DELTA - Střední škola informatiky a ekonomie, s.r.o.
Ke Kamenci 151, Pardubice

MATURITNÍ PROJEKT

TenFingers

Příjmení, jméno: Kárský Lukáš
Třída: 4.A
Studijní obor: Informační technologie 18-20-M/01
školní rok: 2024/2025

Zadání maturitního projektu z informatických předmětů

Jméno a příjmení: Lukáš Kárský
Pro školní rok: 2024/2025
Třída: 4.A
Obor: Informační technologie 18-20-M/01

Téma práce: Interaktivní výuka psaní všemi deseti prsty pomocí miniher
Vedoucí práce: RNDr. Jan Koupil, Ph.D.

Způsob zpracování, cíle práce, pokyny k obsahu a rozsahu práce:
Cíl projektu:
Cílem projektu je vytvořit interaktivní webovou aplikaci, která bude sloužit k výuce psaní všemi deseti prsty. Aplikace bude využívat různé minihry a interaktivní cvičení, aby byla výuka zábavná a efektivní.
Specifikace projektu:

1. **Analýza a návrh:**
    - Bude provedena analýza existujících metod a nástrojů pro výuku psaní všemi deseti prsty.
    - Budou definovány požadavky na systém a vytvořen návrh uživatelského rozhraní.
    - Bude navržena databázová struktura pro ukládání uživatelských profilů a výsledků cvičení.
2. **Vývoj uživatelského rozhraní:**
    - Bude vytvořeno přehledné a intuitivní webové rozhraní, které uživatelům umožní snadný přístup ke všem funkcím aplikace.
    - Budou implementovány různé úrovně obtížnosti a možnost uživatele monitorovat svůj pokrok.
3. **Minihry a interaktivní cvičení:**
    - Bude vyvinuto několik miniher zaměřených na zlepšení rychlosti a přesnosti psaní.
    - Minihry budou navrženy tak, aby byly zábavné a motivující, s postupně se zvyšující obtížností.
    - Bude implementován systém zpětné vazby, který bude uživatelům poskytovat informace o jejich pokroku a chybách.

4. **Technické požadavky:**
    - Aplikace bude navržena s modulární architekturou, která umožní snadné přidávání nových miniher a cvičení.
    - Budou implementována bezpečnostní opatření pro ochranu uživatelských dat.

Požadované výstupy:

- Funkční webová aplikace pro výuku psaní všemi deseti prsty.
- Sada miniher a interaktivních cvičení zaměřených na zlepšení rychlosti a přesnosti psaní.
- Uživatelská a vývojářská dokumentace.
- Prezentace a případně demonstrační video.

Hodnocení: Projekt bude hodnocen na základě následujících kritérií:

- Kvalita a funkčnost uživatelského rozhraní.
- Inovativnost a zábavnost miniher a cvičení.
- Úroveň dokumentace a prezentace projektu.
- Kreativita a efektivnost řešení.

Stručný časový harmonogram (s daty a konkretizovanými úkoly):

- **Září:** Analýza problému, tvorba UML usecase diagramů, návrh vzhledu aplikace, návrh konceptu miniher, návrh postupu po „mapě klávesnice“
- **Říjen:** Vývoj – výukových miniher (frontend)
- **Listopad:** Vývoj backendové části aplikace (registrace a autentizace, logování výsledků, monitorování průchodu)
- **Prosinec:** Vývoj frontendové části, integrace dat z backendu a připravených her
- **Leden:** Finalizace projektu, debugging
- **Únor - březen:** Práce na dokumentaci

Prohlášení

Prohlašuji, že jsem svou práci vypracoval samostatně a použil jsem pouze prameny a literaturu uvedené v seznamu bibliografických záznamů.

V Pardubicích dne _____ _____

Lukáš Kárský

Poděkování
Děkuji vedoucímu mého projektu RNDr. Jan Koupil, Ph.D. za vedení mého projektu, rady, nápady a hlavně trpělivost. Také bych chtěl poděkovat kamarádům a rodině za podporu.

Abstrakt
Tato práce dokumentuje vývoj a používání webové aplikace určené pro samovýuku psaní všemi deseti. Aplikace je vytvořena v Next.js a na backend využívá Supabase. Uživatelé mohou v aplikaci plnit psací cvičení, která pomáhají osvojit si techniku psaní všemi deseti, a také hrát minihry, které slouží k procvičování psaní a rozvoji svalové paměti. V současné době aplikace obsahuje čtyři minihry.

Na svém profilu si uživatel může zobrazit průměrné statistiky z psacích cvičení i podrobné statistiky jednotlivých cvičení.

Klíčová slova
React, Webová aplikace, Next.JS, Typescript, Supabase, TailwindCSS, Unity, C#

Abstract
This work documents the development and use of a web application designed for self-learning touch typing. The application is built with Next.js and uses Supabase for the backend. Users can complete typing exercises that help them master touch typing techniques and play mini-games designed to practice typing and develop muscle memory. Currently, the application includes four mini-games.

On their profile, users can view average statistics from typing exercises as well as detailed statistics for individual exercises.

