

ZÁVĚREČNÁ STUDIJNÍ PRÁCE

dokumentace

**Webová aplikace pro tvorbu a řízení
interaktivních kvízů**



Autor: Václav Stoklasa
Obor: 18-20-M/01 INFORMAČNÍ TECHNOLOGIE
se zaměřením na počítačové sítě a programování
Třída: IT4 IT4
Školní rok: 2025/26 2025/26

Poděkování

Rád bych poděkoval Ing. Petru Grussmannovi za odborné vedení práce a Mgr. Marku Lučnému za poskytnutí nápadu projektu a jeho podporu při zpracování.

Prohlášení

Prohlašuji, že jsem závěrečnou práci vypracoval samostatně a uvedl veškeré použité informační zdroje.

Souhlasím, aby tato studijní práce byla použita k výukovým a prezentačním účelům na Střední průmyslové a umělecké škole v Opavě, Praskova 399/8.

V Opavě 1. 1. 2026

.....
Podpis autora

Abstrakt

Výsledkem práce je webová aplikace QuizIT! pro interaktivní kvízy určené pro školy. Učitelé v aplikaci mohou vytvářet vlastní kvízy s otázkami a odpověďmi a spouštět živá kvízová sezení. Studenti se ke kvízům připojují pomocí unikátního kódu a odpovídají na otázky v reálném čase. Body se počítají podle správnosti odpovědi a rychlosti reakce. Aplikace obsahuje systém žolíků a průběžný žebříček účastníků. Výsledky se aktualizují v reálném čase, takže mají učitelé i studenti neustálý přehled o průběhu kvízu.

Klíčová slova

interaktivní kvízy, webová aplikace, školní výuka, real-time komunikace, bodové hodnocení, žebříček účastníků, uživatelské role

Abstract The result of this project is a web application called QuizIT! designed for interactive quizzes intended for use in schools. Teachers can create their own quizzes with questions and answers and run live quiz sessions. Students join the quizzes using a unique code and answer questions in real time. Scoring is based on the correctness of the answer and the response speed. The application includes a joker system and a live leaderboard of participants. Results are updated in real time, giving both teachers and students a continuous overview of the quiz progress.

Keywords

interactive quizzes, web application, school education, real-time communication, scoring system, live leaderboard, user roles

Obsah

Úvod	3
1 Teoretická a metodická východiska	5
1.1 Interaktivní kvízové aplikace	5
1.2 Architektura databázových aplikací	5
1.3 Základy webového vývoje	6
2 Využité technologie	7
2.1 Django	7
2.2 Wagtail CMS	7
2.3 Microsoft OAuth	7
2.4 PostgreSQL	7
2.5 Socket.IO	8
2.6 Docker	8
3 Způsoby řešení a použité postupy	9
3.1 Založení projektu	9
3.2 Adresářová struktura	9
3.3 Databázový model	10
3.4 Autentizace a autorizace	10
3.5 Kvízy a živá sezení	12
3.6 Real-time aktualizace pomocí Socket.IO	18
3.7 Vzdělávací materiály přes Wagtail CMS	19
3.8 Vizuální stránka aplikace	20
4 Výsledky řešení a výstupy	21
4.1 Funkce aplikace	21
4.2 Splněné a nesplněné cíle	22

ÚVOD

Z vlastní zkušenosti mohu říct, že kvízy patří ve výuce mezi nejoblibější formy ověřování znalostí. Studenti se do nich většinou zapojují více než při klasickém zkoušení. Písemné testy a ústní zkoušení často neposkytují okamžitou zpětnou vazbu a ne vždy zaujmou všechny studenty. Interaktivní kvízové aplikace tento problém řeší. Spojují výuku s herními prvky a zvyšují motivaci studentů. Zároveň učitelům umožňují rychle získat přehled o znalostech celé třídy. Proto se v posledních letech ve školách používají stále častěji.

Cílem této práce bylo vytvořit webovou aplikaci QuizIT! pro interaktivní kvízy, která je snadno použitelná ve školním prostředí. Aplikace umožňuje učitelům vytvářet vlastní kvízy s otázkami a odpověďmi a spouštět živá kvízová sezení. Studenti se do těchto sezení připojují pomocí unikátního kódu a odpovídají na otázky v reálném čase. Hodnocení je založeno na správnosti odpovědi i rychlosti reakce. Součástí aplikace je systém žolíků, průběžný žebříček a aktualizace výsledků v reálném čase pomocí technologie Socket.IO. Pro zvýšení efektivity výuky aplikace umožňuje vytvářet vzdělávací materiály prostřednictvím systému Wagtail CMS. Aplikace je postavena na frameworku Django, využívá databázi PostgreSQL a autentizaci pomocí Microsoft OAuth. Celá webová stránka je navržena jako plně responzivní a podporuje světlý i tmavý režim.

V dokumentaci projektu popisuji postup vytvoření této aplikace. Nejprve se věnuji návrhu databázového modelu a architektuře systému. Následně řeším autentizaci uživatelů, tvorbu a správu kvízů a realizaci živých kvízových sezení. Dále se zaměřuji na integraci systému Wagtail CMS pro správu vzdělávacích materiálů. V závěru se věnuji vizuální stránce aplikace a její responzivitě.

