřazena postavička, kterou si mohou upravit. Po připojení všech hráčů je možné spustit kvíz, který se přepne na celou obrazovku.

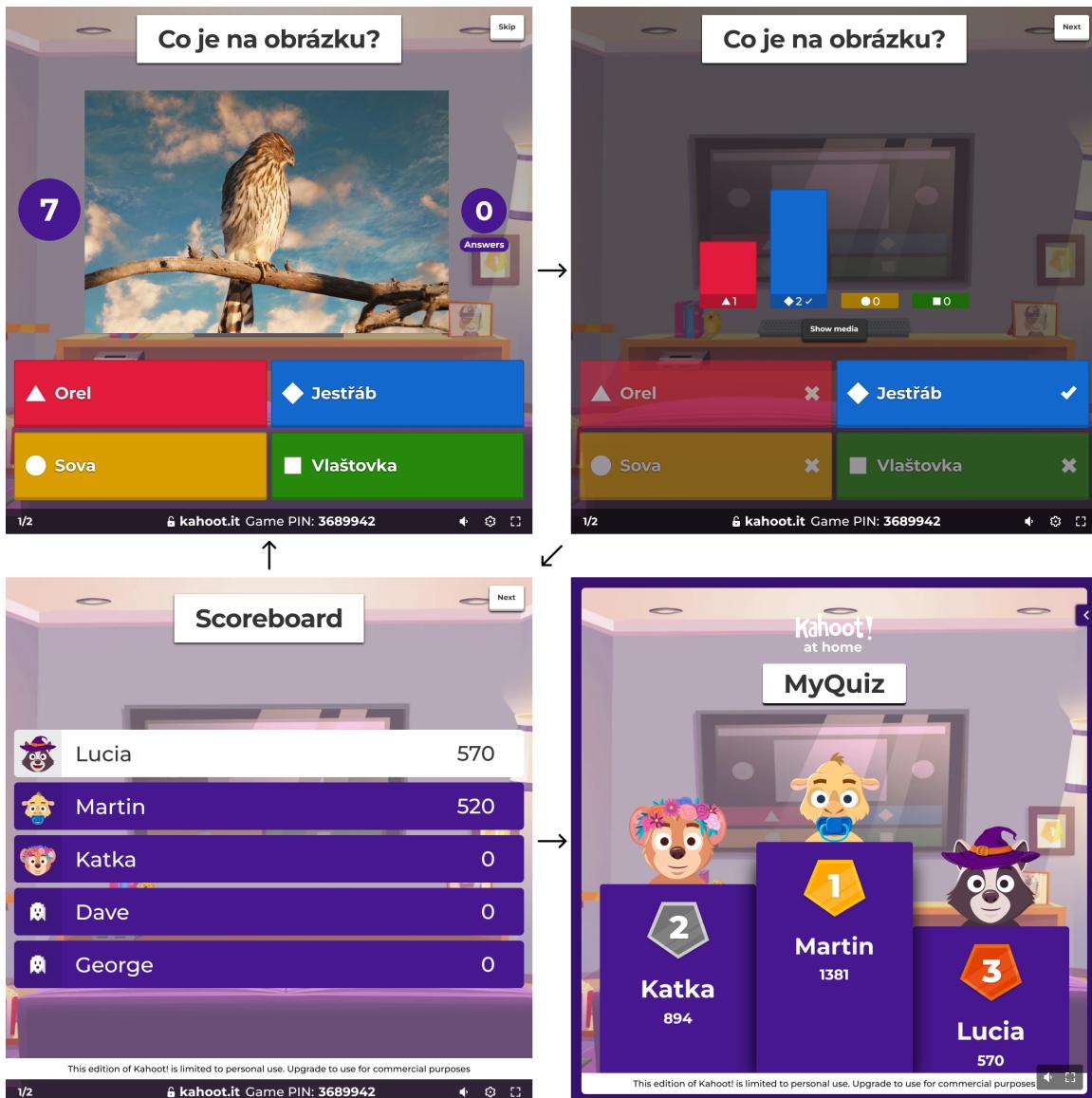
## Klasická hra

V klasickém režimu se nejprve na hlavní obrazovce zobrazí otázka na několik vteřin, poté se objeví odpovědi na obrazovce a barevná tlačítka na zařízeních hráčů. Tlačítka obsahují kromě barvy i odlišné tvary, což zlepšuje uživatelskou přístupnost například pro barvoslepé hráče. Na obrazovce je také zobrazen časovač, počet odpovědí od hráčů, stránkování (počet zodpovězených a zbývajících otázek), možnosti nastavení hlasitosti, přeskročení kola nebo PIN pro pozdější připojení. Lze přepínat mezi celoobrazovkovým režimem a zobrazením v prohlížeči.

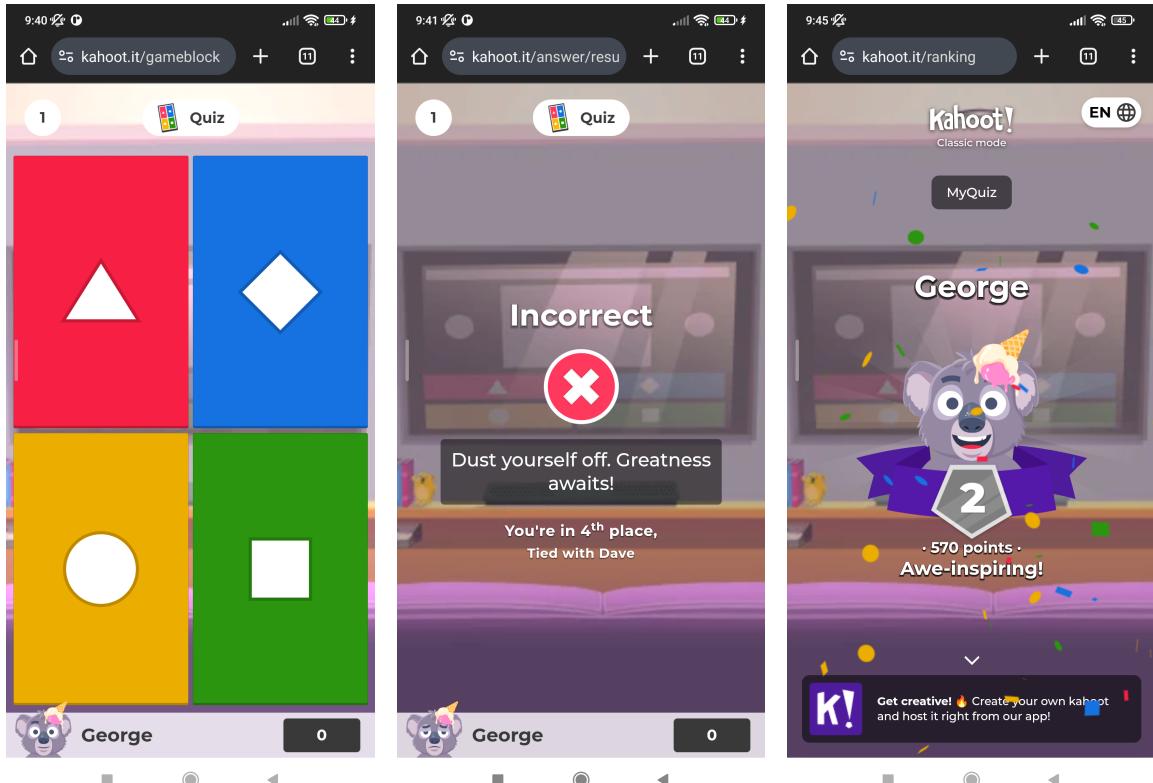
Po uplynutí času nebo po odpovědi všech hráčů se zobrazí správná odpověď a počet odpovědí na jednotlivé možnosti. Následně je zobrazen průběžný žebříček hráčů podle počtu bodů. Hráči na svém zařízení vidí, zda zodpověděli správně, či nikoli, a počet bodů, které dostali za dané kolo. Po zodpovězení všech otázek se na hlavní obrazovce zobrazí finální umístění, zatímco hráči na svých zařízeních vidí pouze vlastní umístění a získané body. Na závěr je možné požádat hráče o zpětnou vazbu, kterou mohou zadat na svých zařízeních. Poté lze zvolit další kvíz se stejnými hráči nebo hru ukončit. Průběh klasické hry na hlavní obrazovce lze vidět na obrázku 3.2. Na obrázcích 3.3a, 3.3b a 3.3c lze pak vidět průběh klasické hry na mobilních zařízeních.

## Týmová hra

Týmový režim probíhá podobně jako klasická hra, avšak hráči jsou rozděleni do týmů náhodně, přičemž toto rozdělení nelze měnit. Po zobrazení otázky a odpovědí na hlavní obrazovce mají hráči 5 vteřin na poradu, než se na jejich zařízeních zobrazí tlačítka s odpovědmi. Hráči odpovídají jednotlivě, přičemž týmy získávají body podle počtu správných odpovědí od svých členů.



Obrázek 3.2: Na obrázku lze vidět průběh klasické hry platformy Kahoot. Nejprve se zobrazí otázka s možnostmi odpovědí. Po uplynutí časového limitu, nebo když všichni hráči odpoví, se zobrazí správná odpověď a počet hráčů, kteří zvolili jednotlivé možnosti. Následně se zobrazí průběžný žebříček hráčů s aktualizovanými body. Tento cyklus se opakuje s každou další otázkou, až do vyčerpání všech otázek. Jakmile dojdou otázky, tak se zobrazí finální umístění hráčů.



(a) Odpovědi na ABCD kvíz      (b) Negativní výsledek odpovědi      (c) Finální umístění hráče

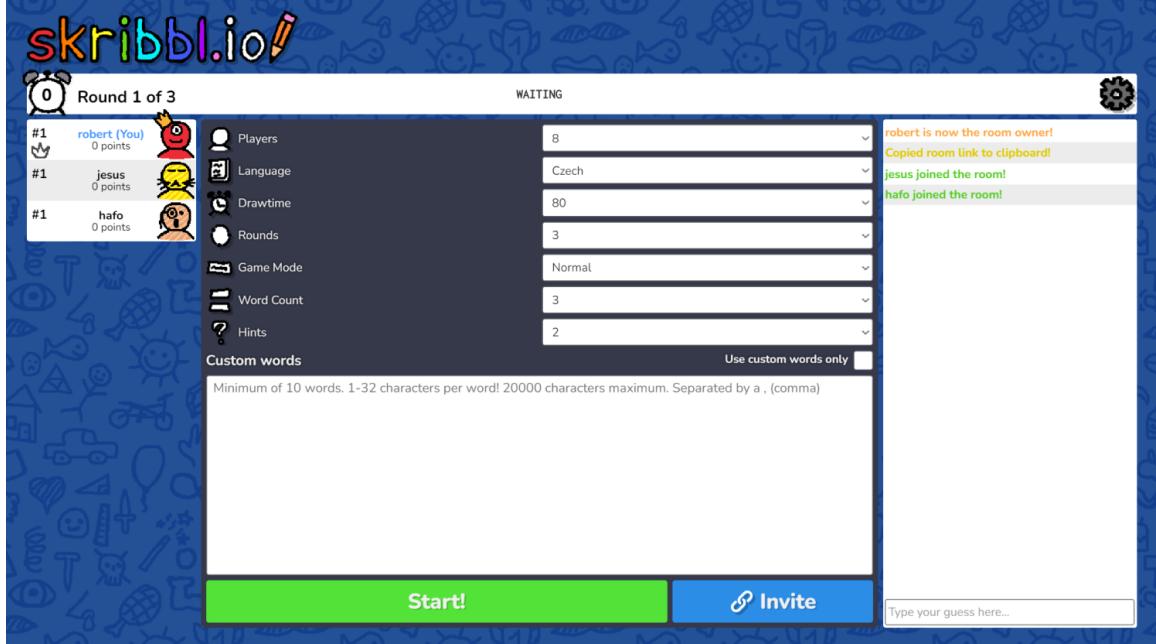
Obrázek 3.3: První obrázek zobrazuje čtyři možné odpovědi na otázku zobrazenou na hlavní obrazovce. Druhý obrázek zobrazuje výsledek zadané odpovědi po ukončení daného kola. Také zde hráč vidí své aktuální umístění. Na posledním obrázku je zobrazeno finální umístění daného hráče a počet získaných bodů.

Z průzkumu vyplývá, že Kahoot je jednou z nejlepších aplikací tohoto druhu na trhu a je pro mě významnou inspirací při vývoji vlastní práce. Mezi její největší přednosti patří jednoduché ovládání a široká podpora různých zařízení, což ji činí přístupnou pro širokou škálu uživatelů. Z pohledu UX je platforma zábavná a poutavá, a to díky interaktivnímu rozhraní, animacím a zvukovým efektům. Nevýhodou je, že některé pokročilejší funkce, například přidávání otázek s audiem nebo puzzle typ otázek, jsou dostupné pouze v placené verzi, což může část uživatelů odradit. Z hlediska UI je design aplikace jednoduchý a moderní.

### 3.1.3 Skribbl

Skribbl.io je interaktivní webová platforma zaměřená na kreslení a hádání slov. Po první návštěvě si uživatel může zvolit buď rychlou hru, nebo hostování vlastní hry. Rychlá hra připojí uživatele k již existující veřejné hře, která může být už rozehraná. Při hostování vlastní hry si uživatel nejprve zvolí přezdívku a upraví vzhled své postavičky. Následně se zobrazí možnosti nastavení hry, jak je znázorněno na obrázku 3.4. Hostující hráč zde může upravit maximální počet hráčů, jazyk slov, časový limit na kreslení a hádání, počet kol (kolikrát každý hráč bude kreslit), herní režim, maximální počet slov pro odpověď a počet nápověd (počet písmen, která se postupně zobrazí). Dále má možnost zadat vlastní slova,

která mohou být přidána ke hře nebo použita výhradně. Pozvánku pro ostatní hráče lze vytvořit pomocí tlačítka „Invite“, které zkopíruje odkaz ke sdílení. Připojující se hráči si zadají přezdívku, vyberou postavičku a následně se mohou připojit do hry. Hru zahajuje hostující hráč kliknutím na tlačítko „Start!“.

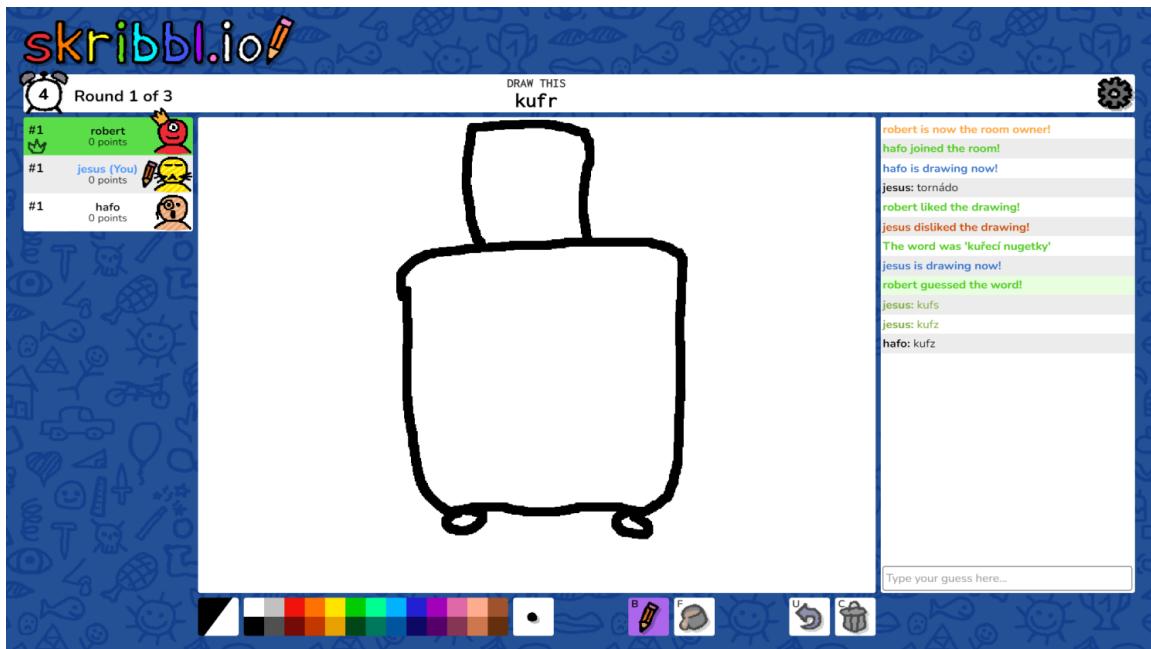


Obrázek 3.4: Na obrázku je vidět první nastavení hry, které hostující hráč vyplní nebo ponechá ve výchozím stavu. Poté rozešle pozvánku hráčům, s kterými chce hrát a jakmile jsou všichni ve hře, spustí hru.

V levé části herního rozhraní jsou zobrazeni všichni hráči, přičemž hostující hráč je označen ikonou korunky. Horní lišta obsahuje časovač aktuálního kola, informaci o pořadí kola a možnost otevřít nastavení. V pravé části obrazovky se nachází chatovací okno, kam hráči zadávají odpovědi. Chat zároveň zobrazuje důležité informace, například připojení nových hráčů, udělení kladného nebo záporného hodnocení obrázku, a také upozornění, když je hráč blízko správné odpovědi, například pokud udělal překlep nebo použil jiné číslo podstatného jména.

Hra probíhá tak, že hráči se střídají v kreslení. Jakmile všichni odmalují, začne nové kolo. Před kreslením si hráč vybírá z trojice slov nebo sousloví, která může kreslit. Tato možnost volby je praktická, protože hráč nemusí znát všechna slova nebo nemusí vědět, jak je nakreslit. Na obrázku 3.5 je ukázáno, jak proces kreslení vypadá. Hráč vidí nahoře slovo, které má kreslit, a pod ním má k dispozici plátno. Pod plátnem jsou nástroje pro kreslení, včetně volby barvy, tloušťky čáry, může si vybírat mezi tužkou a kbelíkem pro vyplňování uzavřené plochy a nakonec zde má tlačítka pro krok zpět nebo vymazání celého plátna.

Hráči získávají body podle toho, jak rychle zadají správnou odpověď. První hráč, který odpoví správně, získává nejvíce bodů, přičemž ostatní získávají postupně méně bodů. Je nutné podotknout, že jakmile někdo odpoví správně, tak se časovač razantně zkrátí, čímž se omezuje čas ostatním hráčům na správnou odpověď. Kreslící hráč také získává body, a to na základě počtu hráčů, kteří jeho kresbu správně uhodli. Tento mechanismus motivuje hráče k tomu, aby se snažili kreslit co nejlépe a nekazili záměrně hru.



Obrázek 3.5: Na obrázku lze vidět herní obrazovku hráče, který zrova kreslí. V horní části je viditelné kreslené slovo, nebo sousloví, zatímco ve spodní části se nachází nástroje pro kreslení a úpravu.

Průzkum ukázal, že největší výhodou platformy skribbl.io je její okamžitá použitelnost bez nutnosti registrace. Z hlediska UX aplikace nabízí jasnou a přehlednou strukturu, ale některé její části, jako například možnosti nastavení hry, by mohly být lépe vysvětleny, nebo zjednodušeny. Z pohledu UI působí rozhraní místy zastarale, což však může být vyváženo jeho funkčností. Mezi pozitivní aspekty bych také zařadil fakt, že pokud nějaký hráč nekreslí, nebo nedodržuje pravidla hry, tak ho ostatní hráči mohou vyloučit ze hry prostřednictvím hlasování, což se hodí převážně ve veřejné hře.

## 3.2 Návrh kvízů

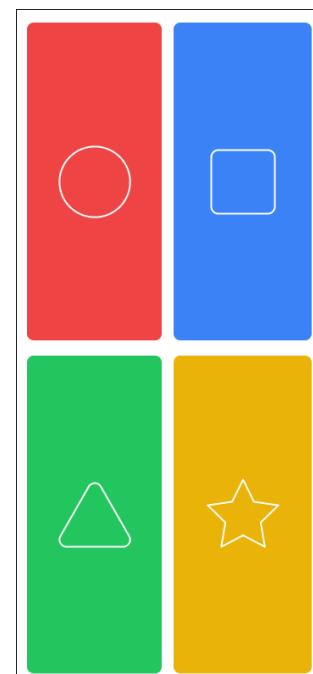
Při návrhu jednotlivých kvízových typů jsem čerpal inspiraci z aplikací uvedených v podkapitole 3.1. Po vytvoření popisných návrhů 13 různých typů kvízů jsem provedl analýzu jejich proveditelnosti a vybral sedm z nich, které nejlépe odpovídají zamýšlenému formátu hlavní obrazovky a mobilních zařízení pro odpovídání. V této sekci jsou představeny všechny navržené kvízové typy. U vybraných z nich jsou také zahrnutý návrhy designů vytvořené v aplikaci Figma. Každý kvízový typ dále obsahuje popis průběhu jednoho kola pro oba režimy, tedy týmový a režim všichni proti všem, a nakonec předběžný systém bodování. Přesnější popis průběhu kvízů bude popsán až dál v podkapitole 3.4, neboť bude téměř stejný pro všechny typy kvízů. Hlavní obrazovka většinou obsahuje časovač a čítač odpovědí, což umožňuje všem hráčům sledovat průběh kola. Každý druh kvízu obsahuje na hlavní obrazovce navíc dvě tlačítka pro přeskočení otázky a ukončení kvízu.

## ABCD kvíz

ABCD kvíz je klasický formát otázek s možnostmi výběru odpovědí, který zahrnuje i možnost výskytu otázky typu **Pravda/Lež**. Jeho princip je velmi podobný formátu používanému na platformě Kahoot, která je blíže popsána v sekci [3.1.2](#). Na hlavní obrazovce je zobrazena otázka spolu s odpověďmi, zatímco hráči vybírají svou odpověď na mobilních zařízeních, jak je vidět na obrázcích [3.6a](#) a [3.6b](#). Kromě barev jsou možnosti odpovědí označeny také tvary, díky čemuž je tento formát přívětivý i pro barvoslepé hráče.



(a) Hlavní obrazovka ABCD kvízu



(b) Mobilní obrazovka

Obrázek 3.6: Na prvním obrázku je ukázána hlavní obrazovka ABCD kvízu, kde lze vidět mimo jiné hlavně otázku a možnosti odpovědí. Na druhém obrázku je zobrazeno mobilní zařízení se čtyřmi tlačítky pro odpověď, které plně využívají prostor obrazovky.

### Průběh hry:

- Všichni proti všem:** Cílem každého hráče je co nejrychleji odpovědět správně. Každý může zadat odpověď pouze jednou.
- Týmový režim:** Jakmile jeden člen týmu vybere odpověď, ostatní v jeho týmu již odpovídat nemohou. Každý tým může zadat odpověď pouze jednou.

Bodování bere v úvahu nejen správnost odpovědi, ale také rychlosť jejího zvolení. Kolo pokračuje, dokud všichni neodpoví, nebo nevyprší časový limit. Jakmile kolo skončí, ukáže se na hlavní obrazovce správná odpověď a kolik hráčů vybral jednotlivé možnosti.

### Otevřené odpovědi

Tento typ kvízu vyžaduje, aby hráči zadali přesnou odpověď do textového pole. Na hlavní obrazovce je otázka a je zde zobrazen počet podtržitek odpovídající počtu písmen v odpovědi. Časem se budou náhodně vyplňovat jednotlivá políčka pro zbylé hráče, aby měli šanci

ještě odpovědět. Tento typ kvízu zahrnuje tři různé typy otázek: normální textová otázka, otázka s využitím obrázku, který se může (ale nemusí) postupně odhalovat, a nakonec otázka s využitím audia. Zmíněná hlavní obrazovka je zobrazena na obrázku 3.7a a mobilní obrazovka na obrázku 3.7b, kde je opět kladen důraz na využití celého prostoru obrazovky, což má minimalizovat chybá stisknutí tlačítka nebo textového pole ze strany hráče. Textové pole bude také zobrazovat hlášky, které oznamují, že se hráč například přepsal o jedno písmenko, nebo zadal odpověď s malou odchylkou.



(a) Hlavní obrazovka kvízu *Otevřené odpovědi*

(b) Mobilní obrazovka

Obrázek 3.7: Na prvním obrázku lze vidět hlavní obrazovku kvízu *Otevřené odpovědi*, která zobrazuje textový typ otázky a políčka odpovídající písmenům správné odpovědi. Na druhém obrázku je zobrazena mobilní obrazovka, která obsahuje textové pole pro odpověď a tlačítko pro její odeslání, přičemž důraz je kladen na efektivní využití prostoru.

### Průběh hry:

- Všichni proti všem:** Cílem každého hráče je co nejrychleji odpovědět správně. Každý má neomezený počet pokusů.
- Týmový režim:** Jakmile jeden člen týmu zadá správnou odpověď, ostatní v jeho týmu již odpovídat nemohou. Každý tým má neomezený počet pokusů.

Bodování bere v úvahu nejen správnost odpovědi, ale také rychlosť jejího zvolení. Kolo pokračuje, dokud všichni neodpoví, nebo nevyprší časový limit. Jakmile kolo skončí, ukáže se na hlavní obrazovce správná odpověď a kolik hráčů ji uholilo.

### Slepá mapa

Slepá mapa kombinuje dvě fáze, které hráče postupně vedou k určení názvu města a jeho polohy na mapě. Tento kvíz je zaměřen na geografické znalosti a byl hodně inspirován

z pořadu *Máme rádi Česko*, který je blíže popsaný v sekci 3.1.1. V první fázi hráči řeší přeházená písmena (tzv. přesmyčku, neboli anagram) pro odhalení názvu města. Na hlavní obrazovce je zobrazena přesmyčka spolu s postupně odhalovanými nápovedami, což lze vidět na obrázku 3.8. Na mobilní obrazovce mají možnost zadat svou odpověď, přičemž časem mohou získat až tři nápovery. Mobilní obrazovka pro vyřešení anagramu je stejná jako v předchozím druhu kvízu. V druhé fázi mají hráči za úkol určit polohu daného města na mapě České republiky nebo Evropy, kterou vyberou na mobilním zařízení a pošlou odpověď, jak je znázorněno na obrázku 3.9.

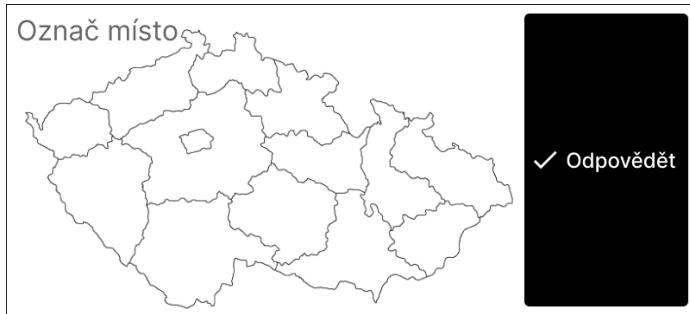


Obrázek 3.8: Na obrázku je vidět hlavní obrazovka 1. fáze, kde se postupně objevují až tři nápovedy pro název města, který je ukrytý v přesmyčce.

### Průběh hry:

- **Všichni proti všem:**
  - **První fáze:** Každý hráč se snaží co nejrychleji správně vyřešit anagram na mobilní obrazovce zadáním správného názvu města. První fáze skončí, až všichni odpoví, nebo vyprší časový limit.
  - **Druhá fáze:** Každý hráč se snaží na mapě označit místo, kde se dané město nachází. Tato fáze není časově intenzivní. Druhá fáze skončí, až všichni odpoví, nebo vyprší časový limit. Na hlavní obrazovce se oproti týmovému režimu mapa nezobrazuje.

Bodování se skládá ze dvou složek: rychlosti odpovědi v první fázi a přesnosti určení polohy v druhé fázi. V první fázi získává hráč bonusovou hodnotu podle pořadí, ve kterém správně odpověděl. Ve druhé fázi se bodování odvíjí od vzdálenosti od správného místa na mapě, která je rozdělena do několika poloměrů. Celkový výsledek hráče se vypočítá jako součin bonusové hodnoty z první fáze a bodového zisku z druhé fáze. Jakmile kolo skončí, objeví se na hlavní obrazovce mapa se správným místem a kolik lidí jej uhodlo, ale ukážou se i neúspěšně vybraná místa.

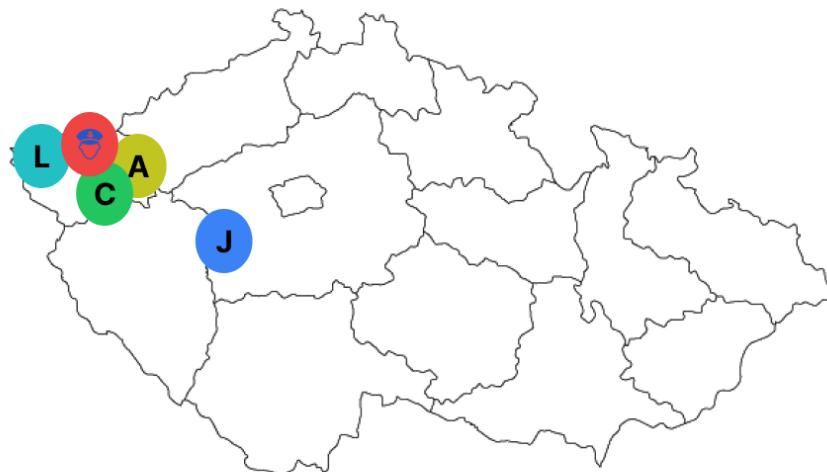


Obrázek 3.9: Na obrázku lze vidět mobilní obrazovku, na které hráč v 2. fázi vybere místo, kde si myslí, že se dané město nachází a poté svou odpověď odešle tlačítkem „Odpovědět“.

- **Týmový režim:**

- **První fáze:** Všichni hráči mohou neomezeně odpovídat, dokud někdo nezadá správnou odpověď. Tým, jehož člen první uhodne správné město, získává výhodu v následující fázi. První fáze skončí po první správné odpovědi.
- **Druhá fáze:** Tým, který uhodnul název města, má přednost v určování jeho polohy na mapě. Všichni členové mohou označit místo, kde si myslí, že se město nachází, ale konečnou odpověď podává kapitán týmu. Odpovědi jednotlivých členů týmu se po zadání objevují na hlavní obrazovce na mapě, jak lze vidět na obrázku 3.10. V případě, že první tým neoznačí správné místo, bude hádat druhý tým, který bude mít výhodu v tom, že ví, kde se dané město nenachází.

Bodování je založeno na správnosti určení polohy v druhé fázi. Body dostane pouze tým, který uhodl správné místo města. Pokud ale oba týmy netrefí správné místo, dostanou oba týmy body dle blízkosti ke správnému místu. Jakmile kolo skončí, zobrazí se správné umístění města na mapě a odpověď kapitána, nebo obou kapitánů v případě neúspěchu.



Obrázek 3.10: Na obrázku je zobrazena mapa České Republiky, která se objeví na hlavní obrazovce v 2. fázi v týmovém režimu. Na mapě se postupně zobrazují zadaná místa všech členů týmu včetně finálního výběru kapitána týmu.

## Kreslení

Kvíz *Kreslení* funguje velice podobně jako platforma Scribbl.io, která je podrobněji popsána v sekci 3.1.3. Kvíz vyžaduje, aby hráči zadali přesnou odpověď do textového pole na mobilním zařízení na základě obrázku kresleného jedním z hráčů. Na hlavní obrazovce, zobrazené na obrázku 3.11a, je možné sledovat aktuální kresbu a počet podtržitek odpovídajících písmenům správné odpovědi. Časem se budou náhodně vyplňovat jednotlivá políčka pro zbylé hráče, aby měli šanci ještě odpovědět. Kreslící hráč si nejprve na mobilním zařízení vybere ze tří možností, kterou bude kreslit. Po vybrání uvidí na obrazovce slovo či sousloví, které má nakreslit, a kreslící plátno. Dále zde má různé nástroje na kreslení, jako je tužka (s možností volby tloušťky), kbelík na vyplnění uzavřených ploch, výběr barvy a nakonec zde jsou tlačítka pro krok zpět a vymazání plátna. Tato mobilní obrazovka je znázorněna na obrázku 3.11b. Textové pole pro odpovědi, shodné s tím na obrázku 3.7a, navíc zobrazuje hlášky, například pokud je odpověď blízko správnému výsledku, ale obsahuje drobnou chybu.