Key words
React, Web application, Next.JS, Typescript, Supabase, TailwindCSS, Unity, C#

Obsah

1 Úvod **8**

1.1 Psaní všemi deseti 8

1.1.1 Svalová paměť 8

1.2 Existující řešení 9

1.2.1 MonkeyType 9

1.2.2 Nedatluj 9

2 Pojmy **10**

2.1 Protected routes 10

2.2 Middleware 10

2.3 React komponenta 10

2.4 Callback funkce 10

2.5 Parent komponenta 10

3 Use case diagram **11**

4 Architektura projektu **12**

4.1 Next.js 13

4.2 Vercel 13

4.3 Supabase 13

5 Databáze **14**

5.1 Struktura databáze 14

5.2 Práce s databází 15

6 Webová aplikace **16**

6.1 Registrace 16

6.2 Přihlašování 17
6.3 Přihlášený / Nepřihlášený uživatel 17
6.4 Psací cvičení 18
6.5 Profil 20
6.6 Minihry 20
6.6.1 Drop Tiles 20
6.6.2 Sandbox 21
6.6.3 Quotes 22
6.6.4 Shape Strike 22
6.7 Administrátor 23
6.7.1 Editor cvičení 23

7 Závěr **24**

1 Úvod

V dnešní době se počítače používají téměř ve všech oblastech života, a proto je důležité umět je efektivně ovládat. Jednou z klíčových dovedností je rychlé a přesné psaní na klávesnici, které může výrazně zvýšit produktivitu a usnadnit práci s digitálními technologiemi.

Tento dokument popisuje webovou aplikaci zaměřenou na výuku psaní všemi deseti. Cílem projektu je pomoci uživatelům zlepšit rychlost a přesnost psaní na klávesnici prostřednictvím interaktivních psacích cvičení a miniher. Aplikace je určena jak pro začátečníky, kteří se chtějí naučit správnou techniku psaní, tak pro pokročilé uživatele, kteří si chtějí zdokonalit své dovednosti.

Hlavní funkce aplikace zahrnují:

- Strukturovaná psací cvičení zaměřená na jednotlivé klávesy a kombinace písmen
- Minihry
- Sledování pokroku a statistiky výkonu uživatele
- Možnost přizpůsobení obtížnosti a obsahu miniher

1.1 Psaní všemi deseti

Psaní všemi deseti je metoda psaní na klávesnici, která využívá všech prstů a umožňuje rychlejší a efektivnější práci s textem. Díky této technice uživatel ztrácí potřebu sledovat klávesnici a může tak dosáhnout vyšší přesnosti a rychlosti při psaní. Základem této metody je správné umístění prstů na tzv. domovské řadě klávesnice a jejich systematické používání při psaní. Výuka této metody většinou začíná psaním jednoho písmene a postupným nabalováním dalších písmen. Tento proces pomáhá rozvíjet svalovou paměť.

1.1.1 Svalová paměť

Svalová paměť hraje klíčovou roli při osvojování psaní všemi deseti. Jedná se o proces, kdy si mozek a svaly postupně zapamatují pohyby prstů na klávesnici, což umožňuje automatické a rychlé psaní bez nutnosti vědomého přemýšlení o poloze jednotlivých kláves. Pokud uživatel nebude nějakou dobu psát všemi deseti, svalová paměť napomůže k tomu, aby se k dovednosti rychle vrátil, i když může zpočátku pociťovat menší nejistotu nebo zpomalení. [1]

1.2 Existující řešení

Protože webové aplikace na psaní všemi deseti již existují, představím zde některé, ze kterých jsem si vzal inspiraci.

1.2.1 MonkeyType

MonkeyType není webová aplikace zaměřená na výuku psaní všemi deseti. Jak uvádí oficiální web, „MonkeyType je minimalistický a přizpůsobitelný test psaní“ [2]. Umožňuje uživatelům měřit rychlost a přesnost psaní a podporuje různé jazyky, včetně programovacích. Ale v aplikaci nenajdete žádné psací cvičení, které vás postupně naučí psaní všemi deseti. MonkeyType je vyvíjen skupinou dobrovolníků a je zdarma bez potřeby vytvoření účtu.

1.2.2 Nedatluj

Nedatluj.cz je online webová aplikace, která lidem umožňuje se naučit psát všemi deseti prsty. Je tvořena základním kurzem, který poskytuje interaktivní prostředí pro nauku správného prstokladu a sérii doplňujících cvičení, která Uživateli umožňuje procvičovat rychlost a správnost svého psaní na klávesnici.

Aplikace nabízí tři typy uživatelských účtů. Bezplatný uživatelský účet Nedatluj, který obsahuje první 4 klávesy kurzu a ukázku z tréninku. Placený účet Nedatluj s plným přístupem veškerého obsahu a školní účet pro učitele nebo žáka. [3]

2 Pojmy

2.1 Protected routes

V Next.js je middleware pro chráněné trasy výkonná funkce, která slouží k zabezpečení určitých stránek nebo tras. Zajišťuje, že přístup k těmto stránkám mají pouze autorizovaní uživatelé, a to kontrolou podmínek, jako jsou autentizační tokeny nebo uživatelské role, ještě před vykreslením stránky. Tímto způsobem lze snadno spravovat autentizaci a autorizaci v celé aplikaci.[4]

2.2 Middleware

Middleware vám umožňuje spustit kód před dokončením požadavku. Na základě příchozího požadavku pak můžete upravit odpověď přepisováním, přesměrováním, modifikováním hlaviček požadavku nebo odpovědi, nebo odpovědět přímo. [5]

2.3 React komponenta

React komponenta je znovupoužitelný kus kódu, který je zodpovědný za vykreslení části uživatelského rozhraní. Tyto komponenty mohou být jednoduché, jako tlačítko, nebo složité, jako celý formulář. Díky modulární povaze React komponent mohou vývojáři rozdělit své aplikace na spravovatelné a znovupoužitelné části, což podporuje čistý a organizovaný kód. [6]