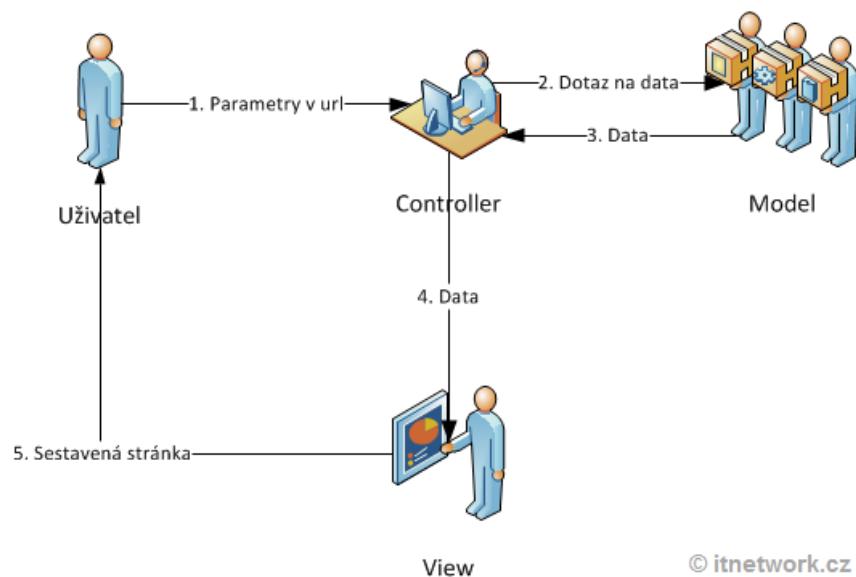
1 TEORETICKÁ A METODICKÁ VÝCHODISKA

1.1 INTERAKTIVNÍ KVÍZOVÉ APLIKACE

Na internetu existuje mnoho kvízových aplikací pro výuku, z nichž nejznámější je Kahoot. Kromě něj jsou dostupné i další aplikace s různými funkcemi a omezeními. S některými z nich jsem se setkal a často mi přišly zbytečně složité, málo přizpůsobitelné nebo dostupné až po zaplacení. Přitom vytváření a správa kvízů nemusí být složité a mohou učitelům dobře posloužit k rychlému a přehlednému ověřování znalostí studentů.

1.2 ARCHITEKTURA DATABÁZOVÝCH APLIKACÍ

Databázové webové aplikace jsou často založeny na architektuře MVC nebo její variantě MVT, kterou využívá framework Django. Aplikace je rozdělena na tři části – modely, pohledy a šablony. Uživatel odešle požadavek přes prohlížeč, který je zpracován v pohledu. Ten podle potřeby získá data z databáze pomocí modelu a předá je šabloně. Ta následně vytvoří výslednou stránku pro uživatele. Toto rozdělení zajišťuje přehlednější a lépe udržovatelný kód. Celou situaci můžeme znázornit diagramem:



Obrázek 1.1: Diagram MVC architektury

1.3 ZÁKLADY WEBOVÉHO VÝVOJE

Webové aplikace fungují na principu architektury klient–server. Klientem je webový prohlížeč, který odesílá požadavky na server pomocí HTTP protokolu. Server tyto požadavky zpracuje, případně načte potřebná data z databáze, a vrátí odpověď ve formě HTML stránky, kterou prohlížeč zobrazí uživateli.

U složitějších aplikací je nutné využívat programování na straně serveru, které umožňuje vytvářet dynamický obsah. Frameworky, jako je Django, usnadňují vývoj těchto aplikací tím, že poskytují hotovou strukturu, práci s databází, formuláři a základní zabezpečení.

2 VYUŽITÉ TECHNOLOGIE

2.1 DJANGO

Django je webový framework napsaný v jazyce Python, který slouží k vývoji webových aplikací. Nabízí nástroje pro práci s databází, například ORM, které umožňuje pracovat s databází pomocí objektů v kódu místo psaní SQL dotazů, a obsahuje také vestavěný administrační panel. Framework využívá již zmíněnou architekturu MVT (Model-View-Template), díky které je kód přehlednější a logika aplikace je oddělena od jejího vzhledu. V projektu byla použita verze Django 4.2.

2.2 WAGTAIL CMS

Wagtail je redakční systém postavený na frameworku Django, který slouží ke správě obsahu webových aplikací. Umožňuje i ne-technickým uživatelům jednoduše vytvářet a upravovat obsah přes přehledné administrační rozhraní. V této aplikaci je Wagtail využit pro správu vzdělávacích materiálů, které se studentům zobrazují před zahájením kvízu nebo po jeho skončení.

2.3 MICROSOFT OAUTH

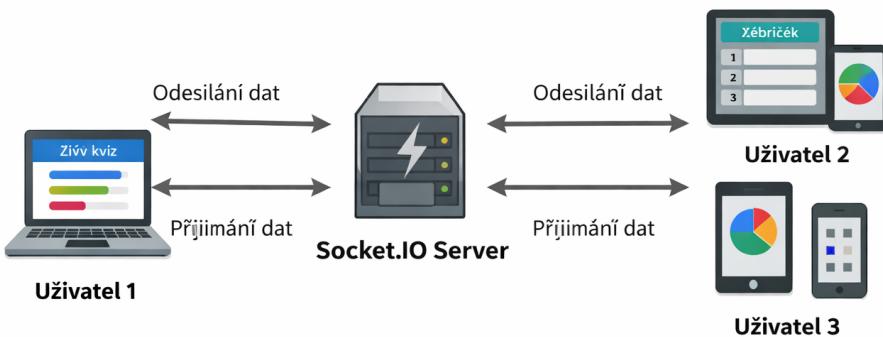
Přihlášení přes Microsoft OAuth je řešeno pomocí klíčů vytvořených ve školním Microsoft Entra ID. Tyto klíče slouží k bezpečnému přihlášení uživatelů a nejsou uložené přímo v kódu aplikace.

2.4 POSTGRESQL

PostgreSQL je relační databáze, která slouží k ukládání dat aplikace. V tomto projektu je využita pro ukládání všech důležitých informací, jako jsou kvízy, otázky, odpovědi, uživatelé a průběh živých sezení. Databáze běží v Docker kontejneru, což usnadňuje její spuštění a zajistí uje stejné prostředí při vývoji i nasazení aplikace.

2.5 SOCKET.IO

Socket.IO je knihovna, která umožňuje komunikaci mezi klientem a serverem v reálném čase. Díky tomu je možné okamžitě aktualizovat data bez nutnosti znova načítat stránku. V této aplikaci je Socket.IO využíván pro živé aktualizace statistik odpovědí, žebříčku a průběhu kvízu během živých sezení. Socket.IO server běží samostatně v Docker kontejneru a komunikuje s Django aplikací.



Obrázek 2.1: Socket.IO

2.6 DOCKER

Docker je nástroj, který umožňuje balit aplikace a jejich závislosti do kontejnerů, aby bylo možné je spouštět stejným způsobem na různých systémech. Díky tomu není nutné složitě nastavovat prostředí pro běh aplikace. V této aplikaci je Docker využit ke spuštění Django serveru, databáze PostgreSQL a Socket.IO serveru pomocí Docker Compose.