Obrázek 3.11: Na prvním obrázku lze vidět hlavní obrazovku kvízu *Kreslení*, která zobrazuje aktuální kresbu a políčka odpovídající písmenům správné odpovědi. Na druhém obrázku je zobrazena mobilní obrazovka, která obsahuje plátno pro kreslení a kreslené slovo či sousloví. V poslední řadě jsou zde také různé nástroje pro kreslení a úpravu.

### Průběh hry:

- **Všichni proti všem:** Cílem každého hráče je co nejrychleji uhodnout správnou odpověď. Každý má neomezený počet pokusů. Hráči se postupně střídají v kreslení. Kreslící hráč se snaží nakreslit obrázek co nejlépe, aby ho ostatní hráči mohli uhodnout.

- **Týmový režim:** Vždy kreslí jeden hráč z týmu a hádají pouze ostatní členové jeho týmu. Jakmile jeden člen týmu zadá správnou odpověď, ostatní v jeho týmu již odpovídát nemohou. Tým má neomezený počet pokusů. Tímto způsobem se budou týmy střídat v kreslení.

V režimu všichni proti všem bere bodování v úvahu nejen správnost odpovědi, ale také rychlosť jejího zvolení. Kreslící hráč dostane body na základě toho, kolik lidí uhodlo jeho obrázek, tedy čím více lidí uhodlo odpověď, tím více bodů dostane. V týmovém režimu dostane tým body tehdy, když člen týmu uhodne kresbu kreslícího hráče téhož týmu. Kolo pokračuje, dokud všichni neodpoví, nebo nevyprší časový limit. Jakmile kolo skončí, ukáže se na hlavní obrazovce správná odpověď a kolik hráčů ji uhodlo.

### Hádej číslo

Kvíz Hádej číslo je založen na otázkách, na které jsou správné odpovědi složitá čísla. Tento kvíz byl inspirován pořadem *Máme rádi Česko*, který je podrobněji popsán v sekci 3.1.1. Na hlavní obrazovce je zobrazena otázka, což lze vidět na obrázku 3.12. Mobilní obrazovka pro zadávání čísla je stejná jako u kvízu Otevřené odpovědi (obrázek 3.7b).



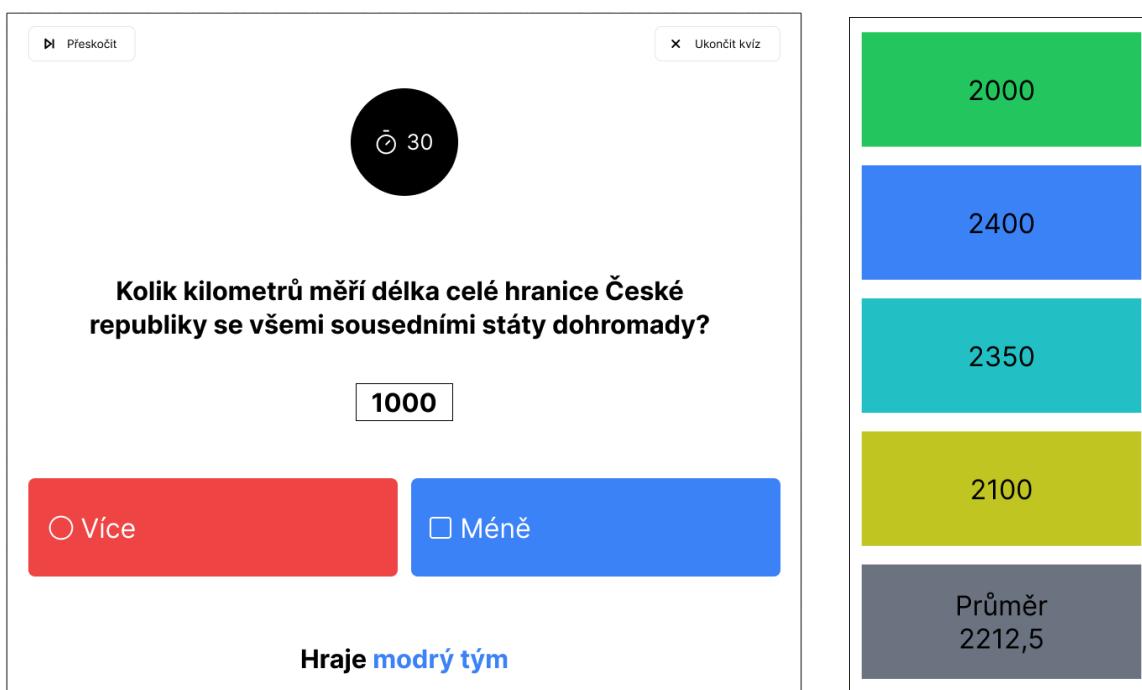
Obrázek 3.12: Na obrázku je vidět hlavní obrazovka kvízu Hádej číslo, která obsahuje otázku a informaci o aktuálně hrajícím týmu (v případě týmového režimu).

### Průběh hry:

- **Všichni proti všem:** Cílem každého hráče je zadat číslo co nejbližše ke správné odpovědi. Každý může zadat odpověď pouze jednou.
- **Týmový režim:**
  - **První fáze:** Odpovídá pouze jeden tým. Každý člen týmu zadá svůj tip na svém mobilním zařízení. Kapitán týmu si z nich následně vybere finální týmovou odpověď, nebo použije průměrnou hodnotu tipů všech členů jak je na obrázku 3.13b. Pokud první tým trefí přesnou hodnotu, druhá fáze se nehraje.

- **Druhá fáze:** Druhý tým vidí odpověď prvního týmu a všichni hráči hlasují, zda je správná odpověď větší, nebo menší pomocí tlačítka „Více“ a „Méně“ jak lze vidět na obrázku 3.13a. Finální odpověď týmu je určena většinovým hlasováním. V případě nerovnodobého hlasování rozhoduje volba kapitána týmu. Mobilní rozhraní pro volbu „Více“ nebo „Méně“ odpovídá rozhraní Pravda/Lež v kvízu ABCD, kde jsou dvě tlačítka vedle sebe přes celou obrazovku.

V režimu všichni proti všem jsou hráči seřazeni podle blízkosti ke správné odpovědi a body dostanou podle výsledku řazení, přičemž kolo končí, jakmile všichni odpoví, nebo vyprší časový limit. V týmovém režimu kapitán vítězného týmu zatočí kolem štěstí a získá tak body pro svůj tým. Kolo štěstí se ovládá na mobilním zařízení kapitána, přičemž jeho otáčení a výsledky jsou současně zobrazovány na hlavní obrazovce. Po skončení kola se na hlavní obrazovce zobrazí správná odpověď a zadáné odpovědi jednotlivých hráčů spolu s počtem hlasů pro každou možnost.



(a) Hlavní obrazovka kvízu Hádej číslo

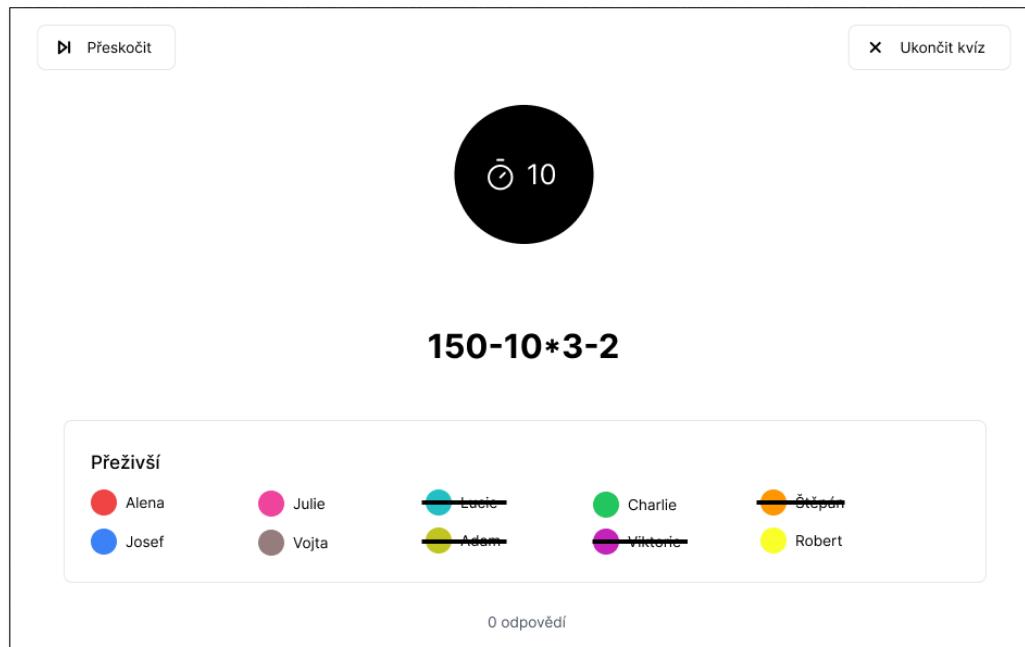
(b) Mobilní obrazovka

Obrázek 3.13: Na prvním obrázku je zobrazena hlavní obrazovka druhé fáze kvízu Hádej číslo v týmovém režimu. Kromě otázky z první fáze je zde navíc ukázána odpověď prvního týmu z první fáze a dvě možnosti na výběr („Více“ a „Méně“), kterou volí druhý tým. Na druhém obrázku je vidět mobilní obrazovka kapitána v první fázi, kde vidí hodnoty zadané svými spoluhráči a vypočítaný průměr. Kapitán si z nich vybere výslednou odpověď.

## Matematický kvíz

V tomto kvízu hráči pod časovým tlakem odpovídají na matematické rovnice, které mohou být buď ve formě klasické číselné rovnice, nebo zapsány slovně (například *počet moudráků + divy světa \* jídelní příbor*). Kvíz funguje v režimu přežití, kde hráči, kteří odpoví špatně, okamžitě vypadávají. Tento režim je inspirován herním módem z aplikace *Quizzland*. Kvíz vyžaduje, aby hráči zadali přesnou odpověď do textového pole na mobilním zařízení, jak bylo

dříve zobrazeno na obrázku 3.7b. Na hlavní obrazovce (obrázek 3.14) je zobrazena aktuální matematická rovnice a seznam hráčů. Hráči, kteří vypadli, jsou označeni přeškrtnutím.



Obrázek 3.14: Na obrázku je vidět hlavní obrazovka Matematického kvízu, kde se nachází číselně, nebo slovně zapsaná rovnice. Rovněž se zde nachází seznam hráčů, kteří stále hrají, nebo již vypadli. Hra probíhá bez přestávek, takže seznam hráčů je viditelný po celou dobu.

### Průběh hry:

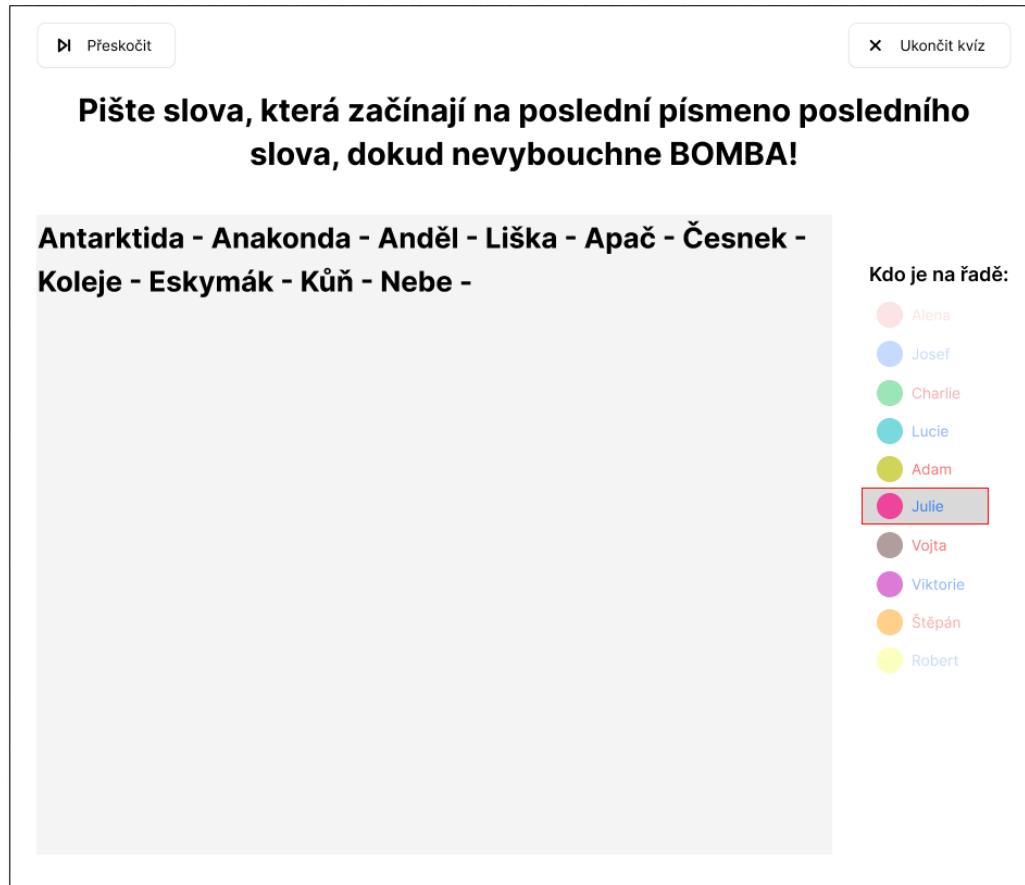
- Všichni proti všem:** Cílem každého hráče je co nejrychleji vypočítat rovnici správně. Pokud zadá chybnou odpověď, ihned vypadává. Každý má pouze jeden pokus na odpověď.
- Týmový režim:** Jakmile jeden člen týmu zadá správnou odpověď, ostatní v jeho týmu již odpovídат nemohou. Pokud však člen týmu zadá špatnou odpověď, vypadává ze hry zatímco zbytek týmu pokračuje.

Ačkoli se jedná o režim přežití, body se přidělují. Jejich váha je však relevantní pouze v případě, že po skončení hry zůstane více přeživších hráčů nebo týmů. Bodování je založeno na rychlosti zadání správné odpovědi, přičemž body získávají pouze hráči, kteří vydrželi až do konce hry. Kolo pokračuje, dokud všichni neodpoví, nebo nevyprší časový limit. Jakmile kolo skončí, okamžitě začíná další kolo a z tohoto důvodu zůstává seznam přeživších a vypadlých hráčů zobrazen na hlavní obrazovce po během celé hry. Hra končí buď po vyčerpání všech otázek, nebo ve chvíli, kdy zůstane poslední přeživší hráč či tým.

### Slovní řetěz

Slovní řetěz je rychlá slovní hra, ve které hráči střídavě zadávají slova do řetězu (každé slovo začíná posledním písmenem předchozího slova). Pokud hráč zadá slovo, které není ve slovníku, musí v časovém limitu zadat jiné. Hráči zadávají slova prostřednictvím textového

pole na mobilním zařízení, stejně jako v předchozím kvízu zobrazeném na obrázku 3.7b. Na hlavní obrazovce, která je na obrázku 3.15, se postupně přidávají zadávaná slova do řetězu a je zde také znázorněno pořadí hráčů a informace o tom, kdo je zrovna na řadě.



Obrázek 3.15: Na obrázku lze vidět hlavní obrazovku kvízu Slovní řetěz pro týmový režim, kde je zobrazen řetěz slov, který hráči vytváří a dále je zde vidět pořadí hráčů a aktuální hráč na tahu.

#### Průběh hry:

- Všichni proti všem:** Cílem každého hráče je co nejrychleji zadat slovo do řetězu. Každý hráč má svůj vlastní časovač, který běží pouze během jeho tahu. Po zadání slova se jeho časovač zastaví a na řadě je další hráč. Pokud hráči čas vyprší, končí pro něj hra. Na hlavní obrazovce jsou vidět všechny časovače hráčů, přičemž aktuální hráč je označen trojúhelníkem nad časovačem, jak je znázorněno na obrázku 3.16. Kromě časovače je u každého hráče zobrazeno jeho jméno, barva a aktuální počet bodů.
- Týmový režim:** Hráči z obou týmů se střídavě snaží zadat slova do řetězu co nejrychleji, zatímco na pozadí tiská bomba. Výbuch bomby nastane pro hráče v neznámý moment. Časovač bomby může být například nastaven na 5 minut. Před samotným výbuchem se hlavní obrazovka začne vizuálně měnit, například začne blikat nebo se přidá efekt kouře. Pokud bomba vybuchne během tahu hráče jednoho týmu, druhý tým automaticky vyhrává.



Obrázek 3.16: Na obrázku jsou zobrazeny všechny časovače hráčů v režimu všichni proti všem ve kvízu *Slovní řetěz*. U časovačů jsou viditelné také přezdívky hračů, jejich barevné označení a aktuální bodové skóre. Trojúhelník nad časovačem označuje hráče, který je právě na tahu.

Body se v obou režimech získávají za správně zadaná slova přidaná do řetězu. V týmovém režimu však body získá pouze tým, kterému nevybuchla bomba. Hra končí, když všem hráčům v režimu všichni proti všem vyprší časovač, nebo vybuchne bomba v týmovém režimu.

### Kombinovaný kvíz

Kombinovaný kvíz spojuje všechny dříve popsané typy kvízů do jedné hry. Tento formát byl inspirován televizním pořadem *Máme rádi Česko*, který je popsán v sekci 3.1.1, a umožňuje hráčům zažít různé herní styly v rámci jedné soutěže. Aby se předešlo zmatkům, jednotlivé typy kvízů jsou odděleny do samostatných fází. Každý typ musí být kompletně odehrán (tj. všechny otázky daného typu), než hra přejde k dalšímu. Například se nejprve odehráje pět otázek ve stylu ABCD, poté pět otázek s **Otevřenými odpověďmi** a tak dále. Hra je vždy zakončena kvízem Uhodni číslo.

#### Průběh hry:

- Všichni proti všem:** Hráči soutěží individuálně a sbírají body na základě své úspěšnosti ve všech fázích. Tento režim neobsahuje kolo štěstí.
- Týmový režim:** Hráči soutěží ve dvou týmech. V tomto režimu je součástí kvízu uhodni číslo na konci hry kolo štěstí, které může výrazně ovlivnit konečný výsledek.

Bodování v každém typu kvízu má stejnou váhu, přičemž body jsou rovnoměrně rozděleny podle počtu otázek v daném typu. Tento systém zajišťuje spravedlivé hodnocení bez ohledu na to, který typ kvízu hráči nebo týmy preferují. V týmovém režimu však může závěrečné kolo štěstí přinést dramatickou změnu ve výsledcích.

### Nevyužité kvízy

- Pantomima a Hádej věc** – Jsou to dva podobné koncepty kvízů, které spočívají v tom, že jeden hráč musí přimět ostatní, aby uhodli dané slovo. V prvním případě pomocí předvádění beze slov (pantomima) a ve druhém případě pomocí popisu, přičemž některá klíčová slova nesmějí být vyslovena. Tyto kvízy by mohly být teoreticky kombinovány do jednoho, přičemž hráči by si sami zvolili, zda budou slova popisovat nebo předvádět. Oba kvízy jsou inspirovány z pořadu *Máme rádi Česko* ze sekce 3.1.1 a zároveň sdílí podobnost s kvízem **Kreslení**, který je popsán v sekci 3.2.

### Průběh hry:

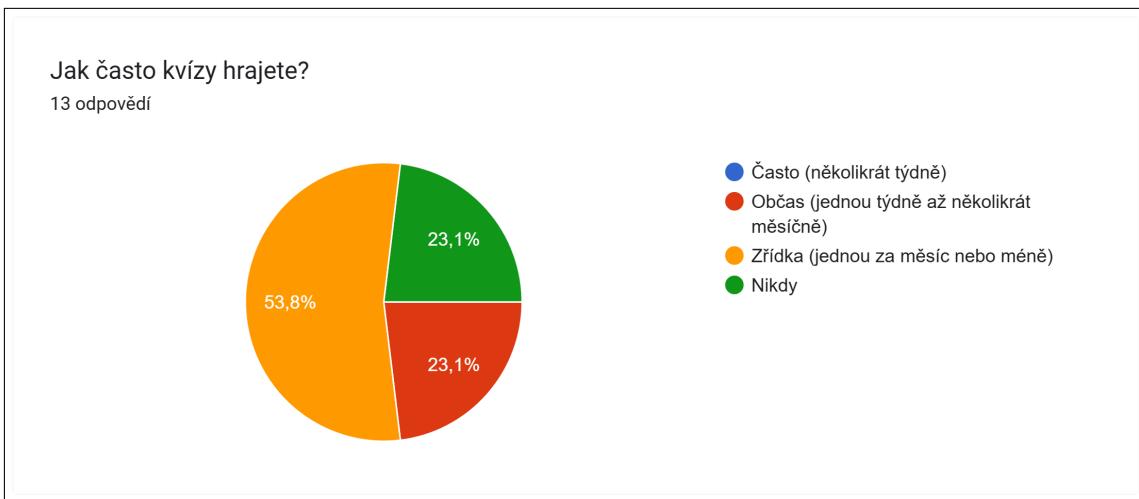
- **Všichni proti všem:** Cílem každého hráče je co nejrychleji uhodnout správnou odpověď. Každý má neomezený počet pokusů. Hráči se postupně střídají v předvádění, nebo povídání a snaží se, aby ostatní hráči uhodli správnou odpověď.
- **Týmový režim:** Kapitán týmu předvádí nebo popisuje po zadaný časový limit, zatímco jeho tým odpovídá na mobilním zařízení. Předvádějící má na svém zařízení nejen pojem, ale i tlačítko pro přeskočení obtížných slov. Počet slov uhodnutých během limitu určuje body pro tým. Poté hraje druhý tým, přičemž celkový počet kol je předem nastaven.

Hlavním důvodem vyřazení těchto kvízů je skutečnost, že by se nedala efektivně využít hlavní obrazovka, protože pozornost hráčů by se soustředila na toho, kdo předvádí nebo popisuje. Tento formát by tak nevyužíval potenciál aplikace.

- **Uhodni název písni, nebo zpěváka/kapelu** – Tento kvíz původně zahrnoval hádání názvu písni nebo interpreta na základě krátké hudební ukázky přehrané na hlavní obrazovce. Hráči by na svých mobilních zařízeních zadávali odpovědi, přičemž by otázky mohly být zaměřeny buď specificky na jednu věc (například název skladby nebo jméno zpěváka), nebo by hráč mohl získat body za jakoukoli správnou informaci o písni (např. plný počet bodů za název, méně za interpreta). Tento kvíz nebyl vyřazen, ale byl přesunut do kvízu **Otevřené odpovědi** popsaného v sekci 3.2.
- **Poznávačka podle obrázku** – Tento kvíz se zaměřoval na rozpoznávání slavných osobností, míst nebo předmětů na základě obrázku, který by se postupně odhaloval na hlavní obrazovce. Hráči by museli co nejrychleji zadat správnou odpověď na svých mobilních zařízeních. Tento kvíz nebyl zcela vyřazen, ale byl přesunut do kvízu **Otevřené odpovědi** popsaného v sekci 3.2.
- **Kdo je to? (stopy)** – Tento kvíz spočívá v identifikaci slavných osobností na základě různých stop, jako jsou fotografie, citáty nebo úryvky z jejich životopisu. Na hlavní obrazovce by se postupně odkrývaly stopy, přičemž hráči by museli co nejrychleji zadat správnou odpověď na svých mobilních zařízeních. Bodování by záviselo na počtu využitých stop — čím méně stop hráč potřebuje k určení správné odpovědi, tím více bodů získá. Tento kvíz nebyl zařazen do aplikace zejména proto, že jeho příprava by byla časově velmi náročná, a také by vyžadovala značné množství prostoru pro ukládání obrázků a dalších multimediálních souborů.

### 3.3 Průzkum trhu

Pro zjištění názorů na navržené kvízy a jejich potenciální hratelnost jsem vytvořil dotazník v nástroji Google Forms. Dotazník měl za cíl zjistit, zda se lidem líbí návrhy kvízů a zda by je byli ochotni hrát. Dotazník vyplnilo celkem 13 respondentů ve věkovém rozmezí 21–24 let. Na obrázku 3.17 je znázorněno, jak často respondenti hrají různé typy kvízů. Na grafu lze vidět, že většina respondentů si alespoň jednou měsíčně nějaký kvíz zahraje, z čehož vyplývá, že jejich zpětná vazba na mé kvízy bude relevantní. Z dalších výsledků vyplývá, že nejčastěji hrají kvízy v hospodách nebo ve škole se svými přáteli a spolužáky. Některí také hrají online kvízy z domova prostřednictvím počítače.



Obrázek 3.17: Graf z dotazníku zobrazující četnost hraní kvízů respondenty.

Jednotlivé typy kvízů byly v dotazníku popsány a doplněny obrázky návrhů vytvořených v aplikaci Figma. Respondenti měli za úkol ohodnotit každý kvíz na škále 1 až 5 hvězdiček, kde 5 představuje nejlepší možné hodnocení a 1 nejhorší. V tabulce 3.1 jsou uvedeny průměrné hodnoty hodnocení jednotlivých kvízů.

	Průměrné hodnocení
ABCD kvíz	4,08
Otevřené odpovědi	3,92
Slepá mapa	3,46
Uhodni číslo	3,46
Kreslení	3,85
Matematický kvíz	3,38
Slovní řetěz	3,15
Kombinovaný kvíz	3,85

Tabulka 3.1: Průměrné hodnocení jednotlivých kvízů respondenty dotazníku, kde 5 je nejlepší hodnocení a 1 nejhorší.

Z výsledků hodnocení vyplývá, že největší oblibě se těší ABCD kvíz, zatímco Slovní řetěz získal nejnižší průměrné hodnocení. Přesto byly všechny kvízy hodnoceny spíše nadprůměrně, a proto jsem se rozhodl naimplementovat všechny typy kvízů.

Na základě zpětné vazby respondentů jsem provedl několik úprav v kvízu **Hádej číslo** v týmovém režimu. Body se již neudělují prostřednictvím kola šestí. Namísto toho se přidělují pouze na základě odpovědí v druhé fázi. Druhý tým získává body, pokud správně odhadne, zda je odpověď větší nebo menší než hodnota zadána prvním týmem. Pokud je odhad špatný, získává body první tým. To znamená, že za první fázi body nejsou a kolo šestí bylo přesunuto do kombinovaného kvízu.

Další změny se týkají hratelnosti a způsobu zadávání odpovědí. Původně všichni hráči v týmu zadávali svůj tip povinně až na kapitána, který poté vybral finální odpověď od svých spoluhráčů, nebo jejich průměr. Nyní však hráči mohou, ale nemusí zadávat své individuální tipy. Tyto tipy se ihned zobrazí kapitánovi na jeho mobilním zařízení, kde má možnost vybrat jednu z navržených odpovědí nebo zadat vlastní. Kapitán může rovněž zvolit

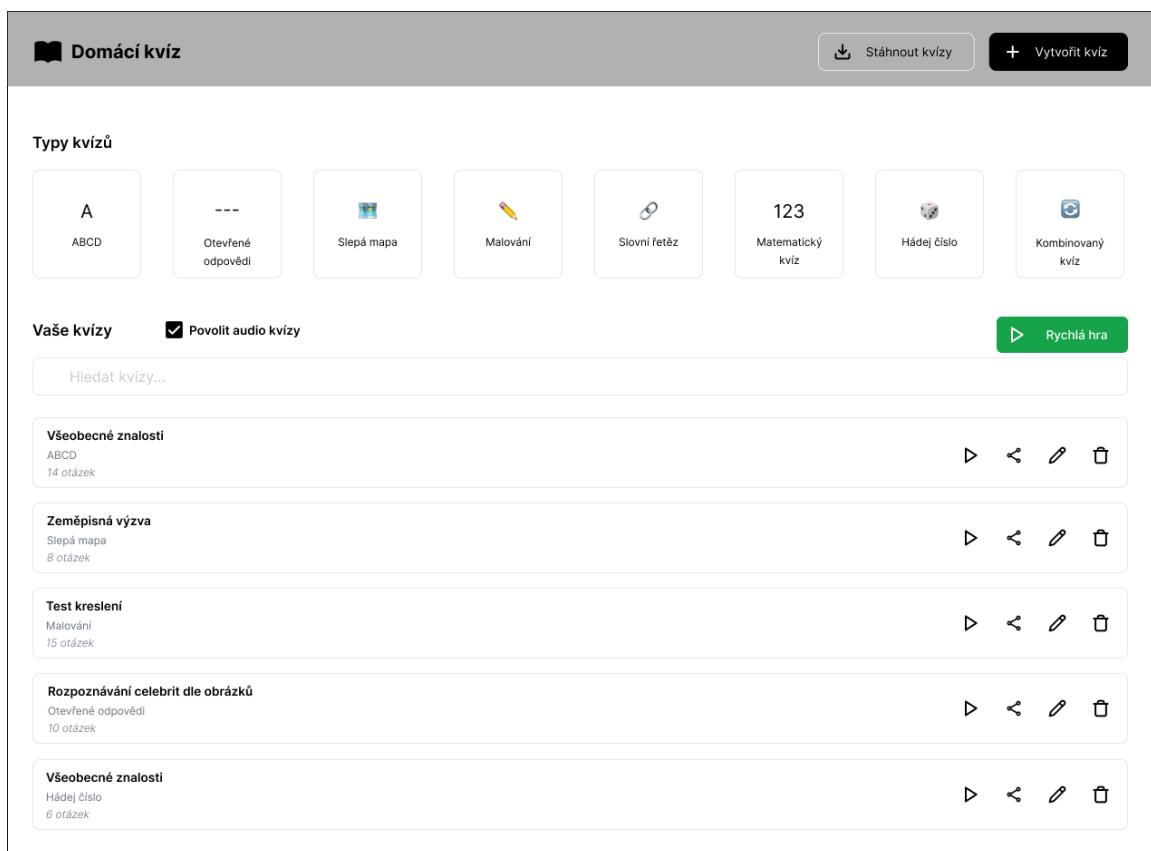
průměrnou hodnotu zadaných tipů jako finální odpověď týmu. Tímto způsobem zajistíme hratelnost i v případě, že nejsou hráči u sebe, ale sedí v různých částech místnosti. Když jsou u sebe, tak se mohou radit a kapitán následně zadá číslo, na kterém se všichni shodnou.

## 3.4 Návrh uživatelského rozhraní

Při návrhu aplikace Domácí kvíz jsem se inspiroval zejména platformou Kahoot, která je podrobněji popsána v sekci 3.1.2. Jednotlivé typy kvízů, které tato aplikace zahrnuje, byly rozebrány v podkapitole 3.2. Tato kapitola se zaměřuje na další klíčové prvky aplikace, jako jsou hlavní strana aplikace, proces stahování a spuštění kvízů, tvorba vlastních kvízů a obecný průběh hry, který bude pro všechny typy kvízů do značné míry stejný.

### Hlavní strana

Aby byla aplikace co nejpřístupnější a intuitivní, rozhodl jsem se vynechat registraci a přihlašování uživatelů. Veškerá funkcionalita aplikace je navržena tak, aby fungovala bez těchto kroků. Po prvním spuštění aplikace se uživateli zobrazí hlavní strana, jež je zobrazena na obrázku 3.18, kde najde seznam předpřipravených kvízů, které lze okamžitě hrát.