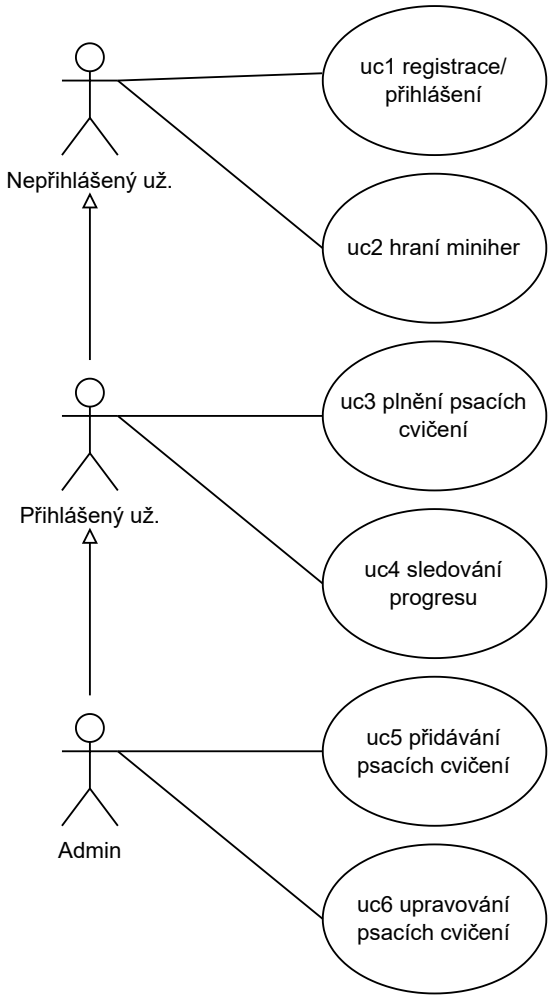
2.4 Callback funkce

Funkce zpětného volání (callback) je funkce, která je předána jiné funkci jako argument a následně je uvnitř této vnější funkce vyvolána za účelem dokončení určité operace nebo akce. [7]

2.5 Parent komponenta

V Reactu je parent komponenta taková komponenta, která obsahuje jednu nebo více vnořených child komponenta. Parent komponenta je zodpovědná za vykreslování child komponenta a předávání dat prostřednictvím props.[8]

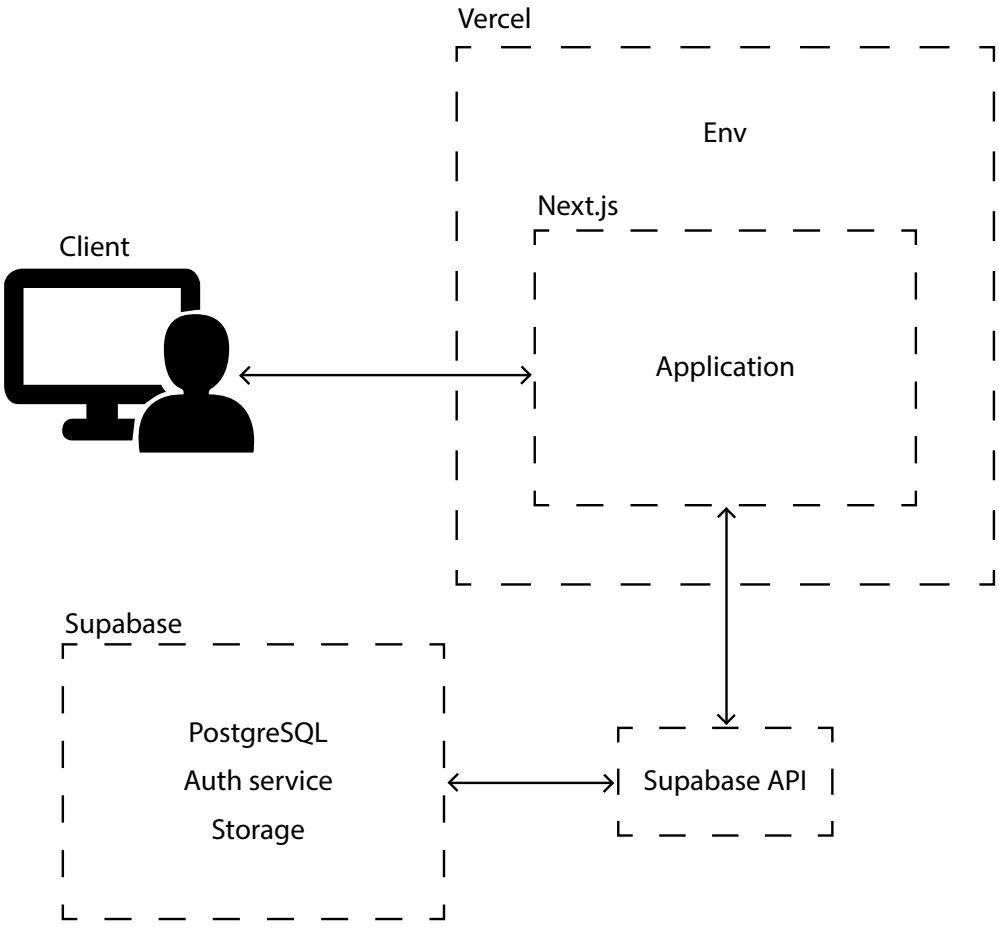
3 Use case diagram



Obrázek 1: Use case diagram

Popis use case daigramu na obrázku 1:
uc1 registrace/přihlášení: Nepřihlášený uživatel se může zaregistrovat nebo přihlásit.
uc2 hraní miniher: Nepřihlášený uživatel má možnost hrát minihry bez ukládání skóre.
uc3 plnění psacích cvičení: Přihlášený uživatel může psát psací cvičení.
uc4 sledování progresu: Přihlášený uživatel může na profilu sledovat statistiky
uc5 přidávání psacích cvičení: Administrátor přidává psací cvičení.
uc6 upravování psacích cvičení: Administrátor upravuje psací cvičení.

4 Architektura projektu



Obrázek 2: Struktura aplikace

Projekt běží na serverless architektuře, což znamená, že neobsahuje vlastní backend, ale využívá službu Supabase. Na obrázku 2 je znázorněna komunikace mezi jednotlivými prvky.