3 ZPŮSOBY ŘEŠENÍ A POUŽITÉ POSTUPY

3.1 ZALOŽENÍ PROJEKTU

Projekt byl založen pomocí frameworku Django příkazem `django-admin startproject kahootapp`. Následně byly vytvořeny jednotlivé aplikace pomocí příkazu `python manage.py startapp`, konkrétně aplikace `quiz` pro práci s kvízy a `home` pro integraci Wagtail CMS.

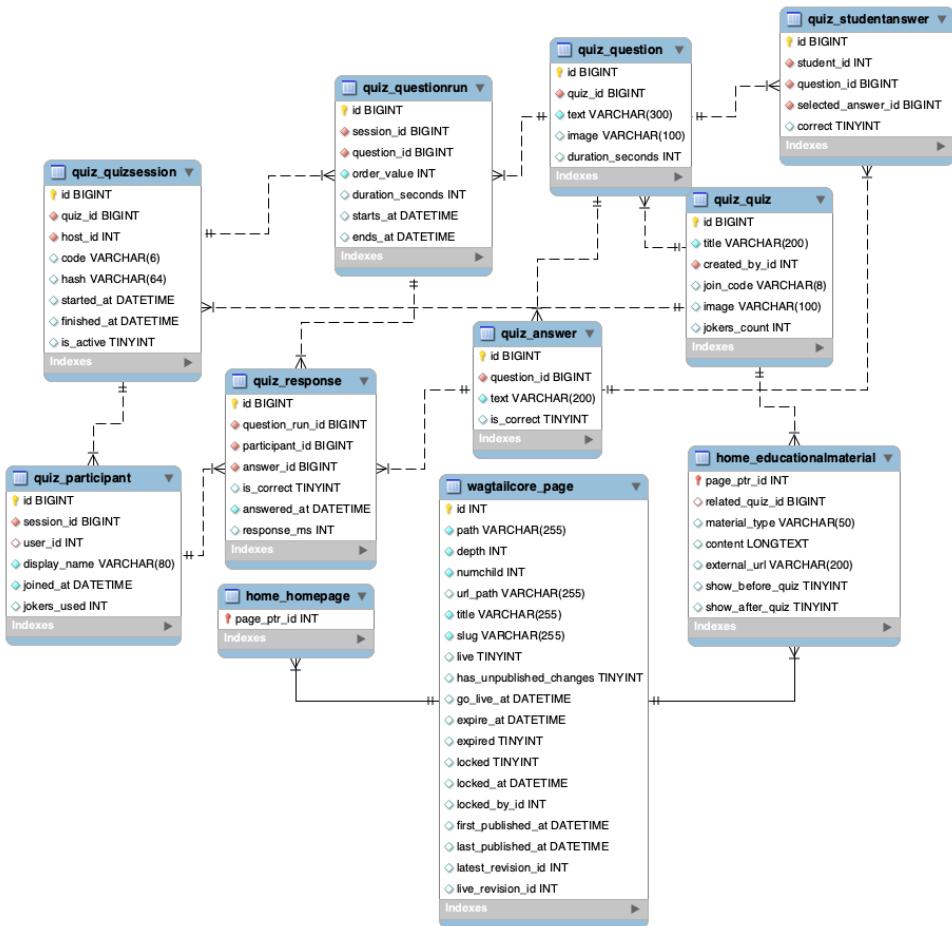
Závislosti projektu byly spravovány pomocí souboru `requirements.txt`, do kterého byly postupně přidávány potřebné knihovny. Na začátku vývoje byla použita databáze SQLite, která je výchozí databází Django a nevyžaduje složité nastavování. Dále byla provedena základní konfigurace projektu a přidání použitých aplikací do nastavení Django.

3.2 ADRESÁŘOVÁ STRUKTURA

```
QuizIT/
├── kahootapp                // hlavní Django projekt
│   ├── settings               // nastavení projektu
│   ├── static                 // statické soubory projektu
│   │   └── templates           // globální šablony
│   ├── quiz                   // aplikace pro kvízy
│   │   ├── migrations          // databázové migrace
│   │   ├── templates            // šablony pro kvízy
│   │   ├── models.py            // databázové modely
│   │   ├── views.py             // řadič
│   │   ├── admin.py              // registrace modelů
│   │   └── socketio_handler.py // Socket.IO integrace
│   ├── home                    // Wagtail CMS aplikace
│   │   ├── migrations          // databázové migrace
│   │   ├── templates            // šablony pro Wagtail stránky
│   │   ├── models.py            // Wagtail modely
│   ├── static                  // statické soubory projektu
│   ├── media                   // nahrané soubory uživatelů
│   ├── data                     // data Docker kontejnerů
│   ├── manage.py                // Django management script
│   ├── requirements.txt         // soubor s požadavky technologií
│   ├── Dockerfile               // slouží pro vytvoření docker image
│   ├── docker-compose.yml       // konfigurace dockeru
│   ├── docker-entrypoint.sh     // entrypoint skript pro Docker
│   └── socketio_server.py        // samostatný Socket.IO server
```

Obrázek 3.1: Adresářová struktura

3.3 DATABÁZOVÝ MODEL



Obrázek 3.2: ER diagram

3.4 AUTENTIZACE A AUTORIZACE

Autentizace uživatelů je důležitou součástí aplikace, protože zajišťuje bezpečný přístup do systému a správné fungování jednotlivých částí aplikace. QuizIT! proto obsahuje uživatelský systém, který se stará o přihlašování uživatelů a omezení přístupu k vybraným funkcím.

Při vývoji jsem nejprve zvažoval využití základního autentizačního systému Djanga, ten je však poměrně jednoduchý a nenabízí dostatečné možnosti pro přihlašování pomocí externích služeb. Z tohoto důvodu jsem se rozhodl použít balíček django-allauth, který umožňuje rozšířenou správu uživatelských účtů.

Uživatelé se mohou přihlásit klasickým způsobem pomocí uživatelského jména a hesla nebo prostřednictvím školního Microsoft účtu, který je řešen pomocí služby Microsoft Entra ID.