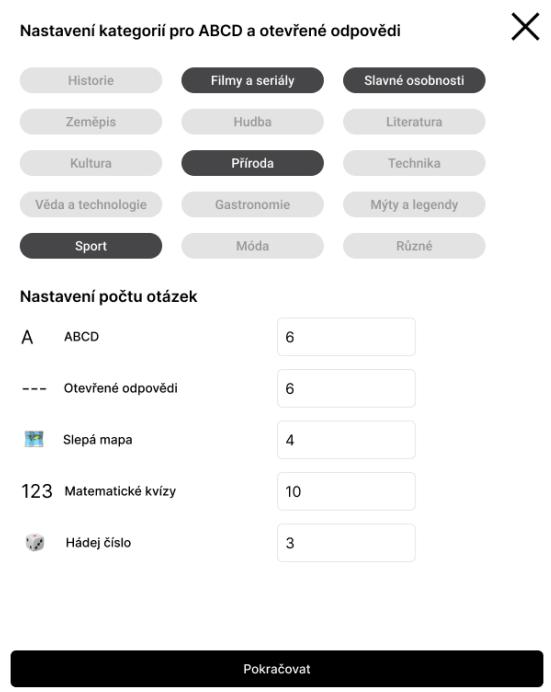
Obrázek 3.18: Na obrázku je zobrazena hlavní strana aplikace Domácí kvíz. Uživatel zde vidí seznam kvízů, které vytvořil, stáhl, nebo které byly předem připravené. Jednotlivé kvízy mají tlačítka pro správu. Hlavní stránka také umožňuje filtrování podle typu kvízu, stahování nových kvízů, přesměrování na tvorbu vlastních kvízů, nebo zapnout rychlou hru.

Uživatel má možnost seznam kvízů procházet pomocí vyhledávače nebo filtrovacích tlačítek podle typu kvízu. Filtrační tlačítka umožňují výběr pouze jednoho typu kvízu, který je třeba specifikovat před spuštěním rychlé hry. Pokud uživatel žádný typ kvízu nevybere a pokusí se spustit rychlou hru, aplikace ho na to upozorní hláškou.

Po spuštění rychlé hry u vybraného typu kvízu se otevře vyskakovací okno (obrázek 3.19a), kde lze nastavit parametry jako počet otázek (pro kvízy ABCD, Otevřené odpovědi nebo kombinovaný kvíz) nebo počet kol (pro kvíz Kreslení). Tato nastavení se automaticky uloží pro budoucí použití. Kvíz Slovní řetěz se spustí bez nutnosti dalších nastavení. Rychlá hra generuje otázky náhodně z databáze, do které přispívají ostatní uživatelé. Aby se předešlo opakování otázek, aplikace si lokálně ukládá ID již hraných otázek.

Hlavní strana také umožňuje uživatelům vyfiltrovat kvízy obsahující audio, což je užitečné v situacích, kdy není k dispozici zvukové zařízení, nebo se bude hrát v tichém prostředí. V pravém horním rohu jsou dvě tlačítka nazvaná „Stáhnout kvízy“ a „Vytvořit kvíz“. Po kliknutí na tlačítko pro stažení kvízů se zobrazí vyskakovací okno, které lze vidět na obrázku 3.19b, kde si uživatel může pomocí vyhledávače najít kvízy vytvořené ostatními a stáhnout si je. Stažené kvízy se automaticky přidají do seznamu na hlavní stránce.

Každý kvíz v seznamu má vlastní tlačítka pro správu, která umožňují spouštění hry, sdílení s ostatními uživateli, editaci nebo odstranění kvízu.

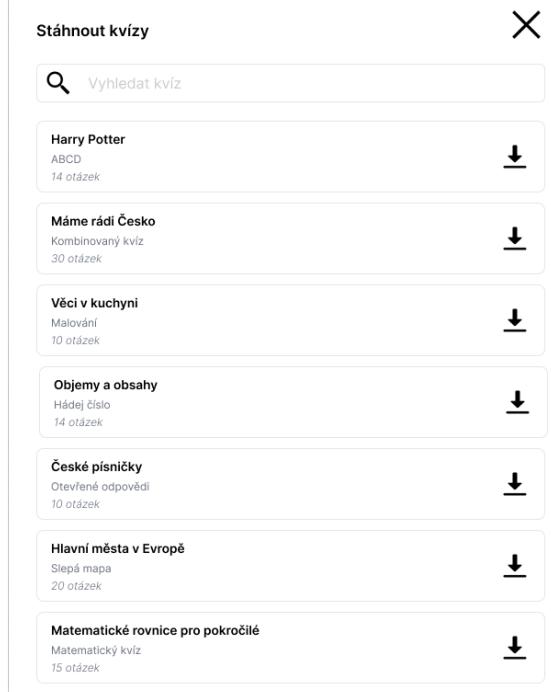


**Nastavení kategorii pro ABCD a otevřené odpovědi**

Kategorie	Počet otázek
A ABCD	6
--- Otevřené odpovědi	6
Slepá mapa	4
123 Matematické kvízy	10
Hádej číslo	3

**Nastavení počtu otázek**

**Pokračovat**



**Stáhnout kvízy**

Quiz	Typ	Počet otázek	Ovládací prvky
Harry Potter	ABCD	14 otázek	
Máme rádi Česko	Kombinovaný kvíz	30 otázek	
Věci v kuchyni	Malování	10 otázek	
Objemy a obsahy	Hádej číslo	14 otázek	
České písničky	Otevřené odpovědi	10 otázek	
Hlavní města v Evropě	Slepá mapa	20 otázek	
Matematické rovnice pro pokročilé	Matematický kvíz	15 otázek	

(a) Nastavení kombinovaného kvízu

(b) Stažení kvízů z databáze

Obrázek 3.19: Na prvním obrázku lze vidět vyskakovací okno pro rychlou hru kombinovaného kvízu, kde uživatel může nastavit kategorie otázek a počet otázek pro hru. Na druhém obrázku je zobrazeno vyskakovací okno pro stažení kvízů ze vzdáleného úložiště, kde lze vyhledávat a stahovat kvízy podle názvu.

## Vytváření kvízů

V horní části rozhraní se nachází rozbalovací seznam, kde si uživatel zvolí typ kvízu, pro který chce vytvořit sadu otázek. Všechny typy kvízů sdílejí některé základní prvky, například to, že uživatel vždy zadá název kvízu a stránka je rozdělena na dvě části. V levé části uživatel definuje detaily otázky, jako je text otázky, odpovědi, délka kola, kategorie apod. Na konci je zde tlačítko pro přidání dané otázky. Pravá část slouží jako náhled kvízu, kde uživatel vidí seznam přidaných otázek a může je mazat, upravovat nebo měnit jejich pořadí. Dále je zde možnost přidat náhodnou otázkou z databáze, která obsahuje otázky vytvořené ostatními uživateli. Nakonec je k dispozici tlačítko pro dokončení kvízu.

Vytváření otázek se však liší podle typu kvízu. Například pro **Slovní řetěz** se otázky nevytvářejí, protože hráči hrají na základě vlastních podmínek a čas je pevně stanoven. Níže je popsán postup pro jednotlivé typy kvízů.

- **Otevřené odpovědi:** Uživatel si vybírá mezi textovou otázkou, obrázkovou otázkou nebo otázkou s audiem. U textové otázky zadává otázku, správnou odpověď, délku kola a kategorii. U obrázkové otázky přidává obrázek a může zvolit postupné odhalování obrázku. U otázky s audiem přidává audio soubor oproti textové otázce. Návrh obrazovky pro tento typ kvízu je znázorněn na obrázku 3.20.

The screenshot shows the 'Create Quiz' interface. At the top, it says 'Vytvořit kvíz' and 'Typ: Otevřené odpovědi'. The left side has a 'Quiz name' input field with placeholder 'Zadejte název kvízu'. Below it, there's a section to add a new question with tabs for 'Text' (selected), 'Image', and 'Audio'. The 'Text' tab has fields for 'Question' ('Co se nachází na obrázku?', 'Kdo namaloval tento obraz?...') and 'Correct answer' ('Zadejte správnou odpověď'). A large button labeled 'Přidat obrázek' is present. Below these are checkboxes for 'Postupné odhalování obrázku' and 'Délka kola' (set to 20s). A dropdown for 'Category' is also shown. The right side shows a preview titled 'Náhled kvízu' with three questions. Question 1: 'Otázka 1' - 'Které město je známé jako 'město světel' a je proslulé svým uměním, kulturou a historickými památkami?' (Munich) with an image of the Mona Lisa and edit icons. Question 2: 'Otázka 2' - 'Co je na obrázku?' (Mona Lisa) with edit icons. Question 3: 'Otázka 2' - 'Kdo zpívá tuto písničku?' (Madonna) with a link to 'LikeAPrayer.mp3' and edit icons. At the bottom are 'Přidat otázku' and 'Dokončit kvíz' buttons.

Obrázek 3.20: Na obrázku je zobrazena obrazovka pro vytváření kvízu typu **Otevřené odpovědi**. Uživatel zde zadá název kvízu a pak v levé části zadává detaily otázek, přičemž si může vybrat mezi textovou, obrázkovou a audio otázkou. Pravá část ukazuje náhled kvízu.

- **ABCD kvíz:** Uživatel si může vybrat mezi otázkou typu ABCD a otázkou **Pravda/Lež**. U ABCD otázky zadává text otázky, čtyři možné odpovědi, správnou odpověď, délku kola a kategorii. U otázky **Pravda/Lež** zadá text otázky, správnou odpověď (pravda nebo lež), délku kola a kategorii.
- **Slepá mapa:** Uživatel si vybírá mezi mapou České republiky a Evropy. Zadává název města, jeho přesmyčku (lze vygenerovat automaticky), až tři nápovědy (nepovinné), přesné umístění města na mapě a délku kola.
- **Kreslení:** Uživatel zadá ke každé otázce tři různá slova, ze kterých si hráč může vybrat. To se hodí z toho důvodu, že například nějaké slovo nezná, nebo neví jak jej nakreslit. Čas pro kolo je pevně daný, ale s každým správným uhodnutím se snižuje, čímž se odstraňuje nutnost definovat časový limit pro každé kolo zvláště.
- **Hádej číslo:** Uživatel přidává otázku, správnou odpověď (musí být číslo) a délku kola.
- **Matematický kvíz:** Uživatel zadá rovnici, která může být napsaná čísla, nebo slovy. Dále zadá její správnou odpověď, která opět musí být číslo, a délku kola.
- **Kombinovaný kvíz:** Tento typ kvízu zahrnuje minimální počet otázek z každého dostupného typu kvízu, přičemž **Slovní řetěz** je přidán automaticky. Uživatel nemůže míchat otázky mezi různými typy kvízů, to znamená, že pořadí typů je pevně stanoven, ale lze měnit pořadí otázek v rámci každého typu. Nutnost zahrnutí každého typu otázky se může ještě v implementaci změnit, ale při návrhu jsem si představoval kombinovaný kvíz ve stylu pořadu *Máme rádi Česko*, popsaného v sekci 3.1.1, kde se hraje spoustu kvízů za sebou a hra je zakončena kvízem **Hádej číslo** s kolem štěstí. Z toho důvodu bych si přál zahrnout všechny kvízy, které aplikace podporuje.

## Průběh kvízu

Uživatel, který spustí kvíz, přejde na stránku pro nastavení parametrů hry, jak je znázorněno na obrázku 3.21. Zde si vybere týmový režim, nebo režim všichni proti všem. Je tu zobrazen QR kód pro připojení, nebo odkaz, který musí hráči zadat do prohlížeče na svém mobilním zařízení. Po připojení se jejich přezdívky a barvy objeví na hlavní obrazovce. V týmovém režimu jsou hráči automaticky zařazeni do jednoho ze dvou týmů, přičemž každý tým má kapitána, kterého lze měnit přímo na hlavní obrazovce. Připravenost hráčů je znázorněna ikonou fajfky vedle jejich přezdívky.

Na konci stránky jsou dvě tlačítka pro spuštění hry. Pokud hostující uživatel má kabelem připojenou velkou obrazovku, projektor, nebo hru spustí na aktuálním monitoru, zapne hru tlačítkem „Spustit zde“. Pokud však nemá kabel na připojení a chce spustit hru například na televizi přes prohlížeč, tak zapne hru tlačítkem „Spustit na jiné obrazovce“, čímž se nejprve zobrazí vyskakovací okno. V okně je uvedena adresa, kterou je třeba zadat do prohlížeče, a stručný návod. Jakmile je hra spuštěna na jiné obrazovce, začne po krátké chvíli automaticky. Po celou dobu zůstává vyskakovací okno otevřené, aby bylo možné hru případně ukončit bez použití ovladače nebo jiných zařízení.

# Rozpoznávání celebrit dle obrázků

Všichni  
proti  
všem

Týmová  
hra

## Modrý tým

- Alena
- Josef
- Charlie
- Lucie
- Adam



Změnit kapitána

## Červený tým

- Julie
- Vojta
- Viktorie
- Štěpán
- Robert



Změnit kapitána



Adresa pro připojení:

**192.168.1.108/play**

► Spustit zde

► Spustit na jiné obrazovce

Obrázek 3.21: Na obrázku je vidět hlavní obrazovka po spuštění vybraného kvízu v týmovém režimu, jehož název je napsán na vrchu. Je zde na výběr měnit režimy a kapitány (v případě týmového režimu). Objevují se zde hráči, kteří se připojují s pomocí zobrazeného QR kódu nebo adresy. Hra se dá spustit na aktuální obrazovce, nebo na jiné s využitím jiné adresy.

Hráči se do kvízu připojí zadáním své přezdívky a vyberou si barvu, jak ukazuje obrázek 3.22a. Po připojení jsou přesměrováni do čekací místnosti, kde vidí svou přezdívku a seznam ostatních hráčů, jak ukazuje obrázek 3.22b. V týmovém režimu uvidí týmy s připojenými hráči, včetně svého zařazení. Pokud je v protějším týmu volno, mají možnost se tam přepnout. Jakmile jsou se svým výběrem spokojeni, mohou potvrdit svou připravenost tlačítkem „Připravit“.

### Připojit se do kvízu

Zadej přezdívku

---

Vyber si barvu

✓ Připojit

### Rozpoznávání celebrit dle obrázků

Adam

---

**Modrý tým**

- Alena
- Josef
- Charlie
- Lucie
- Adam

**Červený tým**

- Julie
- Vojta
- Viktorie
- Štěpán

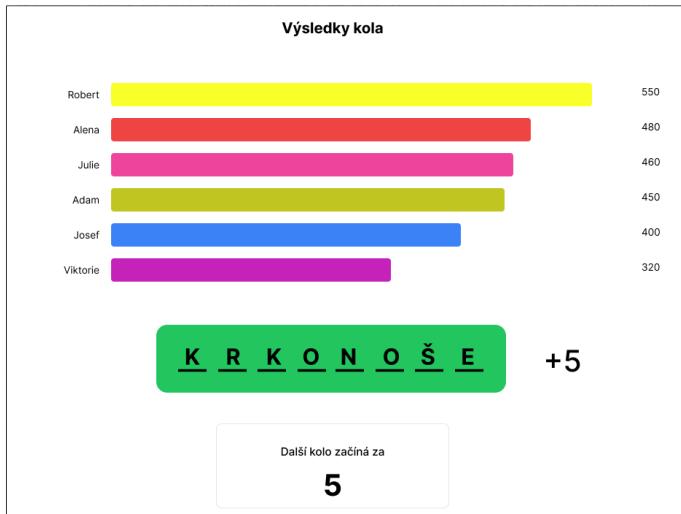
✓ Připojit
✓ Připravit

(a) Připojení do kvízu

(b) Výběr týmu

Obrázek 3.22: Na prvním obrázku lze vidět proces připojení hráče do hry, která je v týmovém režimu. Zadá zde svou přezdívku a vybere si svou barvu. Na druhém obrázku je čekací místnost, kde hráč vidí sebe a dva týmy. Hráč může změnit tým, pokud je v druhém volno. Na konci je tlačítka pro potvrzení připravenosti.

Mezi jednotlivými koly (s výjimkou Matematického kvízu a Slovního řetězu) se zobrazí mezikolo, jehož obrazovka je znázorněna na obrázku 3.23. Tato obrazovka zobrazuje aktuální skóre hráčů (nebo týmů), správnou odpověď na předchozí otázku a počet hráčů, kteří odpověděli správně. Součástí mezikola je také časovač, který odpočítává čas do zahájení dalšího kola. Na mobilním zařízení hráč vidí informaci o tom, zda jejich odpověď byla správná, a případně zprávu, že se čeká na spuštění dalšího kola. Kahoot, popsaný v sekci 3.1.2, rozděloval mezikolo na dvě samostatné části: jednu pro zobrazení správných odpovědí a druhou pro skóre uživatelů. Tyto dvě části jsem spojil do jedné, což zkrátilo dobu čekání na další kolo.



Obrázek 3.23: Na obrázku je vidět obrazovka, která ukazuje aktuální skóre hráčů, správnou odpověď a počet hráčů, kteří ji uhodli. Také je zde časovač, který odpočítává čas do dalšího kola.

Po skončení hry se na hlavní obrazovce zobrazí finální výsledky, včetně pořadí hráčů (nebo týmů) a jejich bodového zisku. Tato obrazovka je znázorněna na obrázku 3.24a. Na mobilních zařízeních hráči vidí pouze své skóre a umístění, jak ukazuje obrázek 3.24b. Na konci hry má hostující uživatel možnost hru ukončit a vrátit se na domovskou stránku. Mobilní zařízení hráčů jsou v tomto okamžiku automaticky odpojena.



Obrázek 3.24: Na prvním obrázku je vidět hlavní obrazovka finálního umístění hráčů v režimu všichni proti všem a na druhém obrázku je zobrazeno finální umístění daného hráče na jeho mobilním zařízení.

## 3.5 Návrh NoSQL databáze

Databázová struktura této diplomové práce je poměrně jednoduchá, jelikož uchovává data pouze o kvízech a otázkách. Jako vhodná se ukázala databáze MongoDB, která je nejen snadno použitelná, ale také díky svému dokumentově orientovanému přístupu umožňuje uchovávat záznamy se společnými i specifickými atributy v jedné kolekci, což zjednoduší implementaci i správu dat.

V této podkapitole je představen návrh dvou kolekcí, a to pro otázky a pro kvízy. Struktura je popsána ve formátu JSON. Nejprve jsou uvedeny společné atributy otázek a následně specifické vlastnosti jednotlivých typů otázek. Výsledná podoba modelu odpovídá reálnému stavu implementace a slouží jako referenční návrh pro další kapitoly.

### 3.5.1 Návrh kolekce otázek

Každá otázka, jak lze vidět v ukázce 3.1, obsahuje text otázky, typ, časový limit, kategorii, odkaz na kvíz, ke kterému náleží, ID zařízení, na kterém byla vytvořena, případný odkaz na původní otázku (pokud byla zkopirována), a metadata s informacemi o četnosti hraní a úspěšnosti odpovědí.

```
{  
    "_id": "<ObjectId>",  
    "question": "<string>",  
    "type": "<enum>",  
    "length": "<int>",  
    "category": "<string>",  
    "part_of": "<ObjectId>", // ID kvízu, ke kterému otázka patří  
    "created_by": "<string>", // ID zařízení, na kterém byla otázka vytvořena  
    "copy_of": "<ObjectId>", // ID původní otázky, pokud byla zkopirována  
    "metadata": {  
        "timesUsed": "<int>",  
        "averageCorrectRate": "<float>"  
    }  
}
```

Ukázka kódu 3.1: Navržená struktura kolekce otázek, která obsahuje všechny společné atributy se vsemi typy otázek, ve formátu JSON v MongoDB.

Otázka typu ABCD, nebo Pravda/Lež, obsahuje mimo společné atributy také pole s možnými odpověďmi a index, který označuje správnou odpověď v daném poli. Tyto atributy jsou znázorněny v ukázce 3.2.

```
{  
    "options": ["Pravda", "Lež"],  
    "answer": "<int>"  
}
```

Ukázka kódu 3.2: Rozšíření otázky typu ABCD a Pravda/Lež ve formátu JSON.

Otzáka typu **Otevřené odpovědi** (ukázka 3.3) obsahuje navíc správnou odpověď, typ a odkaz přiloženého média, pokud nějaký existuje, a v případě přiloženého obrázku je brán důraz i na příznak, který udává informaci o postupném odhalování obrázku.

```
{
  "open_answer": "<string>",
  "media_type": "image/audio",
  "media_url": "<string>",
  "show_image_gradually": "<bool>"
}
```

Ukázka kódu 3.3: Rozšíření otázky typu **Otevřené odpovědi** ve formátu JSON.

Otzáka typu **Matematický kvíz** obsahuje kromě společných atributů také pole jednotlivých matematických rovnic na vyřešení. Toto pole obsahuje zadání rovnice, správnou odpověď a časový limit pro danou rovnici. Tyto atributy jsou znázorněny v ukázce 3.4.

```
{
  "sequences": [
    {
      "equation": "<string>",
      "answer": "<number>",
      "length": "<int>"
    }
  ]
}
```

Ukázka kódu 3.4: Rozšíření otázky typu **Matematický kvíz** ve formátu JSON.

Otzáka typu **Slepá mapa** (ukázka 3.5) obsahuje navíc hledané město a jeho přesmyčku, jeho lokaci na mapě v souřadnicích x a y, typ mapy, obtížnost (která určuje rádius kolem správného místa, který se počítá jako správná odpověď) a až 3 nápovedy.

```
{
  "city_name": "<string>",
  "anagram": "<string>",
  "location_x": "<float>", // X souřadnice v rozmezí 0.0--1.0
  "location_y": "<float>", // Y souřadnice v rozmezí 0.0--1.0
  "map_type": "cz/europe",
  "radius_preset": "<enum>", // "EASY", nebo "HARD"
  "clue1": "<string>",
  "clue2": "<string>",
  "clue3": "<string>"
}
```

Ukázka kódu 3.5: Rozšíření otázky typu **Slepá mapa** ve formátu JSON.

Otzáka typu **Hádej číslo** obsahuje správnou číselnou odpověď. Otázka typu **Kreslení a Slovní řetěz** obsahuje navíc pouze počet kol, jelikož tyto kvízy jsou více dynamické, například u **Kreslení** jsou generována slova až při hraní hry.

### 3.5.2 Návrh kolekce kvízů

Každá položka v kolekci obsahuje své ID, které MongoDB generuje automaticky. Kvíz, jak lze vidět na ukázce 3.6, pak obsahuje název, seznam navázaných otázek s jejich pořadím, typ kvízu, příznak zveřejnění, ID zařízení, na kterém byl vytvořen, a datum vytvoření.

```
{  
    "_id": "<ObjectId>",  
    "name": "<string>",  
    "questions": [  
        {  
            "questionId": "<ObjectId>", // odkaz na ID otázky z její kolekce  
            "order": "<int>"  
        }  
    ],  
    "type": "<enum>", // např. "ABCD", "OPEN_ANSWER", "COMBINED QUIZ", atd.  
    "is_public": "<bool>",  
    "created_by": "<string>",  
    "creation_date": "<timestamp>"  
}
```

Ukázka kódu 3.6: Navržená struktura kolekce kvízů ve formátu JSON v MongoDB.

## 3.6 Návrh API a WebSocketů

API, neboli Application Programming Interface, je rozhraní, které v mé aplikaci slouží k přenosu dat mezi klientskou a serverovou částí aplikace. Serverová vrstva implementuje jednotlivé koncové body (dále jen endpointy), které manipulují s globálním herním stavem nebo s daty uloženými v databázi, jejíž struktura byla popsána výše. Endpointy jsou rozděleny podle oblasti použití a využívají standardní HTTP metody (GET, POST, PUT, DELETE), přičemž důsledně pracují s odpovídajícími stavovými kódy.

Pro návrh API byl využit nástroj SwaggerHub<sup>3</sup>, který nabízí textový editor se specifickou syntaxí pro definování API a generuje přehlednou dokumentaci na základě zapsaného návrhu. Na následujících snímcích obrazovky 3.25 a 3.26 jsou ukázky vygenerované dokumentace, kde jsou dostupné endpointy přehledně rozděleny do několika skupin a opatřeny stručnými popisky. Mimo jiné je na obrázku 3.27 podrobně popsán vybraný endpoint pro změnu jména při čekání na spuštění kvízu. Všechny zmíněné endpointy byly podle tohoto návrhu implementovány a budou v dalších kapitolách sloužit jako referenční rámec.

Kromě klasického API bylo pro některé funkce aplikace nutné navrhnut i komunikaci v reálném čase, realizovanou prostřednictvím WebSocketů (vysvětleno v podkapitole 2.4). Tento způsob přenosu dat umožňuje serveru okamžitě informovat klienty o změnách ve hře, jako je spuštění nové otázky nebo aktualizace skóre, bez nutnosti opakování data-

---

<sup>3</sup><https://app.swaggerhub.com/>

zování. Konkrétní použití vybraných API endpointů a WebSocketů je poté blíže popsáno v implementačních kapitolách 4 a 5.

## Systémové operace Základní systémové endpointy pro poskytování informací o serveru a jeho stavu ^

**GET** /server\_time Získání času serveru 📋 ← ⏪ ↴

**GET** /server\_ip Získání IP adresy a portu serveru 📋 ← ⏪ ↴

**GET** /online\_status Kontrola internetového připojení serveru 📋 ← ⏪ ↴

## Herní řízení Endpointy pro správu herního procesu, progrese kvízu a herních stavů ^

**POST** /activate\_quiz Aktivace kvízového módu 📋 ← ⏪ ↴

**POST** /start\_game Zahájení nové herní instance 📋 ← ⏪ ↴

**POST** /next\_question Přechod na další otázku kvízu 📋 ← ⏪ ↴

**POST** /reset\_game Resetování kompletního stavu hry 📋 ← ⏪ ↴

## Správa médií Endpointy pro správu a manipulaci s mediálními soubory (obrázky a audio nahrávky) ^

**POST** /upload\_media Nahrání mediálního souboru 📋 ← ⏪ ↴

**POST** /delete\_media Odstranění mediálního souboru 📋 ← ⏪ ↴

## Správa hráče Endpointy pro správu připojení hráčů, jejich barev a interakce před začátkem hry ^

**POST** /join Připojení nového hráče ke hře 📋 ← ⏪ ↴

**POST** /change\_name Změna jména připojeného hráče 📋 ← ⏪ ↴

**GET** /available\_colors Získání dostupných barev pro hráče 📋 ← ⏪ ↴

Obrázek 3.25: Dokumentace webové aplikace SwaggerHub, která zobrazuje všechny navržené endpointy pro skupiny „Systémové operace“, „Herní řízení“, „Správa médií“ a „Správa hráčů“, včetně významu každého endpointu a použitých HTTP metod.

## Správa kvízů

Endpointy pro správu kvízů, jejich otázek, kategorií a metadat

^

<b>POST</b>	/check_question	Validace otázky před uložením	   
<b>POST</b>	/create_quiz	Vytvoření nového kvízu	   
<b>POST</b>	/quiz/{quiz_id}/toggle-share	Přepnutí stavu sdílení kvízu	   
<b>POST</b>	/quiz/{quiz_id}/copy	Vytvoření kopie existujícího kvízu	   
<b>PUT</b>	/quiz/{quiz_id}/update	Aktualizace existujícího kvízu	   
<b>GET</b>	/quizzes	Vyhledávání a filtrování dostupných kvízů	   
<b>GET</b>	/get_existing_questions	Vyhledávání a filtrování existujících otázek	   
<b>GET</b>	/quiz/{quiz_id}	Získání kompletního kvízu podle ID	   
<b>DELETE</b>	/quiz/{quiz_id}	Smazání existujícího kvízu	   

## Nedokončené kvízy

Endpointy pro správu konceptů a rozpracovaných kvízů

^

<b>POST</b>	/unfinished_quizzes	Uložení nebo aktualizace konceptu kvízu	   
<b>GET</b>	/unfinished_quizzes	Získání seznamu nedokončených kvízů	   
<b>GET</b>	/unfinished_quizzes/{identifier}	Získání detailu konkrétního nedokončeného kvízu	   
<b>DELETE</b>	/unfinished_quizzes/{identifier}	Odstrannění konceptu nedokončeného kvízu	   

Obrázek 3.26: Dokumentace webové aplikace SwaggerHub, která zobrazuje všechny navržené endpointy pro skupiny „Správa kvízů“ a „Nedokončené kvízy“, včetně významu každého endpointu a použitých HTTP metod.

The screenshot shows a REST API endpoint for changing a player's name. The endpoint is `POST /change_name`. The description states: "Umožňuje hráči změnit své jméno během čekání v lobby před zahájením hry. Ověřuje, zda je nové jméno jedinečné, splňuje požadavky na délku a zda je původní jméno registrováno v systému. Po úspěšné změně aktualizuje stav hry a informuje ostatní připojené klienty." Below the description, there is a "Parameters" section which is currently empty. A "Try it out" button is available. The "Request body" is marked as required and has a type of "application/json". An example value is provided: 

```
{
  "old_name": "string",
  "new_name": "string"
}
```

. The "Responses" section includes two entries: a 200 status with the description "Úspěšná změna jména" and a 400 status with the description "Chyba při změně jména". Both responses have a note "No links".

Obrázek 3.27: Na snímku obrazovky lze vidět příklad zobrazení podrobností o vybraném endpointu. V tomto případě se jedná o endpoint pro změnu jména hráče při čekání na spuštění kvízu. Je zde povinné tělo, které obsahuje staré a nové jméno ve formátu JSON. Nakonec jsou zde zobrazeny možné odpovědi, které mohou být odeslány klientovi po zpracování požadavku. Tyto odpovědi obsahují HTTP návratový kód a zprávy ve formátu JSON.

Na následujících tabulkách je popsán obecný návrh několika klíčových Socket.IO událostí. Jejich názvy se v implementaci nemusí shodovat se zde uvedenými, protože jednotlivé typy otázek mají odlišné vlastnosti. Základní principy komunikace však zůstávají stejné. V první tabulce 3.2 jsou uvedeny události ve směru klient → server, ve druhé tabulce 3.3 naopak ve směru server → klient.

Komunikace prostřednictvím těchto událostí probíhá ve většině případů tak, že hlavní obrazovka odešle na server událost, která následně vyvolá rozeslání odpovídající zprávy všem připojeným hráčům. V některých případech je tok opačný — hráč odešle událost na

server s cílem informovat hlavní obrazovku o své akci. Z tohoto důvodu se v obou tabulkách objevují podobné, případně vzájemně odpovídající události.

Název události	Popis
connect	Automatická událost vyvolaná při úvodním připojení klienta.
disconnect	Automatická událost vyvolaná po odpojení klienta.
join_room	Vytvoření privátního kanálu pro komunikaci s konkrétním hráčem. Používá se pro směrování zpráv.
submit_answer	Obecná událost pro odesílání odpovědí na otázky různých typů. V týmovém režimu můžou mít kapitáni jiné události pro odpovídání.
player_update	Oznámení o změnách stavu hráče, jako je změna jména, nebo tímu.
drawing_update	Přenos aktuálně kresleného obrázku v reálném čase pro kreslící otázky. Obsahuje informace o tazích, barvách a provedených akcích.
time_up	Oznámení o vypršení času na hlavní obrazovce.
phase_transition	Žádost o přechod mezi jednotlivými fázemi u některých herních módů (např. řešení přesmyčky → vybírání místa na mapě).
remote_connected	Oznámení o připojení vzdáleného zařízení (např. TV).
hint_request	Žádost o poskytnutí další nápovědy, či písmena k aktuální otázce.
show_final_score	Žádost o poskytnutí finálního skóre všem připojeným klientům.