Klient komunikuje s aplikací hostovanou na Vercelu přes webový prohlížeč a aplikace komunikuje se službou Supabase přes její API.

4.1 Next.js

Next.js je framework postavený na Reactu, který slouží k tvorbě fullstack webových aplikací. Pro vytváření uživatelského rozhraní se používají React komponenty, zatímco Next.js přidává funkce, jako je například serverové vykreslování (SSR), statická generace (SSG), optimalizace obrázků nebo navigace.

Na pozadí Next.js automaticky nastavuje a spravuje potřebné nástroje, jako je balíčkování, kompilace a další. [9]

4.2 Vercel

Vercel je cloudová platforma, která umožňuje vývojářům vytvářet, nasazovat a spravovat webové aplikace. Specializuje se na hostování front-endových frameworků a statických webů a nabízí funkce jako okamžité nasazení a automatické škálování. Díky integraci s platformami jako GitHub, GitLab a Bitbucket umožňuje nepřetržité nasazování a týmovou spolupráci. Serverless architektura Vercelu zajišťuje automatické škálování aplikací podle aktuální poptávky, aniž by bylo nutné ruční nastavení.

Vercel také stojí za vývojem Next.js, open-source React frameworku. [10]

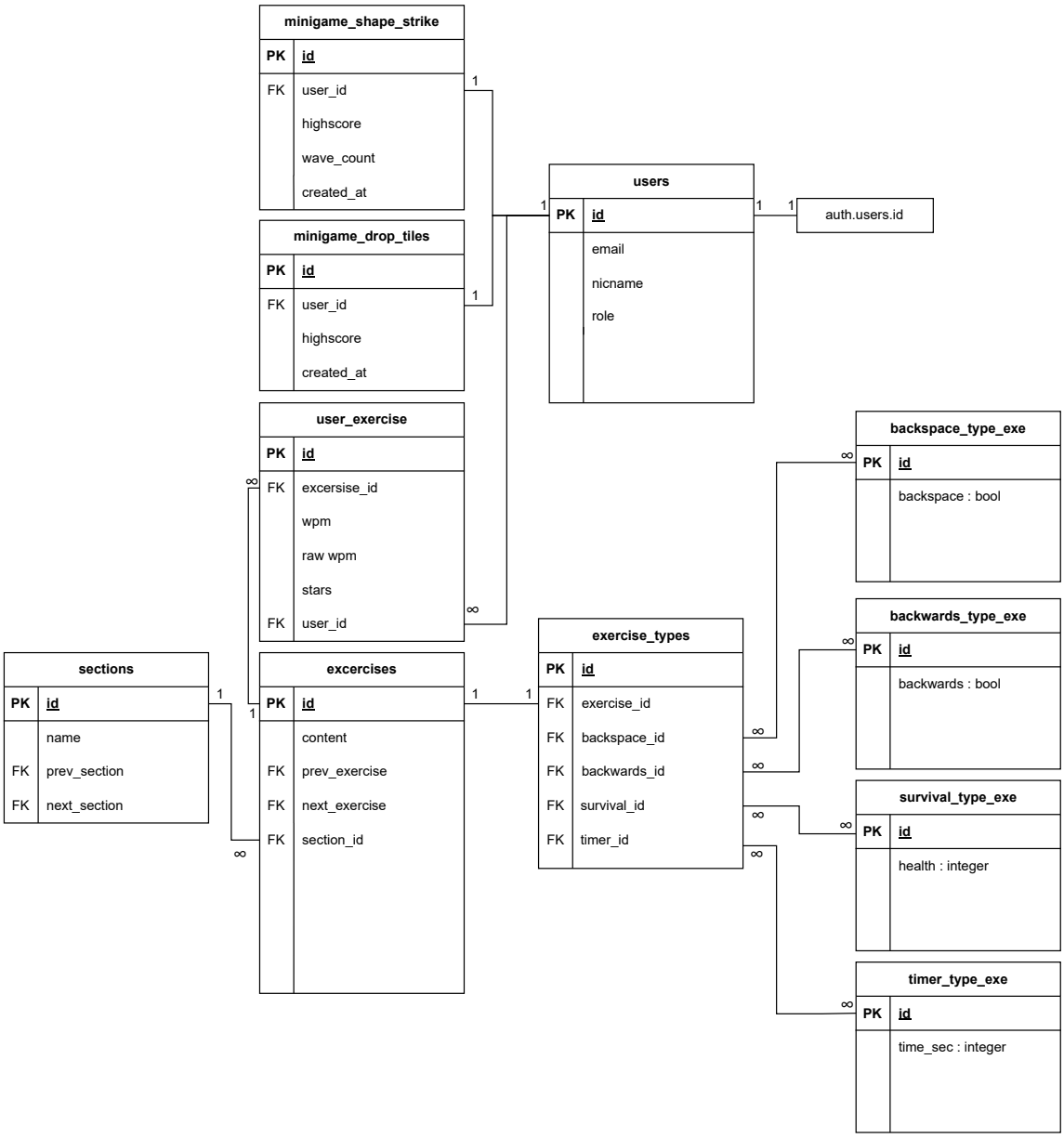
4.3 Supabase

Supabase je open-source backend služba, která poskytuje vývojářům předpřipravenou infrastrukturu pro vytváření webových a mobilních aplikací. Nabízí PostgreSQL databázi s automaticky generovanou REST API, autentizační systém s podporou více poskytovatelů, realtime subscriptions pro okamžité aktualizace dat, integrované úložiště a další. [11]

5 Databáze

V aplikaci je potřeba ukládat velké množství dat, například uživatelské údaje nebo jednotlivá psací cvičení. Tato data jsou ukládána do relační databáze PostgreSQL, která je hostována backend službou Supabase.

