


Katedra informatiky
Přírodovědecká fakulta
Univerzita Palackého v Olomouci

BAKALÁŘSKÁ PRÁCE

Informační systém 

Informační systém pro soukromé učitele angličtiny



2023

Vedoucí práce:
RNDr. Martin Trnečka, Ph.D.

Michael Hajný

Studijní program: Informační technologie,
prezenční forma

Bibliografické údaje

Autor: Michael Hajný
Název práce: Informační systém (Informační systém pro soukromé učitele angličtiny)
Typ práce: bakalářská práce
Pracoviště: Katedra informatiky, Přírodovědecká fakulta, Univerzita Palackého v Olomouci
Rok obhajoby: 2023
Studijní program: Informační technologie, prezenční forma
Vedoucí práce: RNDr. Martin Trnečka, Ph.D.
Počet stran: 51
Přílohy: elektronická data v úložišti katedry informatiky
Jazyk práce: český

Bibliographic info

Author: Michael Hajný
Title: Thesis title
Thesis type: bachelor thesis
Department: Department of Computer Science, Faculty of Science, Palacký University Olomouc
Year of defense: 2023
Study program: Information Technologies, full-time form
Supervisor: RNDr. Martin Trnečka, Ph.D.
Page count: 51
Supplements: electronic data in the storage of department of computer science
Thesis language: Czech

Anotace

Tato bakalářská práce se zaměřuje na návrh a vytvoření informačního systému (IS), který má usnadnit vedení agendy soukromého učitele angličtiny. Hlavním cílem IS bude umožnit evidenci klientů a jejich plateb, sdílení materiálů a zadávání úkolů, přehled učiva (včetně individuálního studijního plánu), poskytování informací o následujících hodinách (den následující hodiny a čas), možnost omluvy studenta a hodnocení žáka. IS bude realizován jako webová aplikace s použitím technologií Next.js a MongoDB. Kromě toho bude IS vhodně řešit evidenci zákonných zástupců v případech, kdy žákem bude nezletilá osoba. To znamená, že bude obsahovat speciální funkce a nástroje pro správu účtů. Výsledkem práce bude informační systém, který bude umožňovat efektivní správu a vedení agendy soukromého učitele angličtiny a bude přínosem pro ty učitele, kteří chtějí efektivněji organizovat svou práci s klienty.

Synopsis

This bachelor thesis focuses on the design and creation of an information system (IS) that will facilitate the management of private English language teacher's agenda. The main goal of the IS will be to enable the recording of clients and their payments, sharing of materials and assignment of tasks, overview of the curriculum (including an individual study plan), providing information about upcoming lessons (day and time of the next lesson), the possibility of student absence notification and evaluation of the student. The IS will be implemented as a web application using Next.js and MongoDB technologies. In addition, the IS will adequately address the registration of legal representatives in cases where the student is a minor. This means that it will contain special features and tools for account management. The result of this thesis will be an information system that will enable effective management and administration of the private English language teacher's agenda and will be a benefit to teachers who want to efficiently organize their work with clients.

Klíčová slova: Figma, Next.js, MongoDB

Keywords: Figma, Next.js, MongoDB



Chtěl bych vyjádřit upřímné poděkování všem, kteří mi pomohli při psaní této bakalářské práce.

Nejprve bych chtěl poděkovat mému vedoucímu práce, panu RNDr. Martinu Trnečkovi, Ph.D., za jeho trpělivost, odborné rady a podnětné nápady, které mi pomohly při vytváření této práce.

Dále bych chtěl poděkovat své rodině a přátelům za jejich podporu a povzbuzení, které mi poskytovali během celého mého studia.

Odevzdáním tohoto textu jeho autor/ka místopřísežně prohlašuje, že celou práci včetně příloh vypracoval/a samostatně a za použití pouze zdrojů citovaných v textu práce a uvedených v seznamu literatury.

Obsah

1	Úvod	8
1.1	Úvodní slovo	8
1.2	Popis funkčnosti výsledného software	8
1.3	Výčet rolí a přidružených funkcionalit	8
2	Návrh UI	10
2.1	Barevné schéma	10
2.1.1	Stránky usnadňující výběr (color pickers)	10
2.1.2	Konkrétní výběr schématu pro UI	11
2.2	Font	11
2.2.1	Konkrétní výběr fontu	12
2.3	Ikony	12
2.3.1	Ikony použité v aplikaci	13
2.4	Prostředí pro návrh UI	13
2.4.1	Figma	13
2.4.2	Můj postup	15
3	Programátorská část	17
3.1	Technologie	17
3.1.1	React	17
3.1.2	Next.js	18
3.1.3	JSX	18
3.1.4	MongoDB a MongoDB Atlas	18
3.2	Rozvržení aplikace	19
3.2.1	Components	19
3.2.2	Models	20
3.2.2.1	Popis schémat	20
3.2.3	Pages	21
3.2.4	Public	22
3.2.5	Styles	22
3.2.6	Utils	22
3.3	Architektura aplikace	22
3.4	Použité knihovny	22
3.4.1	Axios	23
3.4.2	Bcrypt	23
3.4.3	Cookies-next	23
3.4.4	Fs	23
3.4.5	Moment	23
3.4.6	Mongoose	24
3.4.7	Next	24
3.4.8	Path	24
3.4.9	React	24
3.4.10	React-dates	24

3.4.11	React-dom	24
3.4.12	React-icons	24
3.4.13	Styled-components	24
3.5	Připojení do MongoDB Atlas	24
3.6	Připojení stránky z lokální sítě do Internetu	25
4	Uživatelská část	25
4.1	Přihlášení	25
4.2	Student mladší 18 let	26
4.2.1	Řídící panel	26
4.2.2	Hlavní obsah pro studenta	27
4.2.3	Informační panel	28
4.3	Student starší 18 let a zákonný zástupce	29
4.3.1	Informační stránka o platbách	30
4.4	Administrátor	31
4.4.1	Informativní oblast	31
4.4.2	Řídící oblast	32
4.4.2.1	Přidávání uživatele	32
4.4.2.2	Pošta	34
4.4.3	Hlavní obsah se studenty	35
4.4.3.1	Info/edit komponenta	36
4.4.3.2	Komponenta pro změnu lekce	36
4.4.3.3	File komponenta	37
4.4.3.4	Homework komponenta	38
4.4.3.5	Komponenta pro shrnutí lekce	39
4.4.3.6	Komponenta pro vkládání online slovíček	40
4.4.3.7	Zbylé informační prvky na stránce	41
5	Možná rozšíření aplikace	43
	Závěr	45
	Conclusions	46
	A Obsah příloženého datového média	47
	Seznam zkratk	48
	Literatura	49

Seznam obrázků

1	Schéma barev - student	11
2	Schéma barev - administrátor	11
3	Zvolený font pro webovou aplikaci	12
4	Zvolená sada ikon pro webovou aplikaci	13
5	Ukázka rámečků (frames) s layouty	14
6	Ukázka zanoření v rámečku	14
7	Křivka znázorňující pohyb očí uživatele	15
8	Přihlašovací okno	26
9	Řídicí panel	27
10	Hlavní obsah	28
11	Informační panel	29
12	UI zákonného zástupce	30
13	Platby	30
14	Dashboard administrátora	31
15	Přidávání studenta	33
16	Přidávání studenta a jeho zákonného zástupce	34
17	Stránka s poštou	35
18	Ukázka výčtu studentů	35
19	Informace o studentovi v administrátorovi	36
20	Komponenta se změnou lekce	37
21	Komponenta pro práci se soubory	38
22	Komponenta pro práci s úkoly	39
23	Komponenta pro shrnutí lekce	40
24	Komponenta pro vkládání online odkazů na práci se slovíčky	41
25	Oznámení o neúspěšné akci	42
26	Příklad dotazovacího prvku, při destruktivní akci	43

1 Úvod

1.1 Úvodní slovo

V průběhu školního roku 2021/2022 jsem se začal zajímat o problematiku online doučování, když se moje přítelkyně rozhodla doučovat angličtinu. Postupem času jsme si uvědomili, že jedním z největších výzev je udržet efektivní agendu s každým studentem a dohodnout s ním individuální plán. V tu chvíli jsme dostali nápad vytvořit informační systém, který by nám pomohl zefektivnit práci a usnadnit plánování s přípravami na hodiny.

V této bakalářské práci se budu věnovat návrhu a implementaci takového informačního systému. Popíši zde všechny aspekty, které jsou nezbytné pro jeho správnou funkci a rozeberu jednotlivé typy uživatelů, kteří budou systémem využívat. Konkrétně se zaměřím na správce, studenty mladší 18 let a jejich rodiče a studenty starší 18 let.

Závěrem zhodnotím jak se práce vydařila a přidám možná rozšíření, která by informačnímu systému přidala na využitelnosti. Doufám, že tato bakalářská práce bude přínosem nejen pro informační systémy zaměřené na agendu učitele angličtiny, ale i pro další obory, které se potýkají s podobnými komunikačními výzvami.

1.2 Popis funkčnosti výsledného software

Implementace výsledného softwarového produktu umožňuje rozlišovat přihlášení uživatelů podle jejich rolí. Studenti, administrátoři a zákonní zástupci mají každý přístup k jiným informacím a funkcím. Po přihlášení do administrátorského účtu má správce možnost přidávat a upravovat informace o studentech, včetně jejich studijních plánů a úkolů. Administrátor také může vidět, zda se někteří studenti omluvili z hodiny nebo zda byla hodina zaplacená.

Přihlášení do účtu studenta zobrazí informace specifické pro daného studenta, včetně úkolů, následujících hodin (den, kdy následující hodina proběhne a její čas), souhrnu probíraného učiva a připojených souborů. Student může také omluvit svoji nepřítomnost z hodiny a zobrazit základní informace o sobě. Pokud je studentem zcela osoba, může přistupovat ke stejným informacím jako kdyby byl v režimu zákonného zástupce, a tak vidí nezaplacené hodiny a kontaktní údaje na administrátora.

Zákonní zástupci mohou mít přístup k informacím svých dětí (počítá se s tím, že zákonný zástupce pod sebou může mít i více než jedno dítě) a mohou si prohlížet jejich informace týkající se učiva, ale nemohou je měnit. Tato funkce umožňuje rodičům u učitele angličtiny sledovat studijní postup svých dětí a zůstat v kontaktu s administrátorem.

1.3 Výčet rolí a přidružených funkcionalit

Role: Správce/Admin

- Mazání studentů (pouze zneviditelnit a znemožnit přístup, ale pro účely obnovení bude student stále v databázi).
- Možnost zapisovat/shrnout téma minulých hodin pro jednotlivé studenty.
- Možnost zadávat domácí úkoly pro jednotlivé studenty.
- Přidávání individuálních .PDF souborů pro studenty do sekce s materiály.
- Možnost měnit individuální osnovu/plán pro studenta.
- Možnost měnit čas následující výuky.
- Možnost měnit jméno a příjmení.
- Přidávání odkazů na slovíčka (flashcards).
- Možnost potvrzení zaplacené hodiny.

Role: Student mladší 18 let

- Uvidí osnovu, která naznačuje cestu, jakým způsobem bude probíhat studium.
- Uvidí krátké shrnutí hodin v sekci průběžného shrnutí.
- Uvidí domácí úkoly.
- Uvidí .PDF soubory pro tisk na hodiny.
- Uvidí čas následující výuky.
- Možnost omluvit se z hodiny.
- Možnost změny hesla.

Role: Role: Student starší 18 let/Rodič

- Uvidí všechny informace, které jsou viditelné pro studenty mladší 18 let.
- V případě, že je role nastavená na rodiče nemůže obsah měnit, ale v případě přihlášení jako student starší 18 let může s obsahem interagovat.
- Kontaktní informace na administrátora.
- Možnost zaplatit hodinu.

2 Návrh UI

V rámci této bakalářské práce byl jeden z hlavních cílů návrh uživatelského rozhraní pro webovou aplikaci. Pro dosažení tohoto cíle bylo nutné vybrat vhodné prostředí pro návrh, zvolit dobře čitelný a příjemný font pro použití ve webové aplikaci a určit barevné schéma, které bude mít vliv na celkový vzhled uživatelského rozhraní.

2.1 Barevné schéma

Výběr vhodného barevného schématu pro webovou stránku je klíčovým faktorem, který ovlivňuje celkový dojem uživatelů z návštěvy stránky. Správně zvolené barvy mohou přitáhnout pozornost uživatelů, zlepšit čitelnost a usnadnit navigaci po stránce. Naopak, nevhodně zvolené barvy mohou vést k nespokojenosti uživatelů, ovlivnit uživatelskou zkušenost a mohou vést v krajním případě i k opuštění stránky.



