Univerzita Hradec Králové
Fakulta informatiky a managementu
Katedra informačních technologií

Systém pro podporu vzdělávání dětí se znevýhodněními

Bakalářská práce

Autor: Jiří Kumprecht
Studijní obor: Aplikovaná informatika

Vedoucí práce: Mgr. Daniela Ponce, PhD.

Hradec Králové duben 2021

Prohlášení:

Prohlašuji, že jsem bakalářskou práci zpracoval samostatně a s použitím uvedené literatury.

V Hradci Králové dne 29.4.2021 Jiří Kumprecht

Poděkování:

Děkuji vedoucí bakalářské práce Mgr. Daniele Ponce, PhD. za metodické vedení práce, pomoc a cenné připomínky.

Anotace

Bakalářská práce se zaměřila na vývoj systému pro podporu vzdělávání žáků se znevýhodněními pro konkrétní školu Základní školu speciální a praktickou školu Diakonie Českobratrské církve evangelické Vrchlabí. Škola se na autora práce obrátila z nutnosti aktuální situace, kterou zapříčinil vývoj epidemie Covid-19 a požádala jej o vývoj aplikace pro podporu i nové formy vzdělávání, a to zákonem povinné distanční výuky. Pro vytvoření a vývoj aplikace byly podkladem funkční a nefunkční požadavky ze strany školy. Na základě těchto požadavků došlo k vypracování návrhu, podle kterého systém byl v konečné fázi implementován.
Hlavním cílem bylo analyzovat, navrhnout a implementovat systém pro podporu vzdělávání žáků se speciálními vzdělávacími potřebami, a to pro žáky s lehkým, středním a těžkým mentálním postižením.

Klíčová slova

aplikace vzdělávací, aplikace webová, React, žáci se speciálními vzdělávacími potřebami

Annotation

Title: Education support system for children with disabilities

The bachelor's thesis focused on the development of a system to support the education of pupils with disabilities for a specific school Elementary special school and practical school Diakonie of the Czech Brethren Evangelical Church Vrchlabí. The school turned to the author of the work out of the need for the current situation caused by the development of the Covid-19 epidemic and asked him to develop an application to support even new forms of education, compulsory distance education by law. Functional and non-functional requirements from the school were the basis

for the creation and development of the application. Based on these requirements, a design was created, according to which the system was finally implemented.
The main goal was to analyze, design and implement a system to support the education of pupils with special educational needs, for pupils with mild, moderate and severe mental disabilities.

Key words

educational application, web application, React, pupils with special educational needs

Obsah

Seznam obrázků

Seznam tabulek

1 Úvod

V bakalářské práci se autor zaměřil na vývoj systému pro podporu vzdělávání žáků se znevýhodněními pro Základní školu speciální a praktickou školu DČCE Vrchlabí. Škola požádala autora práce o vývoj této aplikace pro klasické prezenční vzdělávání a v současné době na základě aktuální epidemiologické situace pro využití při povinném distančním vzdělávání.

Hlavním cílem byla analýza, návrh a implementace systému pro podporu vzdělávání žáků se speciálními vzdělávacími potřebami.

Pro vytvoření a vývoj aplikace bylo potřeba analyzovat problematiku sběrem funkčních a nefunkčních požadavků, které byly podkladem pro zpracování návrhu. Na základě těchto stanovených požadavků došlo k vypracování návrhu, podle kterého byl systém v konečné fázi implementován.

V bakalářské práci byly vymezeny jednotlivé stupně mentální retardace a jejich charakteristiky. Dále bylo zmíněno, kterými technickými prostředky škola disponuje a jejich prostřednictvím se k vyvíjenému systému přistupovalo. Došlo ke shrnutí zařazených předmětů s kategoriemi, které systém nabízí. Následovalo seznámení s architekturou aplikace, způsobem komunikace mezi jednotlivými částmi systému a použitým návrhovým vzorem.

Následující kapitola doplnila, jakým způsobem se typová cvičení vykonávají a jak jsou strukturovány. Poté se vytvořily modely, ve kterých jsou detailně popsány informace, které se u nich uchovávají. Poslední hlavní kapitola se zaměřila na implementační část vývoje, kde byly charakterizovány jednotlivé použité technologie pro vývoj webové aplikace a dílčí části použité struktury projektu. Poslední podkapitoly byly věnovány popisu implementace jednotlivých funkcí.

2 Cíl práce

Hlavním cílem je analyzovat, navrhnout a implementovat webovou aplikaci pro podporu výuky žáků se speciálními vzdělávacími potřebami, pro žáky s lehkým, středním a těžkým mentálním postižením.
Tento systém má usnadnit vzdělávání i v rámci distanční výuky, která je již součástí povinného vzdělávání.

3 Metodika zpracování

Základní fáze tvorby softwaru jsou úvodní studie, analýza, návrh a implementace. Všechny výše zmíněné fáze byly řádně provedeny v uvedeném pořadí. Pro vývoj aplikace byl zvolen přírůstkový model, který umožňuje rozdělit projekt na menší segmenty a zjednodušuje možnost zavádění změn v průběhu procesu vývoje aplikace. S tímto modelem bylo možné aplikaci vyvíjet po částech prostřednictvím sérií malých vodopádů. Pro uvedení příkladu, jednotlivé vodopády si lze představit jako průběh vývoje jednotlivých typových aktivit. Pokaždé se nejdříve začínalo analýzou typové úlohy, která byla hlavní náplní schůzky se speciálními pedagogy, ve které se specifikovala funkčnost dané aktivity. Na základě analýzy se postupovalo do následující fáze, tedy do fáze návrhu. Z požadavků analýzy byl vytvořen návrh, podle kterého by se aktivita dala implementovat. Implementací je myšlen poslední krok, kterým se dosáhne dokončení dané vyvíjené části. To znamená, naprogramování kódu, který návrh realizuje. Takto se postupovalo i u dalších třech typových aktivit.

4 Analýza aplikace

4.1 Současný stav vzdělávání žáků na ZŠS a PŠ DČCE Vrchlabí

Na Základní škole speciální Diakonie ČCE Vrchlabí probíhá výuka dětí s různými stupni postižení. Žáci zde mají diagnostikované lehké, střední nebo těžké mentální postižení. Ve zmiňované speciální škole se tedy vzdělávají i žáci, kterým míra postižení umožňuje k výuce používat ICT. Většina žáků má k dispozici svůj školní tablet, někteří i notebook s dotykovou obrazovkou. Každá třída disponuje interaktivní tabulí s projektorem. V některých třídách jsou počítače, ke kterým jsou připojeny senzory pro snímání pohybu očí, které mají nahrazovat fyzickém myši. Děti s lehkým mentálním postižením dokáží používat své notebooky, tablety a zvládají práci s webovými aplikacemi, jako jsou například školákov.eu, rysava.websnadno.cz. Na těchto portálech se nachází cvičení vhodná pro žáky prvního stupně základní školy a některé z těchto cvičení však bez problému zvládají i žáci s lehkým mentálním postižením (LMP). S ohledem na vývoj epidemie Covid-19 narůstá požadavek distančního vzdělávání za použití ICT.