5.1 Struktura databáze



Obrázek 3: Struktura databáze

Přehled tabulek na obrázku 3:

- **users** - Údaje o uživatelích
- **minigame_shape_strike** - Záznamy z minihry „Shape Strike“
- **minigame_drop_tiles** - Záznamy z minihry „Drop Tiles“
- **sections** - Sekce, do kterých patří jednotlivá cvičení
- **exercises** - Jednotlivá cvičení
- **user_exercise** - Výsledky napsaných cvičení
- **exercise_types** - Typy cvičení s různými vlastnostmi
- **backspace_type_exe** - Typ cvičení s pravidlem pro backspace
- **backwards_type_exe** - Typ cvičení s pravidlem pro psaní pozpátku
- **survival_type_exe** - Typ cvičení s herním prvkem přežití
- **timer_type_exe** - Typ cvičení s časovým limitem

5.2 Práce s databází

Data z databáze získávám pomocí automaticky generované REST API.

```
export async function UserQuerry(){
    const supabase = createClient();

    const { data, error } = await supabase.from('users').select('nickname,email')

    if(error) return;
    return data;
}
```

Obrázek 4: Získání uživatelských dat z databáze

Na obrázku 4 je vidět, jak z databáze získávám data o uživateli. Nejprve vytvořím objekt supabase, který mi následně umožňuje komunikaci se vším, co Supabase nabízí.

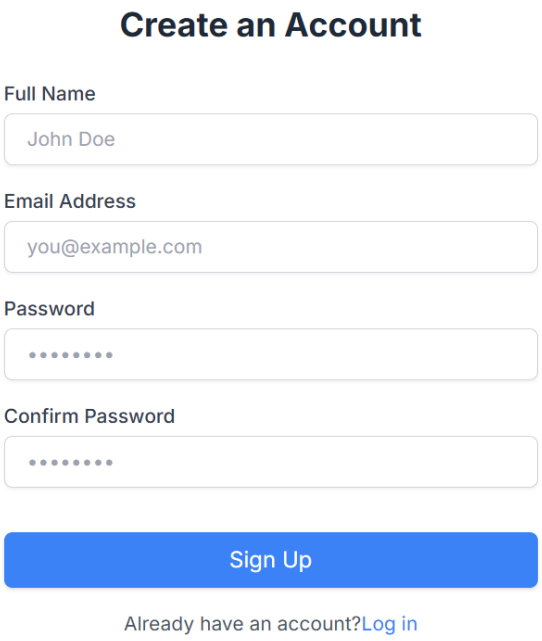
Pro získání dat musím specifikovat název tabulky – v tomto případě „users“ a následně sloupce z tabulky „nickname“ a „email“. Tento dotaz by vrátil všechny uživatele, ale díky RLS (Row-Level Security), která umožňuje uživateli přístup pouze k jeho vlastním datům, vrátí výsledek pouze údaje o daném uživateli.

6 Webová aplikace

Tato část dokumentace popisuje, jak funguje celá webová aplikace. Sekce zahrnuje vše od registrace s přihlášením až po jednotlivé minihry.

6.1 Registrace

Pro registraci aplikace využívá Supabase authentication. Supabase potřebuje email a heslo, které musí uživatel zadat do připraveného formuláře 5. Formulář je ještě rozšířen o pole „Full Name“, které Supabase přijímá jako volitelný údaj, ale ve formuláři je povinné.

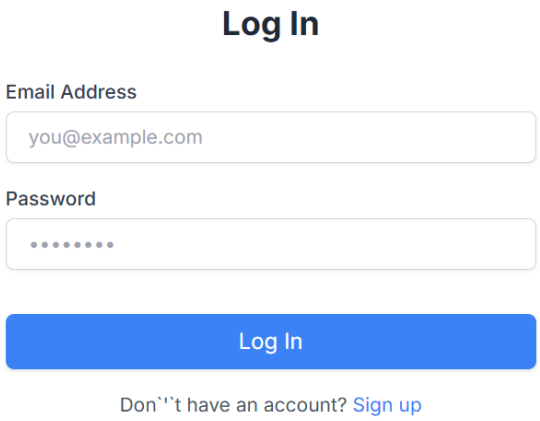


Obrázek 5: Registrační formulář

Po vyplnění se zavolá serverová funkce, která dostane data z formuláře. Funkce zkontroluje data a pokud najde chybu tak ji vypíše. Například, pokud uživatel zadá krátké heslo, zobrazí se mu zpráva, že heslo musí obsahovat alespoň 8 znaků. Když funkce nenajde žádnou chybu, zavolá Supabase funkci signUp a přesměruje přihlášeného uživatele na hlavní stránku.

6.2 Přihlašování

Pro přihlašování je opět připravený formulář 6. Uživatelovi tentokrát stačí vyplnit jen email a heslo.



Obrázek 6: Přihlašovací formulář

Po vyplnění se zavolá serverová funkce, která dostane data z formuláře. Funkce zavolá Supabase funkci signInWithPassword, které předá data a aplikace čeká na to, jestli funkce vrátí chybu. Pokud chybu nevrátí uživatel je přihlášen a přesměrován na hlavní stránku. Když funkce vrátí chybu, vypíše uživateli, že něco zadal špatně a musí zadat údaje znovu.

Z aplikace se uživatel může odhlásit stisknutím tlačítka Sign Out v hlavní nabídce. Stisknutí tlačítka spustí serverovou funkci, která zavolá Supabase funkci signout.

6.3 Přihlášený / Nepřihlášený uživatel

Aplikace se dělí na dvě části, první část tvoří psací cvičení a druhá část obsahuje minihry. Aby mohl uživatel absolvovat psací cvičení, musí být přihlášený. Výsledky psacích cvičení se totiž ukládají s identifikátorem uživatele (aby bylo možné určit, který výsledek patří kterému uživateli) a díky tomu je možné zjistit, na jakém cvičení uživatel skončil a na jakém má pokračovat. Minihry lze hrát i bez přihlášení, protože nepotřebují znát žádné uživatelské informace.