Výběr vhodného barevného schématu je nutné provést na základě několika faktorů, jako jsou cílová skupina uživatelů, zaměření webové stránky, její funkce a vizuální estetika. Například pro webovou stránku určenou pro děti by bylo vhodné zvolit jasná a výrazná barevná schémata, zatímco pro webové stránky zaměřené například na finanční služby by bylo vhodné zvolit spíše konzervativní barvy. V mé bakalářské práci se zaměření barevného schématu míchá a například pro admina jsou barvy spíše potlačené, kdežto pro studenty, jelikož se jedná především o děti, je vybrané schéma mnohem barevnější.

Kromě toho je také důležité zohlednit psychologické aspekty barev a jejich význam. Například zelená barva může být spojována s přírodou, růst a stabilitou, zatímco červená barva může evokovat emocionální reakce, jako je napětí, vzrušení a vášně. Jelikož barevná schémata mohou ovlivňovat to, jak s určitými prvky na stránce budeme interagovat, tak jsem volil pastelové barvy na straně studenta, protože samy o sobě působí pozitivním a klidným dojmem.

2.1.1 Stránky usnadňující výběr (color pickers)



Výběr správného barevného schématu pro webovou stránku není vždy snadný. Existuje však mnoho nástrojů, které mohou s výběrem pomoci, a já jich sám v minulosti využíval. Někdy máme představu o tom, jak by mělo barevné schéma vypadat, ale už nedokážeme říct konkrétní příklady odstínů, nedejbože číselné vyjádření přesných hodnot. V takových případech existují již zmíněné nástroje, které mohou sloužit jako užitečná pomůcka.

Při výběru barev pro webovou stránku je vhodné se držet počtu 3-7 barev, protože více barev by mohlo působit nepřehledně. Následně lze zvolit buď počáteční barvu, ke které se budou ostatní barvy přidávat na základě podobnosti, nebo je taky možnost náhodně generovat sérii barev a vybírat ty, které se nám líbí a uzamknout je pro přesnější doladění oblasti barevného doplnění.



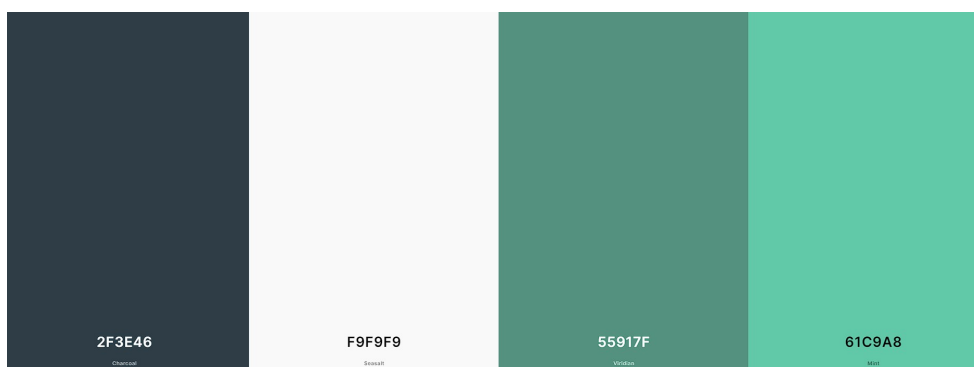
Jednou z takových stránek je [Colours](#) [1], kterou jsem sám při vytváření bakalářské práce využíval. Podobnou službu, možná trochu sofistikovanější, nabízí i webová stránka [Gradientos](#) [2]

2.1.2 Konkrétní výběr schématu pro UI

V případě mé práce jsem tedy zvolil dvě různá barevná schémata: jedno pro studenty  obrázek č. 1 a druhé pro správce viz.  obrázek č. 2. Jejich charakteristikou je, že jsou obecně pastelové a jemné, což může působit příjemně na oko a napomoci k dobré přehlednosti a orientaci na webu.



Obrázek 1: Schéma barev - student 



Obrázek 2: Schéma barev - administrátor

2.2 Font

Další složkou, která nemalou měrou přispívá do přehlednosti a příjemného zážitku s UI, je font. Lze obecně říct, že výběr vhodného fontu na webových stránkách má zpravidla významný vliv na celkový dojem návštěvníka a vnímání obsahu. Správný výběr fontu tak může přispět k větší čitelnosti textu, jasnějšímu vyznění obsahu a k vytvoření správné atmosféry. Proto je důležité pečlivě zvážit

volbu fontu, který bude na webových stránkách použit, s ohledem na typ obsahu, cílovou skupinu a celkovou vizuální koncepci stránek.

V současné době existuje mnoho bezplatných webových stránek, které poskytují nástroje pro manipulaci s vkládaným textem, jako je změna velikosti nebo tloušťky fontu. Tyto nástroje mohou usnadnit výběr vhodného fontu, zejména pro ty, kteří nemají v oblasti typografie zkušenosti.

Jednou z takových stránek je [Google Fonts](#) [3], která nabízí rozsáhlou knihovnu různých fontů k bezplatnému použití. Tato stránka může být vhodnou volbou pro začátečníky v oblasti typografie, kteří hledají snadno dostupné a uživatelsky přívětivé nástroje pro výběr fontu.

Služba [Google Fonts](#) [3] dokonce poskytuje i informativní obsah, který má pomoci při výběru správného fontu, a kroky, jak být zběhlejší při jeho hledání. Dělalí tak na stránce [Google Fonts - Choosing type](#) [4], kde je možné nalézt návody na hledání fontů na základě různých kritérií, jako je nálada, styl a účel. Nachází se zde i základní pravidla kombinování fontů a to, jak si být jistý, že zvolené fonty budou správně fungovat na různých zařízeních a platformách.

2.2.1 Konkrétní výběr fontu

Zde je zvolený font, jež jsem využíval, a který nejlépe vyhovuje stylu a účelu webové aplikace [Google Fonts - Raleway](#) [5]. Na obrázku č. 3 je grafická podoba.



Obrázek 3: Zvolený font pro webovou aplikaci

2.3 Ikony

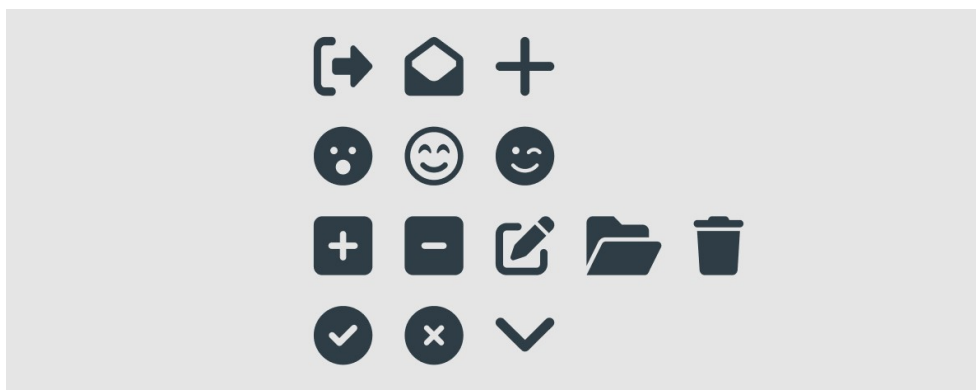
Ikony můžeme vnímat jako externí prvek webových aplikací, který má vliv na uživatelskou zkušenost, produktivitu a celkový vzhled aplikace. Tento grafický element pomáhá rychle identifikovat a použít požadované funkce bez zbytečného hledání nebo pročítání. Jejich vhodný výběr má přímý dopad na vzhled a estetiku aplikace a nemusí být využívány jen pro jejich funkčnost, ale i pro přidání dodatečné estetiky.

Stejně jako s fontem, i zde se nachází opravdu velké množství možností, které nejsou placené a odkud můžeme ikony čerpat. Hodně záleží na zdroji, odkud budeme ikony brát, zda poskytuje sadu, která se hodí k našemu stylu stránek (hrnaté nebo kulaté okraje), a zdali tato ikonová sada obsahuje všechny ikony nezbytné pro obsluhu všech funkcionalit v naší aplikaci.

V aplikaci jsem využil [Font Awesome](#) [6] ikony, které nabízí široké spektrum nástrojů, jež pomáhají filtrovat množství dostupných ikon na této stránce.

2.3.1 Ikony použité v aplikaci




Ukázka vybraných ikon na obrázku č. 4, které mají na stránce své zastoupení.



Obrázek 4: Zvolená sada ikon pro webovou aplikaci

2.4 Prostředí pro návrh UI

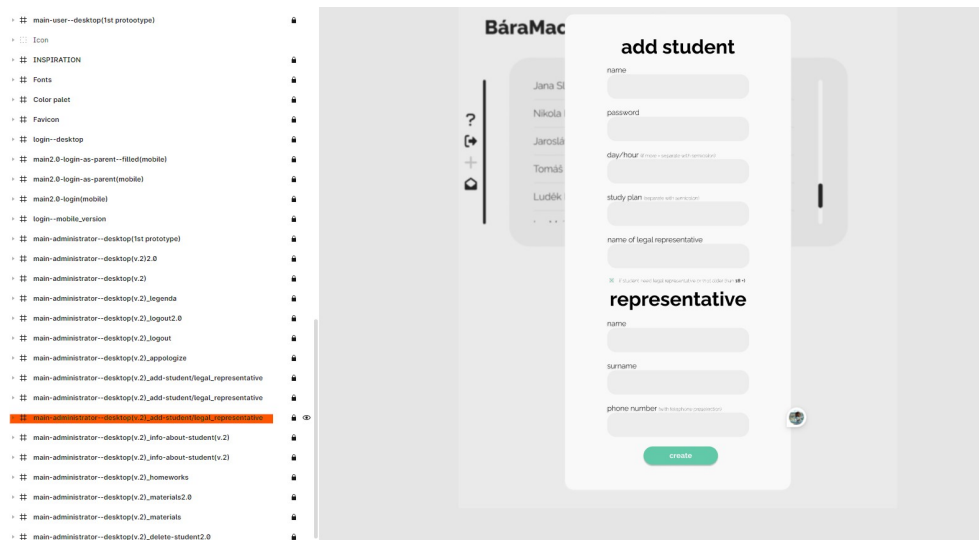
Dostávám se ke konci kapitoly týkající se návrhu uživatelského rozhraní (UI). V tomto bodě se slévají všechny předchozí podkapitoly dohromady, jelikož je připravené barevné schéma spolu s fontem a stejně tak vhodné ikony, které budou odpovídat jednotlivým funkcionalitám na stránce.

 Ted je třeba stránku dále rozvrhnout a navrhnout její celkový design. Ačkoli je obvyklé si stránku nejdříve nakreslit ~~na papír~~ jako drátový model (wireframe), který pomáhá urovnat jednotlivé části stránky a navrhnout její funkcionality, ~~tak~~ já jsem už před začátkem ěl jasno v tom, jak chci aby stránka byla rozvrstvená, takže jsem tento krok vynechal. Vzhledem k tomu, že se jedná o menší projekt, nepovažuji to za velký problém. 

2.4.1 Figma

Pro vytvoření uživatelského rozhraní jsem používal program [Figma](#) [7], jež má tu výhodu, že je dostupný zdarma a pro jednotlivce se zde nachází veškeré nástroje pro střední až mírně pokročilé uživatele v navrhování designu a liší se tedy až v případě kolaborací a firemně licencovaných produktů. Lze samozřejmě použít i jiné prostředí pro tvorbu uživatelských rozhraní a příkladem můžou být nástroje od společnosti Adobe.

V samotném programu Figma [7] lze vytvářet takzvané frames, což jde vidět na obrázku č. 5, a do nich zanořovat objekty, které lze do framů vkládat a tvořit tak layout uživatelského rozhraní.