3.4.1 Balíček django-allauth

Balíček django-allauth slouží v aplikaci jako hlavní nástroj pro autentizaci a správu uživatelských účtů. Zajišťuje základní funkce, jako je přihlašování, registrace uživatelů a práce s hesly, a zároveň umožňuje přihlášení pomocí externích služeb, což zjednoduší celý proces správy účtů.

V projektu je django-allauth nastaven jako hlavní autentizační backend a umožňuje přihlášení pomocí uživatelského jména nebo e-mailu, případně prostřednictvím Microsoft účtu. Po úspěšném přihlášení je uživatel automaticky přesměrován na hlavní stránku aplikace, odkud může pokračovat v její práci.

3.4.2 Autentizace pomocí Microsoft OAuth

Aplikace podporuje přihlášení pomocí Microsoft Entra ID přes technologii OAuth 2.0. Přihlašování pomocí účtu třetích stran je dnes velmi běžné a rychlé, protože uživatelé si nemusí vytvářet nový účet ani pamatovat další heslo. Vzhledem k tomu, že aplikaci vyvíjím pro školní prostředí, přišlo mi vhodné použít Microsoft účty, které se ve škole běžně používají.

Pro tuto funkci jsem vytvořil aplikaci v prostředí Microsoft Entra ID a získal potřebné přihlašovací údaje. Ty jsou uloženy v souboru .env, což umožňuje měnit nastavení bez zásahu do kódu. Přihlášení probíhá pomocí tlačítka „Přihlásit se přes Microsoft“ a po úspěšném ověření je uživatel automaticky přihlášen do aplikace.

The image displays two side-by-side registration and login forms. On the left, the 'Registration' form is shown with fields for 'E-mail (nepovinné)', 'Uživatelské jméno', 'Heslo', and 'Heslo (znovu)'. It features two buttons: a solid blue 'Zaregistrovat' button and a white-outlined 'Mám účet' button. At the bottom is a blue button labeled 'Přihlásit se přes Microsoft'. On the right, the 'Přihlášení' (Login) form is shown with fields for 'Uživatelské jméno nebo e-mail' and 'Heslo'. It features two buttons: a solid blue 'Přihlásit' button and a white-outlined 'Registrace' button. At the bottom is a large blue button labeled 'Přihlásit se přes Microsoft'.

Obrázek 3.3: Prihlasovací a registrační formulář

3.4.3 Systém rolí a oprávnění

Aplikace rozlišuje tři role: admina, učitele a studenty. Systém je založen na Django skupinách, přičemž skupiny „Teacher“ a „Student“ jsou v aplikaci předem definovány. Noví uživatelé jsou při registraci automaticky zařazeni do skupiny „Student“.

Učitelé a admini mají rozšířená oprávnění. Mohou vytvářet kvízy, spouštět živá sezení, sledovat průběžné výsledky a spravovat vzdělávací materiály. Admini mají navíc přístup do administrační části aplikace, kde mohou spravovat obsah a základní nastavení systému. Studenti se mohou pouze připojovat ke kvízům pomocí kódu a odpovídat na otázky.

Omezování přístupu je řešeno kontrolou přihlášení a oprávnění přímo v kódu aplikace. V šablonách se používá filtr

3.5 KVÍZY A ŽIVÁ SEZENÍ

Aplikace umožňuje učitelům vytvářet vlastní kvízy s otázkami a odpověďmi. Kvíz si mohou spustit buď jen pro sebe v jednoduchém režimu, nebo jako živé sezení pro studenty.

Při živém sezení se studenti připojí a postupně odpovídají na otázky. Učitel během kvízu vidí jeho průběh i výsledky jednotlivých studentů.

3.5.1 Tvorba a správa kvízů

Vytváření kvízů

Učitelé vytvářejí kvízy pomocí jednoduchého formuláře. Zadají název kvízu a počet žolíků. Ke každé otázce mohou napsat text, přidat obrázek a nastavit čas na odpověď.

Otzáka může mít až deset odpovědí a alespoň jedna z nich musí být správná. Formulář umožňuje přidat více otásek najednou. Všechny údaje se uloží po odeslání formuláře.

Správa kvízů

V přehledu „Moje kvízy“ jsou zobrazeny všechny kvízy, které učitel vytvořil. U každého kvízu je vidět jeho název a kód, který slouží jako jeho jednoznačné označení v systému. Z tohoto přehledu může učitel kvíz upravit, smazat nebo spustit.

Zároveň je zde možné vytvořit nový kvíz, spustit kvíz pro sebe v jednoduchém režimu nebo zahájit živé sezení pro vybraný kvíz. V přehledu jsou také vidět aktuálně běžící sezení, u kterých může učitel přejít do lobby nebo sezení ukončit.

Počet žolíků za celou hru:

[0 ⏮]

Počet žolíků, které může každý student použít během celého kvízu (0-3).

Otázka 1[Smazat otázku](#)**Text otázky:**

Zadejte text otázky

Obrázek k otázce (volitelné): Choose File no file selected

Zobrazí se u této otázky studentům.

Čas na odpověď (sekundy):

[20 ⏮]

Minimálně 5 sekund, maximálně 300 sekund (5 minut).