Tabulka 3.2: Navržené Socket.IO události pro komunikaci ve směru: klient → server

Název události	Popis
game_state_update	Obecná událost pro aktualizaci stavu hry. Zahrnuje zahájení hry, přechod na další otázku a aktualizace během hry.
answer_feedback	Zpětná vazba k odeslaným odpovědím. Obsahuje informace o správnosti, získaných bodech a statistikách.
round_complete	Oznámení o dokončení kola s výsledky, správnou odpovědí a skóre. Vysílá se po zodpovězení všech hráčů nebo vypršení časového limitu.
drawing_broadcast	Přeposílání aktuálně kresleného obrázku na hlavní obrazovku.
hint_update	Poskytnutí další nápovědy, či písmena hlavní obrazovce.
phase_changed	Oznámení o změně fáze v rámci komplexního herního módu spolu s instrukcemi.
team_update	Aktualizace stavu týmů v týmovém režimu včetně skóre, aktivního týmu, nebo kapitánovy akce.
game_navigation	Instrukce pro přechod mezi hlavními obrazovkami aplikace (např. na finální skóre, další otázku, nebo zpět na hlavní stranu).
remote_status	Informace o stavu připojení vzdáleného zařízení (např. TV).

Tabulka 3.3: Navržené Socket.IO události pro komunikaci ve směru: server → klient

## Kapitola 4

# Architektura a implementace

V této kapitole navazuji na předchozí návrh řešení z kapitoly 3 a je zde ukázáno, jak byly jednotlivé části systému realizovány. Zaměřuju se na technickou architekturu aplikace, způsob spuštění a použití externích nástrojů. Postupně je popsán proces správy kvízů (jejich vytváření, úpravy a sdílení), mechanismus rychlé hry, stejně jako fungování čekací místonosti a příprava ke spuštění samotného kvízu. Zvláštní pozornost je věnována propojení klienta a serveru pomocí WebSocketů, synchronizaci času mezi zařízeními a správě režimů hry.

### 4.1 Architektura a technologie

Aplikace využívá architekturu klient-server, kde klient a server spolu komunikují prostřednictvím HTTP a WebSocketů. Aplikace obsahuje vestavěný webový server, který jsem dříve popisoval v sekci 2.1.1. Server je napsaný v jazyce Python s využitím frameworku Flask, který je přiblížen v sekci 2.1.2. Aplikace také obsahuje klientskou část napsanou v JavaScriptu s využitím knihovny React, která je přiblížena v sekci 2.2.1. Architektura a všechny využité technologie jsou znázorněny na obrázku 4.1.

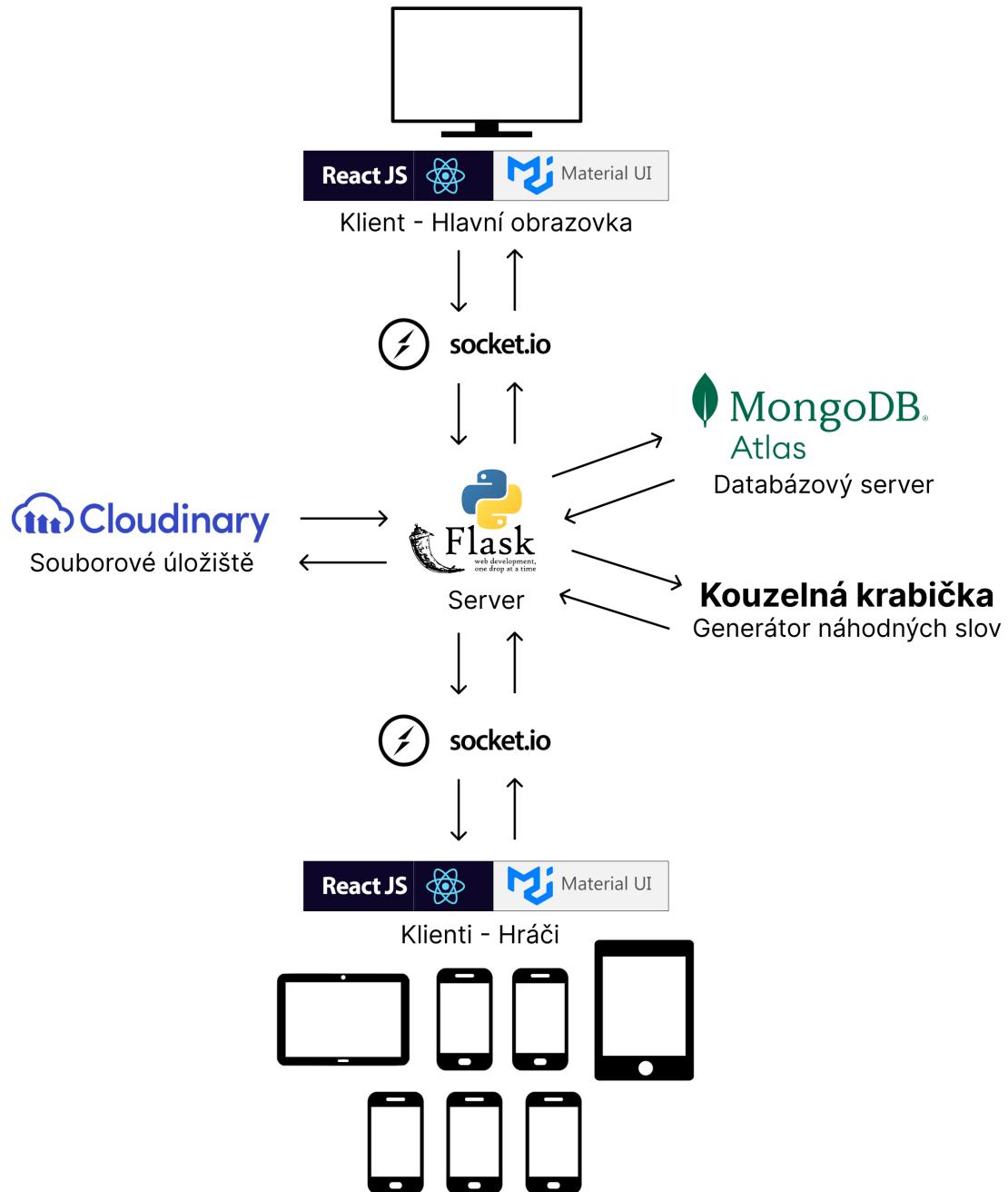
Pro vývoj uživatelského rozhraní s pomocí Reactu jsem využil knihovnu Material UI<sup>1</sup>, která staví na principech Material designu, blíže popsaného v podkapitole 2.2. Tato knihovna poskytuje sadu připravených komponent, jako jsou tlačítka, formuláře nebo dialogová okna, a zajišťuje konzistentní vzhled napříč aplikacemi. Navíc usnadňuje implementaci přepínání mezi světlým a tmavým režimem podle systémového nastavení uživatele, čímž přispívá k modernímu a přístupnému uživatelskému zážitku.

Aplikace je distribuována jako samostatný spustitelný soubor `Home_Quiz.exe` pro Windows, vytvořený pomocí nástroje PyInstaller. Ten umožňuje zabalit Python aplikaci včetně všech potřebných knihoven, Flask frameworku i statických souborů frontendu do jediného balíčku. Po spuštění aplikace se spustí server na pozadí počítače a uživateli se zobrazí okno, ve kterém může zapnout webový prohlížeč s hlavní stránkou aplikace na lokální adrese (typicky `http://127.0.0.1:5000`), a to pomocí knihovny `webbrowser`. Flask je spuštěn v režimu vývoje a PyInstaller se stará i o správu dočasných souborů v systémovém `temp` adresáři. Uživatel tedy může rovnou začít používat aplikaci bez nutnosti přihlašování nebo jiných nadbytečných kroků, což je důležité pro zajištění lepší uživatelské zkušenosti, která je blíže popsána v podkapitole 2.3. Tento distribuční model výrazně zjednodušuje používání aplikace, protože uživatel nemusí manuálně instalovat Python ani žádné závislosti. I přesto, že

---

<sup>1</sup><https://mui.com/>

uživatel má všechno u sebe, je i nadále potřeba internetové připojení pro využívání této aplikace, zejména kvůli komunikaci s databází MongoDB Atlas umístěnou v cloudu.



Obrázek 4.1: Obrázek zobrazuje architekturu klient-server, doplněnou o komunikaci serveru s databází MongoDB, obousměrnou výměnu dat mezi klientem a serverem prostřednictvím WebSocketů, a dále o interakci se službami jako je generátor náhodných slov či úložiště Cloudinary. Klientem může být buď hlavní obrazovka, nebo mobilní zařízení hráčů, protože některé události jsou určeny jednomu z těchto typů klientů.

## Využité technologie pro některé kvízy

Otzázkы typu Kreslení využívají API pro získávání náhodných slov ze stránky Kouzelná krabička<sup>2</sup>, která aktuálně nabízí 3538 jednoslovních podstatných jmen. Tato slova byla přidávána dobrovolníky a následně ověřena vlastníkem stránky. I když nebyla určena pro kvízové hry, jejich jednoduchost je pro tento účel ideální. Navíc hráč vybírá jedno ze tří slov, takže pravděpodobnost výskytu obtížného slova je minimální.

Původně jsem plánoval využít API ze stránky NajdiSlovo.cz, pro kontrolu existence slov ve hře Slovní řetěz, které hledá dané slovo ve dvou slovnících. Nicméně po vyzkoušení jsem došel k závěru, že odpovědi mají příliš dlouhou odevzdu, což bylo nevhodné pro rychlé tempo hry. Místo toho jsem použil statický slovník obsahující přes 260 tisíc českých slov, který publikoval Miroslav Pošta na svém blogu<sup>3</sup>. Slovník vychází z aktualizovaného slovníku Hunspell, používaného například v LibreOffice pro kontrolu gramatiky.

Součástí slovníku jsou dva soubory, jeden s příponou `.dic`, který obsahuje slova s návěstím, a druhý s koncovkou `.aff`, který tato návěští využívá k odvozování různých skloňovacích tvarů. Pro potřeby aplikace jsem převzal pouze soubor s příponou `.dic`, protože obsahuje základní tvary slov bez odvozených tvarů. Tím se zabránil tomu, aby hráči opakovaně zadávali různě skloňovaná slova. Tento slovník je součástí aplikace a uložen v serverové složce `resources`. Oproti původnímu API je kontrola slov tímto způsobem výrazně rychlejší a dokonce rozsáhlejší.

## Databáze

Databáze je hostována na platformě Atlas<sup>4</sup> a využívá NoSQL<sup>5</sup> databázový systém MongoDB<sup>6</sup>. Tuto službu jsem si vybral, protože obsahuje 500 MB úložného prostoru bez časového omezení. Pro testování a počáteční distribuci aplikace je to dostačující, neboť databáze obsahuje pouze kvízy a otázky, které nezabírají mnoho místa. Struktura této databáze byla již blíže popsána při jejím návrhu v podkapitole 3.5.

Komunikace serveru s databází probíhá pomocí knihovny PyMongo. Server nepracuje s tradičními SQL dotazy, ale využívá objektové rozhraní PyMongo pro manipulaci s daty. Hlavní operace zahrnují:

- `find_one()` a `find()` pro vyhledávání záznamů podle zadaných kritérií
- `insert_one()` pro vkládání nových záznamů
- `update_one()` pro úpravu existujících záznamů
- `delete_one()` a `delete_many()` pro mazání záznamů

Databáze je organizována do dvou kolekcí „questions“ a „quizzes“, které fungují obdobně jako tabulky v relačních databázích. Dotazovací jazyk používá filtrační operátory jako `$regex` pro textové vyhledávání, `$nin` pro vyloučení hodnot z výsledků, nebo `$set`

<sup>2</sup><http://slova.cetba.eu/>

<sup>3</sup><http://www.translatoblog.cz/hunspell/>

<sup>4</sup><https://www.mongodb.com/products/platform/atlas-database>

<sup>5</sup>NoSQL (Not Only SQL) je typ databáze, která nevyužívá relační model a je optimalizovaná pro ukládání nestrukturovaných nebo polostrukturovaných dat.

<sup>6</sup>MongoDB je dokumentově orientovaná databáze, která ukládá data ve formátu podobném JSON a je vhodná pro škálovatelné webové aplikace.

pro aktualizaci polí. Pro optimalizaci výkonu je implementováno stránkování pomocí metod `.skip()` a `.limit()` a řazení pomocí `.sort()`. Data jsou ukládána ve formátu BSON, který je binární reprezentací JSON dokumentů.

## Lokální databáze

Pro ukládání lokálních dat aplikace využívá databázi TinyDB, což je jednoduchá NoSQL databáze napsaná v jazyce Python, která zapisuje data do JSON souboru. V aplikaci je inicializována v modulu `local_db.py`, který vytvoří skrytou složku `.homequiz/data` v domovském adresáři uživatele. V této složce se nachází soubor `quiz_database_v2.json` obsahující kolekci `unfinished_quizzes`, jež slouží k automatickému ukládání rozpracovaných kvízů během jejich tvorby.

Práce s lokálním úložištěm je zapouzdřena ve třídě `UnfinishedQuizService`, která poskytuje metody pro správu rozdělaných kvízů. V případě, že se TinyDB nepodaří inicializovat (například kvůli nedostatečným oprávněním k zápisu), aplikace přejde do režimu bez lokálního úložiště. V takovém případě nebude dostupné automatické ukládání při vytváření kvízu a uživatel může přijít o rozdělanou práci při neočekávaném ukončení aplikace.

## Cloudinary

Otzázkы typu *Otevřené odpovědi* mohou v databázi obsahovat odkazy na obrázky nebo audio, které jsou ukládány ve službě Cloudinary<sup>7</sup>, což je specializovaná CDN<sup>8</sup>. Bezplatný tarif zahrnuje 25 kreditů měsíčně, přičemž jeden kredit odpovídá kombinaci 1 GB využitého úložiště, přenosu nebo transformací. Tyto kreditы se každý měsíc obnovují, ale kreditы za úložiště zůstávají vždy utracené podle aktuálně využitého místa.

Komunikace se službou probíhá přes oficiální Python knihovnu `cloudinary`, která zpřístupňuje Cloudinary API. Hlavní operace zahrnují:

- `cloudinary.uploader.upload` pro nahrávání souborů na službu Cloudinary s možností nastavení složky, typu média a dalších parametrů včetně transformací
- `cloudinary.uploader.destroy` pro smazání souboru s využitím `public_id` z jeho URL adresy
- `cloudinary.api.resource` pro ověřování, zda soubor v Cloudinary skutečně existuje

Cloudinary také automaticky provádí transformace obrázků a audia, které snižují jejich rozměry a kvalitu, čímž dochází ke snížení nároků na úložiště. Veškerá interakce s Cloudinary je zapouzdřena v samostatné třídě `CloudinaryService`, což umožňuje jednotné řešení správy médií v celé aplikaci a snadné rozšíření nebo úpravy fungování v budoucnu.

## 4.2 První spuštění aplikace a rychlá hra

Jak už bylo zmíněno výše, uživatel se dostane na hlavní stranu aplikace `DesktopHomePage` přes spustitelný soubor a modální okno. Na obrázku 4.2 lze vidět současný stav hlavní stránky, s tím, že obrázek znázorňuje podporu světlého a tmavého režimu. Při implementaci jsem vycházel z návrhu v sekci 3.4, kde si lze povšimnout téměř identické struktury.

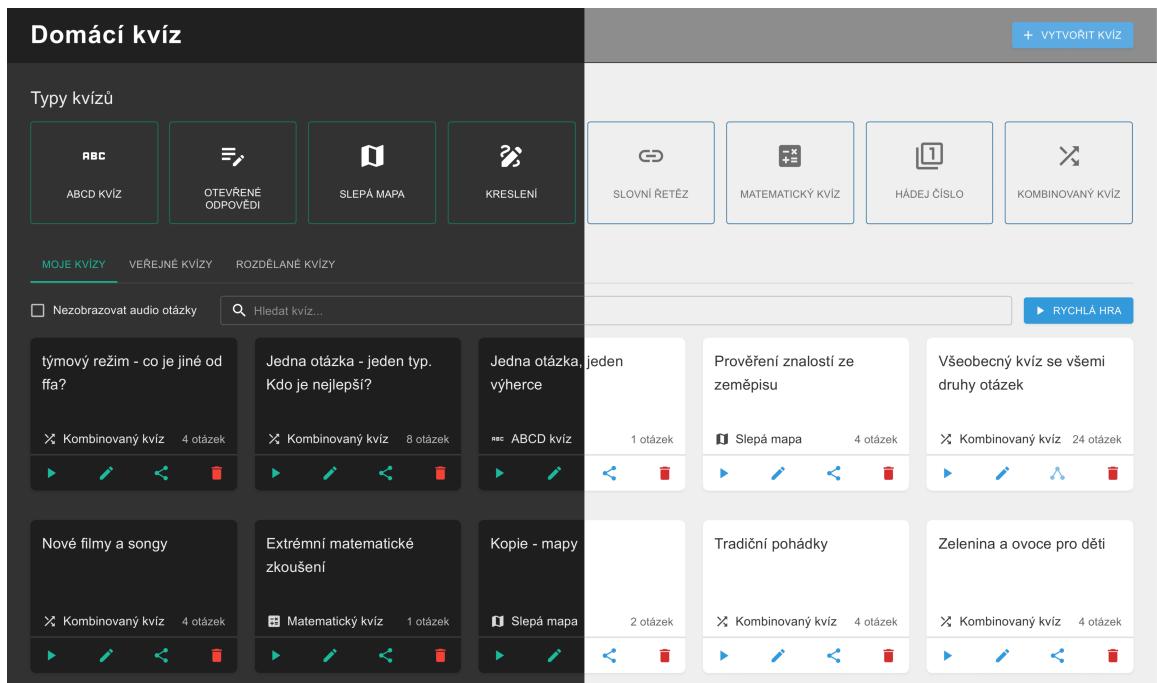
---

<sup>7</sup><https://cloudinary.com/>

<sup>8</sup>CDN (Content Delivery Network) je síť geograficky rozprostřených serverů, která slouží k rychlému a spolehlivému doručování statického obsahu, jako jsou obrázky, videa nebo skripty.

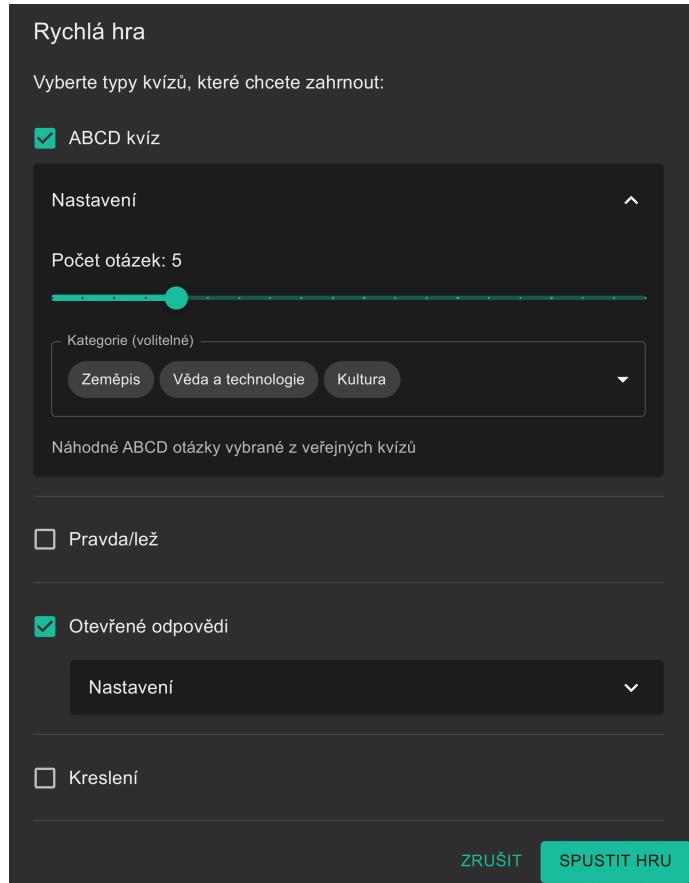
Filtr podle typu kvízu používá komponentu `ToggleButtonGroup` spolu s vlastní komponentou `QuizTypeButton`. Původně plánovanou funkcionalitu pro stahování kvízů jsem nahradil efektivnějším řešením. Díky přidaným záložkám `Tabs` a `Tab` lze nyní využít existující filtr, určený pro vlastní kvízy, i pro kvízy sdílené ostatními uživateli. Původně by bylo nutné řešit tuto logiku zvlášť pomocí vyskakovacího okna, což by bylo zbytečně komplikované. Nyní je možné veřejný kvíz spustit přímo kliknutím na tlačítko nebo si jej zkopirovat pomocí funkce `handleCreateCopy()`. Tím si uživatel vytvoří vlastní verzi, ve které může provádět změny, aniž by ovlivnil původní kvíz.

Karty jednotlivých kvízů `QuizListItem` byly při návrhu protáhlé do šířky, což vedlo k nevyužitému prostoru. Z toho důvodu mají nově čtvercový design, který umožňuje až tři řádky pro název kvízu. Při porovnání obou obrázků 3.18 a 4.2 se zvětšil počet kvízů na stránce dvojnásobně. Karty jsou tvořeny pomocí komponenty `Paper` a strukturovány pomocí `Box`, `Typography` a `Divider` pro oddělení částí. Karty vlastních kvízů obsahují tlačítka `IconButton` pro hraní, editaci, sdílení a smazání. Tato tlačítka jsou doplněna komponentou `Tooltip` pro zobrazení nápovedy. Při stisknutí tlačítka na sdílení se ikonka převrátí a indikuje tak sdílený kvíz. Veřejné kvízy obsahují tedy pouze možnosti pro hraní a vytvoření kopie. Rozdělané kvízy obsahují kvízy, které uživatel na tomto zařízení vytvářel, ale nedokončil je z nějakého důvodu. Ty může buď smazat pomocí funkce `deleteUnfinishedQuiz()`, nebo je dokončit přes funkci `continueUnfinishedQuiz()`.



Obrázek 4.2: Na obrázku je vidět hlavní obrazovka po spuštění aplikace. Díky knihovně MUI je podporován světlý i tmavý režim. Uživatel má k dispozici filtrování kvízů, záložky „Moje kvízy“, „Veřejné kvízy“ a „Rozdělané kvízy“, a také možnosti pro rychlou hru nebo přechod do režimu vytváření kvízu.

Vyhledávání a filtrování kvízů je realizováno pomocí komponenty `TextField` a techniky `debouncing`<sup>9</sup>. Pro notifikace o výsledku akcí jsou využity komponenty `Snackbar` a `Alert`, které poskytují uživateli okamžitou zpětnou vazbu. Pro stránkování kvízů jsem zvolil inkrementální přístup pomocí tlačítka „Zobrazit další“, které načítá další dávku kvízů, jakmile uživatel dosáhne konce seznamu. Tento přístup je uživatelsky přívětivější než tradiční stránkování s čísly stránek, protože umožňuje plynulé procházení obsahu bez přerušení. Tato funkcijsalita je implementována ve funkci `fetchQuizzes()`.



Obrázek 4.3: Na obrázku je vidět přepracované vyskakovací okno pro rychlou hru. Uživatel může vybírat z různých typů kvízů a upravovat jejich specifická nastavení.

Vyskakovací okno `QuickPlayModal` pro spuštění rychlé hry bylo také mírně vylepšeno oproti návrhu na obrázku 3.19b. Původní návrh počítal s tím, že se rychlá hra spustí na základě vybraného typu kvízu z filtru. Aktuální řešení na obrázku 4.3 však funguje samostatně, jelikož nabízí přehled všech typů kvízů s možností přidání do hry pomocí zaškrťávacích políček. Okno je tvořeno dialogovými komponentami z knihovny MUI. Po výběru některého typu kvízu se zobrazí jeho specifické nastavení, jako je počet otázek, kategorie, nebo výběr typu mapy (u `Slepé mapy`). Pro nastavení počtu otázek je použit `Slider` a pro výběr kategorií `Select` s `MenuItem` komponentami. U typů `Kreslení` a `Slovní`

<sup>9</sup>Debouncing je programovací technika, která omezuje frekvenci volání funkcí. V kontextu vyhledávání zajišťuje, že požadavek na server je odeslán až po určité prodlevě od poslední změny v textovém poli (typicky 300-500ms), což zabraňuje nadměrnému zatížení serveru při rychlém psaní.

řetěz se vybírá počet kol a časový limit. Po kliknutí na tlačítko „Spustit hru“ se aktivuje kvíz a uživatel je přesměrován do čekárny.

## Implementace serveru

Vzhledem k absenci uživatelských účtů jsou u jednotlivých otázek a kvízů klíčové databázové atributy `device_id` a `copy_of`, které slouží k určení vlastnictví a identifikaci původu obsahu. Atribut `device_id` reprezentuje zařízení, na kterém byl kvíz s otázkami vytvořen, a `copy_of` obsahuje buď referenci na originální otázku, nebo `null`, pokud je otázka originálem. Nevzniká však strom referencí, protože všechny kopie vždy odkazují přímo na původní otázku. Díky tomu lze například při výběru existujících otázek odfiltrovat duplikáty a zobrazovat pouze originály, což bude blíže popsáno v následující podkapitole 4.3.

Seznam všech API endpointů pro komunikaci s klienty byl představen již při návrhu API v podkapitole 3.6. Následující část se zaměřuje na ty endpointy, které jsou využívány v rámci interakce uživatele s hlavní stránkou aplikace, od získávání seznamu kvízů až po jejich spuštění.

- **GET /quizzes:** Slouží k získání seznamu kvízů s možností filtrování a stránkování. Přijímá parametry jako `filter` (např. „mine“ pro vlastní nebo „public“ pro veřejné kvízy), `type` (např. ABCD, Slepá mapa), `search` pro textové vyhledávání v názvu kvízu, a dále `page` a `per_page` pro stránkování. V rámci třídy QuizService se volá metoda `get_quizzes`, která zajišťuje sestavení MongoDB dotazu podle parametrů a obohacení výsledků o dodatečné informace jako počet otázek nebo přítomnost audio obsahu (to se dále filtruje až na frontendu).
- **POST /quiz/{quizId}/toggle-share:** Přepíná příznak viditelnosti kvízu mezi veřejným a soukromým. V QuizService se volá metoda `toggle_quiz_publicity()`, která ověří, zda je aktuální zařízení (`device_id`) vlastníkem kvízu, a následně přepne příznak `is_public` v databázi.
- **DELETE /quiz/{quizId}:** Slouží ke smazání kvízu. Metoda `delete_quiz()` v QuizService nejprve ověří vlastnictví, poté smaže přidružené mediální soubory, upraví reference všech kopií dané otázky tak, že nejstarší kopie se stává novým originálem a ostatní kopie na ni nově odkazují, následně odstraní všechny otázky daného kvízu z databáze a nakonec smaže samotný kvíz z databáze.
- **POST /quiz/{quizId}/copy:** Vytváří kopii veřejného kvízu. Metoda `copy_quiz()` v QuizService vytvoří nový kvíz s prefixem „Kopie -“, nastaví aktuální zařízení jako vlastníka a zkopiřuje všechny otázky. Pokud již mají referenci `copy_of`, ta zůstává zachována, jinak se vytvoří nová reference na originál. Výsledný kvíz je uložen v MongoDB a klientovi je vráceno jeho ID pro následnou úpravu.
- **GET /unfinished\_quizzes:** Získává seznam rozpracovaných kvízů uložených v lokální databázi zařízení. Zajišťuje to třída UnfinishedQuizService a endpoint se využívá při načítání hlavní stránky nebo přepnutí na záložku „Rozdělané kvízy“.
- **DELETE /unfinished\_quizzes/{identifier}:** Odstraňuje konkrétní rozdělaný kvíz z lokální databáze podle zadанého identifikátoru. Po úspěšném smazání se aktualizuje seznam rozpracovaných kvízů na hlavní stránce.

- **POST /activate\_quiz:** Aktivuje příznak v globálním herním stavu `GameState`, díky kterému se mohou hráči připojit z mobilních zařízení do čekárny. Endpoint je využíván jak po výběru kvízu ke hraní z hlavní stránky, tak při spuštění rychlé hry. Po úspěšné aktivaci je uživatel přesměrován do čekárny pro nastavení a spuštění hry.

### 4.3 Systém správy kvízů

Uživatel se může dostat na stránku `CreateQuizPage` po stisknutí tlačítka „Vytvořit kvíz“ na hlavní stránce. Alternativně se sem dostane při úpravě vlastního nebo nedokončeného kvízu, případně při vytváření kopie veřejného kvízu. V takovém případě je stránka otevřena s existujícím ID vybraného kvízu a zavolá se příslušná funkce pro získání jeho obsahu ze serveru. Rozvržení stránky pro vytváření a úpravu kvízů vychází z návrhu uvedeného v sekci 3.4 a je zachyceno na obrázku 4.4. Oproti návrhu byla mírně upravena horní část, kde jsou ovládací prvky nově umístěny vedle sebe pro efektivnější využití prostoru. Rozdelení na levou formulářovou a pravou náhledovou část zůstalo zachováno.

The screenshot displays the `CreateQuizPage` interface. The left side shows a form for creating a new quiz or editing an existing one. It includes fields for the quiz title ('Název kvízu') and subtitle ('Název kvízu'), a dropdown for question type ('Vyberte typ otázky' with 'Slepá mapa' selected), and a dropdown for map type ('Vyberte typ mapy' with 'ČESKÁ REPUBLIKA' selected). A search bar ('Název města') contains 'Kutná Hora'. Below these are sections for hints ('Přesmyčka') and a map preview ('Umístění vybráno na mapě České republiky'). The right side shows a preview of the quiz ('Náhled kvízu') with four questions. Each question card includes a title ('Otázka 1', 'Otázka 2', 'Otázka 3', 'Otázka 4'), a correct answer ('Správná odpověď'), a hint ('Přesmyčka'), and a list of hints ('Nápovery'). Buttons for 'GENEROVAT PŘESMYČKU' and 'AKTUALIZOVAT OTÁZKU' are located at the bottom of the form and preview sections respectively.

Obrázek 4.4: Na obrázku je vidět obrazovka pro vytváření nebo úpravu kvízu. Uživatel si volí typ otázky, kterou chce tvořit. Levá část zobrazuje formulář pro aktuální typ a pravá část slouží jako náhled kvízu s možnostmi úpravy.

Každý formulář je implementován jako samostatná komponenta ve vlastním souboru pro lepší přehlednost. V následujících bodech je popsána implementace jednotlivých typů otázek včetně změn oproti návrhu. Některé typy otázek nyní nabízejí širší škálu časových limitů (5–120 s) pro větší flexibilitu.