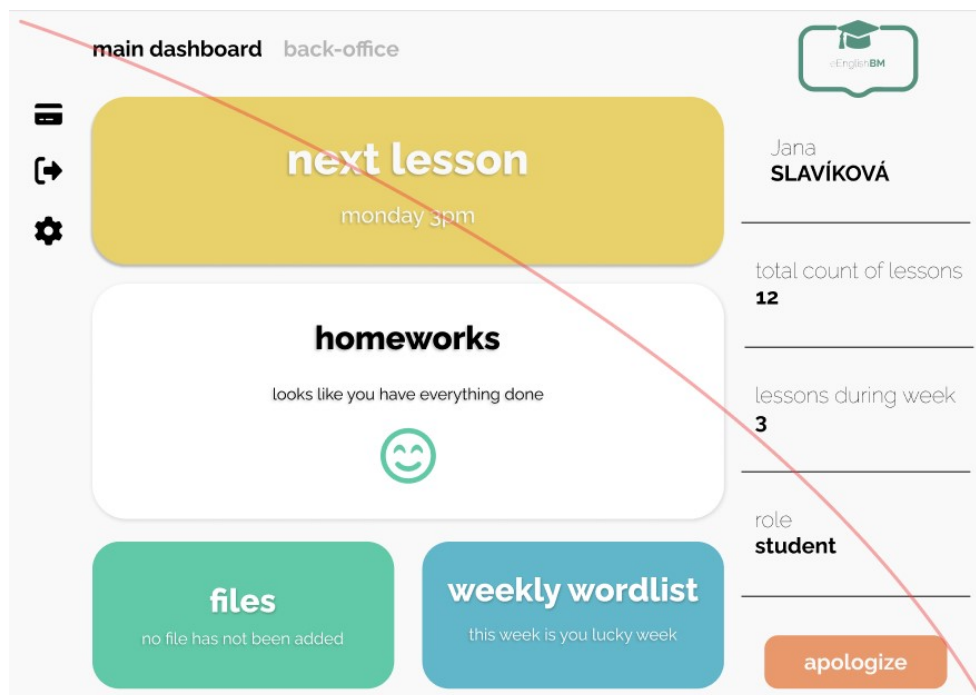
Obrázek 5: Ukázka rámečků (frames) s layouty

Při práci na návrhu uživatelského rozhraní ve Figmě si lze budoucí práci při psaní HTML zjednodušit tím, že objekty, které do layout vkládáme, budeme zanořovat a sjednocovat tak, aby se podobali kontejnerům, které nám pomůžou HTML strukturu zformovat v CSS kodu. Lze vidět na obrázku č. 6. Dokonce ~~přitom~~ potom lze i CSS kod vygenerovat přímo ve Figmě pro dané kontejnery.



Obrázek 6: Ukázka zanoření v rámečku

Při práci na uživatelském rozhraní ve Figmě jsem se snažil neopomenout různá pravidla a konvence týkající se umístění různých prvků na stránce, aby uživatelské rozhraní bylo intuitivní. Jedním takovým příkladem může být křivka, která jde vidět na obrázku č. 7. Křivka znázorňuje pohyb očí uživatele po obrazovce a dále pomáhá s umístěním nejdůležitějších prvků na stránce. Víme totiž, že oči většiny uživatelů mají tendenci přejíždět obrazovku z levého horního rohu do pravého dolního rohu a na této trase by měly být umístěny všechny důležité prvky.



Obrázek 7: Křivka znázorňující pohyb očí uživatele


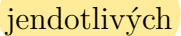

Jak lze na obrázku vidět, tak v levém horním rohu jsou důležité ovládací prvky (přehled plateb, odhlášení, nastavení), dále křivka vede přes „bublinu“ znázorňující čas a den další hodiny a končí na tlačítku pro možnost omluvení se z hodiny. Zmíním ještě, že tohle je nepatrně rozlišné UI od stávající verze, která bude k vidění později v práci a konkrétně se jedná o kapitolu 4.

2.4.2 Můj postup

Nejtypičtějším příkladem, jak začít budovat uživatelské rozhraní pro webové aplikace, je začít ho budovat nejprve pro mobily kvůli responzivitě. ~~Nebo tohle je alespoň poučka, se kterou jsem se osobně mnohokrát setkal [8].~~ Já jsem na to ve své práci šel opačně kvůli funkcionalitám, které jsem potřeboval testovat, a nechtěl jsem se nejdřív zabývat tím, jak je poskládat do mobilní verze.

Jako první jsem si do Figma dal podobné layouty a návrhy projektů pro inspiraci. Potom jsem do projektu ve Figmě přidal i barevná schémata s číselnou

reprezentací, abych mohl barvy využívat, a to samé jsem udělal ještě s fontem a ikonami. Takže pokaždé, když jsem projekt otevřel, měl jsem tam veškeré základní prvky pro tvorbu.

Celý návrh stránek jsem stromově dělil nejdřív na přihlášení, ad a, studenta a následně jsem v jendotlivých děleních pokračoval, až na přihlášení, v rámci role. Například pro studenta jsem vytvořil základní zobrazení všech komponent, které mu umožní kontrolovat informace vložené adminem. Admin mi zabral více času, protože jednotlivé úkony si vždy vyžadovaly vlastní frame, na kterém by interakce s rozhraním šla dobře vidět.

Dále už jsem si jen nachystal frame (rámec) pro desktop s odpovídajícími rozměry, ve kterém jsem začal realizovat první stránku.

Celá stránka i s popisky je možná k vidění na tomto odkazu [Figma Projekt](#).



3 Programátorská část

V této části se pokusím přiblížit nejen strukturu projektu, ale i to, jak jednotlivé části fungují a jaké technologie se na projektu využívají.



3.1 Technologie

Když jsem s projektem začínal, tak jsem se rozmýšlel, které technologie na tvorbu webové aplikace budu chtít využít. Nejmenší cestou odporu byla klasická sestava, kde na straně klienta by byly použity technologie HTML, CSS, JavaScript a na straně serveru potom Apache, který by komunikoval s MySQL databází. ~~Například pomocí jazyka PHP bych komunikoval se serverem a SQL jazyk by obsluhoval typicky MySQL databázi. Cestou nejmenšího odporu to bylo z důvodu, že většinu technologií už jsem znal a nebylo potřeba se učit moc navíc.~~ Nakonec jsem se rozhodl, že chci vyzkoušet jiný přístup. Chtěl jsem vyzkoušet reaktivní aplikace. Podle OpenAI [9] jsou to takové aplikace, které jsou navrženy tak, aby byly co nejvíce responzivní a interaktivní. Tyto aplikace využívají moderních webových technologií a frameworků, které umožňují vysokou rychlost a efektivitu v reakci na uživatelské vstupy.

Reaktivní webové aplikace často využívají JavaScriptové knihovny příkladem by mohly být React nebo Vue.js, které umožňují rychle a jednoduše aktualizovat obsah stránky bez nutnosti obnovení celé stránky. Nad těmito knihovnami jsou vystavěné i celé frameworky, například Next.js nebo Angular, které umožňují snadno vytvářet interaktivní prvky, jako jsou animace, formuláře nebo komponenty, které reagují na uživatelské vstupy.

I v této poměrně nové sféře tvoření webových aplikací už existují některé zaběhnuté kombinace. Příkladem může být React (JavaScriptová knihovna) na straně klienta, Express (framework pro Node.js), který zastupuje serverovou část a dále MongoDB, což je NoSQL databáze.

Chtěl jsem však jít ještě dál a ve své bakalářské práci jsem se rozhodl použít primárně jen dvě technologie, kterými jsou Next.js [10] a MongoDB [11]. Next.js jsem používal pro tvorbu uživatelského rozhraní a komunikaci s MongoDB databází pomocí knihovny Mongoose



Je dobré říct, že express není nutné používat, protože Next.js poskytuje vlastní serverovou stranu a to tak, že funguje na serveru pomocí Node.js a Express.js (má je v sobě integrované), což umožňuje vyvíjet webové aplikace, které jsou plně funkční i bez JavaScriptu na straně klienta.

V praxi pak stačí do projektu vložit URL, kterou získáme u poskytovatele a z routování odstranit předvolbu našeho lokálního serveru.



3.1.1 React



React je open source JavaScriptová knihovna, která se používá k vytváření uživatelských rozhraní pro webové aplikace. Tato knihovna umožňuje tvůrcům aplikací vytvářet dynamické a interaktivní uživatelské rozhraní, která mohou reagovat na

uživatelské vstupy a aktualizovat se v reálném čase bez nutnosti obnovování celé stránky.

React se vyznačuje jednoduchostí a modularitou, což umožňuje snadnou integraci do různých projektů a rychlé vývojové cykly. Knihovna také poskytuje mnoho užitečných funkcí pro práci s komponentami, správu stavu aplikace a efektivní práci s virtuálním DOM. [OpenAI 2023] [9]

3.1.2 Next.js

Next.js je open source framework postavený na React, který umožňuje tvůrcům webových aplikací rychleji a efektivněji vytvářet a nasazovat moderní, interaktivní i vysoce výkonné webové aplikace.

Next.js poskytuje mnoho funkcí, které usnadňují tvorbu webových aplikací, včetně automatického načítání a optimalizace stránek, server-side renderingu a statické generace, předpřipravených konfigurací pro různé typy aplikací, podporu TypeScriptu, hot module replacement pro rychlou iteraci při vývoji a mnoho dalších. [OpenAI 2023] [9]

3.1.3 JSX

JSX je syntaxe, kterou používají některé JavaScriptové knihovny, zejména React, k vytváření uživatelských rozhraní. JSX umožňuje psát kód, který kombinuje HTML a JavaScript do jednoho souboru. Tento způsob zjednodušuje tvorbu uživatelských rozhraní a usnadňuje práci s komponentami. Při použití JSX se kód překládá do standardního JavaScriptu, který pak prohlížeč může interpretovat. JSX umožňuje psát kód, který je srozumitelný a intuitivní, což usnadňuje tvorbu složitých uživatelských rozhraní. [OpenAI 2023] [9]

3.1.4 MongoDB a MongoDB Atlas

MongoDB je open source NoSQL databázový systém, který se používá pro ukládání a správu nestrukturovaných dat. MongoDB je navržen pro snadnou škálovatelnost a flexibilitu, což umožňuje tvůrcům aplikací efektivně pracovat s velkým množstvím dat a rychle reagovat na požadavky uživatelů.

MongoDB používá dokumentový model, což znamená, že data jsou ukládána v BSON formátu, což je binární verze JSON. Tento model umožňuje snadnou správu dat a rychlé dotazování. MongoDB také poskytuje mnoho užitečných funkcí, včetně indexování, replikace a shardingu, což umožňuje snadné zajištění dostupnosti a spolehlivosti dat.

[MongoDB Atlas](#) [12] je cloudová služba nabízená společností MongoDB, která poskytuje správu a hostování MongoDB databází v cloudu. MongoDB Atlas nabízí řadu funkcí, včetně automatického škálování, zálohování a obnovy dat, monitorování výkonu a bezpečnostní funkce, jako je šifrování dat a správa přístupových práv. [OpenAI 2023] [9]

3.2 Rozvržení aplikace

Celý projekt mám ve veřejném repozitáři na GitHubu [Batchelor Thesis Project](#), kde je přístupná i dokumentace a postup, jak aplikaci spustit ~~u sebe~~ v lokální síti, popřípadě odkaz na testovací stránku projektu.

Celý projekt se nachází ve složce app, která má následující dělení.

```
app/  
├── components/  
│   ├── Admin/  
│   │   └── ContentOfStudent/  
│   ├── Login/  
│   └── Student/  
├── models/  
├── pages/  
│   └── api/  
├── public/  
│   └── images/  
├── styles/  
└── utils/
```

Rád bych teď postupně prošel význam složek, které projekt obsahuje a k některým se ještě vrátím později v této práci, protože pro jejich úplné vysvětlení je potřeba uvést je v kontextu některých technologií.

Ještě krátká zmínka ohledně `package.json` souboru, který se nachází na úrovni složek `components`, `models`, `pages` atd. a je takovým záznamníkem o použitých knihovnách, ke kterým se též ještě vrátím. Obecně stromová struktura znázorňuje jen složky a jejich vzájemné uspořádání, ale je zde spousta souborů, které jsem kvůli komplexnosti nevyobrazil. Úplnou strukturu je možné vidět v příloze, která je součástí práce. V případě popisu souborů, které nejsou uvedné, budu jejich lokaci upřesňovat nebo vytvořím zkrácenou strukturu pro lepší přehled.

Jak lze vidět, tak ze struktury není úplně zřejmé, které složky patří klientovi a které se zaměřují na komunikaci se serverem. Částečně je to právě kvůli použité technologii Next.js, která umožňuje tvorbu aplikací a s ním spjaté funkcionality jako routování apod. díky tomu, že je má v sobě zakomponované.


3.2.1 Components

Začnu tedy od vrchu a to u složky `components`, ve které se nachází další podsložky `Admin`, `Student`, což jsou dvě role, které se na stránce objevují, spolu se složkou, která v sobě má přihlašovací komponentu. `Admin` v sobě ukrývá veškeré komponenty, které obsluhují dashboard po přihlášení v roli admina. Složka `Student` zase veškeré komponenty, které jsou potřeba k zobrazení obsahu, jež se vygeneruje na základě studenta, a to buď samotnému studentovi nebo jeho zákonnému zástupci. Ve zkratce se zde nachází to, co uživatel uvidí, tedy struk-

tura a stylování stránky. Některé komponenty v sobě mají i logiku, napojenou na tlačítka, pro odesílání požadavků na server.

3.2.2 Models


Ve složce `Models` se nachází soubory se schémata databázových kolekcí. V databázi se nachází 6 kolekcí, kde některé z nich jsou propojené.