Odpovědi:

Text odpovědi

 Správná ×

Text odpovědi

 Správná ×

Text odpovědi

 Správná ×

Text odpovědi

 Správná ×[+ Přidat odpověď](#)[+ Přidat otázku](#)

Obrázek 3.4: Formulář pro vytváření kvízů

Moje kvízy[Vytvořit kvíz](#)[Spravovat v administraci](#)

Název	Kód	Akce
I/Y	K4MP95QQ	Start live Spustit Upravit Smazat
matematika	SBHRS2RR	Start live Spustit Upravit Smazat

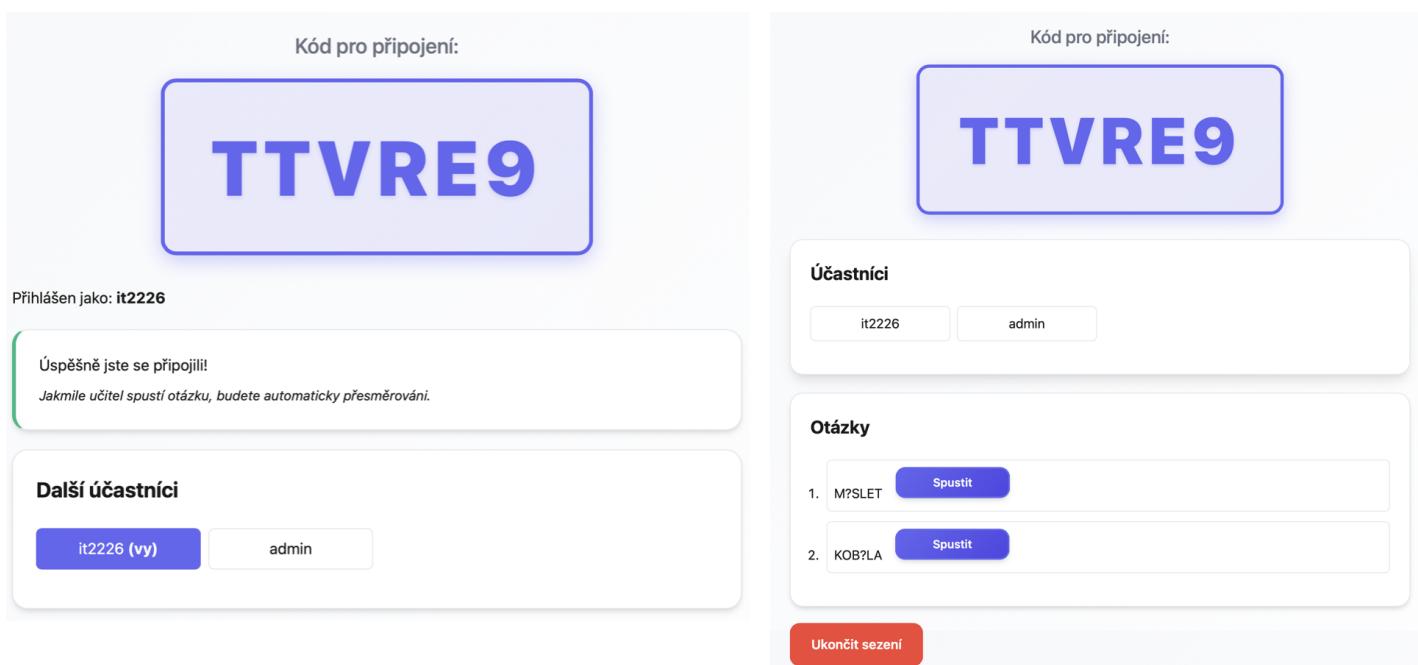
Aktivní sezeníKód: **7VGLLH**[Lobby](#)[Výsledky](#)

Obrázek 3.5: Správá kvízů

3.5.2 Vytváření sezení a připojování studentů

Učitel vytvoří živé sezení kliknutím na tlačítko „Start live“ u vybraného kvízu. Při vytvoření sezení se automaticky vygeneruje šestimístný kód pro připojení a bezpečný hash pro URL adresu. Učitel je následně přesměrován do lobby, kde vidí kód pro připojení, seznam připojených studentů a přehled otázek v kvízu.

Studenti se ke kvízu připojují zadáním šestimístného kódu na stránce „Připojit se ke kvízu“. Po zadání správného kódu jsou přesměrováni do lobby, kde čekají na spuštění kvízu učitelem. V lobby vidí název kvízu, kód pro připojení a seznam ostatních účastníků.



Obrázek 3.6: Lobby z pohledu studenta a učitele před spuštěním kvízu.

3.5.3 Průběh kvízu během živého sezení, bodové hodnocení a žolíky

Zobrazení otázky studentům

Po spuštění otázky učitelem se všem studentům zobrazí stránka s aktuální otázkou. Na obrazovce vidí text otázky, případně obrázek, a seznam možných odpovědí. Zároveň se zobrazuje odpočet času, který má student na odpověď. Student vybírá jednu z možností a svůj výběr potvrdí tlačítkem pro odeslání odpovědi.

I/Y

Otzážka 2: KOB?LA

Zbývající čas: 18 s

I

A

Y

E

Odeslat odpověď

Obrázek 3.7: Nabídka odpovědí na otázku z pohledu studenta

3.5.4 Odpovídání a bodové hodnocení

Po odeslání odpovědi zůstává student na stejné stránce. Systém uloží čas reakce od začátku otázky a podle něj vypočítá počet bodů. Pokud student odpoví špatně, získá 0 bodů. U správné odpovědi platí, že čím rychlejší odpověď, tím více bodů, maximálně až 1000 bodů. Pomalejší správné odpovědi získají méně bodů, minimálně 400. Student po odpovědi vidí body za aktuální otázku i své celkové skóre.

I/Y

Otzážka 4: PEŘ?

Zbývající čas: 25 s

Odpověď přijata. Čekejte, než odpoví ostatní studenti nebo učitel spustí další otázku.

BODY ZA TUTO OTÁZKU:

0

✗ Špatně (0 bodů)

CELKOVÉ BODY:

1725

Obrázek 3.8: Stránka po odeslání odpovědi s pohledu studenta

3.5.5 Použití žolíků

Během odpovídání může student použít žolík, pokud ho má k dispozici. Počet žolíků je nastaven pro celý kvíz a pohybuje se v rozmezí 0–3. Po použití žolíku se smažou dvě náhodné špatné odpovědi, čímž se zjednoduší výběr správné možnosti. Žolík lze použít pouze jednou na otázku a jen před odesláním odpovědi. Po použití se zobrazí zbývající počet žolíků.