- **Otevřené odpovědi** umožňují volitelně přidat audio nebo obrázek (do 5 MB). Náhled obrázku není zobrazen, pouze název souboru pro ulehčení práce. Soubory se nahrávají na Cloudinary rovnou při vytvoření nebo aktualizaci otázky, aby se předešlo zdlouhavému čekání při finálním ukládání kvízu.
- **ABCD kvíz** je implementován v komponentě `QuestionForm`, která přepíná mezi režimy ABCD a Pravda/Lež pomocí funkce `toggleQuestionType()`. Přepínání řídí stavová proměnná `isAbcd` v nadřazené komponentě. V režimu ABCD jsou k dispozici čtyři textová pole pro odpovědi, v režimu Pravda/Lež jsou odpovědi pevně dané.
- **Slepá mapa** využívá komponentu `BlindMapForm`, která slouží jako hlavní formulář pro tvorbu tohoto typu otázky. Pro správu přesmyček je využita pomocná knihovna `AnagramUtils`, která umožňuje jejich automatické generování se zachováním pozice mezer a zajišťuje kontrolu správného složení. Přepínání mezi mapou Česka a Evropy je realizováno pomocí ovládacího prvku `ToggleButtonGroup`. Samotná mapa je zajištěna komponentou `MapSelector`, která umožňuje uživateli kliknutím na tlačítko pro výběr místa zobrazit vyskakovací okno (`Modal`), ve kterém lze označit pozici města na mapě. Vybraný bod je na mapě vizualizován společně s kruhem, jehož velikost určuje bodovací oblast. Ta se mění podle zvolené obtížnosti (lehká nebo těžká), která je nastavena pomocí parametru `radiusPreset`. Obtížnost lze měnit jak v rámci, tak mimo vyskakovací okno. Formulář umožňuje zadat až tři volitelné nápovědy, které se automaticky přeskupují tak, aby mezi nimi nevznikaly mezery. Takže pokud je vyplňena pouze poslední z nich, přesune se automaticky na první pozici.
- **Kreslení**, implementované v komponentě `DrawingForm`, dozalo oproti návrhu několika změn. Uživatel již nezadává tři slova pro jeden tah, protože s ohledem na možný počet až deseti hráčů by to znamenalo vytvořit třicet slov jen pro jedno kolo. Místo toho jsou slova získávána z externího zdroje, jak je popsáno v podkapitole 4.1. Vzhledem k tomu, že uživatel neví, kolik hráčů bude hru hrát, je posuvník pro výběr počtu kol omezen na hodnoty 1–3, aby hra netrvala příliš dlouho. Pokud chce uživatel hrát Kreslení nebo Slovní řetěz na více kol, může si parametry přizpůsobit v režimu rychlé hry, kde již zná počet hráčů. Časový limit pro tah jednoho hráče má nyní spodní hranici 30 s, protože nelze předvídat obtížnost slova a čas na jeho nakreslení.
- **Slovní řetěz** má vlastní formulář, ale slouží jen jako součást kombinovaného kvízu. Uživatel nastavuje pouze časový limit (20–60 s), počet kol nikoli, protože při více hráčích by hra trvala příliš dlouho a uživatel nemá kontrolu nad počtem hráčů. Hraní Slovního řetězu je doporučené v rámci rychlé hry.
- **Hádej číslo** implementované v `GuessANumberForm`, umožňuje validaci celých i desetinných čísel. Oproti návrhu přibylo rozdělení do kategorií, neboť zde mají také své využití.
- **Matematický kvíz** je implementován v komponentě `MathQuizForm` a umožňuje vytvářet sekvence rovnic. Každá sekvence má vlastní časový limit (5–60 s), zadání rovnice (v číselné nebo slovní podobě) a správnou odpověď. Každá otázka musí obsahovat

vat alespoň tří rovnice, čímž rozšiřuje původní návrh, kde jedna otázka představovala pouze jednu rovnici. **Matematický kvíz** však slouží jako vyřazovací hra, proto musí být všechny rovnice spojeny do jedné otázky. Jednotlivé karty rovnic jsou implementovány pomocí komponenty `SortableSequence`, která využívá funkci drag-and-drop díky knihovně `dnd-kit`. Každá karta zároveň obsahuje návod, jak správně zadávat matematické rovnice (mocniny, odmocniny apod.). Nechybí ani tlačítko pro odstranění sekvence pomocí funkce `handleRemoveSequence()`.

- **Kombinovaný kvíz** již nevyžaduje minimální počet otázek z každého typu a nově umožňuje jejich volnou kombinaci i v libovolném pořadí. Typy **Kreslení** a **Slovní řetěz** mají omezení, že v jednom kvízu mohou být maximálně jednou a zároveň nemohou být použity samostatně. Pořadí otázek není pevně stanoveno a lze jej měnit pomocí drag-and-drop. Pokud se v kvízu nachází více typů otázek (kromě kombinace pouze **ABCD** a **Pravda/Lež**), je automaticky označen jako kombinovaný.

Všechny formuláře využívají společnou logiku validace a zpracování dat, která je centralizována ve funkci `handleAddQuestion()` v komponentě `CreateQuizPage`. V případě úpravy již existující otázky se daný formulář předvyplní daty z proměnné `editQuestion`, přičemž následné zpracování probíhá pomocí funkce `handleEditQuestion()`. Validace vstupů probíhá samostatně v každém formuláři, kde se zaměřuje na specifické atributy daného typu otázky. Při vytváření nového kvízu (nikoli při úpravě již existujícího) je aktivní systém automatického ukládání, které se provádí v pravidelných intervalech (každých 30 sekund), ale také při každé změně v počtu otázek. Správa intervalu je řešena pomocí reference, která umožňuje správné zrušení intervalu při odmontování komponenty, čímž se předejde případnému úniku paměti nebo chybám při opuštění stránky. Uživatel je o automatickém ukládání informován textovým oznámením umístěným v horní části obrazovky.

Na pravé straně obrazovky se nachází náhled vytvořeného kvízu, který je realizován komponentou `QuestionPreview`. Ta zobrazuje seznam všech aktuálně přidaných otázek a umožňuje jejich správu. Každá jednotlivá otázka je zobrazena pomocí komponenty `SortableQuestion`, která ukazuje základní informace, především otázku a správnou odpověď, a poskytuje tlačítka pro úpravu nebo smazání. Otázky lze libovolně přesouvat pomocí drag-and-drop mechaniky.

Pro možnost přidat do kvízu již existující otázky z databáze slouží komponenta `Add-ExistingQuestionDialog`. Ta nabízí rozhraní pro vyhledávání (s využitím techniky de-bouncing), filtrování a výběr otázek. V horní části dialogu se nachází vyhledávací pole, přepínač mezi vlastními a veřejnými otázkami a komponenta `Select` pro filtrování podle typu otázky. Rozhraní je rozděleno na dvě části: v horní části se zobrazují aktuálně vybrané otázky pomocí komponenty `QuestionCard`, zatímco ve spodní části je stránkováný seznam dostupných otázek, které lze dále filtrovat. Otázky, které už byly vybrány, jsou automaticky skryty ze spodního seznamu, aby se předešlo duplicitám. Uživatel si tak může pohodlně vybrat více otázek najednou, zkontovalovat jejich správnost, a až poté je přidat do svého kvízu. Otázky lze navíc rozbalit pro zobrazení detailů (např. správné odpovědi) a vzhled posuvníků je sjednocen třídou `scrollbar-Style`, aby odpovídala celkovému designu.

## Implementace serveru

Pro zpracování jednotlivých typů otázek byl navržen systém využívající návrhové vzory Factory a Strategy. Centrální komponentou je třída `QuestionHandlerFactory`, která spravuje registry handlerů a při prvním použití vytváří jejich instance. Metoda `get_handler()`

vrací správný handler na základě typu otázky, přičemž instance handleru je vždy vytvořena jen jednou a dále opakovaně využívána. Celý proces je zapouzdřen a zajišťuje jednotný přístup k logice jednotlivých typů otázek.

Každý handler implementuje rozhraní definované ve třídě `BaseQuestionHandler`, která slouží jako základ podle vzoru Strategy. Tato třída poskytuje metody pro validaci dat, nebo převod otázek do formátu vhodného pro databázi a front-end. Významnou metodou je také `_determine_copy_of()`, která určuje hodnotu atributu `copy_of` pro správné sledování původu otázek při kopírování. Handlerům je umožněno tyto metody přepsat a doplnit specifické chování, například `MathQuizHandler` přidává při formátování seznam rovnic.

Tato architektura výrazně usnadňuje rozšířitelnost. Přidání nového typu otázky totiž vyžaduje pouze vytvoření nového handleru a jeho registraci ve factory. Všechna data jsou ukládána do jediné kolekce v MongoDB, přičemž specifické atributy jsou spravovány jednotlivými handlery.

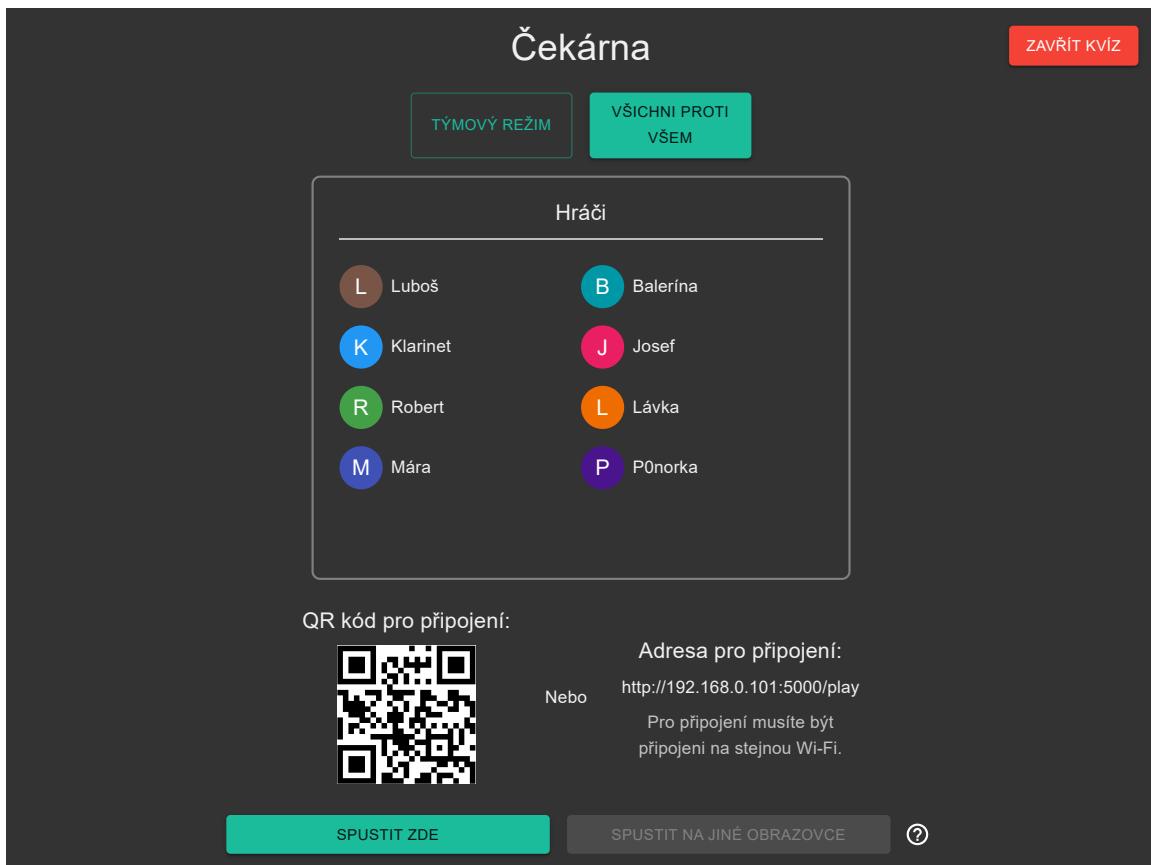
Seznam všech API endpointů pro komunikaci s klienty byl představen již při návrhu API v podkapitole 3.6. V následující části jsou popsány endpointy využívané při tvorbě a úpravě kvízů. Všechny pracují s `device_id` pro určení vlastnictví a s atributem `copy_of` pro správu referencí mezi originálními otázkami a jejich kopiami. Back-endová logika je implementována převážně ve třídách `QuizService` a `UnfinishedQuizService`.

- **POST /create\_quiz:** Endpoint přijímá název kvízu, seznam otázek a typ kvízu. Vytváří nový záznam kvízu v databázi, získá ID a zpracuje otázky pomocí funkce `_handle_quiz_questions()` s využitím odpovídajících handlerů podle typu otázky.
- **GET /quiz/{quizId}:** Načítá existující kvíz pro úpravu. Vrací kompletní data včetně všech otázek a jejich specifických atributů (např. souřadnic pro `Slepou mapu`). Transformace do formátu očekávaného front-endem probíhá v `QuizService`.
- **PUT /quiz/{quizId}/update:** Aktualizuje existující kvíz v databázi. Přijímá upravená data i seznam ID smazaných otázek, aby bylo možné odstranit média z Cloudinary, pokud nejsou použita jinde. Poté se zpracují změny v otázkách včetně mazání nepotřebných médií a na závěr se aktualizuje kvíz i reference `copy_of`.
- **POST /check\_question:** Validuje pouze otázku typu ABCD před přidáním do kvízu. Momentálně probíhá validace jednotlivých typů otázek převážně na front-endu, ale do budoucna je plánováno využití tohoto endpointu při každé úpravě nebo přidání otázky kvůli vyššímu zabezpečení.
- **GET /get\_existing\_questions:** Slouží k vyhledávání existujících otázek pro jejich přidání do kvízu. Lze filtrovat vlastní otázky, nebo veřejné otázky bez duplikátů (pomocí atributu `copy_of` se ukazují pouze originální otázky, případně její veřejné kopie, pokud je originální otázka soukromá). Dále se filtruje podle typu otázky a je zde podpora textového vyhledávání a stránkování. Dynamické typy otázek (`Slovní řetěz`, `Kreslení`) jsou vyloučeny a výsledky jsou seřazeny podle počtu zahrání.
- **POST /unfinished\_quizzes:** Ukládá rozpracovaný kvíz do lokální databáze na zařízení. Třída `UnfinishedQuizService` zajišťuje uložení a vygeneruje unikátní identifikátor pro pozdější načtení.
- **GET /unfinished\_quizzes/{identifier}:** Načítá rozpracovaný kvíz podle identifikátoru. Je volán při návratu k rozdělané práci, kdy front-end obdrží kompletní stav kvízu, jaký byl při posledním uložení.

- **POST /upload\_media:** Zpracovává nahrávání mediálních souborů (obrázky a audio) pro otázky s **Otevřenou odpovědí**. Soubory jsou nahrány prostřednictvím třídy `CloudinaryService` a endpoint vrací odpovídající URL, které jsou později použity při ukládání otázky.
- **POST /delete\_media:** Maže mediální soubory z Cloudinary. Je volán při mazání otázek s médií nebo při nahrazení existujícího média novým. Před odstraněním kontroluje pomocí `CloudinaryService`, zda soubor není využíván jinou otázkou. Pokud ano, ponechá jej a vrátí úspěšnou odpověď.

#### 4.4 Čekací místo a spuštění hry

Jakmile uživatel zapne rychlou hru nebo spustí některý z dostupných kvízů na hlavní stránce `DesktopHomePage`, je přesměrován do čekací místo `RoomPage` s příznakem `isQuickPlayMode`. Tato stránka, zobrazená na obrázku 4.5, se otevírá v režimu celé obrazovky, který je automaticky aktivován po načtení pomocí funkce `enterFullscreen()`. Díky tomu nejsou hráči rušeni prvky prohlížeče a lze naplno využít dostupný prostor.



Obrázek 4.5: Na obrázku je vidět obrazovka čekárny, do které se připojují hráči. Uživatel může přepínat mezi individuálním a týmovým režimem, měnit kapitána nebo přesouvat hráče mezi týmy. Připojení je možné přes QR kód nebo IP adresu. Hru lze spustit přímo nebo na jiné obrazovce s poskytnutým návodem.

Implementace této stránky navazuje na návrh popsaný v sekci 3.4, ale s několika úpravami. Hráči již nepotvrzují připravenost na svém mobilním zařízení, protože tato fáze je koordinována slovní komunikací. V týmovém režimu byla zrušena možnost měnit tým na mobilním zařízení. Správu týmů nyní zajišťuje uživatel na hlavní obrazovce. Právě zde také dochází k první inicializaci IO socketů prostřednictvím funkce `getSocket()` v pomocném modulu `socket.js`, která mimo jiné slouží k synchronizaci času mezi klientem a serverem. K tomu byl implementován mechanismus založený na serverovém čase. Každé zařízení při připojení synchronizuje svůj lokální čas se serverem pomocí funkce `synchronizeTime()`, jež vypočítá časový posun a všechny časové hodnoty následně upravuje funkcí `getServerTime()`. Tento přístup minimalizuje problémy způsobené latencí sítě a zajišťuje, že všechna zařízení zobrazují herní prvky ve stejný okamžik.

V režimu všichni proti všem se v komponentě `PlayersList` zobrazuje tabulka hráčů, přehledně rozdelených do dvou sloupců funkcí `distributePlayersInColumns()`. Komponenta `RoomPage` reaguje na socketové události `player_joined`, `player_left` a `player_name_changed`, které dynamicky aktualizují seznam hráčů. Každý hráč je reprezentován svým jménem a kruhovým avatarem zobrazujícím počáteční písmeno přezdívky a zvolenou barvu, implementovaným komponentou `Avatar` z knihovny MUI.

Při výběru týmového režimu se zobrazuje komponenta `TeamMode`, ve které jsou hráči rozděleni do modrého a červeného týmu. Přepínání týmů je možné pomocí tlačítka vedle každého hráče. Přepnutí týmu má omezení, například není možné odejít z týmu, pokud by v něm nezůstal ani jeden hráč, nebo se přidat do týmu, který už má pět členů. Vedle každého hráče je rovněž tlačítko ve formě hvězdičky, sloužící k označení kapitána týmu. Aktivní může být vždy jen jedna hvězdička v rámci daného týmu a přepínání kapitána je implementováno ve funkci `handleSelectCaptain()`.

Informace o připojení se zobrazují pomocí komponenty `ConnectionInfo`, kde je vygenerován QR kód využitím komponenty z knihovny `qrcode.react`. QR kód i IP adresa směřují na adresu s koncovkou `/play`, která na mobilním zařízení otevírá komponentu `MobileJoinQuizRoom`. Zde hráč zadá přezdívku, vybere si jednu z dostupných 15 barev (automaticky aktualizovaných podle obsazení díky poslechu události `colors_updated`) a připojí se do čekací místnosti. Pokud byl uživatel na tuto stránku přesměrován po dohrání hry, jsou tyto údaje předvyplněny z předchozí hry pomocí `location.state`, aby se zjednodušil proces opětovného připojení a zvýšil uživatelský komfort. Po připojení je přesměrován na komponentu `WaitingRoom`, kde čeká na spuštění hry. Ještě předtím si však může upravit své jméno, pokud si to rozmyslí. Spuštění hry přesměruje hráče na stránku s koncovkou `/mobile-game`, kde hráč vidí komponentu `Loading`, dokud nezačne první kolo.

Kromě spuštění hry na aktuálním zařízení je možné hru spustit i na jiné obrazovce, například na chytré televizi. Vedle hlavního tlačítka pro spuštění hry na jiné obrazovce se nachází komponenta `StartGameTooltip`, která poskytuje uživateli návod, jak na jiném zařízení otevřít stránku `RemoteGamePage`. Stačí do prohlížeče zadat IP adresu s koncovkou `/remote`. Po otevření této stránky je o tom server informován, čímž se na hlavní obrazovce aktivuje možnost spustit hru právě na tomto vzdáleném zařízení. Na vzdálené obrazovce je zatím text, informující, že je potřeba zapnout hru na hlavní obrazovce.

Samotné spuštění hry probíhá odesláním požadavku na server s ID vybraného kvízu nebo konfigurací rychlé hry. V případě úspěchu přijde od serveru událost `game_started`, která spustí přechod na herní stránku s koncovkou `/game`. Režim všichni proti všem vyžaduje minimálně dva a maximálně deset hráčů. V týmovém režimu musí být týmy alespoň 2v2 a nejvíce 5v5. Pokud podmínky nejsou splněny, je na to uživatel upozorněn. Jakmile se přes hlavní obrazovku spustí hra na jiné zařízení, objeví se vyskakovací okno s tlačítkem

pro ukončení kvízu, zatímco na vzdálené obrazovce začne hra s odpočítáváním pomocí komponenty `GameCountdown`. Na televizi nebo jiném zařízení proto není nutné už na nic klikat nebo něco nastavovat. V případě spuštění hry na aktuální obrazovce se objeví odpočítávání.

## Implementace serveru

Centrálním bodem serverové části aplikace je třída `GameState`, která udržuje kompletní stav hry jako singleton. Obsahuje kolekce hráčů, jejich barev, týmové přiřazení, kapitány, skóre, aktuální otázku, informace specifické pro daný typ otázky a spoustu dalších atributů. Při spuštění hry jsou do této instance načteny všechny otázky podle zvolené konfigurace.

Pro obsluhu čekací místnosti, připojování hráčů a zahájení hry je využita knihovna `Socket.IO`, která umožňuje komunikaci v reálném čase mezi serverem a klienty (podrobnosti jsou uvedeny v sekci [2.4.2](#)). Serverová logika je rozdělena na API endpointy a `Socket.IO` události. Přehled všech API endpointů a událostí pro komunikaci s klienty byl představen již v návrhové části v podkapitole [3.6](#). Následující seznam shrnuje jednotlivé související API endpointy, jejich účel a události odesílané klientům.

- **GET /server\_ip:** Získá a vrátí IP adresu serveru ve formátu JSON, sloužící pro zobrazení v připojovacích údajích. IP adresa v lokální síti je určena pomocí funkce `get_local_ip()`.
- **GET /server\_time:** Poskytuje aktuální serverový čas v milisekundách pro časovou synchronizaci mezi zařízeními. Tento endpoint je klíčový pro zajištění, že všechna zařízení pracují se stejným časovým základem při zobrazování otázek a počítání časových limitů. Je zavolán při prvotní inicializaci socketu na klientovi a následně každých 5 minut pro udržení přesnosti.
- **GET /available\_colors:** Vrací seznam aktuálně dostupných barev, které si hráč může vybrat při vstupu do hry. Seznam je omezen na barvy, které nejsou ještě obsazeny ostatními hráči. Endpoint je volán při načtení stránky `/join`.
- **POST /join:** Zpracovává požadavek na připojení nového hráče. Validuje délku přezdívky (3–16 znaků) a kontroluje její unikátnost. Pokud je vstup validní a vybraná barva volná, je hráč přidán do globálního herního stavu `GameState` a je mu přiřazena vybraná barva. Server poté vyšle událost `colors_updated` pro aktualizaci barev na ostatních zařízeních a událost `player_joined` pro zobrazení nového hráče na hlavní obrazovce.
- **POST /change\_name:** Umožňuje hráčům změnit svou přezdívku během čekání na spuštění hry. Nové jméno je validováno a v případě úspěchu aktualizováno v `GameState`, přičemž se zároveň odešle událost `player_name_changed` na hlavní obrazovku.
- **POST /start\_game:** Spouští hru na základě zadанé konfigurace. Endpoint přijímá parametry jako režim hry (týmový nebo všichni proti všem), týmové přiřazení, kapitány a v případě rychlé hry také specifickou konfiguraci typů otázek. Server načte všechny potřebné otázky a inicializuje herní stav. Otázky se budou načítat ze zvoleného kvízu, nebo se v režimu rychlé hry generují dynamicky pomocí modulu `question_generator.py`, který poskytuje funkce jako `generate_random_open_answer_questions()` pro jednotlivé typy. Generátor podporuje filtrování podle kategorií, počtu otázek a mapy ve `Slepé mapě`. Pokud databáze neobsahuje dostatek odpovídajících otázek, systém automaticky vypne filtrování podle kategorie, aby bylo

možné hru i tak spustit. Generátory pro typy `Kreslení` a `Slovní řetěz` však fungují odlišně, protože nepracují s otázkami uloženými v databázi. Tyto generátory jsou blíže popsány v podkapitole 5.4. Po úspěšné inicializaci je herní stav uložen a klientům jsou rozeslány příslušné události (`game_started`, `game_started_remote` a `game_started_mobile`), které obsahují mimo jiné časový údaj pro synchronizovaný začátek hry.

- **POST /reset\_game:** Vrátí herní stav `GameState` do výchozího nastavení, odstraní všechny hráče a připraví server pro novou hru. Endpoint je volán při návratu do hlavní nabídky z čekárny, nebo po ukončení hry. Všichni klienti jsou informováni událostí `game_reset` a vráceni do čekací místnosti.

V následující části jsou popsány základní Socket.IO události, implementované v modulu `base_events.py`, které zajišťují real-time komunikaci mezi zařízeními a na které server naslouchá.

- **connect:** Základní událost vyvolaná při každém navázání spojení mezi klientem a serverem. Server registruje připojení a přiřadí unikátní identifikátor relace (session ID) pro další komunikaci pomocí Socket.IO událostí.
- **disconnect:** Základní událost, která signalizuje, že došlo k přerušení spojení klienta se serverem. Tato událost je automaticky vyvolána při jakémkoli přerušení spojení. Implementace pouze vyčistí serverovou relaci a záměrně nijak nemanipuluje s herním stavem v `GameState`. Je to z důvodu, že Socket.IO připojení může být dočasně přerušeno a znova navázáno (například při slabém Wi-Fi signálu), a přímé odstranění hráče by mohlo způsobit jeho nechteme vypadnutí ze hry.
- **join\_room:** Přiřadí hráče do virtuální místnosti se stejným názvem jako jeho přezdívka. To serveru umožňuje posílat cílené zprávy konkrétním hráčům prostřednictvím funkce `emit()` s parametrem `room=player_name`.
- **player\_leaving:** Událost explicitně oznamující, že hráč opustil hru. Vyvolává ji prohlížeč při události `beforeunload` (např. při zavření nebo obnovení stránky). Využívá se pouze v čekárni, kde nelze hráče ručně odstraňovat ze hry. Při obsluze této události server obratem odešle událost `player_left` na hlavní obrazovku a zajistí, že se hráč nezobrazí vícekrát, pokud by se připojil znovu pod jiným jménem. Bez této události by původní hráč zůstal ve hře bez ovládání, což by zpomalilo její průběh, nebo by bylo nutné restartovat čekací místnost.
- **is\_remote\_connected:** Dotaz hlavní obrazovky na stav připojení vzdálené obrazovky pomocí stejnojmenné události.
- **remote\_display\_connected:** Potvrzení, že vzdálené zobrazení je aktivní. Stejnojmenná událost je odeslána na hlavní obrazovku, kde umožní aktivaci tlačítka pro spuštění hry na vzdálené obrazovce.

# Kapitola 5

## Implementace herních mechanik

V této kapitole se zaměřuji na technickou stránku hratelnosti a jednotlivých herních mechanik, přičemž navazují na předchozí návrh řešení z kapitoly 3. Nejprve popisují základní systém, na němž jsou postaveny všechny typy otázek a který je umožňuje hrát v rámci kombinovaného kvízu. Následně se zaměřuji na implementaci jednotlivých typů otázek, rozdělených podle obtížnosti a stylu. V závěru kapitoly uvádímo možná vylepšení do budoucna a aktuální omezení implementace.

### 5.1 Společný základ všech typů otázek

V této podkapitole se zaměřuji na správu herního stavu s využitím real-time komunikace a na vysvětlení klíčových endpointů, na které budu dále odkazovat. Dále popisují průběh hry, který je společný pro všechny typy otázek, včetně časování založeného na synchronizovaném čase, jehož princip jsem vysvětlil v předchozí podkapitole 4.4, zakončené zobrazením mezivýsledků a finálních výsledků. V neposlední řadě se věnuji také specifickému systému bodování s konkrétními hodnotami, který jsem již nastínil při návrhu v podkapitole 3.2.

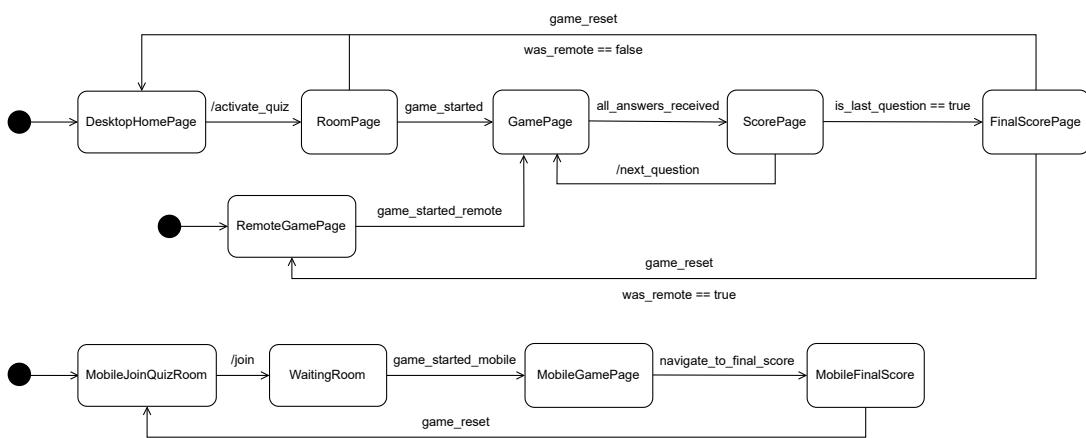
#### Správa herního stavu

Třída `GameState`, částečně představená již v podkapitole 4.4, slouží jako centrální úložiště všech herních dat během aktivní kvízové relace a je využita pro sledování a správu všech aspektů hry. Obsahuje metody jako `reset()` pro návrat do výchozího stavu (zahájení nové relace) a `reset_question_state()` pro vymazání dat specifických pro právě dokončenou otázku, přičemž klíčové informace jako hráči, týmy a skóre zůstávají zachovány. To umožňuje plynulý přechod mezi jednotlivými otázkami. Na tuto logiku navazují i dva klíčové endpointy: `game_start`, podrobně popsaný v podkapitole 4.4, a `next_question`, který zajišťuje přechod na další otázkou a zároveň aktualizuje herní stav.

Při volání `next_question` se nejprve odstraní dočasná data předchozí otázky, zatímco zbytek herního stavu zůstává zachován. U navazujících typů, jako je `Slovní řetěz`, je naopak nutné uchovat některé informace, například pořadí hráčů. U otázek typu `Slepá mapa` a `Matematický kvíz` dochází při přechodu k inicializaci pomocných struktur. Zároveň se určuje, který tým nebo hráč je aktuálně na tahu. Odpověď endpointu obsahuje kompletní objekt nové otázky, informaci o tom, zda jde o poslední otázku v kvízu, identifikaci aktivního týmu a časový údaj pro zobrazení náhledu. U některých typů mohou být součástí odpovědi i doplňující data, například počáteční písmeno nebo slovo u `Slovního řetězu`, případně jméno kreslíře u typu `Kreslení`.

## Průběh hry včetně časování

Během celé hry běží na každé obrazovce časovač, který nelze pozastavit a funguje až do vyčerpání všech otázek. První událost mřená na hlavní obrazovku obsahuje přesný čas, kdy se má spustit náhled první otázky pomocí komponenty `QuestionPreview`. Náhled obsahuje typ otázky a text otázky, aby se hráči mohli připravit před zahájením odpovídání. Na mobilních zařízeních se zobrazuje načítání, dokud nezačne hra. Součástí startovacích událostí je atribut `show_game_at`, který definuje, kdy se má zobrazit samotná herní obrazovka na všech zařízeních. Většinou je to po 5 s, výjimku tvoří typ `Kreslení`, kde je to 8 s, protože kreslící hráč si během náhledu vybírá slovo na kreslení. Na diagramu 5.1 je znázorněna přechodová logika mezi obrazovkami na všech zařízeních.



Obrázek 5.1: Stavový diagram přechodů mezi obrazovkami na všech zařízeních. Šipky znázorňují možné přechody mezi obrazovkami, které jsou vyvolány buď voláním API (označeno prefixem „/“), přijetím události ze serveru, nebo splněním podmínky.

Herní obrazovka `GamePage` používá funkci `renderQuizType()`, která zobrazí konkrétní komponentu na základě typu otázky. Otázka se všemi potřebnými daty je poslána na herní obrazovku při každém přechodu. Zároveň se při přechodu počítá přesný čas ukončení kola na základě aktuálního serverového času a hodnoty `length` následující otázky, která určuje časový limit kola. Výsledný čas je předán ve formě `question_end_time` na herní obrazovku. Univerzální událost `all_answers_received` slouží k ukončení kola na všech zařízeních, ať už v případě vypršení časového limitu, nebo úspěšného dokončení kola. Součástí této události je skóre a různé statistiky.