- `Apology.js`,
- `Homeworks.js`, 
- `Lecture.js`,
- `Payment.js`,
- `PaymentRequest.js`,
- `User.js`,

3.2.2.1 Popis schémat

Já jsem si pro webovou aplikaci zvolil databázi MongoDB, která funguje tak, že se vytvoří nějaká databáze a v ní se definují určité kolekce, což jsou předlohy, které ukazují, jak později konkrétní instance (vyplnění kolekce daty) budou vypadat. Instancím se říká dokumenty.

Základní předlohou/kolekcí je `User.js`, která v sobě má, stejně jako ostatní kolekce, strukturovaná data uložená v různých formátech. Konkrétně `User.js` v sobě ukládá informace popisující uživatelský účet. V této kolekci jsou výčtově uložené tyto informace:

- `_id`: unikátní identifikátor uživatelského účtu
- `role`: role uživatele v systému, zde je uveden jako "student, representative, admin" 
- `name`: křestní jméno uživatele
- `surname`: příjmení uživatele
- `username`: uživatelské jméno uživatele
- `phone`: telefonní číslo uživatele
- `password`: heslo uživatele šifrované pomocí `bcrypt` knihovny
- `students`: pole s informacemi o studentech
- `disabled`: informace o tom, zda je uživatelský účet deaktivován

- `lectures`: odkaz na jednotlivé hodiny a jejich informace
- `legalRepresentative`: odkaz na informace o zákonném zástupci uživatele
- `child`: pole s odkazem na informace o dítěti
- `homeworks`: pole s odkazy na informace o úkolech
- `summary`: pole s informacemi o shrnutích
- `plan`: pole s informacemi o studijním plánu
- `wordList`: odkaz na seznam slov
- `files`: pole s informacemi o souborech

Ve zmíněné kolekci je na několika místech uloženo v poli `id` dokumentu z jiné kolekce, například `Lectures.js` nebo `Homeworks.js`. Takto propojené dokomunity v databázi potom usnadňují samotnou práci s daty a jejich ukládání.



Tedy, když už by měla být jasnější práce s dokumenty vytvořenými na základě kolekcí, už jen přehledově zmíním význam ostatních kolekcí a co se do nich ukládá.

Na základě kolekce `Homeworks.js` se vytvářejí dokumenty domácích úkolů, které obsahují vlastní `id`, nadpis a popis domácího úkolu a jednotliví uživatelé potom v sobě mají pole, ve kterém jsou `id` vytvořených úkolů pro studenta.

Podle kolekce `Lectures.js` se vytvářejí dokumenty lekcí s informacemi, na kterých se student s doučujícím domluvil. Každá hodina má opět svůj identifikátor (alfa-numerické číslo), který je uložen v poli `lectures` u studenta a odkazuje na konkrétní hodinu s konkrétními informacemi, například se začátkem hodiny, koncem hodiny, identifikátorem studenta, polem statusů (jestli hodina proběhla nebo se z ní student omluvil nebo byla-li zrušená správcem), dále polem změn, které v sobě uchovává objekty a ty jsou nositeli informací o změně času dané hodiny.

Další ze zmíněných kolekcí je kolekce s názvem `PaymentRequest.js`, která slouží jako předloha pro dokumenty uchovávající v sobě informace o hodině, jež proběhla, a uživatel posílá požadavek na kontrolu zaplacení lekce s danou částkou. Kolekce, která s tímto souvisí je kolekce `Payments.js`, podle které se vytvářejí dokumenty zaplacených lekcí.

Poslední je kolekce `Apologies.js`, jež je předlohou pro dokumenty s informacemi o lekcích, ze kterých se student chce omluvit. Tato kolekce také obsahuje položku, jejíž hodnota říká, jestli správce danou omluvu viděl.

3.2.3 Pages

Další složkou se soubory typickými pro Next.js je složka `Pages`. Ve složce `pages` najdeme mimo jiné i další složku `api`, v níž běží náš server a provádíme zde i routování naší aplikace. Uvnitř složky `pages` máme i speciální soubory. Soubory


začínající podtržítkem jsou v Next.js považovány za speciální a mají specifický účel. V mém případě používám dva volitelné soubory pro konfiguraci a to jsou `_app.js` a `_document.js`.

Soubor `_app.js` slouží jako kostra webové aplikace, která obaluje všechny komponenty a je zodpovědný za nastavení globálního layoutu a zpracování stránkových změn, jako jsou například změny v URL adrese nebo změny v kontextu aplikace. Tento soubor může být použit k načtení globálních stylů, inicializaci kontextu aplikace nebo například k vytvoření globálního wrapperu pro celou aplikaci.

~~Naproti tomu~~ soubor `_document.js` je zodpovědný za zpracování HTML šablony pro každou stránku, kterou vygeneruje Next.js. Tento soubor může být použit pro nastavení globálních metadat, například pro SEO optimalizaci nebo načtení externích knihoven.

3.2.4 Public

Složka `Public` je dalším typickým prvkem v Next.js a slouží k ukládání statických souborů, které jsou přístupné přímo z webového serveru bez nutnosti překladu pomocí frameworku.

Tento adresář by měl obsahovat veškeré statické soubory, jako jsou obrázky, videa, fonty nebo soubory JSON. Tyto soubory jsou přístupné v aplikaci pomocí cesty začínající `"/`. 

3.2.5 Styles

Složka `Styles` má v tomto projektu jediný účel, a to ten, že v ní definuji stylování, které je pro celý projekt stejné. Nastavuji zde výchozí stav pro stránku.

3.2.6 Utils

Složka `Utils` obsahuje dva soubory, kde první z nich `Colors.js` slouží pro definování barev použitých v aplikaci a druhý `dbMongo.js` obsahuje funkce, které se používají na serveru pro práci s databází.

3.3 Architektura aplikace

Jedná se o client-server architekturu, kde klient pomocí rozhraní webové aplikace posílá požadavky na server, který na základě takových požadavků komunikuje s databází a vrací uživateli patřičná data.

3.4 Použité knihovny

Jak již bylo napsáno, tak soubor `packages.json` obsahuje záznam o všech použitých knihovnách, které jsou v projektu použité a níže je uveden jejich přehled.

- `axios`,

- bcrypt,
- cookies-next,
- formidable,
- fs,
- moment,
- mongoose,
- next,
- path,
- react,
- react-dates,
- react-dom,
- react-icons,
- styled-components.

3.4.1 Axios

Jedná se o knihovnu pro práci s HTTP requesty v JavaScriptu. Umožňuje snadné a efektivní získávání dat z API serverů. Je to možná náhrada za vestavěnou fetch metodu. Odkaz na [Axios](#) [13].

3.4.2 Bcrypt

Knihovna pro šifrování hesel pomocí bcrypt algoritmu, která poskytuje velmi bezpečný způsob ukládání hesel v databázi.

3.4.3 Cookies-next

Knihovna pro práci s cookies v Next.js frameworku, která umožňuje snadnou práci s cookies a ukládání dat mezi uživatelskými relacemi.

3.4.4 Fs

Standardní Node.js knihovna pro práci se soubory a složkami.

3.4.5 Moment

Knihovna pro práci s daty a časy v JavaScriptu. Umožňuje lepší formátování.

3.4.6 Mongoose

ODM (Object Data Modeling) pro MongoDB databáze. Umožňuje snadnou práci s MongoDB databází pomocí definování modelů a dotazů. Odkaz na [Mongoose](#) [14].

3.4.7 Next

React framework pro server-side rendering a tvorbu webových aplikací. Umožňuje snadnou tvorbu dynamických webových stránek s použitím React komponent. Odkaz na [Next.js](#) [10].

3.4.8 Path

Standardní Node.js knihovna pro práci s cestami a soubory na počítači, která umožňuje manipulaci s cestami a získávání informací o souborech.

3.4.9 React

JavaScript knihovna pro tvorbu uživatelského rozhraní, jež se používá se pro tvorbu interaktivních webových stránek a aplikací. Odkaz na [React](#) [15]

3.4.10 React-dates

Knihovna pro práci s kalendářem a daty v React aplikacích. Umožňuje snadnou tvorbu a zobrazení kalendáře a manipulaci s daty.

3.4.11 React-dom

Knihovna pro renderování React komponent v HTML dokumentu, která umožňuje vkládání React komponent do existujícího HTML dokumentu.

3.4.12 React-icons

Knihovna pro použití ikon v React aplikacích. Odkat na [React Icons](#) [16].

3.4.13 Styled-components

Jedná se o knihovnu pro tvorbu a použití CSS stylů v React aplikacích. Díky styled-components jsem například mohl používat `javascript` v `css`, exportovat celé definice stylů, používat proměnné z komponent a podobně. Odkaz na [Styled Components](#) [17].



3.5 Připojení do MongoDB Atlas

Pro připojení do databáze, která nejlépe bude online a pomůže nám tak v pozdější fázi s přemístěním stránky do internetu, potřebujeme poskytovatele. Já

jsem ve své práci využíval služeb [MongoDB Atlas - Register](#), kde je potřeba se zaregistrovat. Po registraci je nutné vytvořit databázi, do které se připojíme z naší aplikace pomocí připojovacího řetězce, který nám MongoDB Atlas poskytne v nastavení naší databáze.

3.6 Připojení stránky z localní sítě do Internetu

Pro připojení webové aplikace do Internetu jsem se rozhodl zvolit standardní řešení. V dnešní době už je možnost připojit stránky se serverem z GitHubu. Funguje to opravdu jednoduše. Je potřeba vybrat si poskytovatele, který bude webovou aplikaci hostovat. V mém případě to byla služba [Vercel](#) [18], ale je možné využít například i [Heroku](#) [19]. Dále je potřeba zjistit, jestli daná služba podporuje technologie, které jsou v aplikaci využity.

V minulosti už jsem měl zkušenosti s tím, že čistě statickou stránku lze hostovat z GitHubu připojením repozitáře v dané službě, takže jsem hned hledal, jestli něco podobného nejde udělat i s celou aplikací. Příkladem stránky, která takové služby poskytuje (připojení statických stránek z GitHubu) je [Netlify](#) [20], která pomocí repozitáře a určení konkrétní složky zajistí pohodlný chod stránek na internetu.

4 Uživatelská část

Jak už bylo zmíněno v předchozích kapitolách, tak se uživatelské rozhraní skládá ze tří částí, a to konkrétně z přihlašovacího, administrátorského a studentského, které má rozdílné prvky v případě přihlášení do role zákonného zástupce. Postupě projdu všechny.

4.1 Přihlášení

Při vstupu na stránku se uživateli zobrazí přehledné přihlašovací okno, které umožňuje přístup ke dvěma možnostem vstupu, což je možné vidět na obrázku č. 8. První způsob vstupu zahrnuje zadání uživatelského jména a hesla, díky čemuž se uživatel přihlásí buď jako student nebo jako administrátor. Druhou možností je přihlášení pomocí telefonního čísla a hesla, což odkáže uživatele na rozhraní pro zákonného zástupce.


The image shows a login form with a light gray background. At the top, the word "login" is written in a bold, black, sans-serif font. Below it, there are two input fields: the first is labeled "username" and the second is labeled "password". Both fields are represented by light gray rounded rectangles. Below the password field, there is a checkbox followed by the text "login as representative". At the bottom of the form is a dark gray rounded rectangle button with the word "login" in white text.

©2022eEnglish

Obrázek 8: Přihlašovací okno

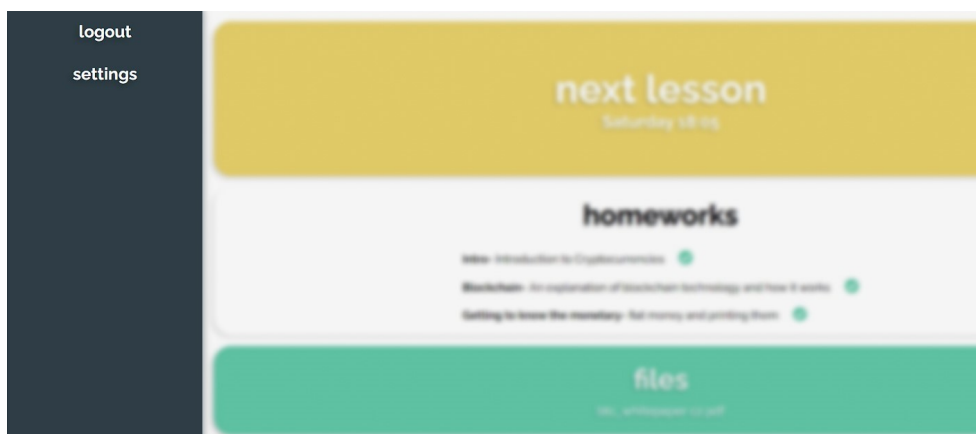
4.2 Student mladší 18 let

Studentské rozhraní je v základu rozdělené do tří hlavních částí. Dvě z těchto částí dále odkazují na další stránky. 