4.2 Definice mentální retardace

Bazalová v publikaci Pipekové [6] charakterizuje mentální retardaci jako celkové snížení intelektových schopností, tedy schopnosti myslet, učit se a přizpůsobovat se svému okolí, k němuž dochází v průběhu vývoje jedince. Jde o trvalý stav, který je buď vrozený nebo získaný. Grossman a kol. v publikaci Emersona [7, s. 12]: „*Mentální retardace byla definována jako významně podprůměrné obecně intelektuální fungování [IQ < 70], které buď vede k současnému zhoršení adaptivního chování, nebo je s ním spojené.*“

4.3 Klasifikace mentální retardace

Podle Švarcové [8] se pro klasifikaci mentální retardace v současné době užívá 10. revize Mezinárodní klasifikace nemocí zpracovaná zdravotnickou organizací v Ženevě, která vstoupila v platnost v roce 1992. „*Podle této klasifikace se mentální retardace dělí do šesti základních kategorií: lehká mentální retardace, středně těžká*

mentální retardace, těžká mentální retardace, hluboká mentální retardace, jiná mentální retardace, nespecifikovaná mentální retardace.“ [8, s. 37]

Tabulka 1 Klasifikace mentální retardace dle MKN-10

KÓD	DRUH	IQ
F 70	Lehká mentální retardace	50–69
F 71	Střední mentální retardace	35–49
F 72	Těžká mentální retardace	20–34
F 73	Hluboká mentální retardace	Nejvýše 20
F 78	Jiná mentální retardace	-
F 79	Neurčená mentální retardace	-

4.3.1 Lehká mentální retardace

Publikace [9] uvádí, že u lehké mentální retardace se jedná o snížení intelektových schopností v pásmu inteligenčního kvocientu (IQ) 50 až 69. Souběžně se objevuje zpoždění v adaptivních funkcích, které se vyskytují před dosažením osmnácti let věku. Tyto děti jsou dle Bazalové [10] znevýhodněné, neboť zdědí nedostatečné vlohy včetně hodnot IQ, které se převážně vyskytují v kombinaci s nepodnětným prostředím. Z důvodu jejich absence v předškolním vzdělávání je mnoho dětí diagnostikováno až při zápisu nebo nástupu do školy, neboť nejvýraznější problémy se projeví v období školní docházky. Děti disponují mechanickým myšlením, oslabenou pamětí, omezenou schopností logického myšlení, nedostatečnou analýzou a syntézou, poruchami pohybové koordinace. S tím souvisí opožděný vývoj jemné a hrubé motoriky. Dále se jedná o zpomalený rozvoj sociálních dovedností, v emocionální oblasti afektivní labilita, impulzivnost, úzkostnost či zvýšená sugestibilita. U dospělých jedinců s LMP se mentální věk pohybuje mezi 9 až 12 let. Hlavní potíže u žáků s LMP Vančová v publikaci Bendové [11] převážně spatřuje ve zvládání školních nároků, a to při teoretické práci ve škole. U žáků se projevují

specifické problémy se čtením a psaním. Důležité je výchovu a vzdělávání zaměřit na rozvoj jejich schopností a kompenzování nedostatků. Lehké mentální postižení je diagnostikováno přibližně u 80–85 % osob s postižením. Tito jedinci většinou dokáží účelně užívat řeč v každodenním životě, verbálně komunikovat a udržovat konverzaci, i přesto, že vývoj řeči bývá v dětském věku opožděn. Jsou schopni navazovat a udržovat vhodné sociální vztahy. Většina jedinců dosáhne úplné nezávislosti v osobní péči, při hygieně, oblékání, stravovacích návycích a v domácích pracích. Rozvoj sebeobsluhy a samostatnosti ve srovnání s normou bývá pomalejší. Většina jedinců zvládne v dospělosti vykonávat jednoduchou manuální činnost a zařadit se do pracovního procesu. Dle Švarcové [8, s. 38]: „*U osob s lehkou mentální retardací se mohou v individuálně různé míře projevit i přidružené chorobné stavy, jako je autismus, a další vývojové poruchy, epilepsie, poruchy chování nebo tělesná postižení*.“

4.3.2 Střední mentální retardace

Podle Švarcové [8] střední mentální postižení (SMP), IQ 35 až 49, je diagnostikována přibližně u 10 % osob s mentálním postižením. V užívání řeči a rozvoji chápání se objevuje výrazné opoždění. Soběstačnost a zručnost jsou výrazně omezeny. Žáci se SMP jsou schopni osvojit si základy čtení, psaní a počítání. Prostřednictvím strukturované výuky a zajištění odborného dohledu jsou tito jedinci v dospělosti schopni vykonávat jednodušší manuální činnost, např. v chráněných pracovních místech. Většina jedinců je plně mobilní, prokazuje schopnost navazovat kontakty, komunikuje s druhými a podílí se na jednoduchých sociálních aktivitách. Někteří skupiny jedinců jsou úspěšnější v senzoricko-motorických dovednostech než ve verbálních schopnostech. Rozdíly se objevují v sociální interakci a komunikaci. Někteří jedinci se SMP jsou schopni navazovat jednoduché konverzace, jiní stěží mluví o svých základních potřebách. Bazalová [10, s. 24]: „*Řeč bývá velmi jednoduchá, slovník obsahově chudý, vyskytují se agramatismy. Vývoj jemné a hrubé motoriky bývá zpomalen a trvale zůstává celková neobratnost, nekoordinovanost pohybů a neschopnost jemných úkonů*.“

Bendová [11] sděluje, že mentální věk u dospělých osob se středním mentálním postižením se pohybuje v rozmezí 6 až 9 let. Úroveň rozvoje řeči je rozdílná, některá

skupina jedinců je schopna jednoduché konverzace, jiní využívají alternativní či augmentativní komunikaci. Další skupina jedinců se mluvit nikdy nenaučí, dokáže vyjádřit své potřeby na základě funkčně používané gestikulace a dalších forem nonverbální komunikace.

4.3.3 Těžká mentální retardace

Dle Valenty a Müllera [12] se jedná o výrazné opoždění psychomotorického vývoje, které je patrné již v předškolním věku. Možnosti sebeobsluhy jsou výrazně a trvale limitované, jedinci nejsou schopni sebeobsluhy, řečový vývoj stagnuje na předřečové úrovni. Charakteristickými znaky jsou poruchy chování v podobě stereotypních pohybů, sebepoškozování, afektů a agrese.

Podle Pipekové [6] pro osoby s těžkým mentálním postižením (TMP), IQ 20 až 34, je patrná značná pohybová neobratnost. Je nutné dlouhodobé osvojování koordinace pohybů. Jsou přidružena tělesná postižení s dalšími vadami. Zároveň se jedná o omezení psychických procesů, poruch pozornosti, nestálost nálad a impulzivitu. Mentální věk u dospělých osob s TMP se nachází mezi 3 až 6 let.

Švarcová v publikaci Bendové [11, s. 14 - 15]: „*I přesto, že jsou možnosti výchovy a vzdělávání těchto dětí značně omezeny, zkušenosti ukazují, že včasná systematická a dostatečně kvalifikovaná rehabilitační, výchovná a vzdělávací péče může významně přispět k rozvoji jejich motoriky rozumových schopností, komunikačních dovedností, jejich samostatnosti a celkovému zlepšení kvality jejich života.*“

4.4 Dostupné ICT nástroje školy

Škola je vybavena mnoha moderními zařízeními. Jak již bylo zmíněno, každá třída disponuje interaktivní tabulí s datovým projektorem a každý žák má svůj školní tablet nebo školní notebook.

4.4.1 Tablet

Tablet je malé snadno přenositelné zařízení s dotykovou obrazovkou. Tablety, které má škola ve výbavě, obsahují různé operační systémy, jako například Android, iOS. Zařízení postrádá fyzickou klávesnici. Ta je nahrazena virtuální klávesnicí, která se v případě potřeby zobrazí na obrazovce.