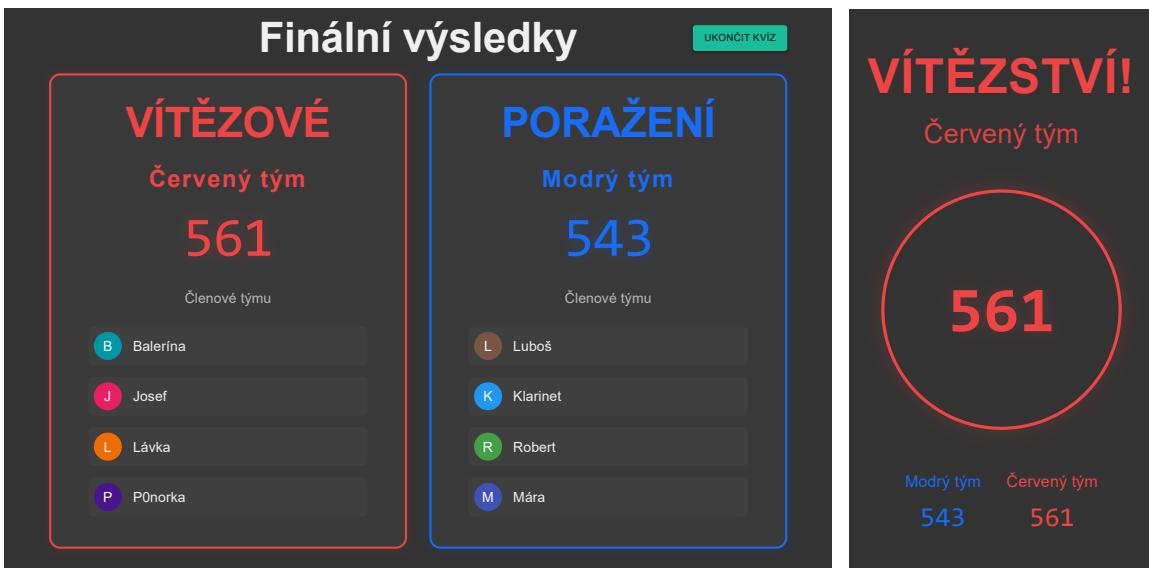
Hráči setrvávají po celou dobu hry v komponentě `MobileGamePage`, přičemž se mění pouze vnitřní obsah. Zatímco hlavní obrazovka počítá čas začátku i konce otázky, mobilní zařízení sledují pouze začátek. Konec otázky je na mobilu rozpoznán výhradně na základě přijetí události `all_answers_received`, která slouží jako globální značka konce kola.

Některé typy otázek, v závislosti na zvoleném režimu (týmový nebo všichni proti všem), probíhají ve více fázích. Každá fáze používá stejný časový limit definovaný atributem `length`, přičemž mezi fázemi je pětisekundová mezifáze. Přesné rozvržení těchto fází je popsáno v příslušných podkapitolách.

Komponenta `ScorePage` na hlavní obrazovce odpočítává 12 s, aby měli hráči dostatek času na prohlédnutí výsledků. Obsahuje komponentu `Leaderboard`, která zobrazuje žebříček všech hráčů v individuálním režimu pomocí funkce `renderPlayer()`, nebo sloupcový

graf v týmovém režimu. Pokud počet hráčů přesahuje výšku žebříčku, aktivuje se automatické scrollování. Žebříček odpovídá vizuálnímu návrhu na obrázku 3.23. Správné odpovědi a statistiky, které se liší podle typu otázky, budou popsány v následujících podkapitolách. Při přechodu do komponenty ScorePage je předán příznak, který značí, zda právě skončilo poslední kolo. Pokud je pravdivý, tak se spustí desetisekundové odpočítávání, po jehož vypršení se aplikace automaticky přepne na poslední obrazovku s celkovými výsledky. Tato změna spustí přechod na finální obrazovku i na mobilních zařízeních. Data pro zobrazení skóre jsou připravena funkcí `get_scores_data()`, která podle režimu formátuje skóre pro všechna zařízení.

Finální obrazovka pro režim všichni proti všem odpovídá návrhu na obrázku 3.24a, s tím rozdílem, že hráči na čtvrtém až desátém místě nyní zobrazují také své avatary, a jsou zarovnáni vedle sebe pod první trojicí. Pokud hrají pouze 2–3 hráči, jejich avatary jsou zvětšeny, aby využily dostupný prostor. První tři hráči jsou nově uspořádáni podle pódiového pořadí: stříbro-zlato-bronz. Na mobilní finální stránce došlo oproti návrhu na obrázku 3.24b k drobné úpravě: jméno hráče je nyní zobrazeno pod jeho avatarem a nad jeho avatarem je jeho umístění, zatímco uvnitř avataru je počet získaných bodů, vše v jeho přiřazené barvě. Finální obrazovky pro týmový režim jsou zachyceny na obrázcích 5.2a a 5.2b. Zobrazuje se zde vítězný i poražený tým (včetně možnosti remízy) spolu s hráči z jednotlivých týmů.



(a) Hlavní obrazovka finálního umístění týmů

(b) Mobilní obrazovka

Obrázek 5.2: Na prvním obrázku je vidět hlavní obrazovka finálního umístění týmů v týmovém režimu a na druhém obrázku je zobrazeno finální umístění týmu daného hráče na jeho mobilním zařízení.

### Bodování podle typu otázky a režimu hry

Systém bodování je důležitou součástí kvízové aplikace, protože přímo ovlivňuje motivaci hráčů i celkový herní zážitek. Bodovací mechanismus je navržen tak, aby zohledňoval různé aspekty výkonu, jako jsou rychlosť, přesnost, týmová spolupráce nebo strategická rozhodnutí, a to v závislosti na typu otázky i zvoleném herním režimu. Přehled bodového ohodnocení jednotlivých typů otázek je uveden v tabulkách 5.1 a 5.2.

<b>Typ otázky</b>	<b>Max. bodů</b>	<b>Mechanismus bodování</b>
ABCD Pravda/Lež Otevřená odpověď	200	100 za správnou odpověď. Rychlejší odpověď přináší více bodů (až 100% bonus).
Hádej číslo	300	Kombinace bodů za umístění (100 bodů odstupňovaných podle umístění a počtu hráčů) a přesnosti: přesná odpověď (200 bodů), do 1% odchylky (150 bodů), do 5% (100 bodů), do 25% (50 bodů).
Slovní řetěz	Proměnlivé	3 body za každé písmeno ve slově. Poslední hráč, který zůstane ve hře, dostane navíc 50 bodů.
Matematický kvíz	75 za rovnici	Polovina bodů za správné řešení, druhá polovina za rychlosť odpovědi. Celkový zisk bodů závisí na počtu vyřešených rovnic.
Slepá mapa	200	Až 100 bodů za správné vyuštění přesmyčky (s odstupňováním podle pořadí a počtu hráčů) a 100 bodů za přesné určení lokace na mapě.
Kreslení	Proměnlivé	Hráč, který kreslí, získává body podle počtu hráčů, kteří správně odpověděli (až 100 bodů) a navíc bonus 50 bodů, pokud uhodnou všechni. Pokud si kreslící hráč nevybere slovo v časovém limitu, dostane pouze polovinu bodů. Hádající získávají 100 bodů za správnou odpověď a mohou získat až 100% bonus podle rychlosti.

Tabulka 5.1: Bodování v režimu všichni proti všem

<b>Typ otázky</b>	<b>Max. bodů</b>	<b>Mechanismus bodování</b>
ABCD Pravda/Lež Otevřená odpověď	200	100 za správnou odpověď. Rychlejší odpověď přináší více bodů (až 100% bonus).
Hádej číslo	300/150	Fáze 1: 300 bodů za přesný odhad týmu. Fáze 2: 150 bodů za správné určení, zda je skutečná hodnota vyšší nebo nižší než odhad soupeře.
Slovní řetěz	200	200 bodů pro tým, jehož protivník nedokáže včas navázat dalším slovem a u něj tak exploduje bomba (vyprší skrytý časový limit).
Matematický kvíz	75 za rovnici	Polovina bodů za správné řešení, druhá polovina za rychlosť odpovědi. Celkový zisk bodů závisí na počtu vyřešených rovnic.
Slepá mapa	200	200 bodů získá tým, jehož kapitán správně určí lokaci na mapě. Pokud žádný tým netrefí přesné místo, získá tým s bližším odhadem 100 bodů.
Kreslení	200	100 bodů pro tým kreslícího, pokud alespoň jeden člen jeho týmu správně uhodne slovo. Rychlejší odpověď přináší více bodů (až 100% bonus).

Tabulka 5.2: Bodování v týmovém režimu

Bodovací systém zohledňuje jednotlivé fáze hry a podporuje různé herní strategie podle typu otázky. Některé otázky probíhají ve více fázích, přičemž každá fáze má odlišnou bodovou hodnotu. Všechny bodové hodnoty jsou definovány v souboru `constants.py` na straně serveru, což umožňuje jejich snadnou změnu. Způsob bodování se liší i podle režimu: v individuálním režimu se hodnotí výkon jednotlivců, zatímco v týmovém režimu hráje roli i spolupráce v rámci týmu a v některých případech i rozhodnutí kapitána. Díky tomu mají jednotlivé otázky odlišnou dynamiku a nutí hráče přizpůsobovat se aktuální herní situaci.

## 5.2 Základní typy otázek

V této podkapitole se věnuji nejjednodušším typům otázek, jejichž implementace vychází z návrhu uvedeného v podkapitole 3.2. Při samotné realizaci jsem však provedl několik úprav s cílem zlepšit přehlednost a využití prostoru. Každý typ nyní obsahuje časovač umístěný v levé části obrazovky a čítač odpovědí v pravé části, což je rozvržení inspirované platformou Kahoot, jak lze vidět na obrázku 3.2. Tento přístup mi umožnil lépe využít centrální oblast obrazovky pro větší komponenty a text otázky, což zlepšuje čitelnost na větších obrazovkách. O aktualizaci čítače odpovědí se stará rodičovská komponenta `GamePage`, která naslouchá události `answer_submitted` a po jejím zachycení inkrementuje čítač.

### 5.2.1 ABCD, Pravda/Lež

Oba typy sdílí stejný základ a mají podobné komponenty jako `ABCDQuiz` a `TrueFalseQuiz` (hlavní obrazovka) a `ABCDQuizMobile`, `TrueFalseQuizMobile` (mobilní zařízení). Mobilní komponenty zobrazují čtyři nebo dvě tlačítka podle typu otázky. Po jejich stisknutí se volá funkce `handleAnswer()` v rodičovské komponentě `MobileGamePage`, která odešle na server událost `submit_answer` s indexem odpovědi a synchronizovaným časem. Server po zpracování odpovědi vrátí klientovi událost `answer_correctness`, podle které se na mobilním zařízení zobrazí jedna ze tří možných obrazovek. Pokud hráč odpověděl správně, zobrazí se komponenta `CorrectAnswer`, která zobrazuje počet získaných bodů a celkové skóre. V případě nesprávné odpovědi se zobrazí `IncorrectAnswer` včetně aktuálního skóre. Pokud hráč nestihne odpovědět v časovém limitu, zobrazí se komponenta `TooLateAnswer`, která ho informuje, že neodpověděl včas, a ukáže mu jeho dosavadní skóre. Tyto obrazovky zůstávají aktivní do začátku náhledu následující otázky.

Na hlavní obrazovce ve `ScorePage` se pod zebříčkem zobrazí komponenty `ABCDAnswers` nebo `TrueFalseAnswers`, které vizuálně zvýrazňují správnou odpověď (pomocí `opacity`) a zobrazují počet hlasujících pro každou možnost.

### Implementace serveru

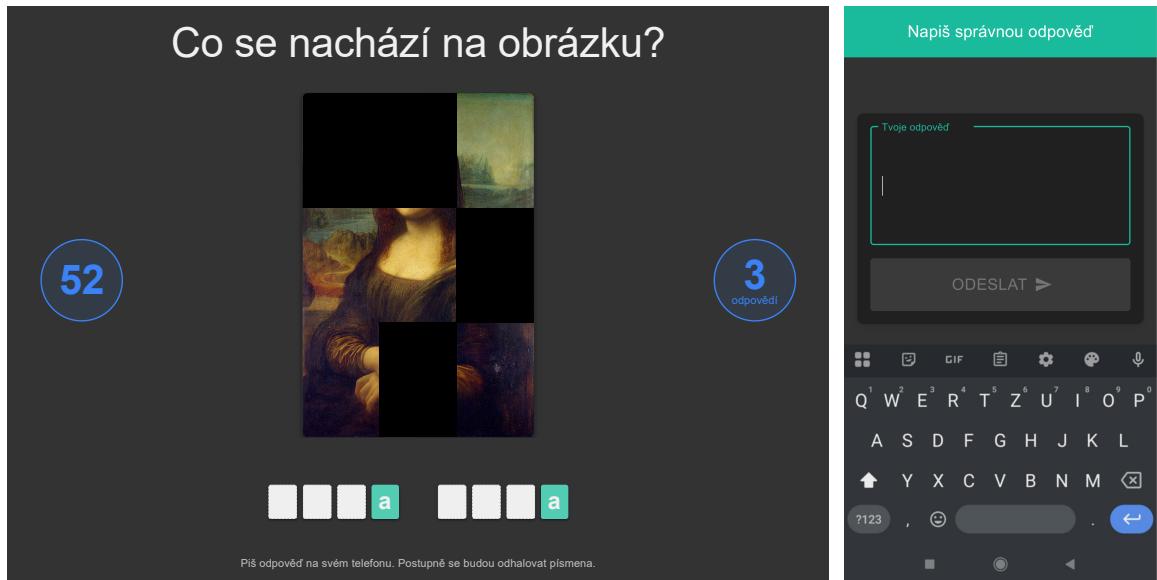
Oba typy kvízů jsou na serveru obsluženy modulem `abcd_events.py`. Hlavní obslužná funkce je `submit_answer()`, která zkонтroluje odpověď hráče, vypočítá bodový zisk na základě správnosti a rychlosti (poměr uplynulého času k časovému limitu otázky, vynásobený 100 body) a aktualizuje herní stav. V týmovém režimu je odpověď jednoho člena rozhodující pro celý tým. Po jejím přijetí server odešle událost `answer_correctness` všem členům týmu, čímž zablokuje ostatní hráče a podpoří týmovou spolupráci.

Po každé odpovědi server vyšle událost pro aktualizaci čítače odpovědí na hlavní obrazovce. Pro případy, kdy vyprší čas a některí hráči neodpověděli, je implementována funkce

`handle_abcd_time_up()`, aktivovaná událostí `time_up` z hlavní obrazovky, která zajistí odeslání výsledků všem klientům.

### 5.2.2 Otevřené odpovědi

Komponenta `OpenAnswerQuiz`, znázorněná na obrázku 5.3a, zobrazuje na hlavní obrazovce rámečky (Paper) reprezentující jednotlivá písmena správné odpovědi, která se postupně odhalují v průběhu času. K tomu využívá pomocné funkce z modulu `letterReveal.js`, které slouží ke strategickému zpřístupňování písmen a zvyšují šanci na úspěšné zodpovězení otázky zbývajícím hráčům. Mechanismus odhalování písmen je založen na dvou klíčových funkcích: `createInitialMask()`, která vytvoří počáteční masku s podtržítky místo písmen (při zachování mezer), a `shouldRevealLetter()`, která určuje, kdy má být další písmeno odhaleno. Algoritmus odhalování je navržen tak, aby začal po 20 % času, skončil před posledními 10 % a zobrazil maximálně polovinu písmen. Tím se zachovává rovnováha mezi nápovodou a výzvou.



(a) Herní stránka na hlavní obrazovce

(b) Mobilní obrazovka

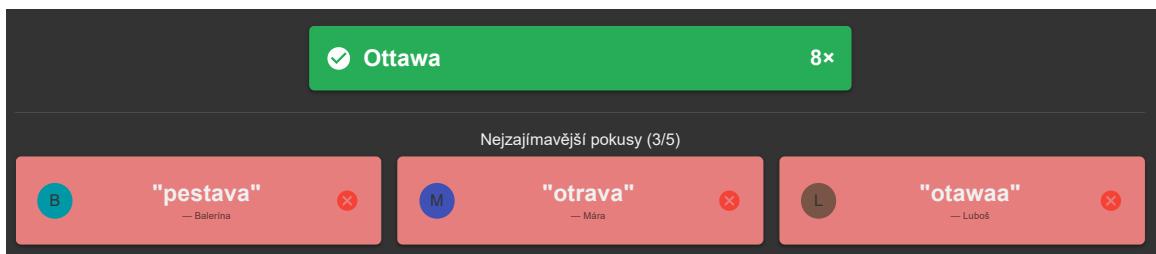
Obrázek 5.3: Na prvním obrázku je vidět herní stránka typu **Otevřené odpovědi** s obrázkem a na druhém obrázku je zobrazena interaktivní obrazovka hráče.

Komponenta podporuje také zobrazení multimediálního obsahu ve formě obrázků nebo zvuků. Obrázky lze zobrazit buď okamžitě (standardní režim), nebo postupně, pomocí komponenty `ImageBlockReveal`. Ta rozdělí obrázek do mřížky  $3 \times 3$  a postupně odkrývá jednotlivé segmenty v náhodném pořadí. Zároveň je zajištěno správné měřítko s ohledem na poměr stran a omezení velikosti (max. 800 px šířky, max. 60 % výšky viewportu (viditelná část webu)). Segmenty jsou tvořeny absolutně pozicovanými komponentami `Box` s černým překryvem, který se postupně zprůhledňuje pomocí CSS přechodů. Odhalování začíná s jedním segmentem a po každých 10 % času se zobrazí další, přičemž celý obrázek je kompletní po 80% trvání otázky.

V případě zvukových otázek se používá HTML5 tag `<audio>` s atributy `autoPlay` a `loop`, který zajistí okamžité a opakované přehrávání. Pokud přehrávání není podporováno, je zobrazena výchozí hláška.

Na mobilních zařízeních zajišťuje komponenta `OpenAnswerQuizMobile` zadávání odpovědí pomocí textového pole a tlačítka pro odeslání, které je aktivní pouze v případě, že pole není prázdné. Po stisknutí se na server pošle událost `submit_open_answer` s textem odpovědi a aktuálním časem. Při nesprávné odpovědi server vrátí událost `open_answer_feedback`, která hráči zobrazí hlášku o délce či podobnosti odpovědi (příliš krátká, dlouhá, nebo téměř správná) a umožní pokus opakovat. V rámci rozvržení je textové pole umístěno nad polovinou obrazovky, aby se nepřekrývalo s klávesnicí, což lze vyzkoušet z obrázku 5.3b. Hlášky o nesprávné odpovědi se zobrazují nad textovým polem, kde mají volný prostor. Pole je zároveň automaticky zaměřeno, takže hráč může začít psát ihned bez další interakce.

Komponenta výsledků `OpenAnswerResult`, jak lze vidět na obrázku 5.4, zobrazuje správnou odpověď a počet hráčů, kteří odpověděli správně. Navíc prezentuje sekci „Nejjednodušší pokusy“, která zobrazuje až tři nejjednodušší nesprávné odpovědi.



Obrázek 5.4: Na obrázku je vidět komponenta využitá u typu Otevřené odpovědi i Kreslení, která se nachází pod žebříčkem v mezikole na hlavní obrazovce. Kromě správné odpovědi jsou zde zobrazeny také tři nejjednodušší odpovědi od hráčů.

## Implementace serveru

Serverová logika Otevřených odpovědí je součástí modulu `open_answer_events.py` kde je hlavní funkcí `submit_open_answer()`, která provádí porovnání hráčovy odpovědi se správnou hodnotou na základě normalizace (převod na malá písmena, odstranění mezer). Pro hodnocení podobnosti nesprávných odpovědí se využívá knihovna `difflib` (konkrétně třída `SequenceMatcher`), která určuje míru shody a poskytuje kontextovou zpětnou vazbu, což umožňuje hráčům dostávat smysluplné nápovědy při nesprávných pokusech.

Postupné odhalování písmen zajišťuje funkce `reveal_open_answer_letter()`, která náhodně vybírá dosud neodhalené znaky (s výjimkou mezer) a přidává je do množiny `revealed_positions`, přičemž jejich počet nepřekročí 50 % celkové délky odpovědi. Server informuje hlavní obrazovku o nově odhalených písmenech prostřednictvím události `open_answer_letter_revealed`.

Součástí logiky je také správa množiny `correct_players` obsahující hráče, kteří již odpověděli správně. V týmovém režimu se otázka považuje za splněnou, jakmile odpoví správně právě jeden hráč z každého týmu, zatímco v individuálním režimu až po správné odpovědi všech hráčů. Ukončení otázky zajišťují funkce `show_open_answer_results()` (v případě splnění) nebo `handle_open_answer_time_up()` (po vypršení času). Obě funkce odesírají výsledky včetně správné odpovědi, bodového zisku a nejjednodušších nesprávných odpovědí, které jsou seřazeny pomocí funkce `sort_player_answers_by_dissimilarity()`. Ta vybírá odpovědi s nejmenší textovou podobností, opět na základě `SequenceMatcher`.

### 5.2.3 Hádej číslo (režim všichni proti všem)

V režimu všichni proti všem komponenta `GuessANumberQuiz` na hlavní obrazovce zobrazuje pouze otázku, časovač a čítač. Hlavní herní mechanika spočívá v co nejpřesnějším odhadu, tedy čím blíže je hráčova odpověď ke správnému číslu, tím více bodů získává. Přesné uhádnutí je navíc odměněno maximálními body a speciální grafikou na mobilní obrazovce.

Mobilní komponenta `GuessANumberQuizMobile` obsahuje číselné vstupní pole optimizované pro mobilní zařízení (s nastavením `type="number"`), které automaticky vyvolává numerickou klávesnici, a odpovědi validuje tak, aby bylo možné odeslat pouze platné číselné hodnoty. Po stisknutí tlačítka se odešle událost `submit_number_guess` s tipem na server.

Po skončení kola se hráčům zobrazí detailní zpětná vazba prostřednictvím komponenty `GuessANumberPlacement`. Hráč musí po odeslání svého tipu vyčkat na ostatní v komponentě `Loading`, protože se bodování provede až na základě všech odpovědí. Následně se na obrazovce objeví informace, mezi které patří: barva pozadí (zelená při přesném zásahu, jinak modrá), textové a vizuální hodnocení přesnosti (například „Přesně!“, „Velmi přesně!“ nebo procentuální vyjádření), srovnání hráčova tipu se správnou odpovědí a zobrazení získaných bodů za přesnost spolu s celkovým počtem bodů.

Komponenta `GuessNumberResult` v mezikole zobrazuje v tomto režimu tři tipy s nejvyšší přesností, seřazené podle vzdálenosti od správné odpovědi. U každého tipu je uvedeno jméno hráče, jeho odpověď a případné zvýraznění přesného zásahu. Hráč s nejbližším tipem je navíc označen štítkem „NEJBLÍŽE“.



Obrázek 5.5: Na obrázku je vidět komponenta typu `Hádej číslo` v režimu všichni proti všem, která se nachází pod žebříčkem v mezikole na hlavní obrazovce. Kromě správné odpovědi jsou zde zobrazeny také tři nejbližší odpovědi od hráčů.

### Implementace serveru

Odpovědi hráčů v tomto režimu zpracovává modul `guess_number_events.py`. Při přijetí tipů funkci `submit_number_guess()` jsou odpovědi uloženy spolu s barvou a identifikací hráče. Po vypršení času nebo odeslání všech odpovědí je spuštěna funkce `handle_all_number_guesses_received()`, která seřadí odhady podle vzdálenosti od správné hodnoty.

Samotné bodování probíhá ve funkci `send_individual_guess_results()`, která každému hráči přiřadí body za pořadí a přesnost. Přesnost se určuje na základě normalizovaného rozdílu mezi tipem a správnou hodnotou, přičemž výsledek je omezen na 100 % a chráněn proti dělení nulou. Přesné či velmi blízké odpovědi získávají bonusové body, jak bylo popsáno v sekci 5.1, a hráč obdrží zpětnou vazbu s textovým/procentuálním hodnocením v rámci události `answer_correctness`. Hráči, kteří nestihnou odpovědět, jsou zařazeni na konec pořadí a dostanou odpovídající zprávu.

## 5.3 Pokročilé typy otázek

V této podkapitole se věnuji složitějším typům otázek, jejichž implementace vychází z návrhů uvedených v podkapitole 3.2. Při samotné realizaci jsem však provedl několik úprav, které budou dále popsány. Časovač a čítač jsou rozvrženy a fungují stejně jako u základních typů otázek v podkapitole 5.2. Co činí tyto typy složitějšími, je to, že obsahují více fází a Matematický kvíz funguje jako vyřazovací hra.

### 5.3.1 Hádej číslo (týmový režim)

V týmovém režimu tohoto typu jsem odstranil mechaniku kola štěstí, která měla původně sloužit jako náhodný přísun bodů ke konci hry. Cílem bylo sjednotit bodování napříč typy otázek, přičemž nevylučuji, že tuto mechaniku bude možné v budoucnu volitelně aktivovat.

Komponenta `GuessANumberQuiz` v týmovém režimu využívá dvoufázový systém, který jej odlišuje od individuálního režimu. V první fázi se na hlavní obrazovce zobrazí otázka a v reálném čase také tipy všech členů aktivního týmu, jak si lze všimnout na obrázku 5.6a. Ve druhé fázi je zobrazen finální tip kapitána prvního týmu, na který druhý tým reaguje hlasováním, zda je správná odpověď větší nebo menší. Výsledky hlasování jsou zobrazovány v reálném čase prostřednictvím čítačů u obou možností, které se aktualizují na základě události `second_team_vote`, aby mohl neaktivní tým také sledovat vývoj hry.



(a) Herní stránka na hlavní obrazovce

(b) Mobilní obrazovka

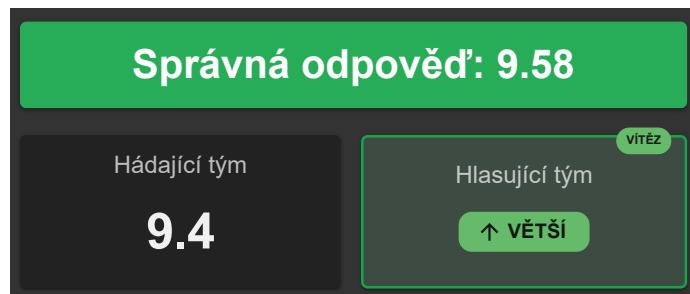
Obrázek 5.6: Na prvním obrázku je vidět herní stránka typu Hádej číslo v první fázi týmového režimu a na druhém obrázku je interaktivní rozhraní kapitána aktivního týmu.

Přechod mezi fázemi zajišťuje komponenta `PhaseTransitionScreen`, která zobrazuje pětisekundový odpočet a informuje o tom, který tým bude v další fázi aktivní. Tato přechodová obrazovka je synchronizována s mobilními zařízeními pomocí přesných časových razítek ze serveru, které značí začátek druhé fáze. Na jejich základě komponenta `GuessANumberQuiz` vypočítá čas zbývající do konce druhé fáze a pokud některý z hráčů nestihne hlasovat, odešle se událost `time_up` pro ukončení kola.

Mobilní komponenty se adaptivně mění v závislosti na aktuální fázi hry a roli hráče:

- **GuessANumberQuizMobile** – Pro běžné členy týmu v první fázi, shodná s individuálním režimem.
- **TeamCaptainGuessNumberMobile** – Speciální rozhraní pro kapitána, který vidí tipy svého týmu (aktualizované událostí `team_guesses_update`) včetně automaticky spočítaného průměru. Kapitán může zvolit průměr, tip konkrétního hráče nebo zadat vlastní odpověď, jak je znázorněno na obrázku 5.6b, pomocí komponenty `RadioGroup`.
- **MoreLessVoteMobile** – Rozhraní pro druhý tým ve fázi hlasování, které umožňuje hlasovat, zda je správná odpověď větší nebo menší než finální tip prvního týmu. Komponenta je podobná jako u typu `Pravda/Lež`, ale zde má hráč možnost měnit svá rozhodnutí, dokud všichni neodpoví, protože tlačítka využívají komponentu `Radio`.
- **TeamWaitingScreen** – Informační obrazovka pro neaktivní tým.
- **PhaseTransitionMobile** – Přechodová obrazovka mezi fázemi s časovým odpočtem a informací o tipu prvního týmu. Zároveň značí, který tým je na řadě.

Komponenta `GuessNumberResult` zobrazuje v mezikole správnou odpověď, tip prvního týmu a výsledky hlasování druhého týmu. Vítězný tým je zvýrazněn zeleným rámečkem a štítkem „VÍTĚZ“. V případě přesného tipu je tato skutečnost navíc vizuálně zvýrazněna. Tuto komponentu lze vidět na obrázku 5.7.



Obrázek 5.7: Na obrázku je vidět komponenta typu `Hádaj číslo` v týmovém režimu, která se nachází pod žebříčkem v mezikole na hlavní obrazovce. Zobrazuje správnou odpověď, tipy kapitánů a označení vítězného týmu.

## Implementace serveru

Týmový režim je na serveru implementován v modulu `guess_number_events.py`. Ve funkci `submit_number_guess()` jsou shromázděny tipy od všech členů aktivního týmu (kromě kapitána) a odeslány kapitánovi pomocí události `team_guesses_update`. Zároveň je odeslána událost `team_guesses_submitted` na hlavní obrazovku pro zobrazení tipů. Poté, co kapitán odešle svůj tip, provede se funkce `submit_captain_choice()`. V případě přesného zásahu získává tým dvojnásobek bodů a hra okamžitě přechází do mezikola. Tuto změnu signalizuje událost `answer_correctness` s příznakem `is_guess_exact`.

Přechod do druhé fáze je řízen událostí `phase_transition` s časovým razítkem. Hráči druhého týmu hlasují prostřednictvím události `submit_more_less_vote`, přičemž server zpracovává jednotlivé volby, umožňuje jejich přepis a průběžně aktualizuje čítače na hlavní

obrazovce. Po obdržení všech hlasů server vyhodnotí výsledek funkcí `handle_all_votes_completed()`. V případě rovnosti hlasů se dává přednost hlasu kapitána. Pokud kapitán nehlasoval, vítězí první tým.

Konečné výsledky jsou odeslány funkcí `send_team_correctness_results()`. Pokud dojde k vypršení časového limitu, server použije speciální logiku definovanou ve funkci `handle_guess_number_time_up()`: v první fázi použije průměr týmových tipů jako finální odpověď, ve druhé fázi vybere odpověď na základě dostupných hlasů.

### 5.3.2 Slepá mapa

Slepá mapa je komplexní typ otázky, který probíhá ve dvou až třech fázích v závislosti na herním režimu. Komponenta `BlindMapQuiz` ukazuje v první fázi přesmyčku názvu města spolu s postupně odhalovanými nápovědami, jak bylo dříve navrženo na obrázku 3.8. Nápovědy jsou uloženy ve třech polích v databázi a zviditelnějí se v určených časových intervalech (po 20 %, 40 % a 60 % trvání otázky) s využitím komponenty `Fade`. Nápovědy jsou načítány na základě události `request_next_clue`, která vyvolá odeslání události `blind_map_clue_revealed` zpět na klienta.