- řídicí panel,
- hlavní obsah pro studenta,
- informační panel, 

4.2.1 Řídicí panel

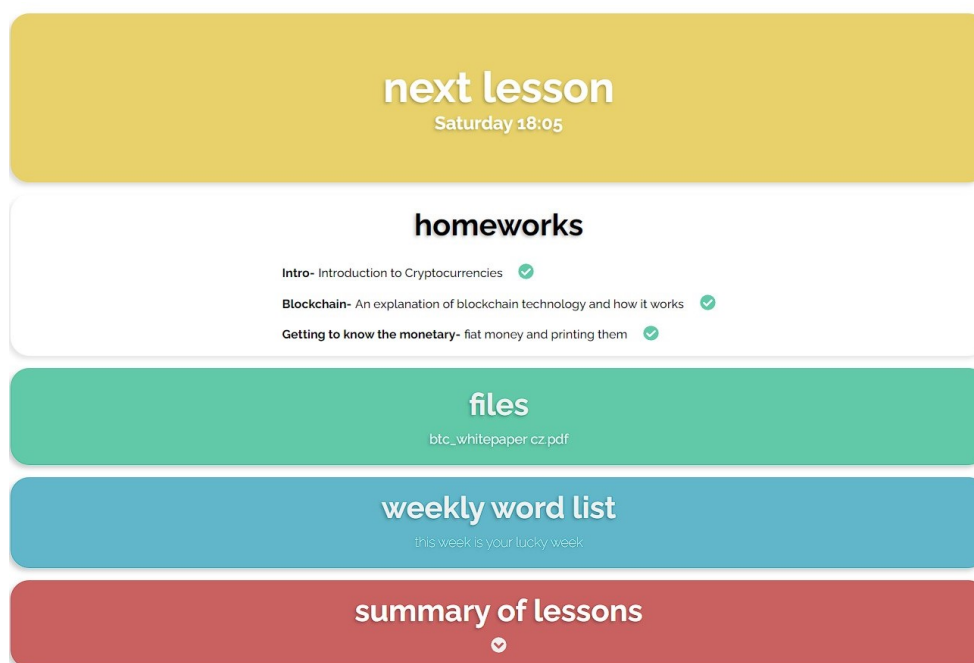
Řídicí panel v roli studenta mladšího 18 let nabízí jen dvě možné akce a těmi jsou odhlášení nebo přesunutí uživatele na stránku s nastavením, kde si může dělat změny týkající se účtu. To jaké akce jsou v roli zákonného zástupce nebo studenta staršího 18 let přiblížím později. Řídicí panel lze vidět na obrázku č. 9.



Obrázek 9: Řídicí panel

4.2.2 Hlavní obsah pro studenta

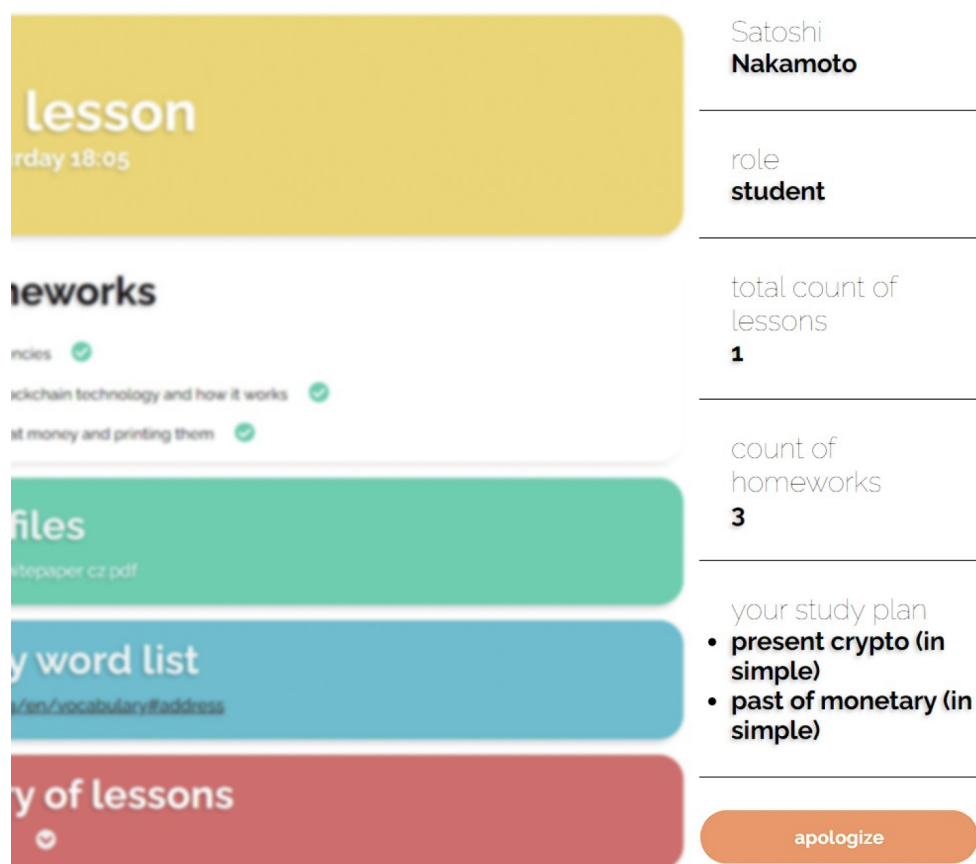
V hlavním obsahu pro studenta se nachází informace, které jsou pro něj už zajímavější a užitečné. Student zde může vidět plánovaný čas a den dalšího vyučování, seznam zadaných úkolů a také má přístup k souborům, které mu byly administrátorem poskytnuty. Dále je zde k dispozici týdenní slovníček, který umožňuje studentovi otestovat své vědomosti a shrnutí probírané látky. Celkově se jedná o důležité prvky pro správnou organizaci a přípravu studenta na vyučování. Student má samozřejmě možnost s obsahem interagovat a například úkoly, které mu učitel zadá, jsou možné po splnění odškrtnout. Dále si uživatel může stáhnout soubory a rozkliknout týdenní wordlist jež odkazuje přímo na stránku. Důležité je zmínit i to, že další lekce ukazuje čas a den i změněné hodiny, takže když administrátor/učitel hodinu změní, tak se to u studenta projeví. Popisované funkcionality z odstavce jsou k vidění na obrázku č. 10



Obrázek 10: Hlavní obsah

4.2.3 Informační panel

Informační panel slouží jako krátké shrnutí pro studenta, kde v horní polovině může vidět své jméno a příjmení spolu s rolí. Druhou polovinu tvoří informace o počtu hodin, úkolu a jeho osobním studijním plánu. Pod těmito informacemi je i tlačítko, které jej odkáže na stránku, kde se může omluvit z hodiny. Stránka, která slouží pro omluvení se z hodiny disponuje kalendářem, který je přístupný pouze pro data, ve kterých mají studenti naplánovanou hodinu. Po vybrání konkrétního data, ze kterého se student chce omluvit, je zpráva o omluvě zaslána adminovi, který získá informaci o identitě studenta a konkrétním datu, ze kterého se omlouvá. Tato funkce pomáhá studentům jednoduše a efektivně řešit omluvy z vyučovacích hodin. V kalendáři je také pohlídané, jestli se student snaží omluvit v den, kdy má hodina proběhnout. Tato funkce umožňuje učiteli mít dostatek času pro případné úpravy plánu výuky, včetně posunu lekcí na dřívější termíny. Díky této funkci má učitel možnost se snadno a rychle domluvit se svými studenty na změnách v plánu. Informační panel viz. obrázek č. 11.



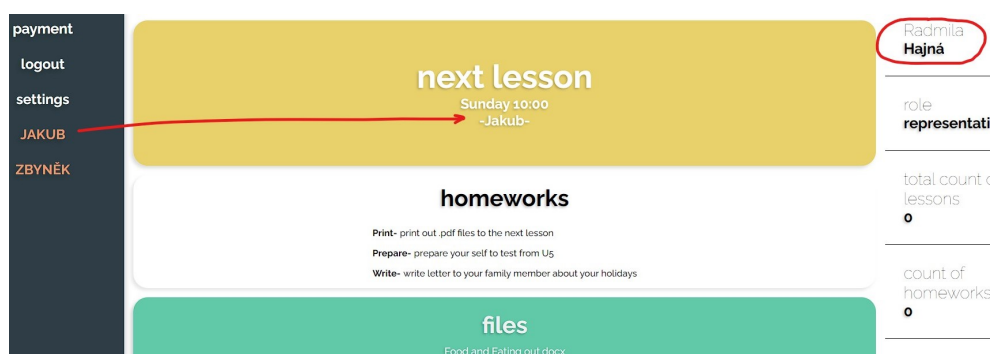
Obrázek 11: Informační panel

4.3 Student starší 18 let a zákonný zástupce

Student starší 18 let se liší od toho, který není plnoletý pouze tím, že má v ovládacím panelu ještě jeden odkaz týkající se plateb, viz. obrázek č. 12.

Role zákonného zástupce a plnoletého studenta je v tomto ohledu stejná. Oba dva můžou přistupovat k nezaplaceným hodinám a informacím potřebným k uhrazení těchto hodin nebo kontaktním údajům na učitele. Kde se však tyto dvě role liší je v tom, že zákonný zástupce má ve svém uživatelském rozhraní dítě/děti a jejich obsahy, mezi kterými se může překlíkávat.

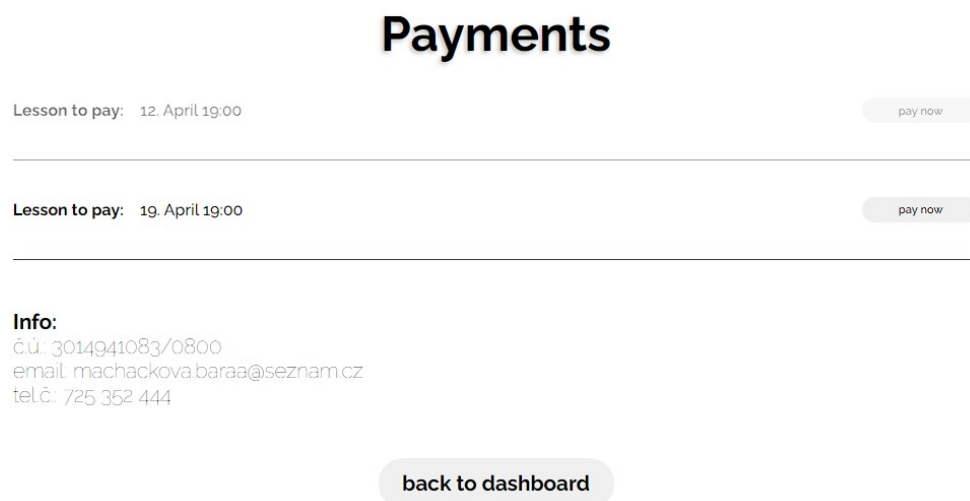
Dalším rozdílem mezi plnoletým studentem a zákonným zástupcem je ten, že zákonný zástupce nemůže odklikávat úkoly svých dětí a například nemá možnost omluvy sama sebe. To můžou jen děti.



Obrázek 12: UI zákonného zástupce

4.3.1 Informační stránka o platbách

Na této stránce, jak titulek napovídá, se nechází informace o platbách a také jsou zde kontaktní údaje na učitele. Abych byl více konkrétní, tak se zde nachází veškeré nezaplacené hodiny a hodiny, které čekají na potvrzení o zaplacení. Funguje to tak, že pokud hodina proběhla a nebyla zrušená, tak se po uplynutí času, který hodina měla trvat přiřadí do hodin, které jsou nezaplacené a zobrazí se právě na této stránce. Student nebo zákonný zástupce potom vidí veškerá data hodin, která proběhla a k nim uvedenou částku, která je nesplacená. Jakmile plnoletý student nebo zákonný zástupce hodinu splatí, tak datum hodiny s částkou zašednou a jeví se jako platba, která čeká na potvrzení administrátorem. Jakmile administrátor platbu ve svém bankovníctví zkontroluje, že opravdu přišla, tak potvrdí, že platbu viděl a na straně studenta v sekci plateb tato zašedlá hodina s částkou zmizí. Vizualizace odstavce na obrázku č. 13.