4.4.2 Notebook

Notebook je lehce přenosné zařízení, které disponuje velkým displejem. Notebooky, které škola svým žákům zajistila, mají dotykovou obrazovku a monitor se dá překlopit o 360 stupňů, proto je lze zároveň využít jako tablet. Oproti tabletům mají všechny notebooky operační systém Windows 10.

4.4.3 Interaktivní tabule

Interaktivní tabule je dotykové zařízení, které dokáže snímat dotyky po celé její rozsáhlé ploše. Připojuje se k ní počítač a datový projektor. Datový projektor slouží pro zobrazení obsahu, který produkuje počítač. Ve výukových hodinách je nejčastěji využívaným ICT nástrojem. V některých třídách je tabule nainstalovaná přímo na stěnu, a v ostatních se využívá specializovaných pojezdných stojanů, určených pro uchycení interaktivní tabule.

4.4.4 Senzor očí Tobii PC Eye Go

Senzor pro snímání pohybu očí nahrazuje klasickou počítačovou myš. Škola jej využívá pro studenty, kteří nejsou schopni klasickou myš použít, tedy pro žáky s tělesným postižením. K počítači se zařízení připojuje pomocí USB konektoru. Podporuje operační systém Windows 10, což dané škole vyhovuje, jelikož tento systém na svých počítačích používá.

4.5 Funkční požadavky navrhované školou

Pro zjištění funkčních požadavků vyvíjené aplikace si autor práce domluvil osobní schůzku s několika speciálními pedagogy dané školy a formou explorativní metody – metodou volného rozhovoru s nimi komunikoval. Škola si stanovila za cíl používat k výuce jednu komplexní aplikaci, která bude vhodná pro děti se speciálními vzdělávacími potřebami, která bude zohledňovat jejich individuální schopnosti a aktuální možnosti.

4.5.1 Seznam požadavků vycházející z rozhovoru

Požadavky vycházející z rozhovoru ze strany školy jsou v této kapitole postupně rozepsány. V aplikaci budou výuková cvičení zařazena do předmětů, do kterých

náleží. V každém předmětu je možné filtrovat cvičení na základě úrovně mentálního postižení. U každé aktivity se bude sledovat počet jejího dokončení. Dalším požadavkem školy je vyhledávat cvičení pomocí chytrého vyhledávače, např. zadáním názvu cvičení. V aplikaci se budou rozlišovat tři úrovně postižení, jimiž jsou lehké, střední a těžké mentální postižení. Cvičení budou tvořena na základě školního vzdělávacího programu (ŠVP), tematických plánu a učebních osnov každého předmětu vycházející z rámcového vzdělávacího programu (RVP) pro ZŠ. Pro žáky s LMP budou navoleny následující předměty: Český jazyk, Anglický jazyk, Matematika, Prvouka, Přírodověda. Pro žáky se SMP budou dostupné předměty: Matematika, Český jazyk, Svět a já, Společnost a já. S ohledem na vzdělávací potřeby žáků s TMP, kdy dochází k modifikaci učiva, budou zvoleny tyto předměty: Rozumová výchova, Smyslová výchova, Řečová výchova, Zdravotní tělesná výchova. Hudební výchova bude dostupná pro všechny stupně postižení. Do samotné aplikace bude přístup možný třemi způsoby.

Dostupnými rolemi jsou host, učitel nebo žák. Vstup do aplikace jako host registraci nevyžaduje. Pro přístup k aplikaci v rolích učitele nebo žáka bude registrace nutná. Uživatel přihlášený v roli učitele, bude mít zpřístupněny následující funkce:

- Bude mít možnost registrovat své žáky a toto je jediný způsob, kterým se bude moci vytvořit uživatel s rolí žáka. Žáci mají účet pro přihlášení k aplikaci z domu nebo ze svých školních zařízení.
- Při registraci žáka lze nastavit jeho úroveň postižení, která bude použita k filtraci jemu vhodných aktivit.
- Může editovat profily svých žáků.
- Bude si uchovávat svůj seznam žáků.
- Před spuštěním cvičení má učitel možnost vybrat si žáka, kterému cvičení spustí. Tato funkce byla vložena z důvodu, aby si učitel mohl ukládat výsledky žáků, a přitom nepřihlašovat individuálně každého žáka, který bude zrovna u interaktivní tabule pracovat.
- Po dokončení aktivity žákem bude moci uložit vlastní poznámku o průběhu.

- Učitel může zadávat domácí úkoly jednotlivým žákům. Z důvodu individuality a rozmanitosti schopností jednotlivých žáků není potřeba implementovat funkci zadávání hromadných domácích úkolů.
- Bude mít možnost vytvářet vlastní aktivity na základě dostupných typových úloh. Může je upravit, sdílet, odstranit.

Funkce žáka:

- Spouštět cvičení z domácího úkolu.
- Výsledky dokončených cvičení z domácího úkolu se ukládají společně s počtem chyb.

Funkce hosta jsou dostupné i všem ostatním rolím:

- Spouštět všechna cvičení dle uvážení.
- Filtrovat cvičení dle předmětů, kategorií a úrovně postižení.
- Vyhledávat cvičení podle názvu.

Kategorie předmětů

Kategorie předmětů byly zvoleny v souladu se ŠVP školy, pro kterou je aplikace vyvíjena. Aplikace obsahuje předměty s jednotlivými kategoriemi, případně podkategorie, které vycházejí z tematických částí učebních osnov, které jsou povinnými standardy pro výchovu a vzdělávání.

Pro vzdělávací oblast matematika byly vybrány následující kategorie: předmatematické pojmy, které se dále člení na jednotlivé podkategorie: porovnávání, třídění, tvoření skupin, řazení, množství a tvary. Dalšími kategoriemi této vzdělávací oblasti jsou: číslo a prvek, číselná řada, sčítání a odčítání, porovnávání, násobení, rozklady čísel.

Druhou vzdělávací oblastí je český jazyk. Do této oblasti patří tyto kategorie: abeceda, přiřazování obrázku ke slovu, měkké a tvrdé souhlásky, párové souhlásky, ú nebo ů, čtení s porozuměním. Kategorie abeceda se dělí na dvě podkategorie: seřaď písmeno podle abecedy, hláska „x“ nás probudí.

Třetí vzdělávací oblastí je anglický jazyk. Předmět se člení do následujících kategorií: dny, týdny, měsíce, pozdravy, rodina, roční období, zvířata, čísla, barvy, ovoce a zelenina, jídlo a části těla.
Pro vzdělávací oblast prvouka byly navoleny tyto kategorie a jejich podkategorie:

- místo, kde žijeme
    - bydliště, škola, významná místa, okolí, dopravní výchova
- lidé kolem nás
    - rodina, povolání, pravidla chování
- lidé a čas
    - hodiny, dny, týdny a měsíce, roční období
- rozmanitost přírody
    - počasí a proměny přírody, domácí zvířata, hospodářská zvířata, živá a neživá příroda, ovoce a zelenina
- člověk a jeho zdraví
    - péče o zdraví, lidské tělo, etapy lidského života, první pomoc

Vzdělávací oblasti vlastivěda byly přiřazeny kategorie:

- místo, kde žijeme
    - domov, škola, Česká republika, zahraničí, okolí
- lidé kolem nás
    - pravidla chování, finanční gramotnost
- lidé a čas
    - orientace v čase, zvyky a tradice, památky

Pro vzdělávací oblast hudební výchova byly navoleny kategorie:

- vokální a instrumentální činnosti
    - zpěv písní, hra na tělo, Orffův instrumentář
- hudebně pohybová činnost
    - tanec, rytmus
- poslechové činnosti

    - poslech hudby, skladatelé

Poslední přidanou vzdělávací oblastí je tělesná výchova, které byly přiřazeny kategorie:

- činnosti ovlivňující zdraví
    - pohybové dovednosti, letní sporty, zimní sporty
- bezpečnost a hygiena
    - nářadí, úbor, pravidla jednání

Všem předmětům byla přidána vlastní kategorie „Nezařazeno“, pro aktivity, které nelze do výše zmiňovaných kategorií jednoznačně zatřídit.