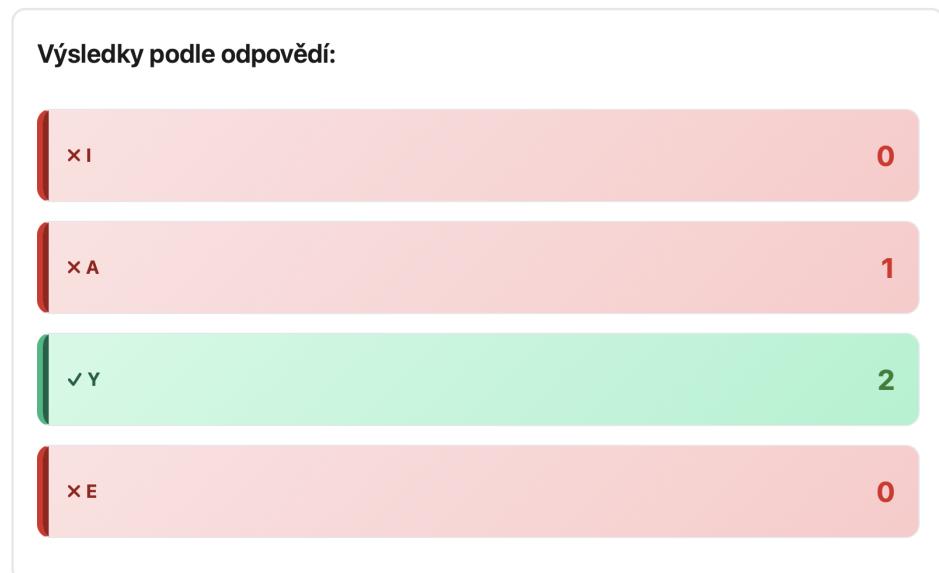


Obrázek 3.9: Tlačítko na využití žolíka po jehož použití se odebere polovina špatných odpovědí

3.5.6 Průběžné hodnocení a statistiky

Během kvízu má učitel přehled o tom, jak studenti odpovídají. Vidí, kolik studentů už odpovědělo, a postupně se mu aktualizuje stav odpovědí.

Jakmile odpoví všichni studenti nebo vyprší časový limit, zobrazí se výsledky otázky. Učitel vidí průběžný žebříček podle bodů a také tabulku „Kdo jak odpověděl“, kde je u každého studenta uvedeno, zda odpověděl správně, špatně, nebo neodpověděl vůbec. Poté může učitel spustit další otázku a studenti jsou automaticky přesunuti dál.



Průběžný žebříček:

#	Účastník	Body
1.	Marek	1686
2.	it2226	823
3.	admin	541

Kdo jak odpověděl:

Účastník	Odpověď	Stav
Marek	Y	Správně ✓
it2226	A	Špatně ✗
admin	Y	Správně ✓

Obrázek 3.10: Pruběžné výsledky studentů z pohledu učitele

3.5.7 Výsledky a žebříček

Po dokončení všech otázek učitel klikne na tlačítko „Ukončit sezení“. Tím se všem účastníkům zobrazí finální výsledky kvízu. Na stránce je vidět pódium s prvními třemi místy a jejich celkovým počtem bodů. Pod ním je zobrazen celý žebříček všech studentů seřazený podle získaných bodů. Učitel si může výsledky stáhnout do souboru CSV. Po skončení kvízu se studentům mohou zobrazit také doplňující vzdělávací materiály.

Stáhnout CSV		
2. místo Marek 2183 bodů	1. místo it2226 2437 bodů	3. místo admin 1462 bodů
Celkový žebříček		
#	Účastník	Skóre
1.	it2226	2437
2.	Marek	2183
3.	admin	1462

Obrázek 3.11: Výsledková listina z pohledu učitele

3.6 REAL-TIME AKTUALIZACE POMOCÍ SOCKET.IO

Pro živá kvízová sezení aplikace využívá technologii Socket.IO. Díky ní se informace během kvízu aktualizují v reálném čase bez nutnosti obnovovat stránku. Učitel i studenti tak okamžitě vidí změny, například průběh odpovídání, žebříček nebo stav aktuální otázky.

3.6.1 Architektura a implementace

Aplikace používá samostatný Socket.IO server, který běží odděleně od Django serveru. Django server odesílá informace o průběhu kvízu na Socket.IO server, který je následně rozesílá připojeným uživatelům.

Každé živé sezení má svou vlastní „místnost“ (room). Do této místnosti se připojí učitel i studenti daného kvízu. Díky tomu dostávají aktualizace pouze účastníci konkrétního sezení.

Prohlížeče studentů i učitele se k Socket.IO serveru připojují pomocí JavaScriptu. Ten naslouchá událostem ze serveru a automaticky aktualizuje obsah stránky. Jedná se například o počty odpovědí, průběžný žebříček, tabulkou účastníků nebo zbývající čas na odpověď. Změny se tak zobrazují okamžitě bez nutnosti ručního obnovení stránky.

Server navíc každou sekundu odesílá aktualizace, aby měl učitel vždy aktuální přehled o průběhu kvízu, i když studenti zrovna neodpovídají.

3.6.2 Fallback mechanismus

V případě, že se nepodaří připojit k Socket.IO serveru nebo dojde k výpadku spojení, aplikace automaticky přepne na záložní řešení. V tomto režimu klient pravidelně získává aktuální stav kvízu pomocí AJAX dotazů na Django server.

Tento mechanismus zajišťuje, že aplikace zůstane funkční i při problémech s real-time připojením. Aktualizace nejsou tak plynulé jako při použití Socket.IO, ale kvíz může bez problémů pokračovat.

3.7 VZDĚLÁVACÍ MATERIÁLY PŘES WAGTAIL CMS

Aplikace umožňuje učitelům vytvářet a spravovat vzdělávací materiály, které jsou propojené s konkrétními kvízy.

Vzdělávací materiály jsou spravovány pomocí systému Wagtail CMS. Učitelé mohou vytvářet různé typy materiálů, například text, video, externí odkaz nebo dokument. Ke každému materiálu lze nastavit, zda se má zobrazit před kvízem, po kvízu, nebo v obou případech. Materiály jsou vždy přiřazeny ke konkrétnímu kvízu.

Studentům se materiály zobrazují v přehledné podobě jako rozbalovací sekce. Vidí pouze ty materiály, které patří ke kvízu, kterého se účastní, a které jsou označeny jako publikované.