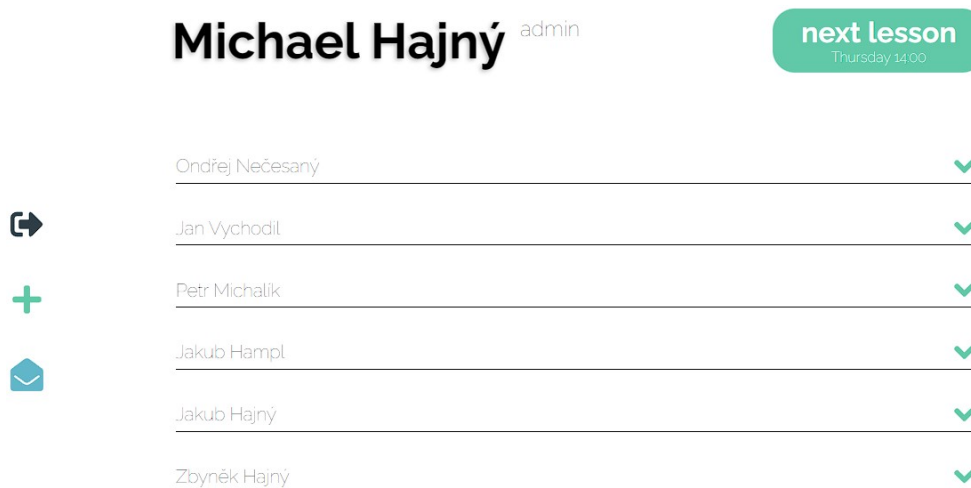


Obrázek 13: Platby

4.4 Administrátor

Administrátor, dále jen admin, je tedy role, která se stará o vytváření studentů, přidělování studentům hodiny, psaní domácích úkolů, vkládání podkladů k jednotlivým studentům, dále taky kontrolu omluvených hodin a plateb, viz. obrázek č. 14.

Stránka, na kterou se admin přihlásí, má též svá dělení. Rozdělena je na tři oblasti. V desktopové verzi se informativní oblast nachází úplně navrchu, kde je možné vidět, kdo se do role admina přihlásil a dále kdy doučujícího čeká následující hodina. Další oblast je řídicí, která se v desktopové verzi nachází vlevo, ale při zmenšení obrazovky na verzi tablet či mobil je hned pod oblastí s informacemi o následující hodině. V tuhle chvíli jsou zde jen tři možné akce, které admin může dělat a těmi jsou odhlášení, přidání studenta a zobrazení si pošty. Poslední je oblast se studenty, která se nachází uprostřed. V této oblasti má admin k dispozici různé funkce pro zajištění výuky. V dalších podkapitolách se budu jednotlivým oblastem věnovat a popíšu, jak s nimi admin může pracovat.



Obrázek 14: Dashboard administrátora

4.4.1 Informativní oblast

Jak tedy bylo v úvodu této kapitoli řečeno, tak oblast primárně umožňuje adminovi získat rychlý přehled o následující hodině. V případě adminem změněného času některé z hodiny se komponenta s následující hodinou aktualizuje. To samé platí i když admin hodinu zruší nebo se student z hodiny omluví. Ve všech případech by tady mělo být datum a čas následující hodiny, který počítá se všemi možnými změnami.

4.4.2 Řídící oblast

V rámci ovládacího panelu či řídicí oblasti, jsou zatím implementovány pouze tři funkcionality a výčtově se jedná o odhlášení, přidání uživatele a poštu. Vzhledem k budoucím plánům je však přidání dalších prvků možné. Obecně umožní administrátorům provádět více úkonů a manipulovat s různými daty. Celkový rozsah funkcionalit bude záviset na potřebách uživatelů a specifikách aplikace. Na konci této práce budu zmiňovat možné rozšíření aplikace o různé funkcionality které plánuji. Například jednou z takových věcí, kterou na konci plánuji více rozebrat, je osobní kalednář administrátora, aby graficky viděl, jak ten jeho týden či měsíc vypadá.

4.4.2.1 Přidávání uživatele

Stránka, pomocí které admin přidává uživatele, obsahuje jednoduchý formulář, viz. obrázek č. 15 a obrázek č. 16. Standardně admin vyplní jméno, příjmení, ze kterého se vygeneruje uživatelské jméno, dále už je přednastavené i unikátní heslo. V dalším kroce admin zadá plán pro studenta, který v čase může libovolně měnit. Admin také musí zadat den s časem, ve kterém chce, aby hodina probíhala. Lekcí samozřejmě může navolit i více.

Jestliže není vytvářený student plnoletý, tak admin zaškrtně políčko, které se týká jeho zletilosti. V případě, že je student nezletilý, tak je potřeba vložit údaje o zákonném zástupci. Pro případ, že by administrátor chtěl vkládat dva studenty, kteří spadají pod jednoho zákonného zástupce, tak se to řeší tím, že do pole, které je určené pro mobilní telefon, se vloží stejné číslo.

V případě nezadání některého z polí je administrátor obeznámen o tom, že dané pole není vyplněné. Při vytváření nové hodiny na stránce pro administrátora je to komplikovanější. Abych zachoval strukturu dat, použil jsem na stránce kalendářovou strukturu, ale administrátor v podstatě vybírá pro studenta pouze den, ve kterém se hodina bude konat. Konkrétní datum je vlastně zcela nepodstatné, protože data proběhlých hodin se stejně budou počítat až od data vytvoření studentova účtu a první započítaná hodina, kterou student bude platit, bude první hodina, která proběhla v týden založení studenta.

add student

name

name

surname

surname

username (auto)

username

password (auto)

LSztiMpL8w

study plan

(separate string with cor

count of hours

dd.mm.r

--:--

--:--

add new lesson day +

if student is under aged or need legal representative ☐

create

back to dashboard

Obrázek 15: Přidávání studenta

dd.mm.rr --:-- --:--

add new lesson day +

if student is under aged or need legal representative ☒

representative

name

name

surname

surname

password (auto)

BZ83xj43KP

phone number

create

back to dashboard

Obrázek 16: Přidávání studenta a jeho zákonného zástupce

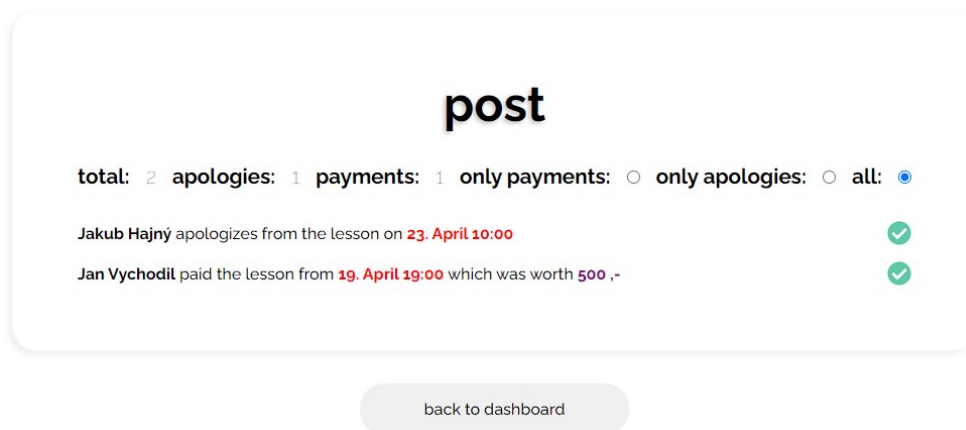
4.4.2.2 Pošta

Jedná se o část webové aplikace, kam správci chodí veškeré notifikace týkající se omluvenek a plateb, viz. obrázek č. 17. Administrátor může na této stránce kontrolovat omluvené hodiny a částky zaplacených lekcí.

Struktura je vždy stejná a má následující tvar: Kdo posílá zprávu, jakou informaci zpráva nese, pro které datum je zpráva určená a v případě platby je zde uvedená i částka.

Jedním z možných způsobů **kontroly** zaplacených lekcí, které proběhly, je použití naivního ověřovacího mechanismu, který je implementován prostřednictvím pošty. Tento mechanismus funguje následovně: když student absolvoval hodinu,

zobrazí se u něj v systému nezaplacená hodina. Tuto hodinu si student může označit jako zaplacenou tím, že zašle zprávu, ve které oznámí, že provedl platbu. Tato zpráva se poté zobrazí administrátorovi v jeho poště. V této zprávě je uvedeno, o jakou hodinu se jedná. Administrátor poté ověří v bankovním systému, zda platba proběhla a potvrdí ji. Po této kontrole je hodina u studenta označena jako zaplacená a nezobrazuje se již jako čekající na potvrzení a zmizí.



Obrázek 17: Stránka s poštou

4.4.3 Hlavní obsah se studenty

Následující text popisuje poslední ze tří sekcí, která se nachází v adminově dashboardu, a to sekci se studenty, viz. obrázek č. 18. Tato sekce poskytuje výčet všech studentů, kteří jsou registrováni v systému. Po rozbalení každého jednotlivého studenta jsou zobrazeny různé funkce spojené s tímto studentem, které mohou být aplikovány a ovlivní studentův profil, ať už přímo nebo nepřímo.

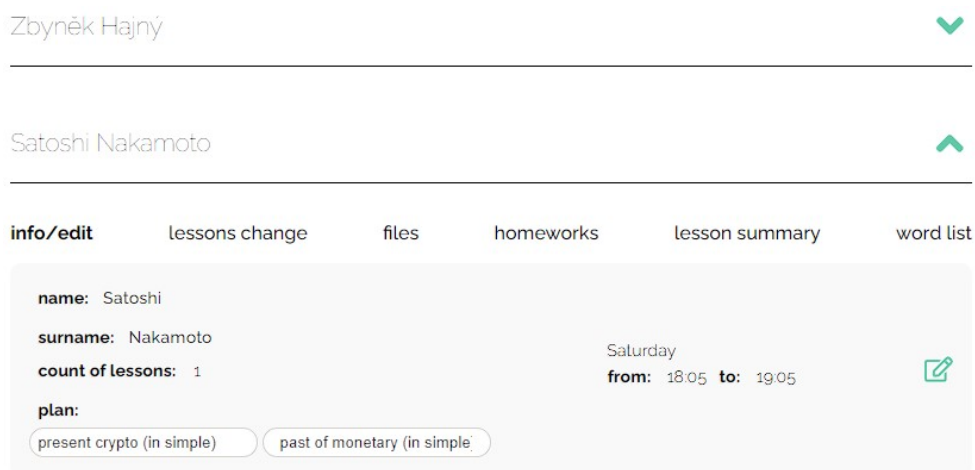


Obrázek 18: Ukázka výčtu studentů

4.4.3.1 Info/edit komponenta

Po rozkliknutí studenta, se jako vychozí zobrazí info/edit komponenta, viz. obrázek č. 19. Tato komponenta obsahuje základní informace o studentovi, které může správce aplikace editovat. V info/edit je možné vidět jméno a příjmení studenta, dále počet hodin, studijní plán a datum lekcí s časy.

Každá komponenta, která se zobrazí po kliknutí na studenta, má podobnou strukturu. Nejprve se zobrazí obsah, který je možné upravit pomocí ovládacích prvků, jež se nachází na pravé straně. V případě info/edit komponenty se po stisknutí tlačítka pro úpravy zvýrazní prvky, které je možné upravovat, a navíc se objeví tlačítko s názvem „disable“. Toto tlačítko slouží k tomu, aby se student v aplikaci již nadále nezobrazoval, ale jeho údaje zůstaly v databázi pro případnou obnovu.

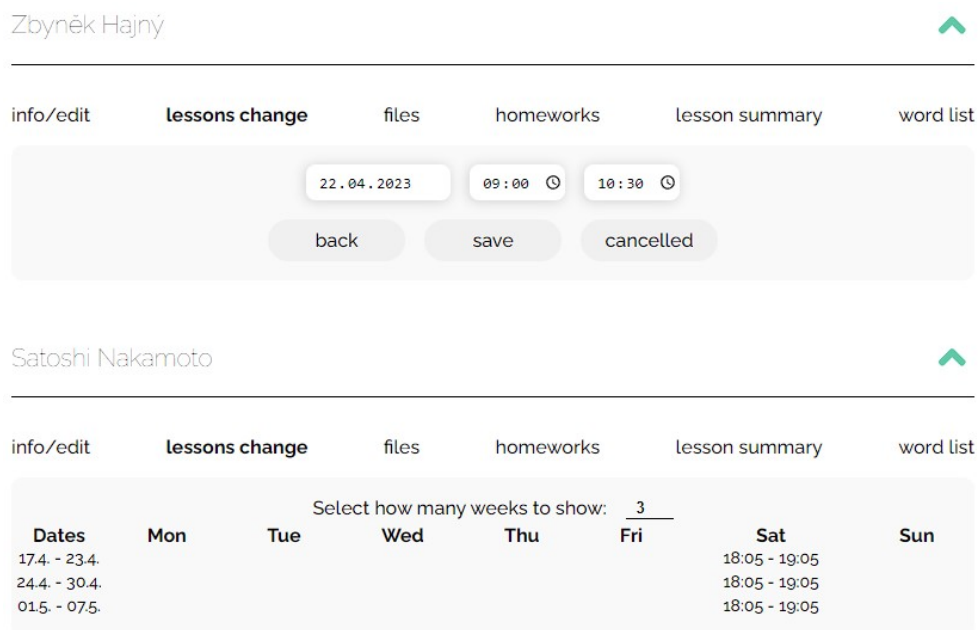


Obrázek 19: Informace o studentovi v administrátorovi

4.4.3.2 Komponenta pro změnu lekce

V komponentě pro změnu lekce se zobrazí n týdnů pod sebou v závislosti na počtu, který si lze zvolit na vrchu komponenty, viz. obrázek č. 20. Každý týden má přiřazený čas lekce, který se zobrazí v den, ve kterém se daná hodina koná. Tato komponenta slouží administrátorovi pro úpravu či zrušení hodiny. Tedy v případě potřeby lze vyučovací hodinu jednoduše změnit nebo odstranit.