4.6 Nefunkční požadavky

V nejvyšší řade jsou kladeny požadavky na bezpečnost užívání webové aplikace. Pro zajištění bezpečnosti je vyžadováno, aby komunikace klienta se serverem probíhala šifrovaně, například za použití SSL certifikátu.
Z ohledu dostupnosti je požadováno, aby bylo možné k aplikaci přistoupit odkudkoli z libovolného zařízení za použití internetového prohlížeče.
Z hlediska přístupnosti uživatelských účtu, je nutné zajistit způsob autentizace a pro přidání oprávnění uživatelům provádět různé operace v aplikaci bude zajištěna autorizace.
Dále se kladl důraz na udržitelnost. Systém byl vhodně rozdělen do komponent, aby bylo možné v případě potřeby opravit pouze tu komponentu, ve které se vyskytne problém a nemusí být upravován celý rozsah aplikace.
Pro možnost rozšiřování aplikace bez zasahování do již funkčních části systému se dbalo na požadavek rozšiřitelnosti. Rozšiřitelnost umožnuje přidat novou funkcionalitu nebo upravovat již stávající funkce, aniž by se ovlivnil existující systém.
Pro plynulý chod aplikace pro více uživatelů současně je zajištěno, aby se požadavky uživatelů vzájemně neblokovaly.

5 Návrh aplikace

Z důvodu předpokládaného používání vyvíjené aplikace - na tabletech, počítačích – tudíž na zařízeních s rozdílnými operačními systémy, se autor práce zaměřil na tvorbu webové aplikace s responzivním designem, aby bylo možné aplikaci využívat na všech zmíněných zařízeních. Na základě předchozích zkušeností si autor práce zvolil technologie, které dobře zná a ovládá.

5.1 Architektura aplikace

V této kapitole se popisuje použitá architektura aplikace.

5.1.1 Architektura Klient-server

Architektura spočívá v rozdělení aplikací, které mohou zastupovat roli klienta nebo serveru. Pokud aplikace právě zasílá http požadavek, pak zastává roli klienta. Aplikace, která je připravena naslouchat požadavkům a patřičně je obsluhovat, zastává pak roli serveru.

5.1.2 Frontend

Frontend ve volném překladu znamená „přední část“ a je označením pro část webové aplikace, která je viditelná pro návštěvníky webových stránek, e-shopu či jiné webové prezentace. K tomuto tématu se vyjadřuje i společnost Welcome to the Jungle [1]: *„Front-end vývojáři vytváří kód pro uživatelské rozhraní webové stránky a prvky, které uživatelům umožňují s danou stránkou interagovat. Mají na starosti funkčnost, intuitivnost a jednoduchou ovladatelnost vizuálního prostředí stránky.“*

5.1.3 Backend

Jan Štráfelda [2] k tomuto termínu uvádí: *„Jako backend se označuje část webové aplikace, která slouží k administraci webu a ke zpracování dat.“* Dále sděluje, že v případě redakčního systému backend umožňuje vytváření a editaci článků, zakládání dalších účtů či dokonce manipulaci se strukturou celého webu. [2]

5.1.4 REST API

Jedná se o způsob komunikace mezi částmi frontend a backend. Webový server je nastaven tak, aby obsluhoval http požadavky klienta a vracel např. data ve formátu JSON. Požadavky směřují na určitou URL adresu a mají http metodu. Na základě těchto dvou informací se požadavek dostane na konkrétní API rozhraní serveru, které ho obslouží.

5.1.5 Návrhový vzor Model View Controller

Návrhový vzor MVC spočívá v rozdělení aplikace na 3 části: z hlediska aplikační logiky, prezentace uživateli a používaných dat. Tyto jednotlivé části představují termíny: model, view a controller. **Model** obsahuje datovou část aplikace. Reprezentuje data v podobě kódu, která jsou uložena v databázi. Hlavním úkolem vrstvy **view** je prezentovat data uživateli. Poskytuje běžnému uživateli prostředí, které umožňuje interagovat s aplikací, a tím ji i ovládat. Poslední části je **controller**. Ten zastává funkci jakéhosi prostředníka, který části model a view propojuje.

5.2 Návrh aktivit

Při navrhování aktivit bylo zapotřebí spolupracovat se speciálními pedagogy. Autor práce si z tohoto důvodu domluvil další schůzku, která se tohoto problému týkala. Speciální pedagogové navrhovali, jakým způsobem by různé aktivity s žáky chtěli provádět. Bylo podrobně rozebráno, jak by tedy vypadal průběh plnění jednotlivých aktivit. Takto se probralo téměř celé spektrum aktivit, a to ze všech předmětů, které by daní pedagogičtí pracovníci chtěli při výuce využít. Autor práce si poté prostudoval postupy aktivit, které učitelé navrhli, aby našel jejich spojitosti i odlišnosti a snažil se je nějakým způsobem kategorizovat a vytvořit z nich typové úlohy. Tento postup sjednocení vzájemně si blízkých aktivit byl velice prospěšný, protože se tak zredukovalo množství typových aktivit z 6-ti na 4 typové aktivity.

Všechny aktivity se prozatím ovládají pouze kliknutím. Funkce tažení není použita z důvodu, aby aktivity mohli využívat i žáci, kteří mají somatické postižení a neudrží dotyk s obrazovkou po celou dobu tažení, zvláště na velké interaktivní tabuli. Po dokončení libovolné aktivity je možno spatřit počet chyb, které byly nasčítány

během celého cvičení. Odměnou a motivací pro děti bude animace po dokončení úkolu.

Níže autor postupně popisuje, které typové aktivity z rozhovoru s pracovníky školy vyplynuly, a jak se zamýšlí jejich využití.

5.2.1 Typová aktivita - Řazení

Typová úloha „Řazení“ vnikla po zvážení, jakým způsobem by se dalo vytvořit cvičení do předmětu matematiky, kategorie číselná řada. Typová úloha obsahuje otázku, ve formě textu, a případně dodatečného obrázku. Dále se v této úloze nachází až 5 možností, které musí být poklikané v určitém pořadí, aby byl úkol úspěšně splněn. Pořadí možností bude možné nastavit. Možnosti u této aktivity jsou tvořeny obrázkem a skrytou hodnotou, která obrázek reprezentuje. Tato typová úloha bude obsahovat režim – Porovnávání velikosti obrázku pro tvorbu aktivit v kategorii – Porovnávání velikosti, kde se nahraje jeden obrázek a ten se použije pro všechny možnosti v různých velikostech. Tato typová úloha se dá použít i ve více předmětech než jen v matematice. Mohou se např. nahrát obrázky dítěte, dospívajícího, dospělého a starší osoby. Otázka může znít: „*Seřaď od nejmladšího/nejstaršího*“.

Další náměty: Jaká jsou vývojová stádia motýla? V jakém pořadí nadcházely události historie? Seřaď noty podle jejich doby. Obecně se dá použít pro všechny úlohy, kde zadání abstraktně zní – V jakém pořadí funguje / se provádí to či ono?