Tady vzniká problém, protože je nutné zajistit, aby nepřihlášený uživatel neměl přístup na určité stránky. Tento problém se řeší pomocí chráněných tras (protected routes 2.1), které umožňují omezit přístup na základě stavu přihlášení uživatele. V middleware 2.2 se zjišťuje, jestli je uživatel přihlášený, a pokud ne, je přesměrován na přihlašovací stránku. Jak je vidět na obrázku 7.

```
const {
  data: { user },
} = await supabase.auth.getUser()

if (user) {
  const data = await supabase.from('users').select('role').eq('id', user.id)
  if(request.nextUrl.pathname.startsWith('/admin') && data.data && data.data[0].role !== 'admin'){
    const url = request.nextUrl.clone()
    url.pathname = '/'
    return NextResponse.redirect(url)
  }
}


if (
  !user &&(
  request.nextUrl.pathname.startsWith('/sections') ||
  request.nextUrl.pathname.startsWith('/profile')||
  request.nextUrl.pathname.startsWith('/exercise/*')||
  request.nextUrl.pathname.startsWith('/admin')
  )
) {
  // redirect user for login
  const url = request.nextUrl.clone()
  url.pathname = '/login'
  return NextResponse.redirect(url)
}
```

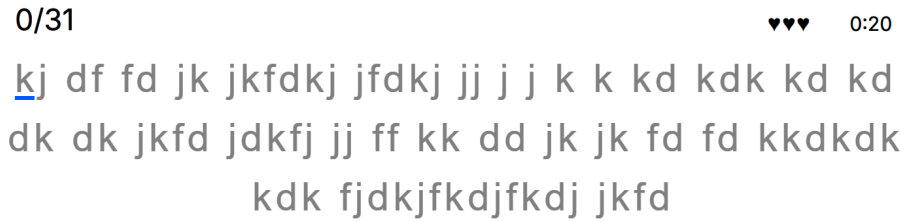
Obrázek 7: Middleware

Přístup k administrační stránce je řešen obdobně, ovšem vyhodnocuje se také role uživatele uložená v databázi.

6.4 Psací cvičení

Cvičení mají standardní formát výuky psaní všemi deseti. Jsou rozdělena do sekcí po čtyřech, někdy třech písmenech, přičemž každé písmeno má čtyři cvičení. Po dokončení čtyř cvičení se k písmenu přidá další. Písmena jsou rozdělena tak, aby se uživatel nejdříve naučil psát prostřední řadu klávesnice („home row“), poté horní řadu a nakonec spodní řadu písmen. Cvičení také mohou obsahovat nějaké „výzvy“. Výzvy jsou zatím čtyři: omezený čas, psaní pozpátku, zakázaný backspace a omezený počet chyb.

Po rozkliknutí cvičení se z databáze získá text cvičení a pokusí se načíst hodnoty výzev. Pokud cvičení obsahuje výzvy, použije se jejich hodnota, jinak se přiřadí prázdná hodnota. Po načtení hodnot se zavolá React komponenta2.3 TypingText, která přijímá text a výzvy jako parametry. V komponentě se text rozdělí na řádky a vykreslí. Výzvy se spustí podle toho, zda mají nějakou hodnotu, nebo jsou prázdné.



Obrázek 8: Psací cvičení

Na obrázku 8 je psací cvičení z první sekce. V levém horním rohu je zobrazen počet slov ve cvičení a napravo sekce pro zobrazení výzev. Cvičení na obrázku 8 obsahuje dvě výzvy: omezený čas, který se začne odpočítávat, jakmile uživatel začne psát, a omezený počet chyb. Zbytek tvoří text, který musí uživatel napsat, aby cvičení úspěšně dokončil. Písmeno, které právě píše, je podtrženo.

Jakmile uživatel dopíše nebo poruší nějakou výzvu, dostane se na vyhodnocování. Aplikace vyhodnocuje rychlost, přesnost a chybovost. Data pro vyhodnocování jsou z callback funkce 2.4 OnCompletion, která se zavolá v TypingText komponentě. Komponenta TypingText uchovává všechna data o psaní, například chyby a čas. Po dopsání tato data předá jako argument do funkce OnCompletion, která je předá do parent komponenty 2.5. Tam se data následně vloží jako argument do komponenty TypingTextStats, která z nich vypočítá zajímavé statistiky, např. počet napsaných slov za minutu a zobrazí je uživateli. Způsob výpočtu slov za minutu můžete vidět na obrázku 9.

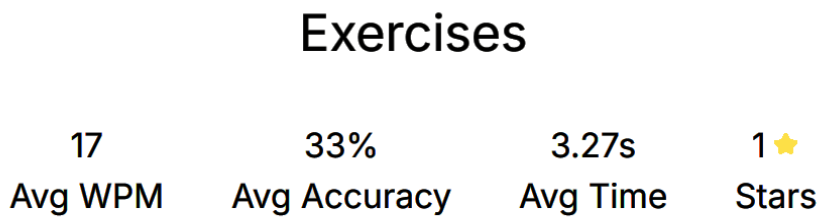
```
export function calculateWPM(startTime:number, endTime:number, words:number){
    return Math.round(words / ((endTime - startTime) / 60000));
}
```

Obrázek 9: Funkce na výpočet slov za minutu

Současně s vyhodnocováním jsou data ukládána do databáze. Pokud uživatel dané cvičení již dříve dokončil, aplikace porovná předchozí a nové výsledky a uloží lepší z nich.
K ukládání dat do databáze se využívají serverové funkce InsertUserExercise a UpdateUserExercise.