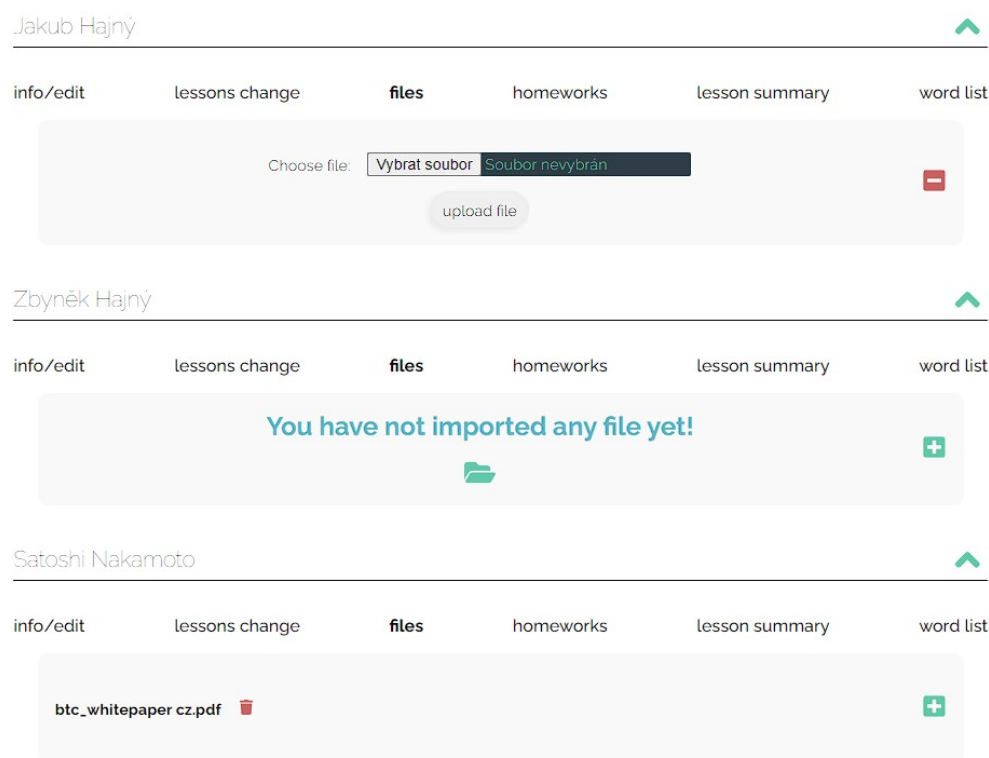
Při zrušení hodiny se nadále v daný den nebude lekce zobrazovat a při změně se to projeví i v komponentě edit změnou barvy písma.



Obrázek 20: Komponenta se změnou lekce

4.4.3.3 File komponenta

Jedná se o komponentu, která umožňuje přidávat a odebírat soubory danému studentovi, viz. obrázek č. 21. Jsou zde dva výchozí stavy, které jsou závislé na souborech. Jestli není žádný soubor vložený, tak se zobrazuje text, který říká, že jsme zatím žádný soubor nevložili a v případě vloženého souboru/souborů je zde výčet. Vpravo se opět nachází ovládací prvek, pomocí kterého můžeme zobrazit prvek pro vkládání souborů.

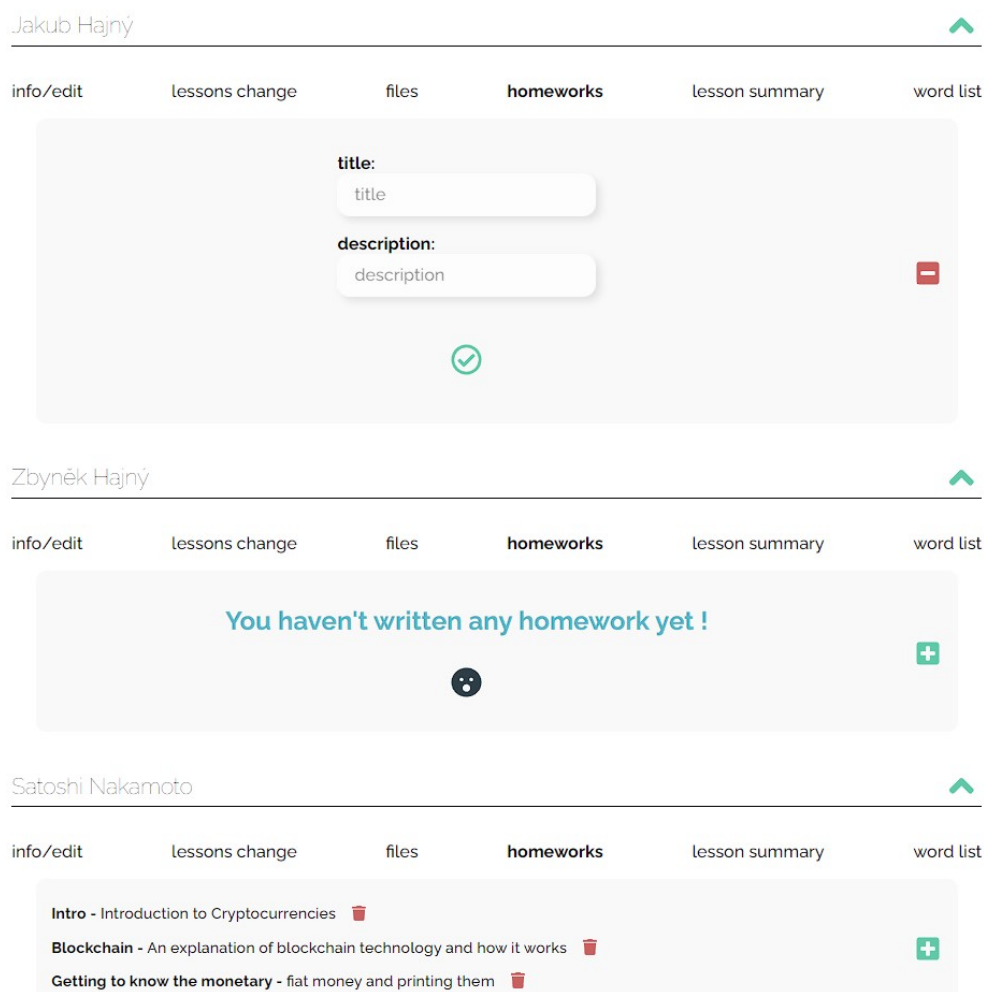


Obrázek 21: Komponenta pro práci se soubory

4.4.3.4 Homework komponenta

Práce s domácími úkoly v komponentě homeworks je velice intuitivní, viz. obrázek č. 22. Opět je zde výchozí zobrazení stejné jako u souborů a to znamená, že pokud soubory existují, tak se ukáže jejich výčet a my máme možnost je odstranit nebo se zobrazí text, který nám říká, že žádné domácí úkoly ještě nebyly pro studenta napsané.

Tlačítko na přidání úkolu nám odkryje formulář, ve kterém jsou dva vstupy pro text. První vstup je pro název úkolu a ten druhý pro popis.

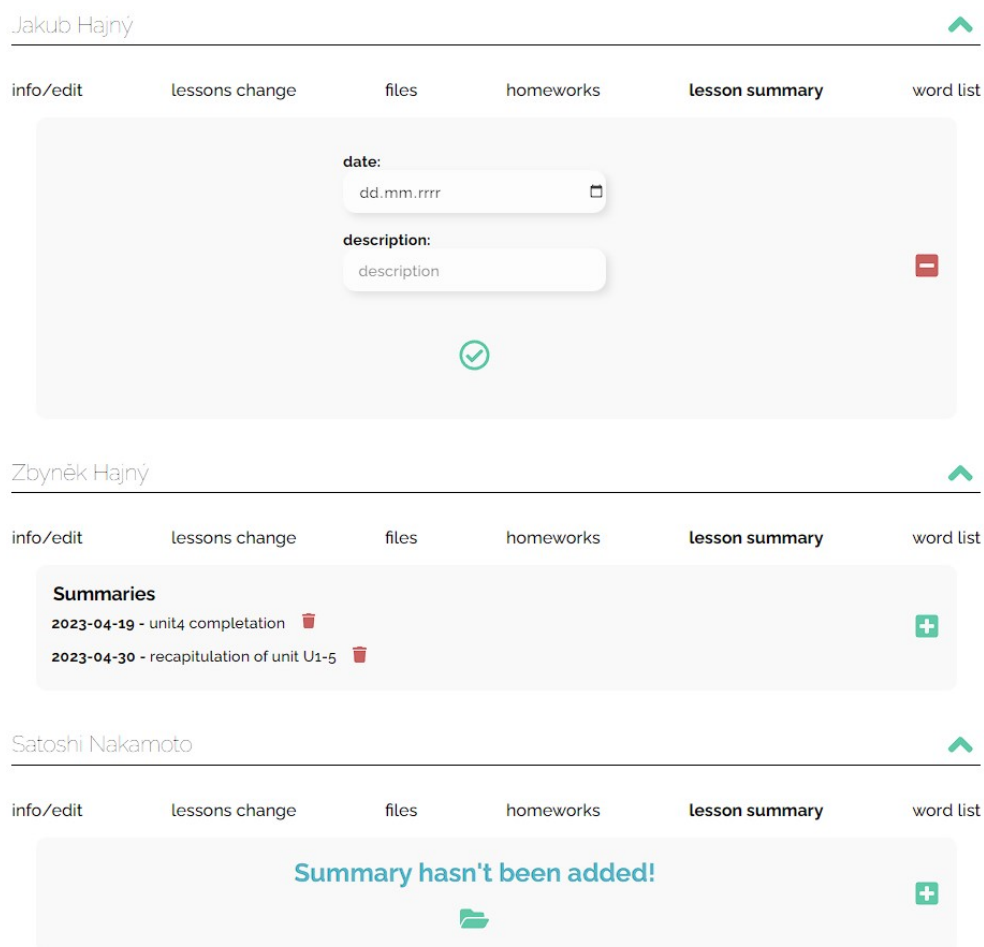


Obrázek 22: Komponenta pro práci s úkoly

4.4.3.5 Komponenta pro shrnutí lekce

Komponenta se funkcionálně podobá práci s domácími úkoly, ale je zde rozdíl ve formuláři, kde první vstup už není pro text, ale admin si vybírá datum, ke kterému se chce se shrnutím vyjádřit. Vizualizace na obrázku č. 23.