5.2.2 Typová aktivita - Třídění

Typová úloha ‚Třídění‘ přišla na řadu při vymýšlení průběhu cvičení v kategorii Matematické představy – Třídění. Aktivita obsahuje alespoň 2 skupiny ve formě obrázků a také obsahuje až 5 možností, které se do skupin třídí. Vždy se nejdřív označí skupina, do které chcete následně kliknutím přesouvat možnosti. Využít se dá napříč všemi předměty.

Nápady na cvičení: Roztřiď podle barvy, podle tvaru, podle ročního období, ovoce/zelenina..

5.2.3 Typová aktivita – Hledej

Typová aktivita ‚Hledej' je zatím poslední navrženou typovou úlohou. Její princip je naprosto jednoduchý. Nejprve je zadaná otázka ve formě textu a případně doplňujícího obrázku. Možnosti jsou tvořeny obrázky. Správné odpovědi po kliknutí zezelenají, špatné zčervenají. Tím je zajištěna okamžitá zpětná vazba a odpověď.
Náměty na cvičení: Které obrázky začínají na písmeno „x"? - kde za x je možno dosadit libovolné písmeno od A do Z. Najdi, který obrázek se nehodí mezi ostatní.

5.2.4 Typová aktivita – Pexeso

Tato aktivita by se skládala maximálně z dvaceti kartiček, respektive deseti dvojic. Kartičky ve dvojici nemusí mít stejné obrázky. Jako pozadí herní plochy je možné nahrát vlastní obrázek, který je během hraní černobílý. Po dohrání úlohy se obrázek za odměnu zbarví. Tuto aktivitu je zamýšleno použít dvěma způsoby. Buď by se dala hrát v režimu, kdy jednotlivé kartičky jsou již odhalené, a žák pouze přiřazuje správné dvojice. Nebo by šlo hrát klasickým stylem, tedy se zakrytými kartičkami. Po kliknutí na kartičku by se otočila a odhalila se. V režimu se zakrýváním kartiček si učitel smí zvolit vlastní obrázek, který kartičky budou zobrazovat v zakrytém stavu.

5.3 Návrh modelů

Níže v podkapitolách jsou uvedeny návrhy jednotlivých entit, které sloužily jako podklad pro jejich implementaci.

5.3.1 Učitel

Pro model učitele bylo uvažováno použít následující vlastnosti:

- **name** pro křestní jméno,
- **surname** pro příjmení učitele,
- **email** pro emailovou adresu učitele, se kterou se bude přihlašovat,
- **password** pro uchování zašifrovaného hesla z bezpečnostních důvodů,
- **registrationToken** pro uchování jednoznačného řetězce, který bude součástí odkazu pro ověření registrace,

- **isVerified** pro hodnoty: false – neoveřený uživatel nebo true – oveřený uživatel,
- **createdAt** pro datum registrace,
- **activePupilsId** pro ID žáka, který je u daného učitele jako aktivní žák.

5.3.2 Žák

Návrh modelu žáka obsahuje vlastnost **name** pro jméno, **surname** pro příjmení, **username** pro uživatelské jméno, které má každý žák jedinečné, slouží k přihlašování, většina žáků se SVP běžně email nemá, **password** pro zašifrované heslo, **createdAt** pro datum a čas registrace, **degreesOfDisability.lmp, degreesOfDisability.smp** a **degreesOfDisability.tmp** obsahují pravdivostní hodnoty, které určují preferované obtížnosti aktivit daného žáka. Žákovi se v aplikaci zobrazují pouze aktivity, které korespondují s těmito úrovněmi postižení. Poslední vlastnost je **teachersId** – ta slouží pro uchování ID učitele, který profil žáka vytvořil.

5.3.3 Výsledek

Návrh model výsledku obsahuje vlastnost **note** pro vlastní poznámku učitele o průběhu cvičení. Vlastnost **mistakes** uchovává číselnou hodnotu a označuje počet chyb provedených v aktivitě. **createdAt** pro datum a čas zaznamenání výsledku, **activityId** pro ID aktivity, aby bylo možné dohledat, ke které aktivitě se hodnocení vztahuje, **pupilsId** uchovává ID žáka, kterého učitel právě hodnotí. Poslední vlastnost je **teachersId** pro ID učitele, který záznam výsledku vytvořil.

5.3.4 Aktivita

Návrh model aktivity je základní stavební kámen všech typových aktivit. Je to všeobecný model, který obsahuje základní vlastnosti, a to **title** pro název aktivity, **desc** pro popis aktivity, **subject** pro předmět, kterému je aktivita určena, **category** pro případnou kategorii, **typeOfActivity** pro uchování textové informace, o kterou typovou aktivitu se jedná, **typeActivityId** pro ID typové aktivity, **playedCount** pro počet odehrání, **levels.lmp, levels.smp** a **levels.tmp** obsahují pravdivostní hodnoty, které aktivitu zařazují do určitých úrovní postižení. **isValid** pro uchování

informace, zda je uložená aktivita validní, tj. je správně vytvořená a je povoleno její sdílení. **isShared** pro pravdivostní hodnotu, zda se aktivita sdílí, **createdAt** pro datum a čas vytvoření aktivity a poslední **teachersId** pro ID učitele, který aktivitu vytvořil. Modely typových aktivit obsahují již samotná data pro hraná aktivit.

5.3.5 Typová aktivita – Třídění

Návrh typové aktivity – Třídění – obsahuje pouze potřebná herní data. V tomto případě pole **tasks**, které představuje všechny úlohy dané aktivity. Každá úloha obsahuje **text** a **img**, které uchovávají informace o zadání. Dále obsahuje 2 pole:

- **options** – prvky pole jsou reprezentovány objekty, které představují jednotlivé možnosti úlohy. Každá možnost má své **id**, **groupId** – tato hodnota odkazuje na skupinu, do které možnost patří. Vlastnosti **text** a **img** přestavují textovou hodnotu a adresu obrázku samotné možnosti, které jsou viditelné při hraní úlohy.
- **groups** – elementy pole jsou objekty představující jednotlivé skupiny hrané úlohy. Obsahuje **id**, které je potřebné, aby možnosti mohly jednoznačně patřit do určité skupiny, **text** a **img**, které popisují skupiny textem či obrázkem při hraní úlohy.

5.3.6 Typová aktivita – Řazení

Návrh typové aktivity – Řazení – obsahuje, jako všechny typové úlohy, pouze herní data. Pole **tasks** obsahuje všechny úlohy, které jsou ve cvičení zahrnuty. Každá úloha obsahuje zadaní ve formě textu – **text,** či obrázku – **img.** Dále jsou přítomny vlastnosti:

- **sizeComparison** – pravdivostní proměnná, která určuje, zda se jedná o úlohu, ve které jde o porovnávání velikosti obrázku.
- **lowestToHighest** – přestavuje logickou hodnotu, která určuje správné pořadí možností. Hodnota true pro vzestupně, false pro sestupně.
- **options** – pole, které obsahuje možnosti. Každá možnost má **img –** adresu obrázku, případně **text**. Také obsahuje vlastnost **value –** tato hodnota udává číselné pořadí možnosti. Nabývá hodnot od 1 do max. počtu možností úlohy.