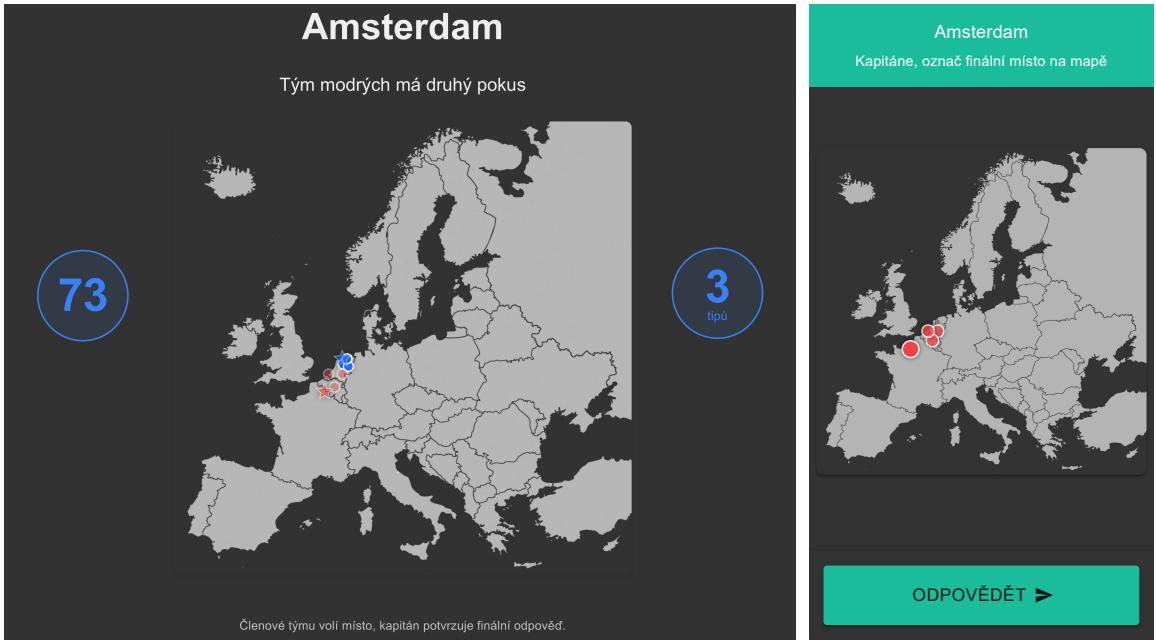
Po dokončení první fáze následuje pětisekundový přechod, realizovaný komponentou `BlindMapPhaseTransition`, synchronizovaný pomocí časových razítek ze serveru. Tato fáze zobrazuje správnou odpověď na přesmyčku a v týmovém režimu informuje o tom, který tým bude ve druhé fázi aktivní.

Ve druhé fázi se chování liší podle herního režimu. V týmovém režimu zobrazuje komponenta `TeamMap` na hlavní obrazovce tipy všech členů aktivního týmu v reálném čase, včetně speciálního označení tipu kapitána hvězdičkou. Funkcionalita `captainPreviews`, aktualizovaná událostí `captain_preview_update`, umožňuje ostatním hráčům sledovat aktuální pozici kurzoru kapitána ještě před jeho finálním potvrzením, což podporuje týmovou spolupráci. Pokud aktivní tým sice správně vyřeší přesmyčku, ale netrefí lokalitu, přechází hra do třetí fáze. Ta je z hlediska funkce i vzhledu stejná jako ta druhá, ovšem s aktivací druhého týmu, který má možnost reagovat. Tipy prvního týmu jsou na mapě v této fázi zobrazeny pro lepší orientaci, jak lze vidět na obrázku 5.8a. V režimu všichni proti všem je druhá fáze minimalistická a zobrazuje pouze název města a počet hráčů, kteří již odpověděli.

Mobilní komponenty se dynamicky přizpůsobují fázi a roli hráče:

- `BlindMapRoleHandler` – Řídící komponenta, která rozhoduje o zobrazené komponentě podle aktuální fáze a týmu. Zajišťuje přechody mezi fázemi a odesílá na server odpovědi na přesmyčku, nebo souřadnice na mapě.
- `BlindMapQuizMobile` – Obrazovka pro první fázi luštění přesmyčky, která je stejná v obou režimech. Funguje stejně jako komponenta pro odpovídání typu `Otevřené odpovědi` ze sekce 5.2.2, ale zahrnuje navíc zelenou informační obrazovku po správném zodpovězení.
- `BlindMapLocationMobile` – Obrazovka pro druhou a případně třetí fázi určování lokace na mapě. Implementuje interaktivní mapu s podporou pozicování a tlačítka pro odeslání odpovědi. V týmovém režimu zobrazuje tipy ostatních členů týmu (díky poslechu události `blind_map_location_submitted`) a umožňuje kapitánům zasílat náhled pozice (posílá událost `captain_location_preview` po každém stisknutí na mapu). Tato funkcionalita je zobrazena na obrázku 5.8b.
- `TeamWaitingScreen` – Informační obrazovka pro neaktivní tým.

- **BlindMapPhaseTransitionMobile** – Přechodová obrazovka mezi fázemi synchronizovaná se serverem, obsahuje specifické informace pro aktuální fázi a režim.



(a) Herní stránka na hlavní obrazovce

(b) Mobilní obrazovka

Obrázek 5.8: Na prvním obrázku je vidět herní stránka typu **Slepá mapa** ve třetí fázi v týmovém režimu zobrazující vybrané lokace hráčů i kapitánů (hvězdičkou) v reálném čase. Na druhém obrázku je zobrazena interaktivní obrazovka kapitána aktivního týmu v 2. fázi, který vidí vybrané pozice spoluhráčů.

Komponenta **BlindMapResult** zobrazovaná v mezikole ukazuje interaktivní mapu s vyznačenými tipy všech hráčů nebo týmů. Barvy označují příslušnost ke konkrétnímu týmu či hráci podle režimu a kapitáni jsou odlišeni ikonou hvězdy. Správná lokalita je znázorněna zeleným kruhem s tolerančním rádiusem podle obtížnosti. Původní návrh počítal i se zobrazením avatarů, ale tato možnost byla zrušena kvůli přehlednosti.

### Implementace serveru

Serverová logika je implementována v modulu `blind_map_events.py` pomocí sady vzájemně provázaných funkcí. Funkce `submit_blind_map_anagram()` zpracovává odpovědi na přesmyčku. V týmovém režimu získává možnost určovat polohu ten tým, který odpoví správně jako první. V režimu všichni proti všem se body přidělují podle pořadí správných odpovědí, počítané funkcí `calculate_anagram_points()`. V obou režimech se odesílá zpětná vazba a přechodová událost.

Ve druhé fázi se odhady pozic zpracovávají funkcí `submit_blind_map_location()`. Správnost je určována pomocí funkce `check_location_guess()` na základě eukleidovské vzdálenosti a tolerance dle obtížnosti (`radius_preset`).

V týmovém režimu rozhodují pouze finální odpovědi kapitánů. Pokud kapitán prvního týmu odpoví správně, tým získá body a kolo končí. V opačném případě přebírá iniciativu druhý tým v rámci třetí fáze. Pokud žádný tým neuhodne přesně, použije se funkce

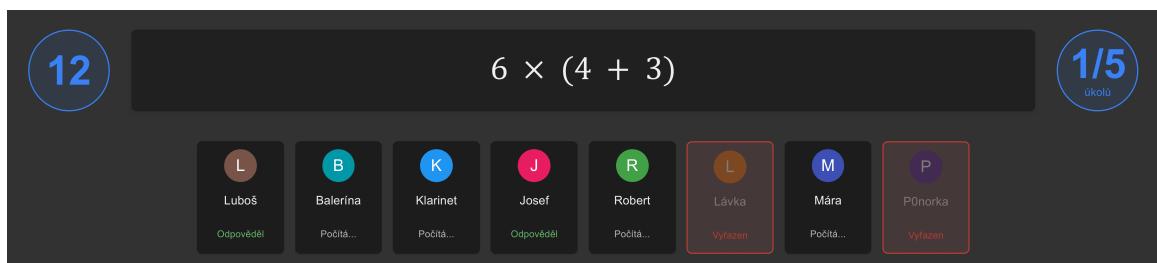
`calculate_distance()` a body získá tým s přesnějším odhadem. Výsledky jsou odeslány prostřednictvím události `answer_correctness`. V režimu všichni proti všem každý hráč získává body podle správnosti svých odpovědí v obou fázích. V případě vypršení času je implementováno ošetření v `handle_blind_map_time_up()`, které řeší různé scénáře včetně situace, kdy žádný tým nevyřešil přesmyčku, nebo kdy oba kapitáni nevybrali finální polohu, a to tak, že se ukončí kolo a nikdo nezíská body.

Zajímavým prvkem je funkce `handle_captain_preview()`, která se zavolá po stisknutí lokace kapitánem v rámci události `captain_location_preview` a umožňuje mu ukázat vlastní pozici na mapě spoluhráčům v reálném čase. Dále server reaguje na událost `request_next_clue` zasláním další nápovedy (`blind_map_clue_revealed`) a při ukončení kola připraví výsledky pro mezikolo, které odešle všem hráčům.

### 5.3.3 Matematický kvíz

Oproti návrhu uvedenému v sekci 3.2 hra končí až po vyčerpání všech rovnic nebo po eliminaci všech hráčů. Body jsou udělovány všem hráčům, kteří odpověděli alespoň na jednu rovnici správně. Původně bylo zvažováno, že body získají pouze ti, kteří přežijí všechny rovnice, případně poslední přeživší, avšak tento přístup byl spíše nespravedlivý.

Na hlavní obrazovce zajišťuje komponenta `MathQuiz` zobrazení jednotlivých rovnic pomocí funkce `renderMathEquation()` z modulu `mathRenderer.js`, která převádí textovou reprezentaci výrazů do čitelného matematického formátu. Funkce správně zobrazuje druhé odmocniny, mocniny a běžné operátory. Stav jednotlivých hráčů je vizuálně indikován pomocí karet ve spodní části obrazovky, které informují o tom, zda hráč již odpověděl, stále přemýšlí, nebo byl vyřazen. Tuto stránku lze vidět na obrázku 5.9. V týmovém režimu jsou hráči rozděleni do dvou barevně odlišených sekcí, což usnadňuje orientaci v průběhu hry.



Obrázek 5.9: Na obrázku je vidět herní stránka typu **Matematický kvíz** v režimu všichni proti všem, která zobrazuje aktuální rovnici, odpočet času a stav hráčů (zda odpověděli, přemýšlejí, nebo byli vyřazeni)

Tento typ kvízu jako jediný využívá knihovnu `useTimer` pro řízení časovače, protože umožňuje dynamické restartování, které je v tomto případě potřeba. Funkce `restart()` umožňuje nastavit nový čas a spustit odpočet s novou hodnotou. Každá rovnice má definovaný časový limit (`length`). Pokud všichni hráči nebo oba týmy odpoví dříve než 3 sekundy před koncem, časovač se zrychlí a restartuje na 3 sekundy, aby hráči nemuseli čekat dlouho na další rovnici, ale zároveň se na ni mohli připravit. Po uplynutí času se odešle událost `math_sequence_completed` a následně server vyšle zpět událost `math_sequence_change`, která inicializuje novou rovnici, včetně nového časovače.

Komponenta `MathQuizMobile` na mobilních zařízeních poskytuje numerické vstupní pole, stejně jako u typu **Hádej číslo** ze sekce 5.2.3, a zpětnou vazbu v podobě barevného

rozhraní. Zelená obrazovka značí správnou odpověď, červená chybnou odpověď nebo vypršení času. V týmovém režimu se správná odpověď jednoho člena vizuálně promítne i spoluhráčům pomocí události `math_quiz_update`, která obsahuje stav každého hráče včetně příznaku `hasAnswered`, který může být aktivován i na základě odpovědi spoluhráče.

Po skončení hry (vyřešení všech rovnic nebo eliminaci všech hráčů) se v mezikole zobrazí komponenta `MathQuizResult`, která shrnuje statistiky, jako je „Matematika kola“ (hráč s nejvyšším počtem správných odpovědí) a „První matematická oběť“. Pro jednotlivé hráče je k dispozici buď komponenta `MathQuizCorrectAnswer` se získanými body, když hráč odpověděl správně na všechny rovnice, nebo `MathQuizEliminatedAnswer` se získanými body a s informací, na které rovnici byl hráč vyřazen.

## Implementace serveru

Serverová část, implementovaná v modulu `math_quiz_events.py`, zajišťuje dynamiku eliminačního systému prostřednictvím sady událostí `Socket.IO` a podpůrných funkcí.

Zpracování odpovědí probíhá v rámci události `submit_math_answer`, kde server porovná odeslanou odpověď s očekávanou hodnotou. Odpovědi jsou předem normalizovány (nahrazení desetinných čárek tečkami, převod na `float`). V případě správné odpovědi server:

- vypočítá počet bodů na základě rychlosti odpovědi pomocí procentuálního využití časového limitu,
- odešle zpětnou vazbu pomocí události `math_feedback`,
- v týmovém režimu informuje celý tým o úspěchu událostí `math_quiz_update`.

V případě nesprávné odpovědi je hráč vyřazen a přidán do množiny `eliminated_players` a obdrží informaci o vyřazení. Po vypršení času na rovnici je vyvolána funkce `handle_sequence_completed()`, která:

- v individuálním režimu vyřadí všechny hráče, kteří neodpověděli,
- v týmovém režimu eliminuje tým, pokud žádný jeho člen neodpověděl správně,
- přechází na další rovnici aktualizací proměnné `current_sequence`, zaznamenáním času začátku nové rovnice a odesláním událostí všem klientům.

Průběžnou aktualizaci stavu klientů zajišťuje funkce `broadcast_math_quiz_update()`, která odesílá současný stav hry všem klientům pomocí události `math_quiz_update`. Tato funkce také kontroluje, zda nastaly podmínky pro urychlení kvízu, jako například eliminace všech hráčů, všech týmů, nebo pokud jeden tým odpověděl správně a druhý byl kompletně eliminován. V takových případech se buď přeskočí na další rovnici, nebo dojde k ukončení hry voláním funkce `handle_math_quiz_completed()`. Výše bodového zisku je určena konstantou `POINTS_FOR_MATH_CORRECT_ANSWER` a dále upravena koeficientem dle rychlosti odpovědi v rozmezí 0,5–1,0. Tímto způsobem jsou zvýhodněni rychlejší hráči, přičemž je zachována celková vyváženosť hry.

## 5.4 Dynamické typy otázek

V této podkapitole popisují typy otázek, které se od ostatních liší zejména tím, že jejich herní obsah není součástí databáze, ale generuje se dynamicky před začátkem každé hry.

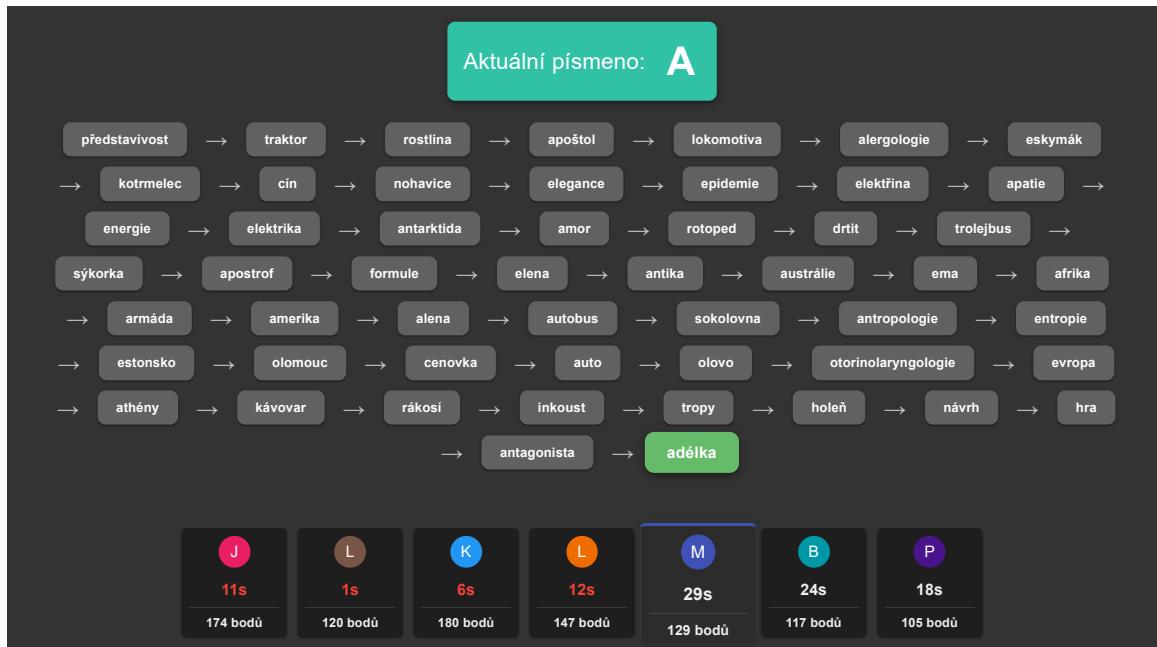
Jedná se zejména o získání slov pro kreslení nebo počátečního slova pro řetěz z externího API, jak bylo popsáno v sekci 4.1. Tyto kvízy lze spustit bez předchozí přípravy ze strany uživatelů, neboť otázky vznikají až za běhu. Implementace navazuje na návrhy uvedené v podkapitole 3.2.

#### 5.4.1 Slovní řetěz

Na hlavní obrazovce je implementována komponenta `WordChainQuiz`. V horní části se zobrazuje aktuální písmeno v barevném kontejneru jako výrazný navigační prvek. Centrální část obsahuje aktuální řetězec slov reprezentovaný komponentami `Chip`, propojenými šipkami. Poslední slovo je zvýrazněno větší velikostí a jinou barvou. Pokud je délka řetězce větší než dostupná výška obrazovky, komponenta automaticky scrolluje, aby bylo poslední slovo vždy viditelné.

V týmovém režimu je vizualizováno střídání hráčů pomocí komponent `Paper` a `Avatar`, přičemž prostřední hráč představuje aktuálního hráče, dva nalevo a dva napravo jsou minulí a nadcházející. Barevné ohraničení (modré a červené) značí týmovou příslušnost. V posledních 40 % kola je aktivován efekt jiskření (`renderSparkles()`) a v posledních 10 % bliká pozadí červeně, čímž je zesílen pocit časové tísně.

V individuálním režimu se místo rotace zobrazují individuální časovače všech aktivních hráčů. Po přijetí události `word_chain_update` se časovač pro aktuálního hráče zastaví a spustí se nový pro následujícího pomocí funkce `startTimeForPlayer()`. Eliminovaní hráči jsou skryti, aby nerušili přehled o aktuálním stavu. Ukázku tohoto režimu lze vidět na obrázku 5.10.

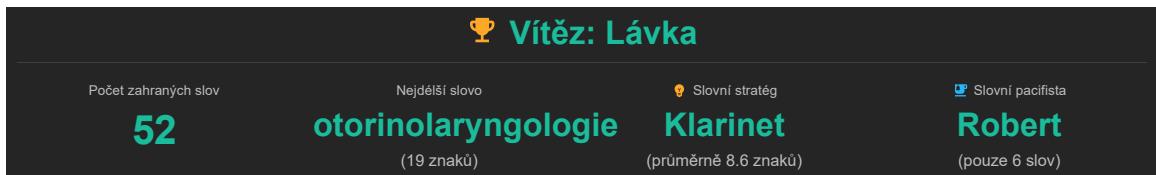


Obrázek 5.10: Na obrázku je vidět komponenta typu `Slovní řetěz` v režimu všichni proti všem, která zobrazuje aktuální hrané písmeno, celý řetězec a hráče s body a zvýrazněného aktuálního hráče.

Mobilní rozhraní se dynamicky přizpůsobuje aktuální fázi hry:

- **WordChainQuizMobile** – Rozhraní pro aktivního hráče, umožňuje zadat slovo a zvýrazňuje počáteční písmeno. Implementuje jednoduchou validaci prvního písmene, zatímco validace diakritiky a existence slova probíhá na serveru. Na základě poslechu události `word_chain_update` se ví, kdo je na řadě. Po vyřazení se zobrazuje červená obrazovka s textem „Vyřazen“.
- **WordChainWaitingScreen** – Obrazovka pro neaktivní hráče, zobrazující aktuální písmeno, jméno hráče na tahu a poslední přidané slovo. Tato komponenta zajišťuje, že i mimo svůj tah mají hráči přehled o průběhu hry.
- **WordChainResult** – Personalizovaná obrazovka výsledků, která na základě funkce `playerRole()` přiřazuje hráči jednu ze čtyř rolí („Slovní strateg“, „Slovní pacifista“, „Vítěz“, „Účastník hry“). Pro každou roli zobrazuje odlišnou barvu pozadí a ikonu, spolu s individuálními statistikami jako počet přidaných slov, celkový počet znaků a průměrná délka slova.

Po skončení hry je na hlavní obrazovce zobrazen souhrnný přehled prostřednictvím komponenty `WordChainResults`, která kalkuluje a vizualizuje různé statistiky související s průběhem hry. Zobrazen je celkový počet zahránych slov, nejdélší zaznamenané slovo a v režimu všichni proti všem také speciální individuální ocenění. Mezi ně patří „Slovní strateg“, který značí hráče s nejvyšší průměrnou délkou slov a „Slovní pacifista“, jenž označuje hráče s nejmenším počtem příspěvků, který však přispěl alespoň jedním slovem. Ukázka statistik režimu všichni proti všem je zachycena na obrázku 5.11.



Obrázek 5.11: Na obrázku je vidět komponenta typu `Slovní řetěz` v režimu všichni proti, která se nachází pod žebříčkem v mezikole na hlavní obrazovce. Jsou zde zobrazeny různé statistiky, které mohou být pro hráče zajímavé.

## Implementace serveru

Serverová logika pro `Slovní řetěz` je implementována v modulu `word_chain_events.py`. Základem je kontrola existence slova v českém slovníku, který je načten do množiny. Při kontrole se využívá funkce `remove_diacritics()`, která převádí některé znaky s diakritikou na jejich základní varianty (např. „í“ → „i“) kvůli nízkému výskytu slov s těmito znaky. Speciální pozornost je věnována písmenům, na která není možné navázat (q, w, x, y, ü). V takových případech se náhodně vygeneruje nové platné písmeno.

Tah hráče je zpracován funkcí `submit_word_chain_word()`, která ověřuje, že: slovo záčíná správným písmenem, má alespoň 3 znaky, nebylo již použito a je obsaženo ve slovníku. Pokud je slovo platné, je přidáno do řetězce, vypočítají se body, nastaví se další hráč a všem klientům je rozeslána událost `word_chain_update` s aktuálním stavem hry.

Správa pořadí hráčů je odlišná podle herního režimu. V týmovém režimu funkce `get_next_player()` střídá hráče mezi týmy a v rámci týmu používá kruhovou rotaci podle

indexu s využitím modulo operace. V individuálním režimu se přeskakují eliminovaní hráči a hra pokračuje podle původního pořadí.

Eliminace hráče probíhá tak, že se přidá hráč do množiny `eliminated_players` a na řadu přichází další hráč. Funkce `check_game_end()` vyhodnocuje konec hry v individuálním režimu (zůstane jeden hráč), zatímco v týmovém režimu hru ukončuje funkce `handle_word_chain_time_up()` po vypršení časovače bomby.

Otzázkы jsou generovány dynamicky před začátkem hry v endpointu `start_game` pomocí funkce `generate_word_chain_questions()` z modulu `question_generator.py`. Ta využívá externí API pro získání náhodného počátečního slova (jejich počet závisí na počtu kol), přičemž kontroluje platnost koncového písmene. Pro případ selhání API je implementována záložní logika, která využívá lokální seznam základních českých slov, aby bylo možné hru spustit i bez externího zdroje. Generátor zároveň inicializuje pořadí hráčů a výchozí stav hry. V týmovém režimu se náhodně volí délka kola v rozsahu 120–240 sekund, aby byl časový limit nepředvídatelný.

#### 5.4.2 Kreslení

Tento typ vychází z návrhu uvedeného v sekci 3.2, přičemž jedinou změnou je odstranění možnosti měnit tloušťku čáry, která nemá na malém displeji moc velké využití. Implementace zahrnuje real-time přenos kresby, systém výběru slov a postupné odhalování písmen, které je identické jako u typu **Otevřené odpovědi** popsánému v sekci 5.2.2.

Komponenta `DrawingQuiz` na hlavní obrazovce zobrazuje v reálném čase kresbu hráče prostřednictvím události `drawing_update_broadcast`, která se spouští pouze při skutečné změně kresby. Tím se předchází blikání a výkonnostním problémům zaznamenaným v raných fázích vývoje. Kreslicí plocha, na které se promítá kreslený obrázek, je rozvržena adaptivně tak, aby vyplnila celou obrazovku, přičemž proporce obrázku odpovídají zařízení kreslícího hráče. Pod kreslicí plochou je umístěna sekce pro postupné odhalování písmen hádaného slova.

Mobilní komponenty poskytují dvě odlišná rozhraní podle role hráče:

- **WordSelectionScreen** – Obrazovka pro výběr jednoho ze tří náhodně vygenerovaných slov kreslícím hráčem. Výběr musí proběhnout v časovém limitu, jinak dojde k bodové penalizaci. Po výběru se slovo odesílá na server událostí `select_drawing-word` a následně je zobrazena maskovaná verze slova na hlavní obrazovce. Kreslíř je přesměrován na `DrawerWaitingScreen`, případně přímo na `DrawerQuizMobile`, pokud již penalizace nastala.
- **DrawerWaitingScreen** – Obrazovka zobrazující vybrané slovo kreslíře v zamaskované podobě, s možností jeho odkrytí, dokud nezačne kolo. Slouží k ochraně proti podvádění nahlížením na mobilní zařízení.
- **DrawerQuizMobile** – Interaktivní rozhraní pro kreslení, implementované s využitím HTML5 `<canvas>`. Obsahuje několik pokročilých funkcionalit:
  - Duální implementace pro dotykové a myši ovládání pomocí optimalizovaných obslužných funkcí, které přesně detekují pozici dotyku/myši s využitím škálovacích faktorů pro různá rozlišení obrazovek.
  - Dynamická správa velikosti plátna s funkcí `updateCanvasSize()`, která zajišťuje konzistentní zobrazení na různých zařízeních a při rotaci obrazovky, včetně zachování nakreslených dat při změně velikosti.

- Výběr nástrojů včetně tužky a výplně, s pokročilým algoritmem `floodFill()`, který implementuje prohledávání do šířky (BFS) pro efektivní vyplňování oblastí s nastavitevnou tolerancí barev.
- Paleta 15 předvolených barev s intuitivním grafickým rozhraním a okamžitou vizuální zpětnou vazbou aktuálně vybrané barvy.
- Správa historie kreslení pomocí funkce `saveToHistory()`, která ukládá stavy plátna pro funkci zpět s optimalizací paměti (omezení na max. 20 stavů).
- Optimalizovaný přenos dat pomocí `sendDrawingUpdate()`, která využívá throttling (omezení četnosti na každých 50ms) a kompresi obrázků (60% kvalita PNG) pro vyvážení mezi kvalitou a síťovým provozem.
- Funkce pro detekci prázdného plátna `isCanvasEmpty()`, která efektivně kontroluje alfa hodnoty pixelů pro rozhodnutí, zda ukládat stav do historie.
- Soukromé zobrazení slova s přepínačem viditelnosti, který umožňuje hráči zkontolovat slovo, aniž by bylo viditelné pro ostatní hráče, kteří by mohli nahlížet na jeho obrazovku.

Komponenta řeší také problémy, jako je nepřesnost dotyku (pomocí scale faktorů), správa paměti (omezení historie) a problém s blikáním při aktualizaci plátna (přenos pouze změněných dat). Z důvodu složitosti a časové náročnosti implementace byla využita asistence GitHub Copilot pro kreslicí plátno a jeho funkce.

- `DrawingAnswerQuizMobile` – Rozhraní s textovým polem určené pro hádající hráče, které funguje obdobně jako u typu **Otevřené odpovědi**.
- `DrawerResultScreen` – Výsledková obrazovka kreslíře, která zobrazuje počet úspěšných hádajících, získané body a vizuální zpětnou vazbu na základě úspěšnosti kresby.

Na hlavní obrazovce v mezikole je použita komponenta `DrawingAnswerResult`, která zobrazuje stejně výsledky jako u typu **Otevřené odpovědi**, jak lze vidět na obrázku 5.4.

## Implementace serveru

Serverová logika je implementována v modulu `drawing_events.py` a je soustředěna kolem tří hlavních oblastí funkcionalit: správa výběru a zobrazení slov, real-time přenos kresby a zpracování odpovědí hádajících hráčů. Výběr slov zajišťuje funkce `select_drawing_word()`, která ověřuje platnost výběru, zaznamenává případnou penalizaci pozdního výběru (příznak `is_late_selection`) a aktualizuje herní stav. Vybrané slovo se odesílá kreslíři v plném znění, zatímco na hlavní obrazovce se zobrazí jeho maskovaná verze.

Real-time přenos kresby je zajištěn funkcí `drawing_update()`, která přeposílá kreslicí data na hlavní obrazovku prostřednictvím události `drawing_update_broadcast` pouze při změně obrazu. Zpracování odpovědí implementuje funkce `submit_drawing_answer()`, která porovnává odpověď se správným slovem. Bodování provádí funkce `calculate_drawer_stats()`, která v individuálním režimu uplatňuje penalizaci za pozdní výběr a uděluje bonus kreslíři, když všichni hráči uhodli jeho kresbu.

Po skončení kola jsou výsledky distribuovány funkcií `show_drawing_results()` nebo `handle_drawing_time_up()` v případě vypršení času. Obě obsahují detailní statistiku pro kreslíře v poli `drawer_stats` a přehled o „nejzajímavějších“ odpovědích hráčů.

Otzádky jsou generovány dynamicky funkcií `generate_drawing_questions()` v modulu `question_generator.py`, která je volána v rámci endpointu `start_game`. Pro každé kolo

vytváří úlohu s trojicí slov získaných z externího API. Výběr kreslíře se řídí režimem hry: v týmovém režimu se střídají týmy rovnoměrně a každý hráč musí kreslit alespoň jednou, zatímco v individuálním režimu má každý hráč příležitost kreslit před opakováním cyklu.

## 5.5 Nedostatky implementace a možná vylepšení

Jedním z hlavních technických omezení současné verze je nedořešené obnovení po odpojení hráče v průběhu hry. Aplikace sice po obnovení stránky nebo výpadku správně zobrazí aktuální herní obrazovku (od další otázky po znovupřipojení) a zachová bodování, ale výsledkové komponenty mohou být nesprávně zobrazeny (např. *TooLateAnswer* i při včasné odpovědi). Toto chování by mělo být možné vylepšit bez zásadního zásahu do fungování aplikace, ale jeho otestování by si vyžádalo více času. Připojení nového hráče v průběhu hry zatím není podporováno, a zejména u typů, kde jsou hráči předem rozděleni do kol, nebo mají předem definované pořadí (*Kreslení* a *Slovní řetěz*) to ani nebude možné bez většího zásahu do struktury kódu.

V počáteční fázi vývoje jsem uvažoval i o možnosti vytváření kvízů a hraní v offline režimu. Vzhledem k tomu, že aplikace využívá přímé spojení s databází, se však ukázalo jako obtížné tuto funkcionality realizovat bez zásadního přepracování architektury. Stejně tak jsem plánoval implementaci automatického ukládání při úpravě kvízů. Problémem však byla kolize mezi autosave objektem ve formátu JSON na klientovi a databázovou verzí otázek, což by vyžadovalo rozsáhlé řešení synchronizace.

Z uživatelského hlediska by mohly být přínosné některé doplňkové funkce, které jsem nestihl realizovat. Například tlačítka v čekací místnosti pro správné nastavení přiblížení, možnost zobrazit při vytváření *Otevřených odpovědí* náhled vloženého obrázku, nebo přidání nápovědy s výpisem typů otázek u kombinovaných kvízů na hlavní stránce. U některých typů by šlo vylepšit vizuální stránku výsledků, například v *Matematickém kvízu* zobrazit v mezikole správné výsledky jednotlivých rovnic.

V oblasti serverové logiky zůstaly nevyužité některé připravené prvky. Například atribut `correct_rate` v metadatech otázek, který by mohl sloužit ke sledování úspěšnosti odpovědí a identifikaci nevhodných či obtížných otázek. Funkcionalita nebyla zahrnuta kvůli nízké prioritě a vyšší náročnosti. Časem jsem zjistil, že by bylo vhodnější vést typ *Pravda/Lež* samostatně, místo jeho sdílení s *ABCD*, což zpětně zkomplikovalo logiku tvorby otázek.

Z pohledu budoucího vývoje se nabízí několik optimalizací. V režimu rychlé hry by bylo možné zabránit opakování již hraných otázek (pokud nebyly vytvořeny na stejném zařízení) tím, že se do lokální databáze budou ukládat ID již odehraných otázek. Lokální úložiště, které si aplikace již dnes vede, by šlo využít i pro zrychlené načítání vlastních kvízů s využitím jejich ID. Tato možnost však zatím nebyla aktivována, protože v databázi není dostatek otázek, aby to přineslo výrazné benefity.