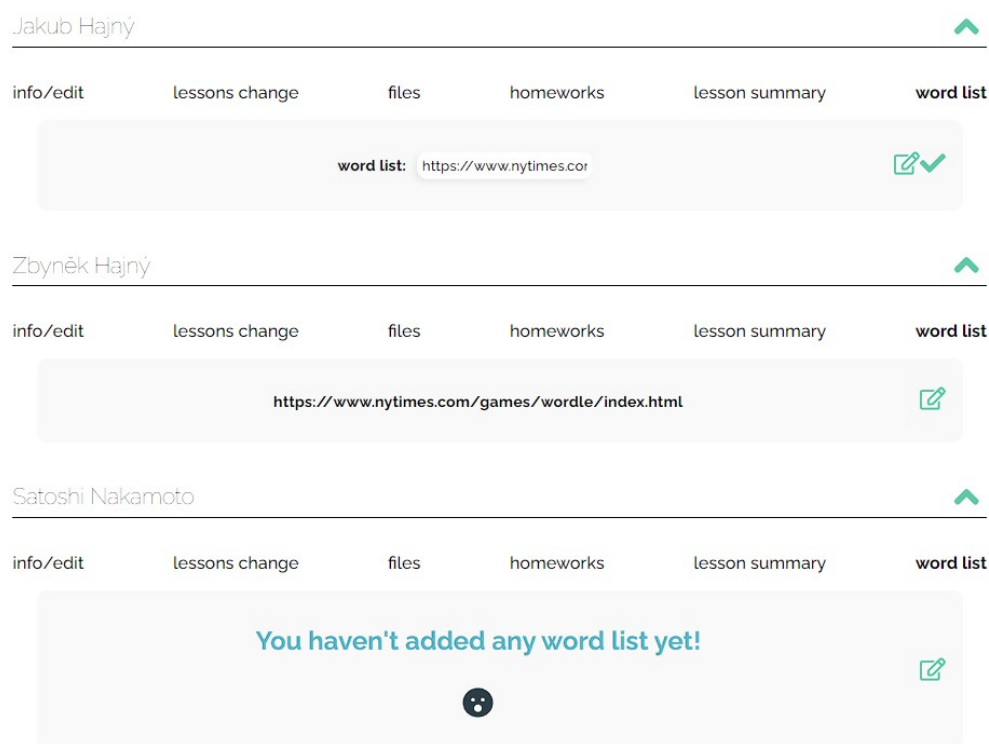




Obrázek 23: Komponenta pro shrnutí lekce

4.4.3.6 Komponenta pro vkládání online slovíček

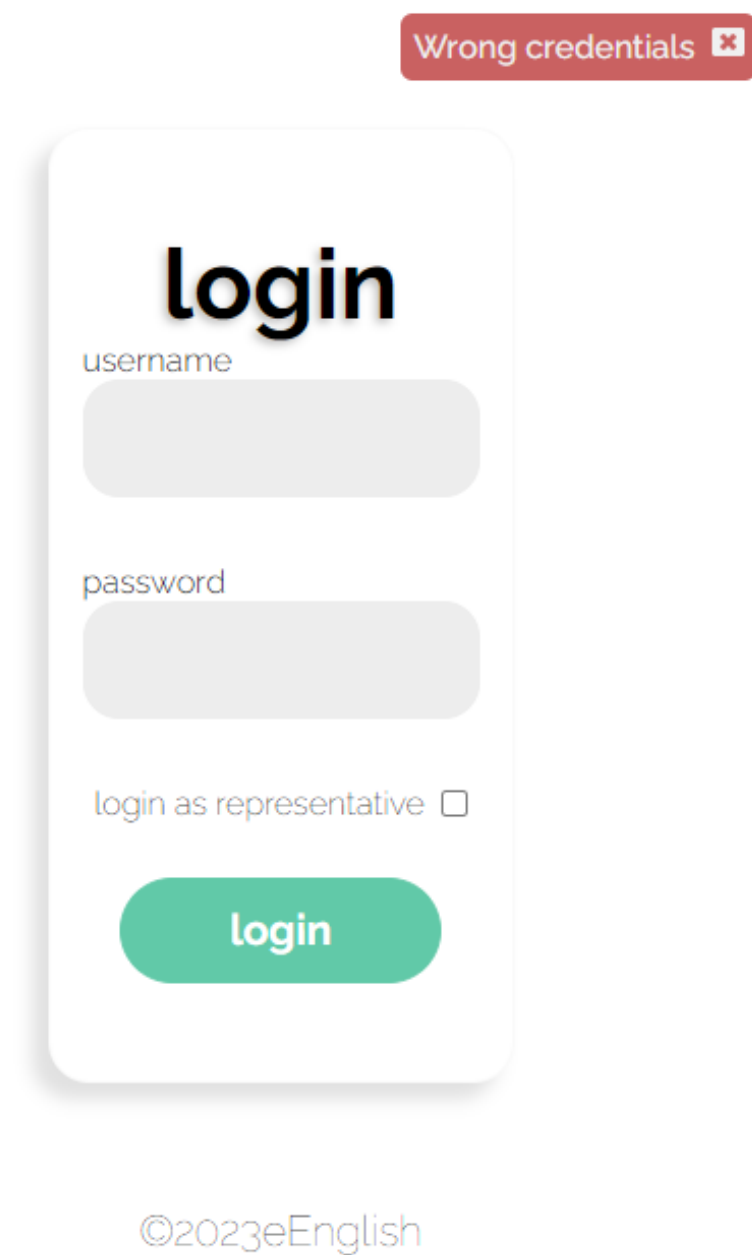
Wordlist komponenta slouží pro vkládání týdenního online wordlistu, ale i jiných přidružených her pro slovíčka, viz. obrázek č. 24. Při vytváření této komponenty jsem si říkal, že by bylo dobré nemít pouze jednoúčelovou komponentu, která je zaměřená na slovíčka, ale že by zde mohl být místo toho obecný vstup pro odkazy, které by učitel angličtiny chtěl studentům poslat. Zatím jsem tedy zůstal u jednoho odkazu, který učitel může měnit, ale do budoucna nevyklučuji, že by komponenta mohla projít úpravou.



Obrázek 24: Komponenta pro vkládání online odkazů na práci se slovíčky

4.4.3.7 Zbylé informační prvky na stránce

V mé aplikaci se také často používají prvky zpětné vazby, které informují uživatele o provedených akcích a mohou být buď pozitivní nebo negativní, viz. obrázek č. 25. Tyto prvky zahrnují například oznámení o úspěšně provedených úkonech nebo varování o neplatných nebo neúspěšných interakcích s prvky na stránce.



Obrázek 25: Oznámení o neúspěšné akci

Ve webové aplikaci jsou též umístěny prvky pro dotazování, které mají za úkol ověřit, zda chce uživatel opravdu provést danou akci, například před provedením destruktivních úkonů, viz. obrázek č. 26.

Obrázek 26: Příklad dotazovacího prvku, při destruktivní akci





5 Možná rozšíření aplikace

Na závěr mé bakalářské práce, bych chtěl zmínit, že je zde určitě prostor pro různá rozšíření. Aplikace je od samého začátku myšlena jako projekt, který se bude reálně používat, a tak na ní hodlám pokračovat i po ukončení bakalářského studia. V mých očích, aplikace, tak jak si ji představuji, není rozhodně dokončená, ale pro účel obhajoby jsem se veškeré funkcionality zadání snažil naplnit, aby bylo možné práci obhájit. Zkrátka je napsaná tak, aby to, co sem stihl, bylo prezentovatelné, ale další rozšíření určitě plánuji.

Uvedu zde několik funkcionalit a rozšíření, které mě napadaly během psaní bakalářské práce. Postupně jsem je zapisoval, aby bylo možné na nich dále pracovat. Některé z těchto funkcionalit by na integraci nebyly ani časově náročné, a tak jsem s nimi již počítal v průběhu strukturování práce, což by mělo usnadnit jejich implementaci. Na druhou stranu jsou zde i rozšíření, která by zabrala spoustu času, a tak jsem je uvedl do výčtu níže, abych se k nim mohl vrátit.

- Možnost integrace [React Native](#) [21] pro konvertování webové aplikace do mobilní aplikace. Lze to provést využitím funkcí API React Native, které umožňují přizpůsobit existující kód webové aplikace pro použití v mobilním prostředí.
- Vyhledávání v uživateli pro případ nadměrného množství uživatele a většího pohodlí v selekci konkrétního uživatele namísto dlouhého skrolování a manuálního vyhledávání uživatele.
- Možnost přidávat soubory a odkazy všem naráz (například online anglické hry a podobně).
- Platební brána.
- Integrovaní nějakých anglických her přímo v aplikaci (story dice, spelling game, wordle, apod.).

- Kalendář na straně admina, kde by viděl všechny hodiny, které přes týden má, aby věděl, na které dny si může přidat další studenty.
- Změna lekce permanentně. 
- Při změně hodiny adminem dostane uživatel notifikaci (email, aplikace).
- Úprava sekce s odkazy na online slovíčka na obecnou komponentu pro odkazy.
- Při zakládání nového uživatele, dát adminovi vědět, že nastavuje datum na den a čas, kdy už nějakého studenta má. Klidně ho tam ale může vložit v případě, že by na danou hodinu měl dva a více studentů zároveň ~~-je to spíše informativní.~~
- V modu pro zákonného zástupce udělat přepínač na češtinu, kdyby se jednalo o osobu, která anglicky neumí.
- Ve formuláři pro přidání studenta dodělat kolonku pro kontakt na zákonného zástupce nebo dítě.
- Integrované zprávy s jednotlivými studenty 

Závěr

S ohledem na stanovený cíl této bakalářské práce se podařilo vytvořit informační systém pro soukromého učitele angličtiny, který umožňuje efektivní a pohodlné vedení agendy. IS usnadňuje evidenci klientů a jejich plateb, sdílení materiálů, zadávání úkolů, vytváření individuálního studijního plánu, poskytování informací o následujících hodinách, umožňuje omluvy studentů a poskytuje přehledné shrnutí lekce.

Realizace IS, jako webové aplikace, zahrnuje vhodné řešení evidování zákonných zástupců v případech, kdy žákem je nezletilá osoba. Tento informační systém je snadno přístupný a použitelný pro všechny uživatele.

Na základě výše uvedeného mohu potvrdit, že cíl této bakalářské práce byl naplněn a vytvořený informační systém splňuje požadavky kladené na něj v rámci této práce.

Conclusions

With regard to the stated objective of this bachelor thesis, an information system for a private English teacher was created that facilitates efficient and convenient management of agenda. The IS simplifies client and payment records, sharing materials and assigning tasks, creating individual study plans, providing information on upcoming lessons, enabling student absences, and providing a clear summary of the lesson.

The implementation of the IS as a web application includes a suitable solution for recording legal representatives in cases where the student is a minor. This information system is easily accessible and usable for all users.

Based on the above, I can confirm that the objective of this bachelor thesis has been achieved and the created information system meets the requirements set for it within this work.

A Obsah přiloženého datového média

Zde je uveden stručný popis obsahu přiloženého datového média, tj. jeho závazné adresářové struktury, důležitých souborů apod.

bin/

Obsahuje zdrojové kódy webové aplikace v archivu ZIP.

doc/

Obsahuje text práce ve formátu PDF a zdrojové texty v archivu ZIP.

src/

Obsahuje kompletní zdrojové kódy webové aplikace v archivu ZIP.

readme.txt

Obsahuje instrukce pro spuštění webové aplikace

Literatura

- [1] *Coloors.co*. [online]. [cit. 2022-4-23]. Dostupný z: <https://coolors.co/>.
- [2] *Gradientos.co*. [online]. [cit. 2022-4-23]. Dostupný z: <https://freestuff.dev/alternative/coolors.co/>.
- [3] *Google Fonts*. [online]. [cit. 2022-4-23]. Dostupný z: <https://fonts.google.com>.
- [4] *Google Fonts - Know how*. [online]. [cit. 2022-4-23]. Dostupný z: https://fonts.google.com/knowledge/choosing%5C_type.
- [5] *Google Fonts - Raleway*. [online]. [cit. 2022-4-23]. Dostupný z: <https://fonts.google.com/specimen/Raleway?query=raleway>.
- [6] *Font Awesome*. [online]. [cit. 2022-4-23]. Dostupný z: <https://fontawesome.com/search>.
- [7] *Figma*. [online]. [cit. 2022-4-23]. Dostupný z: <https://www.figma.com/>.
- [8] Michálek, Martin. *Vzhůru do responzivního webdesignu*. Praha: Vydavatelství vlastní, 2017. Dostupný také z: <https://www.vzhurudolu.cz/kniha-responzivni-design/>.
- [9] OpenAI. *OpenAI Chat GPT*. 2021. Dostupný také z: <https://openai.com/>.
- [10] Rauch, Guillermo. *Next.js*. 2016. Dostupný také z: <https://nextjs.org/>.
- [11] Inc., MongoDB. *MongoDB*. 2009. Dostupný také z: <https://www.mongodb.com/>.
- [12] Inc., MongoDB. *MongoDB Atlas*. 2021. Dostupný také z: <https://www.mongodb.com/cloud/atlas>.
- [13] Allen, Mike. *Axios*. 2017. Dostupný také z: <https://axios-http.com/docs/intro>.
- [14] *Mongoose*. Dostupný také z: <https://www.mongodb.com/developer/languages/javascript/getting-started-with-mongodb-and-mongoose/>.
- [15] Walke, Jordan. *React*. 2013. Dostupný také z: <https://react.dev/>.
- [16] *React Icons*. 2020. Dostupný také z: <https://react-icons.github.io/react-icons/>.
- [17] *Styled Components*. Dostupný také z: <https://styled-components.com/>.
- [18] Inc., Vercel. *Vercel*. 2021. Dostupný také z: <https://vercel.com/>.
- [19] Salesforce.com, Inc. *Heroku*. 2021. Dostupný také z: <https://www.heroku.com/>.
- [20] Inc., Netlify. *Netlify*. 2021. Dostupný také z: <https://www.netlify.com/>.

- [21] Facebook, Inc. *React Native*. 2021. Dostupný také z: <https://reactnative.dev/>.