5.3.7 Typová aktivita – Hledej

Návrh typové aktivity – Hledej – obsahuje pole **tasks**, kde jsou informace potřebné ke všem jednotlivým úlohám aktivity. Obsahuje vlastnosti **text** a **img**, které umožňují uchovávat zadání. Dále obsahuje možnosti jednotlivých aktivit ve formě objektů v poli **options**. Každá možnost má své jednoznačné **id**, vlastnost **isCorrect**, která nabývá hodnoty true, pokud je možnost správná. Pokud je nesprávná, uchovává hodnotu false. Vlastnosti **text** a **img** jsou určeny pro uložení reprezentace možnosti. Uchovávají tedy obrázek, případně text možnosti.

5.3.8 Typová aktivita – Pexeso

Návrh typové aktivity – Pexeso – je zatím v rozpracované fázi vývoje. U každé úlohy je ale zamýšleno uchovávat následující informace:

- **hiddenStyle** – pravdivostní proměnná, která uchovává informaci o tom, zda budou kartičky zakryté (klasické pexeso) nebo zda budou otočené a tudíž čitelné.
- **itemsCover** – proměnná uchovávající textový řetězec. Hodnotou by měla být URL adresa obrázku, který bude použit jako zadní obrázek kartiček. Pokud bude hodnotou prázdný řetězec, použije se výchozí obrázek.
- **backgroundImg** – proměnná, která uchovává textový řetězec. Uchovává URL adresu obrázku, který má být vložen na pozadí hrací desky. Postupným hraním pexesa se obrázek odkrývá a po nalezení všech dvojic pexesa se za odměnu zbarví. Barevnost obrázku se řešila použitím technologie CSS.
- **items** - pole, které obsahuje objekty, které reprezentují jednotlivé kartičky. Každý objekt obsahuje atributy: **img –** kde je uložena URL adresa obrázku, který je na kartičce vykreslen, **value -** jedinečná identifikace kartičky, jedná se o číselnou hodnotu, **match –** obsahuje číselnou hodnotu, která odkazuje na hodnotu **value** některé jiné kartičky. Tím je zajištěna provázanost mezi kartičkami. Každá kartička tedy ví, se kterou kartičkou tvoří dvojici. Dále se uchovává informace **show**, která nabývá hodnoty true, právě tehdy, když je kartička odhalená/viditelná. Poslední vlastností uchovávanou u objektu kartičky je **isGone**. Ta určuje, zda byla již kartička správně spárována s jinou.

6 Implementace

Pro realizaci aplikace byly použity tyto zásadní technologie: Node.js, Express.js, MongoDB, React. Společně tyto technologie tvoří koncept MERN.

6.1 Použité technologie

V následujících kapitolách jsou charakterizovány použité technologie pro vývoj webové aplikace.

6.1.1 HTML, CSS

Termín HTML přestavuje Hyper Text Markup Language. Jedná se tedy o značkovací jazyk, který je jedním ze zásadních jazyků umožňující vytváření webových stránek. CSS znamená Cascading Style Sheets. Tento jazyk slouží pro vzhledovou stylizaci elementů HTML dokumentu.

6.1.2 React

Na oficiálních stránkách React [13] je uvedeno, že tento termín představuje javascriptovou knihovnu pro tvorbu grafických rozhraní. Tato knihovna je vyvíjena společností Facebook a její komunitou.

Máca [3] uvádí, že na rozdíl od jiných kompletních frameworků (např. Angular) se zaměřuje pouze na jednu specifickou oblast, a pokud je na ni nahlíženo z hlediska MVC architektury, tvoří právě část view, tedy vrstvu pohledu, která má za úkol prezentovat data uživateli.

Podle Ming Soon, Jon Chan a kol. [4] se aplikace React skládá z několika komponent, z nichž každá je zodpovědná za výstup malé opakovaně použitelné části HTML kódu. Jednotlivé komponenty mohou být vnořovány do jiných komponent, což umožňuje vytvářet složité aplikace, které jsou vytvořeny z dílčích stavebních bloků. Komponenty mohou také udržovat vnitřní stav.

6.1.3 Node.js

O této technologii Delbono Emanuele [5] uvádí, že Node.js je platforma postavená na JS běhovém prostředí prohlížeče Google Chrome pro jednoduché vytváření rychlých a škálovatelných síťových aplikací. Node.js také využívá neblokujícího I/O modelu,

založeného na událostech, díky kterému je velmi efektivní. Node.js je ideální pro datově náročné aplikace, které běží v reálném čase napříč distribuovanými zařízeními.

6.1.4 Express.js

Na stránkách vývojářů [14] je uvedeno, že Express.js je minimální a flexibilní framework pro vývoj webových aplikací, vytvářených v Node.js, který poskytuje obrovskou škálu funkcí pro webové a mobilní aplikace.

6.1.5 MongoDB

MongoDB je multiplatformní dokumentová databáze pracující s dokumenty namísto tabulek, které jsou běžné u relačních databází. Dokumenty ukládá ve formátu BSON, který je svou syntaxí velmi podobný formátu JSON.

6.1.6 Mongoose

Mongoose je knihovna pro Node.js a MongoDB. Velice usnadňuje práci s MongoDB. Dokáže spravovat vztahy mezi daty, umožňuje vytvářet schémata datových objektů, validuje schémata a dokáže transformovat objekty napsané v Javascriptu na jejich reprezentaci v MongoDB.

6.2 Struktura projektu

Při zakládání projektu pro backendovou část aplikace se v kořenové složce projektu inicioval nový projekt příkazem **npm init**. Pro vytvoření frontendové části se použil příkaz **create-react-app**.

Na backendu se nachází soubor app.js, který se spouští pro běh serveru. Dále obsahuje složky **controllers –** obslužné funkce příchozích http požadavků, **models –** kódová reprezentace dokumentů uložených v databázi, **validators –** zde jsou soubory s kódem pro validaci vstupů na straně serveru, **routes –** obsahuje soubory s kódy, které definují rozhraní serveru, respektive jednotlivá API a typy požadavků, které server obslouží, **uploads** pro nahrané obrázky a složka **middlewares** obsahuje dílčí obslužné funkce, kterými může požadavek putovat. V tomto případě se zde nachází kód pro autentizaci a autorizaci požadavků.

Frontend obsahuje hlavní složku **src**, ve které se nachází složky:

- **contexts –** v ní je zamýšleno mít soubory, obsahující datový objekt, který může být dostupný pro všechny komponenty, které uzavírá.
- **hooks –** v této složce jsou soubory, které definují „vlastní háčky“. Mohou to být například části kódu, které by byly použity na více místech v projektu, a díky přístupu s háčky se zamezí duplicitě kódu. Autor práce je využil také k rozdělení datové části od komponent.
- **images –** obsahuje obrázky, které jsou použity v aplikaci na pevně určených místech a není potřeba hledat jejich cestu v databázi. Nejedná se o obrázky např. použitých v aktivitách, které by byly dynamicky vykreslovány.
- **shared –** obsahuje složky **pages** a **components,** v nichž se nahází stránky a komponenty, které jsou dostupné všem uživatelům. Komponentami zde jsou například horní lišta a její dílčí části, seznam předmětů, UI elementy.
- **teacher –** v této složce jsou stránky a komponenty, ke kterým má přístup pouze učitel.
- **utils –** obsahuje soubory s pomocnými funkcemi například pro validaci vstupů nebo pro práci s daty aktivit.