Kromě toho jsem narazil na některá omezení způsobená technickými limity webových prohlížečů. Například spuštění aplikace v režimu celé obrazovky (F11) na vzdálené obrazovce není možné provést automaticky bez přímé interakce uživatele, což může být rušivé při promítání hry na televizi. U některých typů otázek s textovými odpověďmi by také stálo za zvážení přidání možnosti zrušit odhalování písmen jako nápovědy, pokud by to hráčům vadilo. Navíc by se v čekací místnosti v týmovém režimu dala odstranit správa kapitánů, pokud nemají ve vybraném kvízu žádnou speciální roli.

Přestože se jedná o různé drobnosti, aplikace má solidní základ a většinu zmíněných vylepšení je možné doplnit bez zásadního přepracování. Identifikace těchto bodů může sloužit jako odrazový můstek pro další vývoj.

# Kapitola 6

## Testování aplikace

Testování představuje klíčovou fázi vývoje, která slouží k odhalování chyb, ověření funkčnosti a zajištění spolehlivosti výsledné aplikace. V této kapitole se zaměřuji na testování multiplatformní responzivity, uživatelské testování, zpětnou vazbu od testerů a vyhodnocení funkčnosti aplikace.

### Testování multiplatformní responzivity a kompatibility

Důležitou součástí vývoje bylo ověření správného zobrazení a fungování aplikace na různých typech zařízení. Aplikace je navržena tak, aby byla spouštěna na počítači s operačním systémem Windows prostřednictvím spustitelného souboru `.exe`, který inicializuje server a otevře webový prohlížeč s hlavní obrazovkou. Hráči se do hry připojují přes své mobilní telefony, tablety nebo notebooky pomocí prohlížeče, což vyžaduje responzivní rozvržení a bezproblémovou funkčnost napříč zařízeními.

Během vývoje jsem aplikaci testoval na operačním systému Windows 11. K testování front-endu jsem používal prohlížeče Google Chrome a Opera GX s využitím vývojářských nástrojů jako konzole, kam se zapisovaly ladící výstupy přes `console.log()` v JavaScriptu. Oba prohlížeče také umožňují simulovat různá rozlišení obrazovky, což značně usnadnilo ladění responzivity.

Při testování se ukázalo, že aplikace může mít problémy na starších chytrých televizích nebo pomalejších mobilních zařízeních. To je nutné akceptovat jako technologické omezení. Aplikace podporuje světlý i tmavý režim. Většina vývoje probíhala v tmavém režimu, ale po dokončení jsem aplikaci důkladně otestoval i ve světlém režimu. Kromě několika drobných úprav nebyly potřeba žádné zásadní změny a vzhled aplikace zůstal atraktivní v obou variantách.

Testování back-endové části probíhalo převážně ručně pomocí vestavěných nástrojů jazyka Python a prostředí Visual Studio Code. Dále jsem testoval správné ukládání kvízů a otázek do databáze MongoDB a funkčnost nahrávání souborů přes službu Cloudinary.

Jako pilotní testování jsem zorganizoval online hru se svými přáteli, kdy jsme hráli kvízy typu **Kreslení** a **Slovní řetěz**. Sdílel jsem hlavní obrazovku přes platformu Discord, zatímco ostatní hráli na svých zařízeních. Díky veřejné IP adrese bylo možné připojení i z různých domácností. Test proběhl úspěšně, získal jsem první zpětnou vazbu a začal připravovat otázky pro osobní testování. Součástí přípravy bylo i to, že jsem předal aplikaci jedné osobě pro vytvoření vlastního kvízu, který jsme později využili při osobním testování.

## Počáteční zpětná vazba

Z počátečního testování vzešlo několik konkrétních podnětů. Například nebylo možné změnit jméno hráče po připojení do čekací místnosti. Při obnově stránky pro zadání nového jména zůstal původní hráč zapsaný, což vedlo k duplicitám. Tento problém jsem následně opravil. Ve hře *Kreslení* se vyskytl problém s použitím nástroje kbelík, protože kvůli ohraničení plátna nebylo možné kreslit zcela k okrajům a výplň nefungovala správně. Tuto situaci jsem vyřešil úpravou plátna. Na základě zpětné vazby jsem přidal zobrazení typu otázky během náhledu, aby hráči nebyli zaskočeni změnou herního formátu. Dále jsem prodloužil mezikolní obrazovku z 8 na 12 sekund, protože hráči nestíhali včas přečíst výsledky a vyhodnocení odpovědí.

Během testování editoru kvízů se projevil problém s automatickým ukládáním. Při kombinaci úprav a vytváření nových otázek došlo k narušení struktury JSON souboru, což vedlo k nevalidnímu formátu se dvěma kořenovými objekty. Tato chyba znemožnila další načítání kvízu. Problém jsem identifikoval a opravil. Také jsem zjistil, že nebylo možné vkládat soubory kvůli chybějícímu přístupovému klíči ke službě Cloudinary. Po jeho doplnění jsem vytvořil finální .exe soubor, který byl použit pro osobní testování.

## 6.1 Průběh testování s uživateli

Testování hratelnosti proběhlo ve dvou skupinách po šesti účastnících. V jedné ze skupin jsem nebyl přítomen, zatímco v té druhé jsem se účastnil i jako hráč. První skupinu tvořila rodina s různými věkovými kategoriemi, druhou skupinu pak vrstevníci kolem dvaceti let. S vedoucím druhé skupiny jsme se domluvili na jednotném postupu: každý z nás připravil kombinovaný kvíz obsahující všechny typy otázek (jeden s 38 otázkami, druhý s 24), a každá skupina odehrála oba tyto kvízy. Aby byly výsledky relevantní, vedoucí skupiny se účastnil pouze kvízu, který nevytvořil, a tudíž neznał správné odpovědi. Zároveň byl zodpovědný za uvedení do pravidel, spuštění hry a řízení jejího průběhu.

První kvíz byl odehrán v režimu všichni proti všem, který sloužil k seznámení hráčů s aplikací a jednotlivými typy otázek. Druhý kvíz proběhl v týmovém režimu 3v3. První skupina hrála na chytré televizi s využitím funkce aplikace pro zobrazení na vzdálené obrazovce, zatímco druhá skupina použila notebook připojený přes HDMI k velkému monitoru. Všichni hráči byli připojeni ke stejné Wi-Fi síti. Kvízy trvaly přibližně 45 a 30 minut. Po skončení byli všichni účastníci (kromě vedoucích skupin) požádáni o vyplnění připraveného dotazníku zaměřeného na vyhodnocení herního zážitku. Dotazník i jeho výsledky jsou popsány v podkapitole 6.3.

Testování správy kvízů proběhlo samostatně u čtyř účastníků. Šlo především o ověření srozumitelnosti a intuitivnosti práce s formuláři. Testování probíhalo vzdáleně tak, že jsem testerům zaslal spustitelný soubor aplikace a spojili jsme se přes Discord, kde sdíleli obrazovku. Během testu jsem zadával úkoly a žádal je, aby komentovali svůj postup. Zároveň jsem je instruoval, aby se na nic neptali, protože mě zajímalo, zda je rozhraní intuitivní. Úkoly byly následující:

- Spusťte aplikaci a přejděte na stránku pro tvorbu kvízu.
- Vytvořte kombinovaný kvíz s alespoň jednou otázkou od každého typu.
- Změňte pořadí otázek, jednu smažte a jinou upravte.
- Dokončete a zveřejněte kvíz.

Každé testování trvalo přibližně 5 minut. Po dokončení dostali účastníci krátký dotazník se šesti otázkami. Pět z nich mělo podobu škály 1–5 (s krajními body definovanými přídavnými jmény) a v šesté otázce měli možnost volně napsat návrhy na vylepšení. Popsaný dotazník s výsledky je uveden v podkapitole 6.3.

Později jsem poskytl aplikaci kamarádovi, který si chtěl zahrát kvíz s rodinou. Skupina se však nedokázala připojit z mobilních zařízení do čekací místnosti. Aplikaci jsem poté zaslal dalším třem osobám, kde dvěma fungovala bez problémů a jednomu opět ne. Po podrobnějším zkoumání jsem zjistil, že problém způsobovalo bezpečnostní nastavení systému. U uživatelů, u kterých připojení fungovalo, vyskočilo standardní upozornění na povolení komunikace ve firewallu. U ostatních spuštění blokoval antivirus (například program ESET) a žádné firewall okno se nezobrazilo. Z toho důvodu jsme museli ručně povolit aplikaci ve firewallu (ve Windows Defenderu nebo antiviru) a již vše fungovalo správně.

## 6.2 Zpětná vazba uživatelů

Testování přineslo množství odezvy pro možná vylepšení aplikace. Některé návrhy se týkaly samotného herního průběhu, jiné ovládání nebo rozhraní editoru kvízů. Většina nápadů a připomínek byla smysluplná a jejich případná implementace by přispěla ke zlepšení uživatelského zážitku.

Ve skupině, které jsem se neúčastnil, zazněly návrhy na větší kontrolu nad tempem hry. Hráči by ocenili možnost dočasně pozastavit hru, například na mezikolní obrazovce, pokud by si potřebovali odskočit nebo přerušit hru. Podle některých hráčů by mezikolní obrazovka mohla trvat déle (např. 30 sekund), ale s možností jejího přeskočení pomocí tlačítka na mobilu, ideálně na základě hlasování. Dalším požadavkem byla možnost znova se připojit do rozběhlé hry v případě náhodného odchodu (například při stisknutí tlačítka zpět na mobilu). Také by uvítali zobrazení QR kódu a URL adresy pro připojení přímo na vzdálené obrazovce, jelikož v současnosti se zobrazuje pouze na hlavním zařízení. Při výběru slova v kvízu *Kreslení* by si přáli zrušit potvrzovací tlačítko a nahradit jej automatickým výběrem po kliknutí. Dále navrhovali, aby aplikace nevyžadovala diakritiku při psaní *Otevřených odpovědí*, protože její zadávání na mobilu je nepohodlné.

V mé skupině byly požadavky pro zobrazení pravidel k jednotlivým typům otázek, například v čekací místnosti, zejména zda je povolena pouze jedna nebo více odpovědí. Některým hráčům během týmové hry chyběla informace o tom, ve kterém týmu se nacházejí (například v mezikole). Tato informace je aktuálně zobrazena pouze v čekací místnosti a po skončení hry.

Na základě těchto poznatků byla aplikace rozšířena o zobrazení QR kódu a URL adresy také na vzdálené obrazovce a dále bylo do čekací místnosti přidáno tlačítko, po jehož stisknutí se objeví stručná pravidla pro jednotlivé typy otázek v obou režimech.

Při testování editoru kvízů zazněly návrhy na zvýšení pohodlí při tvorbě tématického kvízu. Například předvyplnění kategorie podle předchozí otázky, nebo zachování stejného časového limitu, pokud uživatel vytváří podobné otázky. Také zazněla žádost o možnost odstranit připojený soubor z otázky bez nutnosti mazat celou otázku. Jeden z testerů by uvítal přepínač mezi světlým a tmavým režimem namísto automatického řízení podle systémových preferencí.

### 6.3 Závěrečné hodnocení aplikace

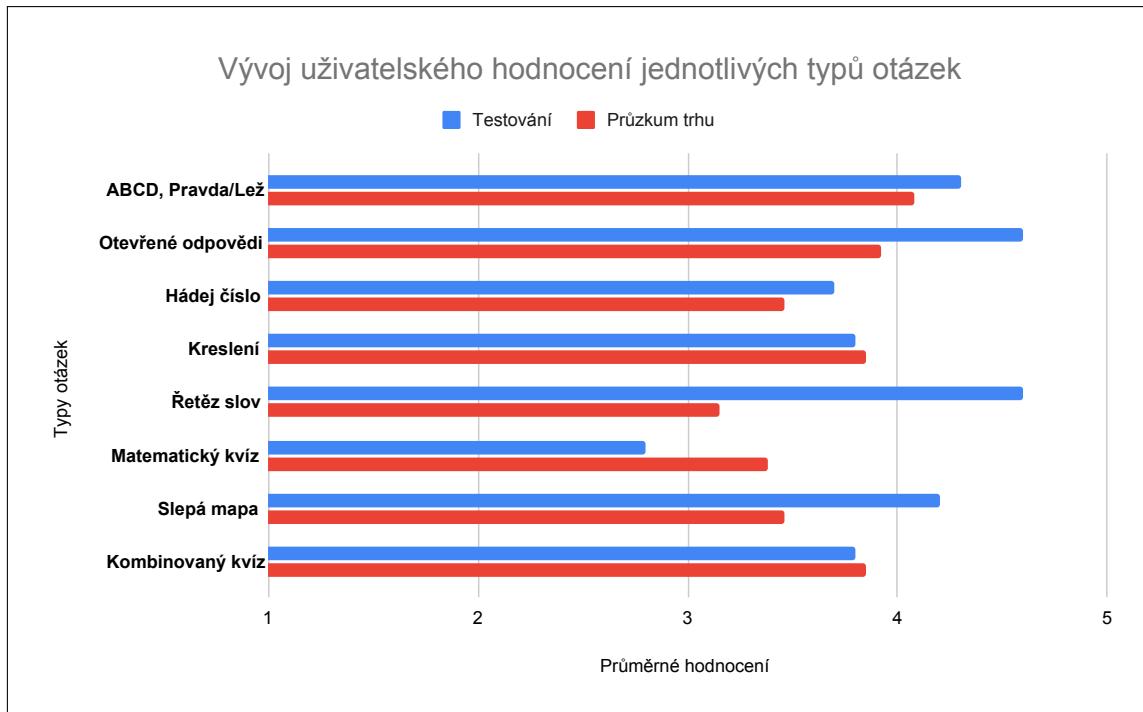
V této podkapitole shrnuji výsledky uživatelských dotazníků a jejich průměrné hodnoty. Dotazníky obsahovaly převážně škálové otázky, u nichž respondenti hodnotili na stupnici od 1 do 5, kde 1 znamenala nejhorší a 5 nejlepší hodnocení. Některé otázky však umožňovaly i textové odpovědi.

Dotazník zaměřený na správu kvízů byl stručný. Respondenti ohodnotili proces tvorby kvízu průměrnou známkou 4,25. Pochopitelnost jednotlivých formulářů získala 4,5 a úpravy otázek i sdílení kvízu obdržely maximální hodnocení. Žádný z respondentů nezaznamenal výraznější problémy.

První část dotazníku k testování hratelnosti byla zaměřena na zjištění předchozích zkušeností respondentů. Ukázalo se, že všichni účastníci mají s kvízy zkušenosti a hrají je alespoň jednou měsíčně, přičemž preferují osobní hraní před online formou. Nejoblíbenějšími typy byly vědomostní kvízy a kreslení, což ukazuje, že šlo o vhodnou testovací skupinu. Kromě již zmíněných platform z průzkumu v podkapitole 3.1 respondenti uvedli i znalost aplikací jako Dobyvatel nebo AZ kvíz.

Další část dotazníku se týkala uživatelského rozhraní. Hodnocení ovládání bez předchozích instrukcí dosáhlo průměrné známky 3,5 (většina ale zvolila 4). Grafický design byl hodnocen pozitivně s průměrem 4,3. Snadnost připojení byla ohodnocena 4,1, což mírně snížily technické potíže na starších televizích a mobilních zařízeních. Jediný zmíněný problém s navigací souvisel s náhodným stisknutím tlačítka „zpět“ na mobilu, kdy se uživatel ocitl zpět v čekací místnosti bez možnosti návratu do hry.

Respondenti dále hodnotili jednotlivé typy otázek z hlediska zábavnosti. Průměrná hodnocení jsou uvedena v grafu 6.1, kde jsou také porovnána s výsledky z průzkumu trhu uvedeného v podkapitole 3.3.



Obrázek 6.1: Graf zobrazující průměrné hodnocení jednotlivých typů otázek z testování srovnанé s výsledky z průzkumu trhu. Stupnice: 1 = nezávavné, 5 = velmi zábavné.

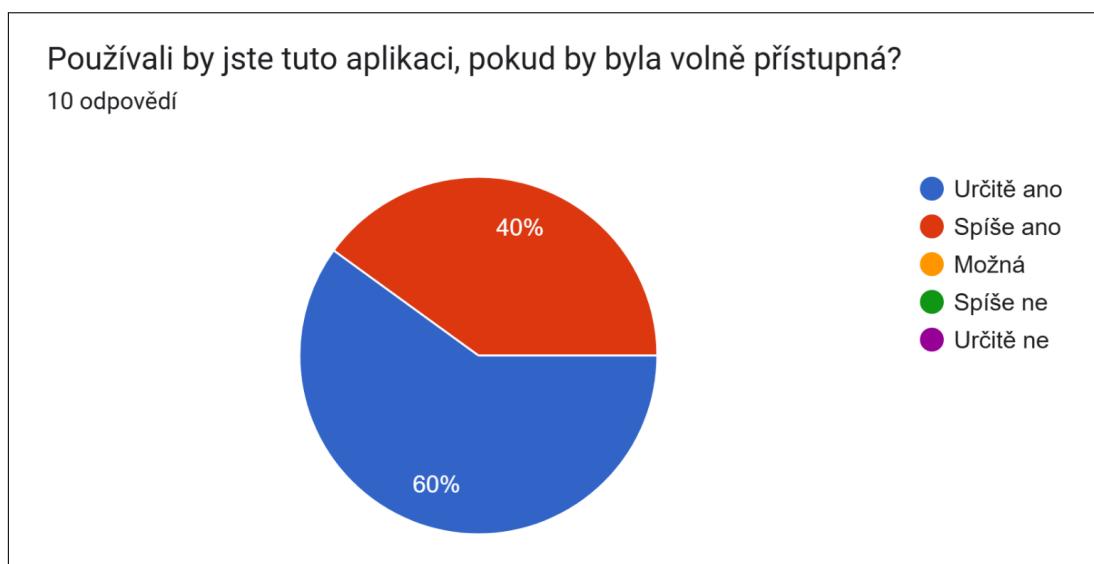
Ačkoliv se obě skupiny respondentů překrývaly pouze u pěti osob, lze porovnáním získat alespoň orientační závěry. U většiny typů otázek došlo ke zvýšení průměrného hodnocení, což může souviset se skutečností, že testovací subjekty jsou častými hráči a kvízy je baví. Výrazné rozdíly byly zaznamenány u **Matematického kvízu**, který si vedl hůře především kvůli vysoké obtížnosti mnou vytvořených rovnic, a u **Slovního řetězu**, který zaznamenal výrazný nárůst obliby, čemuž odpovídá skutečnost, že jsem si na tomto typu dal hodně záležet při implementaci.

Systém bodování byl hodnocen jako spravedlivý s průměrným skóre 4,3. Preference mezi týmovým režimem a režimem všichni proti všem byly vyrovnané. Odhalování písmen u vybraných typů otázek bylo hodnoceno jako užitečné (4,4) a obtížnost **Slepé mapy** byla označena známkou 3,8, což naznačuje, že šlo o lehčí typ otázky, aniž by to bylo vnímáno negativně.

Stabilita aplikace byla hodnocena převážně pozitivně, ale objevily se i problémy. V obou skupinách došlo k situaci, kdy jeden hráč omylem obnovil stránku během hry typu **Slovní řetěz** v týmovém režimu, čímž se nemohl znova připojit do aktuálního kola a to způsobilo výbuch bomby u něj. V jednom případě se tento typ otázky vyskytl na konci kvízu, takže hra ihned skončila. V druhém případě se dotyčnému obnovila herní obrazovka po dokončení kola a hra mohla pokračovat, ale občas se mu zobrazily nesprávné výsledkové komponenty.

Na závěr dotazníku odpovídali respondenti na otázky celkového hodnocení. Otázka „**Do poručili byste tuto aplikaci ostatním?**“ získala průměrnou hodnotu 4,7. Mezi nejčastěji uváděné přednosti patřila multiplatformnost bez nutnosti instalace, atraktivní a kompaktní design a svižná odezva. Aplikace byla především hodnocena jako zábavná. Mezi návrhy na další typy otázek zazněly například otázky ze sportu nebo filmové ukázky. Pokud by byla aplikace placená, čtyři respondenti uvedli, že by za ni byli ochotni zaplatit až 200 Kč, přičemž průměrná částka byla 125 Kč. Respondenti také odpovídali na otázku, zda by aplikaci aktivně využívali, pokud by byla volně dostupná. Výsledky této otázky jsou znázorněny v grafu 6.2 a ukazují, že většina uživatelů by o jejím používání vážně uvažovala.

Na závěr byli respondenti požádáni o celkové hodnocení aplikace na škále od 1 do 10, kde 10 značí maximální spokojenost. Průměrná udělená známka činila velmi příznivých 9,1.



Obrázek 6.2: Graf z dotazníku, který zobrazuje zájem o používání aplikace, pokud by byla volně dostupná.

# Kapitola 7

## Závěr

Cílem této diplomové práce bylo vytvořit kvízovou aplikaci pro operační systém Windows, která umožní hráčům soutěžit v jedné místnosti prostřednictvím lokální sítě. V rámci teoretické části jsem se zaměřil na studium technologií, které umožňují komunikaci v reálném čase a vývoj webových aplikací. Důležitou součástí teoretické části bylo také zkoumání existujících kvízových aplikací a jejich funkcionalit, což mi poskytlo užitečnou inspiraci a podklady pro návrh aplikace.

Při návrhu aplikace jsem vytvořil intuitivní uživatelské rozhraní jak hlavní obrazovky, která zobrazuje průběh hry, tak mobilních zařízení, na nichž hráči odpovídají. Důraz byl kladen na jednoduché ovládání a variabilitu disciplín s možností jejich kombinace, aby aplikace zůstala zábavná a rozmanitá i při opakovaném hraní. Návrh zohledňuje také možnost tvorby a sdílení vlastních kvízů, přičemž použitelnost jednotlivých disciplín byla ověřena pomocí uživatelského průzkumu. Mimo jiné bylo rovněž nutné navrhnout API a události knihovny Socket.IO pro obousměrnou komunikaci mezi serverem a klienty. Pro ukládání a správu kvízových dat jsem navrhl vhodnou strukturu kolekcí pro databázi MongoDB.

Aplikace byla implementována jako responzivní webová aplikace s vestavěným serverem ve frameworku Flask a klientskou částí vytvořenou pomocí Reactu. Aplikace zahrnuje řadu technických řešení včetně synchronizace přes WebSockets, správy kvízů a hratelnosti různorodých typů otázek. Herní mechaniky byly navrženy tak, aby umožnily snadné rozšíření o další disciplíny do budoucna. Pro lepší představu o fungování aplikace jsem připravil video<sup>1</sup>, demonstrující správu kvízů a průběh hry v týmovém i individuálním režimu. Aplikace je plně funkční a odpovídá stanoveným požadavkům.

Pro ověření hratelnosti a zábavnosti jednotlivých disciplín byly uspořádány dvě testovací seance, každá se šesti účastníky, kteří si zahráli dva kvízy v reálných podmírkách s mobilními zařízeními. Pro otestování funkcí souvisejících s tvorbou, úpravou a sdílením kvízů byl připraven samostatný scénář testovaný čtyřmi uživateli. Přijaté návrhy na vylepšení jsou shrnutы v testovací části a tvoří základ pro budoucí vývoj. Jedním z hlavních bodů dalšího vývoje je zajištění podpory pro opětovné připojení hráče po výpadku.

Při vývoji jsem rozšířil své dovednosti o komunikaci v reálném čase mezi serverem a klienty, a naučil jsem se přemýšlet o architektuře systému s ohledem na budoucí rozšířování. Oproti své bakalářské práci jsem tentokrát věnoval větší pozornost dlouhodobé udržitelnosti a jednoduchému nasazení bez nutnosti hostingu. Pozitivní reakce testerů mě velmi potěšily a motivovaly k dalšímu rozvoji aplikace i případnému zveřejnění pro širší veřejnost.

---

<sup>1</sup><https://youtu.be/q07QE9WSRpw>

# Literatura

- [1] ALEXEY, K. *Flask aplikace pro tvorbu kvízů online*. 2023. Bakalářská práce. Vysoká škola ekonomická v Praze, Fakulta informatiky a statistiky. Dostupné z: [https://vskp.vse.cz/89627\\_flask-aplikace-pro-tvorbu-kvizu?title=V&page=15](https://vskp.vse.cz/89627_flask-aplikace-pro-tvorbu-kvizu?title=V&page=15). [cit. 2024-12-02].
- [2] ANTOSHA, A. *Webová aplikace pro kvíz v reálném čase online*. 2020. Bakalářská práce. České vysoké učení technické v Praze, Fakulta elektrotechnická. Dostupné z: <https://dspace.cvut.cz/handle/10467/87702>. [cit. 2024-12-02].
- [3] AWATI, R. a GILLIS, A. S. *What is a web server?* online. Prosinec 2024. Dostupné z: <https://www.techtarget.com/whatis/definition/Web-server>. [cit. 2025-01-02].
- [4] CINCOVIĆ, J. a MARIJA, P. Comparison: Angular vs. React vs. Vue. Which framework is the best choice? In: *ICIST 2020 Proceedings: Proceedings of the 10th International Conference on Information Society and Technology* online. ISOS Conference Proceedings Series. Belgrade, Serbia: Information Society of Serbia - ISOS, 2020, s. 250–255. ISSN 2738-1447. Dostupné z: <https://www.eventiotic.com/eventiotic/files/Papers/URL/50173409-699e-4b17-8edb-9764ecc53160.pdf>. [cit. 2025-02-26].
- [5] COBURN, J. Getting Started with Flask. In: *Build Your Own Car Dashboard with a Raspberry Pi: Practical Projects to Build Your Own Smart Car* online. Berkeley, CA: Apress, 2020, s. 143–170. ISBN 978-1-4842-6080-7. Dostupné z: [https://doi.org/10.1007/978-1-4842-6080-7\\_6](https://doi.org/10.1007/978-1-4842-6080-7_6). [cit. 2025-01-03].
- [6] DEVELOPER RELATIONS TEAM. *What are WebSockets ws:// and wss:// connections* online. 03. září 2023. Dostupné z: <https://www.pubnub.com/guides/websockets/#h-0>. [cit. 2025-02-27].
- [7] GOOGLE DESIGN. *When Material Made Its Global Debut* online. Dostupné z: <https://design.google/library/material-design-launch-2014>. [cit. 2025-05-13].
- [8] GRINBERG, M. *Flask-SocketIO* online. Dokumentace. 2018. Dostupné z: <https://flask-socketio.readthedocs.io/en/latest/>. [cit. 2025-02-27].
- [9] HASAN, M. F. *Understanding Server-Sent Events (SSE) with Node.js* online. 11. září 2024. Dostupné z: <https://itsfuad.medium.com/understanding-server-sent-events-sse-with-node-js-3e881c533081>. [cit. 2025-02-27].
- [10] HELLODARWIN. *UX: 10 ways to improve the "User eXperience" of your website* online. 22. června 2024. Dostupné z:

<https://hellodarwin.com/blog/ux-10-ways-increase-user-experience-website>. [cit. 2025-02-25].

- [11] INTERACTION DESIGN FOUNDATION - IxDF. *What is User Interface (UI) Design?* online. 2. června 2016. Dostupné z: <https://www.interaction-design.org/literature/topics/ui-design>. [cit. 2025-01-04].
- [12] KABAMBA, H. M.; KHOUZAM, M. a DAGENAIS, M. R. Vnode: Low-Overhead Transparent Tracing of Node.js-Based Microservice Architectures. *Future Internet* online, Prosinec 2024, sv. 16, č. 1. ISSN 1999-5903. Dostupné z: <https://doi.org/10.3390/fi16010013>. [cit. 2025-01-03].
- [13] KRUG, S. *Don't Make Me Think, Revisited: A Common Sense Approach to Web Usability*. 3. vyd. USA: New Riders Publishing, 2014. ISBN 978-0-321-96551-6.
- [14] MUDRA, P. *Flask aplikace pro tvorbu kvízů* online. 2024. Bakalářská práce. Vysoká škola ekonomická v Praze, Fakulta informatiky a statistiky. Dostupné z: [https://vskp.vse.cz/92890\\_flask-aplikace-pro-tvorbu-kvizu?page=1](https://vskp.vse.cz/92890_flask-aplikace-pro-tvorbu-kvizu?page=1). [cit. 2024-12-02].
- [15] MUI TEAM. *React templates* online. Dokumentace. Dostupné z: <https://mui.com/material-ui/getting-started/templates>. [cit. 2025-05-13].
- [16] NAYAK, A. *Exploring Real-Time Communication in Web Development: Short Polling vs. Long Polling* online. 17. listopadu 2023. Dostupné z: <https://medium.com/@canoopnayak1/exploring-real-time-communication-in-web-development-short-polling-vs-long-polling-ec571f5e8af8>. [cit. 2025-02-27].
- [17] POLEY, C. *Angular vs React vs Vue - Detailed Framework Comparison* online. 30. srpna 2024. Dostupné z: <https://www.tiny.cloud/blog/vue-react-angular-js-framework-comparison/>. [cit. 2025-02-26].
- [18] REDDY, K. S. P. a UPADHYAYULA, S. *Beginning Spring Boot 3: Build Dynamic Cloud-Native Java Applications and Microservices* online. 2. vyd. Apress Berkeley, CA, listopad 2022. ISBN 978-1-4842-8792-7. Dostupné z: <https://doi.org/10.1007/978-1-4842-8792-7>. [cit. 2025-01-04].
- [19] SHAH, V. *Angular Vs React Vs Vue: Which One To Choose* online. 05. února 2025. Dostupné z: <https://www.tatvasoft.com/blog/angular-vs-react-vs-vue/>. [cit. 2025-02-26].
- [20] UNGER, R. a CHANDLER, C. *A Project Guide to UX Design: For user experience designers in the field or in the making*. 2. vyd. USA: New Riders Publishing, 2012. ISBN 978-0-321-81538-5.
- [21] USERTESTING TEAM. Usability Testing vs. A/B Testing. *UserTesting Blog* online, October 2023. Dostupné z: <https://www.usertesting.com/blog/usability-testing-vs-ab-testing>. [cit. 2025-05-13].

- [22] XU, A. Short/long polling, SSE, WebSocket. *EP 39: Accounting 101 in Payment Systems* online. 31. prosince 2022. Dostupné z: <https://blog.bytebytogo.com/i/93929190/shortlong-polling-sse-websocket>. [cit. 2025-02-27].
- [23] ZANEVYCH, O. ADVANCING WEB DEVELOPMENT: A COMPARATIVE ANALYSIS OF MODERN FRAMEWORKS FOR REST AND GRAPHQL BACK-END SERVICES. *Grail of Science* online, Březen 2024, č. 37, s. 216–228. Dostupné z: <https://doi.org/10.36074/grail-of-science.15.03.2024.031>. [cit. 2025-01-03].

## Příloha A

# Přehled odevzdaných souborů na NextCloudu

```
/HomeQuiz
└─ /HomeQuizClient - zdrojové soubory klientské části
    └─ /docs
        └─ /jsdoc
            └─ index.html - vygenerovaná dokumentace klientské části
└─ /HomeQuizServer - zdrojové soubory serverové části
    └─ /docs
        └─ /app
            └─ index.html - vygenerovaná dokumentace serverové části
└─ /HomeQuizLatex - zdrojové soubory technické zprávy
└─ HomeQuizReport.pdf - technická zpráva
└─ HomeQuizVideo.mp4 - video zobrazující funkcionalitu aplikace
└─ HomeQuiz.zip - zip obsahující spustitelný soubor aplikace
    └─ HomeQuiz.exe - nejnovější verze (nebyla důsledně otestována)
    └─ HomeQuizTested.exe - verze použitá při testování s uživateli
└─ README.md - návod na instalaci/spuštění
```