Vzdělávací materiály

Před začátkem kvízu si můžete prostudovat následující materiály:

The screenshot shows a Wagtail CMS page for a 'Text' material type. At the top, it says 'I/Y' and 'Typ: Textový materiál'. Below this is a section titled 'Klíčové principy psaní i/y' containing a bulleted list of rules for writing 'i' and 'y'. The list includes: 'Měkké souhlásky (c, j, ž, š, č, ř, d', t, ň)': Vždy měkké i (např. člověk, říka).; 'Tvrde souhlásky (h, ch, k, r, d, t, n)': Vždy tvrdé y (např. kytka, tygr).; 'D, T, N': Rozlišují se podle výslovnosti (např. dítka, tik vs. dým, tin).; 'Vyjmenovaná slova (po obojetných souhláskách b, f, l, m, p, s, v, z)': Píšeme y (např. byt, líny, my).; 'Koncovky': Záleží na vzoru (jarní/mladý) nebo shoda přísudku s podmětem (mužský rod životný = i, ostatní = y/e).'. Below this is another section titled 'Typy textů a cvičení' with a bulleted list: 'Doplňovačky': Vkládání i/y do slov (např. "kino" vs. "kytka").; 'Příklady podle vzorů': Procvičování koncovek ("jarní vs. mladý", "pěkní vs. mladí").; 'Texty s nevyjádřeným podmětem': Psaní i/y v přísudku (např. "Všichni jsme šli" - měkké i).; 'Přejatá slova': Výjimky (princ, kino).

Obrázek 3.12: Vzdělávací materiál

3.8 VIZUÁLNÍ STRÁNKA APLIKACE

Aplikace je navržena jako plně responzivní, takže se přizpůsobuje různým velikostem obrazovek. Rozhraní je použitelné na počítačích, tablettech i mobilních telefonech. Pro menší zařízení jsou upraveny rozměry prvků, velikosti písma a navigace, aby byla aplikace přehledná a dobře ovladatelná.

Součástí aplikace je také podpora světlého a tmavého režimu. Uživatel si může režim přepnout pomocí tlačítka v rozhraní a zvolený režim se uloží do prohlížeče, takže zůstane zachován i při dalším otevření aplikace. Barevné schéma je řešeno pomocí CSS proměnných, což zajistí jednotný vzhled celé aplikace.

Styly aplikace byly optimalizovány pomocí znovupoužitelných CSS tříd a proměnných. Díky tomu se podařilo výrazně zmenšit velikost CSS souboru a zjednodušit jeho údržbu.

4 VÝSLEDKY ŘEŠENÍ A VÝSTUPY

4.1 FUNKCE APLIKACE

4.1.1 Uživatelský průvodce pro učitele

Učitel se může do aplikace přihlásit nebo zaregistrovat buď klasicky pomocí e-mailu a hesla, nebo přes školní Microsoft účet. Po přihlášení má přístup ke všem funkcím pro správu kvízů.

V sekci „Moje kvízy“ může vytvořit nový kvíz. Zadá jeho název, přidá otázky s možnými odpověďmi a označí správné možnosti. U každé otázky nastaví čas na odpověď v rozmezí 5–300 sekund a také počet žolíků pro celý kvíz (0–3). Ke kvízu i jednotlivým otázkám může přidat obrázky.

Ke kvízu může učitel připojit vzdělávací materiály. V administraci vytvoří materiál, například text, video, dokument nebo externí odkaz. Materiál pak přiřadí ke konkrétnímu kvízu a nastaví, zda se má zobrazit studentům před kvízem, po kvízu, nebo v obou případech.

Hotový kvíz může učitel spustit dvěma způsoby. Bud' si ho spustí jen pro sebe v jednoduchém režimu, nebo vytvoří živé sezení pro studenty pomocí tlačítka „Start live“. Při živém sezení se vygeneruje unikátní kód, pomocí kterého se studenti připojí.

Během kvízu učitel sleduje průběžné statistiky odpovědí, aktuální žebříček a přehled toho, kdo jak odpověděl. Po dokončení všech otázek ukončí sezení a všem účastníkům se zobrazí finální výsledky s pódiem. Výsledky si může stáhnout do CSV souboru pro další zpracování.

4.1.2 Uživatelský pruvodce pro studenty

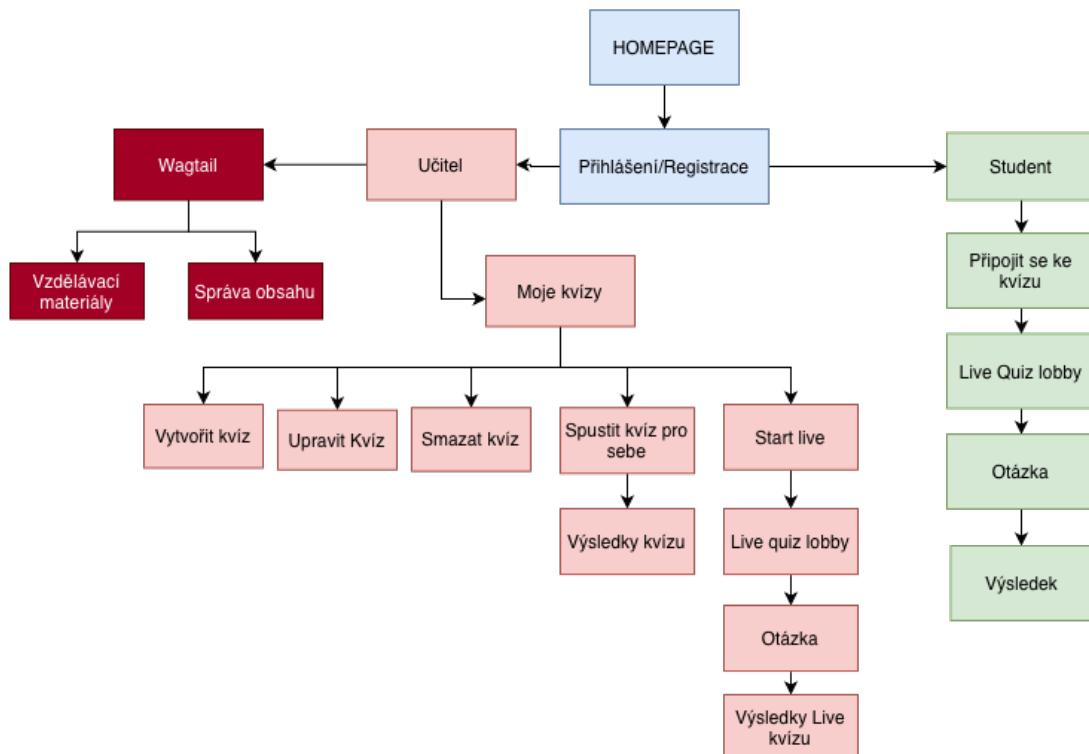
Student se může do aplikace přihlásit nebo zaregistrovat klasicky, případně pomocí školního Microsoft účtu. Po přihlášení se dostane na hlavní stránku, kde zadá kód kvízu, který obdrží od učitele.