6.3 Aplikační vrstva

V následujících podkapitolách jsou podrobně probrány dílčí části webové aplikace, konkrétně jejich implementace, jak na straně klienta, tak na straně serveru. Také se zde vyskytují zkrácené URL adresy. Z důvodu častého využívání těchto adres v této práci, jsou adresy frontendu zkráceny do tvaru „F:/prihlaseni“, který ve skutečnosti reprezentuje adresu http://localhost:3000/prihlaseni. Pro adresy na backendu je obdobný přístup. Adresa „B:/api/auth/register-teacher“ tedy reprezentuje skutečnou adresu http://localhost:5000/api/auth/register-teacher.

6.3.1 Zasílání požadavků

Protože téměř ve všech podkapitolách se vyskytuje zasílání požadavků z klienta (webového prohlížeče) na server, je potřeba nejprve zmínit, jakým stylem se zasílání

provádí. Autor práce využil funkce Fetch API, kterou disponují všechny moderní webové prohlížeče s aktivní podporou.

Při rozmýšlení, jak konkrétně provést implementaci, bylo bráno v potaz to, že požadavků bude velké množství. Tudíž byl pro zasílání požadavků vytvořen vlastní háček tzv. *custom hook*, které obecně React od verze 16.8. podporuje. Autor práce pojmul zasílání požadavků modernějších způsobem, kde je graficky vizualizováno odeslání a čekání na odpověď ve formě drobné animace. Uživatel díky tomu pozná, že se v aplikaci cosi načítá. V případě neúspěchu je schopný vrátit zprávu o této události.

Samotný háček se nachází v souboru „/hooks/**http-hook.js**“. Tento háček je schopný vracet objekt s těmito vlastnostmi:

- **sendRequest** – tato funkce přijímá 4 parametry, jako první je zapotřebí definovat URL adresu, kam má požadavek směřovat. Dále je možnost zadat typ požadavku (GET, POST, PATCH, DELETE), umožňuje vkládat tělo požadavku a nastavit hlavičku.
- **isLoading** – V této proměnné se uchovává pravdivostní hodnota. Nabývá hodnoty *true,* jestliže byl poslán požadavek, a zároveň ještě nedorazila žádná odpověď, a to jak o úspěchu, či neúspěchu.
- **error** – Tato proměnná slouží pro uchování zprávy o neúspěšném pokusu o provedení akce na serveru, jinými slovy, jakmile požadavek nebyl úspěšně dokončen. Tato zpráva je zobrazována uživateli ve vyskakovacím okně.
- **clearError** – je pomocnou funkcí, která vynuluje proměnou **error**. Volá se v případě pokusu o zavření vyskakovacího okna s chybovou hláškou.

6.3.2 Hlavní stránka (nepřihlášený uživatel)

Hlavní stránka neboli index, se nachází na adrese „F:/“. Přístup je k ní umožněn pouze v nepřihlášeném stavu. Nachází se na ní základní informace o webové aplikaci. Obsahuje 3 tlačítka. Tlačítko „Vstoupit“, které uživatele zavede přímo do aplikace bez nutnosti vlastnit účet. Tlačítko „Přihlásit se“, které přesměruje uživatele na adresu „F:/*prihlaseni“,* kde se po vyplnění formuláře může přihlásit, je-li registrovaný.

Poslední tlačítko je „Registrace učitele“, které přesměrovává na adresu „F:/*registrace*“, kde se dají vytvářet účty učitelů.

6.3.3 Registrace učitele

Jak již bylo zmíněno výše, formulář pro registraci učitele se nachází na adrese „F:/*registrace*“. Při přechodu na tuto adresu se vykreslí komponenta src\shared\pages\Auth\Signin**Signin.js.** Ve formuláři Jsou vyžadovány následující informace: Jméno, příjmení, email, heslo, heslo pro ověření. Všechny uživatelem zadané vstupy se validují jak na straně klienta, tak na straně serveru. Více informací o validacích na frontendu je uvedeno v samostatné kapitole, jelikož jsou použity nejen ve formuláři pro registraci učitele. Aby bylo možné formulář odeslat, je potřeba mít zadané vstupy validní. Po odeslání požadavku pro registraci na REST API server na adresu „B:/api/auth/register-teacher“ se tedy provede validace vstupů na serveru scriptem „validators**registerTeacher.js**“, kde se kontroluje, zda je zadané jméno, příjmení, zda heslo obsahuje minimálně 5 znaků a zda se obě zadaná hesla shodují. Dále se testuje, zda zadaný email není již zaregistrovaný. V tom případě, by se proces registrace zrušil a byla by zaslána frontendu hláška o tomto stavu. Pokud je emailová adresa volná, zašifruje se uživatelem zadané heslo do nečitelného řetězce znaků, aby v databázi bylo heslo nečitelné z bezpečnostních důvodů.

6.3.4 Registrace žáka

Registrace žáka je dostupná pouze pro přihlášené učitele. Každý třídní učitel registruje pouze žáky své třídy. V horní liště se nachází ikona žáka se seznamem. Po kliknutí na tuto ikonku budete přesměrováni na adresu „F:/*seznamzaku*“, kde každý učitel vidí své registrované žáky, ale tomu se autor věnuje v práci později. Důležité je, že se pod tímto seznamem nachází tlačítko **Přidat nového žáka**, které přinutí prohlížeč přejít na adresu „F:/*registracezaka*“. Na této adrese se nachází formulář, který je velmi podobný tomu pro registraci učitele. Namísto emailu se vyžaduje uživatelské jméno a je potřeba specifikovat, které postižení žák má. Tyto zvolené možnosti mají vliv na zobrazované aktivity danému žákovi. Odeslání formuláře vyvolá zaslání požadavku na adresu serveru „B:/api/teacher/register-pupil“. Na

serveru se nejprve zjistí, zda požadavek pochází od uživatele, který je přihlášený a zastává roli učitele. Více o této problematice je zmíněno v kapitole Autorizace a autentifikace. Následně server provede validaci zadaných vstupů z formuláře. Poté požadavek doputuje do finální části pro zpracování požadavku. Zde se pomocí zadaného uživatelského jména v databázi zjistí, zda se uživatelské jméno již nevyskytuje u jiného žáka. Pokud je volné, zašifruje se heslo žáka a následně se celý jeho objekt, který ho reprezentuje, uloží do databáze zavoláním jeho metody **save**.

6.3.5 Přihlášení uživatele

Stránka pro přihlašování se nachází na adrese „F:/prihlaseni“, na ní se nachází formulář se dvěma vstupními poli. Je potřeba zadat email, pokud jste učitel, nebo uživatelské jméno, přihlašuje-li se žák. V poslední řadě je vyžadováno heslo. Po odeslání formuláře na adresu serveru s cestou „B:/api/auth/login“ server vstupy uživatele zvaliduje. Pomocí prvního vstupu se pokusí najít účet učitele, pokud nenalezne, pokusí se najít žáka se zadaným uživatelským jménem. Pokud je uživatel nalezen, provede se porovnání hesel. Nejprve se rozšifruje heslo, které má uživatel v databázi přiřazené, a to se porovná s heslem z formuláře. Shodují-li se, vygeneruje se uživateli token, který je blíže popsán v následující kapitole. Ten se spolu s daty uživatele pošle zpět klientovi ve formátu JSON.

6.3.6 Autentifikace a autorizace

Autentifikaci a autorizaci autor práce řeší s pomocí modulu **jsonwebtoken.** Autentifikace je důležitá pro zjištění a uchování informace, zda je uživatel přihlášený. Pokud má ale přihlášený uživatel právo např. manipulovat s určitými daty v databázi, potřebuje se autorizovat.