6.5 Profil

K profilu má přístup pouze přihlášený uživatel. Na profilu si může zobrazit své údaje, přičemž nejzajímavější částí jsou výsledky z psacích cvičení.



Obrázek 10: Statistiky

Výsledky se počítají z uživatelem dokončených cvičení. Pokaždé, když přejde na profil, načtou se z databáze výsledky a vypočítají se statistiky zobrazené na obrázku 10.

6.6 Minihry

Minihry neslouží k výuce psaní všemi deseti, ale spíš k procvičování a trénování svalové paměti.

6.6.1 Drop Tiles

Minihra Drop Tiles je inspirovaná hrou Piano Tiles. Ve hře se objevují padající písmena a uživatel musí stisknout správnou klávesu. Hra končí, pokud uživatel stiskne klávesu, která není na obrazovce, nebo pokud nechá písmeno spadnout mimo obrazovku.

V minihře lze zvolit jednu ze tří obtížností: Easy, Medium nebo Hard. Aktuálně obtížnost ovlivňuje pouze počáteční rychlost padání písmen a násobitel skóre.

Písmena padají v osmi sloupcích a každý sloupec může obsahovat pouze písmena odpovídající prstu podle pravidel psaní všemi deseti. Například levý malíček obsluhuje písmena: „Q, A, Z/Y“, což znamená, že v levém sloupci mohou padat pouze tato písmena.
V minihře běží interval, který každou sekundu vybere náhodné písmeno a přiřadí ho do odpovídajícího sloupce.

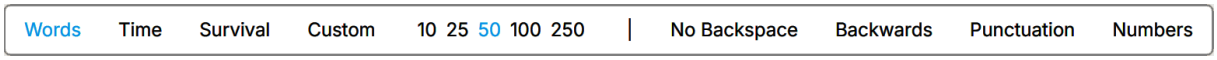
Písmena se pohybují pomocí CSS. Padání funguje prostřednictvím funkce requestAnimationFrame, která volá funkci updateTiles. Ve funkci updateTiles se vypočítá časový rozdíl mezi snímky, projde se pole obsahující všechna padající písmena a jejich pozice se upraví přičtením hodnoty rychlost × čas mezi snímky.

Pro získání skóre musí uživatel stisknout klávesu ve vyznačené oblasti. Uvnitř této oblasti je specifický bod, kde lze dosáhnout perfektního skóre. Výpočet skóre zajišťuje funkce handleScore, která na základě pozice určuje, zda bylo písmeno při stisku klávesy v požadované oblasti. Pokud bylo písmeno stisknuto ve vyznačené oblasti, kombo se zvýší o jedna a ke skóre se přičte hodnota číslo × kombo. V opačném případě se kombo vynuluje.

Po dokončení hry se skóre přihlášeného uživatele uloží do databáze pouze v případě, že dosud žádné skóre nemá, nebo pokud je nové skóre vyšší než předchozí.

6.6.2 Sandbox

Sandbox je psací cvičení, které si uživatel může upravit podle svých potřeb. Nabízí širokou škálu možností, jak si cvičení přizpůsobit můžete vidět na obrázku 11.



Obrázek 11: Konfigurátor textu

Uživatel si může nastavit počet slov vygenerovaných v textu, zvolit jazyk nebo si nechat vytvořit text obsahující pouze vybraná písmena k procvičování. Dále si může aktivovat výzvy, které jsou součástí psacích cvičení.

Sandbox využívá TypingText komponenety pro zobrazování textu. Při každé změně parametru, například délky nebo jazyka, se vygeneruje nový text pomocí funkce generateStringJson nebo generateStringCustom. Tento nový text se následně uloží do useState React hooku, čímž se aktualizuje obsah komponenty TypingText a zobrazí se nový text.

Funkce generateStringJson generuje text z uživatelem zvoleného jazyka. Prochází JSON soubor a náhodně vybírá slova, dokud nedosáhne požadovaného počtu slov.

Funkce generateStringCustom generuje text z písmen zadaných uživatelem. Náhodně vybírá písmena a skládá z nich slova s proměnlivou délkou, dokud není dosaženo požadovaného počtu slov.

Po dopsání textu se uživateli vykreslí TypingTextStats komponent, který dostane data z TypingText komponentu a ukáže uživateli statistiky z psaní.

6.6.3 Quotes

Quotes je minihra, kde uživatel opisuje hlášky z her, filmů nebo od významných lidí. Hlášky se generují podle uživatelem vybraného jazyka. TypingText komponent dostane náhodnou hlášku vygenerovanou z JSON souboru, kterou zobrazí uživateli. Ten si potom může vybrat jestli si nechá vygenerovat novou nebo ji napapíše. Po napsání hlášky, TypingTextStats komponent zobrazí statisktiky.

6.6.4 Shape Strike

Shape Strike je minihra vytvořená v Unity a exportovaná do WebGL. Z pravé strany obrazovky se postupně objevují nepřátelé se svými názvy. Hráčovým úkolem je co nejrychleji napsat správné jméno nepřítele, než dosáhne druhé strany obrazovky. Za každé správně napsané jméno získává hráč skóre podle počtu znaků ve jméně.

Celá minihra funguje na systému vln. V každé vlně se zobrazí předem určený počet nepřátel, kteří jsou definováni na začátku vlny. S rostoucím počtem přežitých vln se mohou objevit nepřátelé s různými vlastnostmi. Na začátku každé vlny se vygeneruje pole nepřátel, které obsahuje informace o tom, kteří nepřátelé se mají zobrazit. Toto pole se následně postupně prochází, takže nepřátelé se objevují v pořadí, v jakém jsou v poli uvedeni.