Po zadání správného kódu se připojí do lobby. Zde vidí název kvízu, kód pro připojení a seznam ostatních účastníků. Pokud učitel přidal vzdělávací materiál, student si ho může v lobby otevřít a projít ještě před začátkem kvízu.

Jakmile učitel spustí otázku, zobrazí se studentovi text otázky, případně obrázek, možnosti

odpověď a odpočet času. Student vybírá jednu odpověď a odešle ji. Pokud má k dispozici žolík, může ho během odpovídání použít ke zjednodušení výběru.

Po odeslání odpovědi student vidí, kolik bodů za otázku získal, a také své celkové skóre. Po dokončení všech otázek se zobrazí finální výsledky s pódiem prvních tří míst a celkovým žebříčkem. Pokud jsou ke kvízu připojeny studijní materiály pro zobrazení po kvízu, student je zde může znova otevřít.



Obrázek 4.1: Struktura podstránek

4.2 SPLŇENÉ A NESPLŇENÉ CÍLE

Cílem práce bylo vytvořit funkční webovou aplikaci pro interaktivní kvízy a zároveň si při jejím vývoji osvojit a lépe pochopit principy a technologie, které byly v projektu použity. Většinu stanovených cílů se mi podařilo splnit a výsledná aplikace je funkční a použitelná. Přesto si uvědomuji, že kód není v některých částech napsaný zcela optimálně a do budoucna by bylo vhodné ho dále zpřehlednit a vylepšit.

V současné době je možné si aplikaci vyzkoušet jejím naklonováním z mého veřejného GitHub repozitáře a spuštěním přiložených příkazů. Díky využití Dockeru je aplikaci možné spustit na zařízeních s různými operačními systémy.

ZÁVĚR

Cílem práce bylo vytvořit webovou aplikaci pro interaktivní kvízy určenou pro školní prostředí. Aplikace slouží učitelům k tvorbě a správě kvízů a studentům k jejich vyplňování v reálném čase. Celé řešení je postaveno na framework Django a pro správu vzdělávacích materiálů využívá Wagtail CMS. Data jsou ukládána do databáze PostgreSQL a komunikace mezi uživateli probíhá pomocí technologie Socket.IO. Pro přihlašování uživatelů je použita autentizace přes Microsoft účet a aplikace je připravena ke spuštění pomocí Dockeru.

Základem aplikace jsou kvízy, které může učitel vytvářet, upravovat a spouštět. Během živého kvízu mají učitelé přehled o průběžných výsledcích a pořadí účastníků. Studenti se ke kvízům připojují pomocí kódu a odpovídají na otázky přímo v prohlížeči. Body jsou udělovány podle správnosti a rychlosti odpovědi. Součástí aplikace je také možnost využívat žolíky a zobrazovat si studijní materiály před nebo po skončení kvízu.

Aplikace je responzivní a lze ji používat na počítači i mobilním zařízení. Podporuje světlý i tmavý režim a umožňuje export výsledků do souboru CSV.

Aplikace je založována a dostupná ve veřejném GitHub repozitáři na adrese:

https://github.com/Stoklasavasek/Rocnikovy_Projekt

LITERATURA

- [1] *Django Documentation* [online]. Django Software Foundation, 2005- [cit. 2024-12-20]. Dostupné z: <https://docs.djangoproject.com/>
- [2] *Django Tutorial* [online]. Django Software Foundation, 2005- [cit. 2024-12-20]. Dostupné z: <https://docs.djangoproject.com/en/stable/intro/tutorial01/>
- [3] *django-allauth Documentation* [online]. [cit. 2024-12-20]. Dostupné z: <https://docs.allauth.org/>
- [4] *Socket.IO Documentation* [online]. Socket.IO, 2014- [cit. 2024-12-20]. Dostupné z: <https://socket.io/docs/v4/>
- [5] *python-socketio Documentation* [online]. [cit. 2024-12-20]. Dostupné z: <https://python-socketio.readthedocs.io/>
- [6] *Wagtail CMS Documentation* [online]. Wagtail, 2014- [cit. 2024-12-20]. Dostupné z: <https://docs.wagtail.org/>
- [7] *PostgreSQL Documentation* [online]. PostgreSQL Global Development Group, 1996- [cit. 2024-12-20]. Dostupné z: <https://www.postgresql.org/docs/>
- [8] *Microsoft identity platform documentation* [online]. Microsoft, 2020- [cit. 2024-12-20]. Dostupné z: <https://learn.microsoft.com/en-us/entra/identity-platform/>
- [9] *Docker Documentation* [online]. Docker, Inc., 2013- [cit. 2024-12-20]. Dostupné z: <https://docs.docker.com/>
- [10] *Real Python - Django Tutorials* [online]. Real Python, 2012- [cit. 2024-12-20]. Dostupné z: <https://realpython.com/tutorials/django/>
- [11] *W3Schools Online Web Tutorials* [online]. W3Schools, 1998- [cit. 2024-12-20]. Dostupné z: <https://www.w3schools.com/>
- [12] *MDN Web Docs* [online]. Mozilla Foundation, 2005- [cit. 2024-12-20]. Dostupné z: <https://developer.mozilla.org/>
- [13] *Stack Overflow* [online]. Stack Exchange, 2008- [cit. 2024-12-20]. Dostupné z: <https://stackoverflow.com/>