Pole nepřátel se generuje podle systému nakupování. Na začátku každé vlny se zjistí, kolik vln hráč přežil, a na základě toho systém obdrží určitou částku „peněz“. Systém poté náhodně vybírá z dostupných nepřátel, přičemž každý nepřítel má svou cenu. Pokaždé, když je nepřítel vybrán, přidá se do pole a z rozpočtu se odečte jeho cena. Tento proces se opakuje, dokud systém nevyčerpá všechny dostupné peníze na nákup nepřátel pro danou vlnu.

Systém zobrazování nepřátel vybere jeden ze tří bodů, které jsou umístěny mimo pravou stranu obrazovky. Poté si zvolí náhodný časový interval mezi 0,5 s a 3 s. Po uplynutí tohoto času se nepřítel objeví na jednom z vybraných bodů a začne se pohybovat směrem doleva. Tento proces se opakuje pro každého nepřítele ve vlně.

Pokud se nepřítel dostane na levou stranu obrazovky, hráči se zobrazí nová scéna se statistikami. Tato scéna ukazuje počet zabitých nepřátel podle jednotlivých typů, celkové skóre a číslo vlny, ve které hráč skončil. K dispozici je také tlačítko Restart, které hráče vrátí na resetovanou herní scénu, kde může začít novou hru od první vlny.

Pro zobrazování WebGL v Reactu se používá balíček React Unity WebGL. Do hooku useUnityContext se přidají odkazy na WebGL build, a následně se tento hook předá jako argument do komponenty Unity, která poté spustí a zobrazí hru. [12]

6.7 Administrátor

Administrátor má v aplikaci přístup k admin stránce, na kterou se dostane po přihlášení k účtu s rolí administrátora. Na této stránce může využívat admin nástroje, přičemž aktuálně je k dispozici editor psacích cvičení.

6.7.1 Editor cvičení

Editor psacích cvičení je nástroj, který umožňuje přímo v aplikaci přidávat, upravovat a odebírat psací cvičení. Díky němu není nutné zasahovat přímo do databáze, což pomáhá předcházet chybám. Jelikož psací cvičení a jejich sekce jsou strukturovány jako linked list, takže při přidávání cvičení doprostřed seznamu je nutné upravit i ostatní prvky, aby se neporušilo správné propojení. Editor tento proces automaticky spravuje, čímž zjednodušuje práci administrátorům.

Nástroj také obsahuje generátor textu, který usnadňuje tvorbu obsahu. Administrátor si vybere délku textu, jazyk a sadu písmen, ze kterých se mají slova skládat. Generátor následně projde JSON soubor a vybere slova, která se skládají jen z vybraných písmen. Ze slov následně vytvoří text, dlouhý podle zadané délky textu.

7 Závěr

Při práci na projektu byly analyzovány a prozkoumány existující metody výuky psaní všemi deseti, včetně dostupných online nástrojů a softwarových řešení. Na základě této rešerše byla navržena a implementována vlastní webová aplikace, která se zaměřuje na interaktivní a efektivní výuku psaní.

K ukládání uživatelských dat, autentizaci a správě obsahu využívá Supabase. Tento backend poskytuje robustní databázové řešení postavené na PostgreSQL, které umožňuje efektivní správu uživatelských účtů, výsledků cvičení a statistik.

V současné době lze plnit psací cvičení a hrát minihry. Aministrátor může přidávat a upravovat psací cvičení. Uživatel si může zobrazit průměrné statistiky z všech psacích cvičení i podrobné statistiky u jednotlivého cvičení.

Seznam obrázků

1 Use case diagram 11

2 Struktura aplikace 12

3 Struktura databáze 14

4 Získání uživatelských dat z databáze 15

5 Registrační formulář 16

6 Přihlašovací formulář 17

7 Middleware 18

8 Psací cvičení 19

9 Funkce na výpočet slov za minutu 19

10 Statistiky 20

11 Konfigurátor textu 21

Odkazy

[1] typingcom. *The Secret to Typing Without Looking at Your Fingers*. Accessed: 7 March 2025. 2022. URL: https://www.typing.com/blog/muscle-memory/.

[2] Miodec. *About*. Accessed: 3 March 2025. 2020. URL: https://monkeytype.com/about.

[3] Nedatluj.cz. *Co je Nedatluj.cz?* Accessed: 3 March 2025. 2024. URL: https://www.nedatluj.cz/obchodni-podminky.

[4] Turingvang. *Nextjs Middleware Protected Routes*. Accessed: 25 March 2025. 2024. URL: https://medium.com/@turingvang/nextjs-middleware-protected-routes-bcb3df06db0c.

[5] NextJs. *Middleware*. Accessed: 25 March 2025. 2025. URL: https://nextjs.org/docs/app/building-your-application/routing/middleware.

[6] React Masters. *What are Components and Types of Components in React JS*. Accessed: 25 March 2025. 2024. URL: https://medium.com/@reactmasters.in/what-are-components-and-types-of-components-in-react-js-4e2642b136a2.

[7] MDN. *Callback function*. Accessed: 25 March 2025. 2024. URL: https://developer.mozilla.org/en-US/docs/Glossary/Callback_function.

[8] liva jorge. *Understanding React Parent Components: Best Practices and Common Techniques*. Accessed: 25 March 2025. 2023. URL: https://medium.com/@livajorge7/introduction-9e84391f4b83.

[9] Nextjs. *Introduction*. Accessed: 7 March 2025. 2025. URL: https://nextjs.org/docs#what-is-nextjs.

[10] Justin Gage. *What does Vercel do?* Accessed: 8 March 2025. 2023. URL: https://vercel.com/blog/what-is-vercel.

[11] Sophie Becker. *What is Supabase?* Accessed: 8 March 2025. 2023. URL: https://blog.boldtech.dev/what-is-supabase-vs-firebase/.

[12] Jeffrey Lanters. *Introduction*. Accessed: 24 March 2025. 2024. URL: https://react-unity-webgl.dev/docs/introduction.