Aby se mohl uživatel prokázat, když posílá požadavek na server například o smazání výsledku žáka, potřebuje s tím samým požadavkem v hlavičce zaslat i token. Token je textový řetězec, který obsahuje zašifrované informace, tudíž pro ostatní nečitelné. Na serveru lze do tokenu vložit a zašifrovat jakékoliv informace. V popsané aplikaci dává smysl zašifrovat id uživatele, email/uživatelské jméno, a informaci, zda se jedná o učitele. K zašifrování je potřeba zadat libovolný textový řetězec, s pomocí

něhož se informace zašifrují, a následně pouze s tímto stejným řetězcem lze informace uložené v tokenu rozšifrovat zpět do čitelné podoby.
Tento token uživatel získá při přihlášení do aplikace a je následně uložen v lokálním úložišti klientova prohlížeče. Použitý přístup nemusí být zcela bezpečný, avšak tokenu lze nastavit jeho expiraci, a tím se míra nebezpečí částečně redukuje. Do lokálního úložiště se ukládá objekt, který obsahuje následující vlastnosti:

- name, surname – jméno a příjmení pro zobrazení v horní liště
- username, userId – id uživatele pro autorizaci
- isTeacher, isPupil – pravdivostní hodnoty, pomocí kterých jsou vykreslovány chtěné komponenty
- isLoggedIn – pravdivostní hodnota určující stav přihlášen/nepřihlášen
- activePupilsId – tato proměnná uchovává id žáka, kterého má učitel nastaveného jako aktivní – tudíž případné výsledky, po dohrání aktivity, se ukládají k žákovi s tímto ID
- token – textový řetězec obsahující zašifrované informace o přihlášeném uživateli.

Odhlášení je řešeno pouhým smazáním obsahu v lokálním úložišti. Tedy smazáním výše zmíněného objektu.
Při příchodu požadavku na server, tak pokud je potřeba, provádí se autentizace pomocí souboru B:\middlewares**isAuth.js**. Zde se získá token z hlavičky požadavku. Pokud existuje, uživatel je autentizován a jsou z tokenu rozšifrována potřebná data. Neexistuje-li, autentifikace neuspěla.
Některé API, které jsou přístupné jen učitelům, používají soubor B:\middlewares**isTeacher.js**. Po úspěšné autentifikaci, tzn. po dokončení rozšifrování informací z tokenu, se ověří, zda se mezi informace nachází vlastnost **isTeacher**, která nabývá hodnoty true.

6.3.7 Validace vstupních dat

Validace vstupních dat je důležitou součástí každé webové aplikace, do které je uživateli umožněno pracovat s daty.

Validace na klientovi

Tento přístup uživateli přináší příjemný zážitek z používání aplikace, např. vyplňování formulářů. V autorově aplikaci je validace formulářů řízena souborem „F:\src\hooks**form-hook.js“** s použitím pomocných funkcí nacházejících se zde „F:src\utils\validators.js“. U každého vstupu si autor práce eviduje:

- isValid - informace, zda je aktuální hodnota vstupu validní
- isTouched – Uchovává informaci, že se uživatel pokusil vstup vyplnit. Je potřebná z důvodu, aby se uživateli neukazovaly chybové hlášky, např. „email je zadaný v nesprávném tvaru“, když se ještě uživatel ani nepokusil o jeho vyplnění.
- value – obsahuje samotnou hodnotu vstupu

Nakonec existuje ještě poslední proměnná – formIsValid, ta je společná pro všechny vstupy a nabývá hodnoty true, právě tehdy, když proměnné isValid u všech vstupů obsahují true. Pokud je proměnná formIsValid true, formulář je možné odeslat.

Validace na serveru

Použití validace pouze na straně klienta může být velmi nebezpečné. Požadavky nemusí přicházet pouze z frontendové části aplikace, ale mohou být zasílány např. pomocí softwarového nástroje Postman. Tudíž je nezbytné, aby server obsahoval svůj validační systém.

Validace pro registraci učitele se nachází v souboru „B:\validators**registerTeacher.js**“. K její implementaci je využíván modul **express-validator**. Na jednotlivých vstupech z formuláře se může validovat různými dostupnými metodami z modulu, nebo je možné si vytvořit vlastní validaci. Pro jméno a příjmení je vyžadováno, aby obsahovali alespoň 1 znak. U emailu byla použita metoda **isEmail** pocházející z modulu. Pokud je vstupem doopravdy emailová adresa, server se pokusí v databázi vyhledat zadanou adresu, aby zamezil

dvojité registraci jedné adresy. Jestli se některý z těchto vstupů vyhodnotí jako nevalidní, vrátí se frontendu chybová hláška specifikující problém.

6.3.8 Rozložení obsahu

V následujících podkapitolách jsou popsány základní komponenty aplikace, které jsou vykreslovány přihlášeným uživatelům. Pouze v případě hraní aktivit nejsou vykreslované. Jedná se o hlavní lištu, prostor pro obsah konkrétní stránky a patičku.

Horní lišta

Šablona aplikace jako první obsahuje horní lištu, kde se úplně vlevo nachází ikonka sovičky, která po kliknutí odkazuje na adresu „F:/“. V liště následuje ikonka lupy, která přesměruje uživatele na adresu „F:/hledat“. Dále je zde jméno a příjmení uživatele, které odkazuje na profil uživatele, buď „F:/ucitel/:ucitelID“ nebo „F:/zak/:zakID“. Je-li v aplikaci host, je místo jména napsáno „Vítaný Návštěvník“. V případě hosta jméno nikam neodkazuje. Učiteli jsou zobrazeny ještě 2 možnosti. Zaprvé, je přítomna ikonka, která odkazuje na seznam žáků – „F:/seznamzaku“ a je mu zpřístupněna komponenta pro změnu aktivního žáka. Tato horní lišta se nachází na všech stránkách, kromě hraní aktivit. Pod horní lištou se nachází prostor pro samotný obsah konkrétní stránky.

Obr. 1 Rozložení obsahu

6.3.9 Hlavní stránka (přihlášený uživatel)

Hlavní stránka se nachází na adrese „F:/“. Vlevo na obrazovce se nachází navigace ve formě výpisu všech předmětů. Uprostřed obrazovky se zobrazují nejnověji přidané aktivity. Komponenta, která se stará o vykreslování nových aktivit, se nachází v adresáři právě zde „src\shared\pages\Home\NewActivities\ **NewActivities.js**“ a zasílá požadavek na API „B:/api/get-new-activities“. Ten je zachycen na serveru s příslušným rozhraním. Na backendu se v try bloku naleznou nejnovější aktivity. Limit získaných aktivit z databáze je nastaven na hodnotu 6. Po úspěšném načtení vrací server odpověď ve formátu JSON.

6.3.10 Stránka předmětu

V této kapitole je popsán způsob implementace vykreslování stránky src\shared\pages\SubjectPage**SubjectPage.js.**

Adresa pro vykreslení sekce předmětu může obsahovat následující informace F:/“název předmětu“/“kategorie“. Pokud se nenachází údaj o kategorii, ale pouze o předmětu, jsou na obrazovku vykresleny kategorie, dostupné pro daný předmět. Tento případ nastává tehdy, kdy uživatel klikne vlevo v navigaci na název předmětu. Bude přesměrován na adresu F:/“název předmětu“. Uvidí tedy kategorie předmětu, kterého si zvolil. Po kliknutí na kategorii bude klient přesměrován na F:/“název předmětu“/“kategorie“. Kategorie, dostupné pro předměty, se nachází v souboru „src\utils**activity-ulils.js**“. Jakmile je známa kategorie, může začít vyhledávání aktivit. Vyhledávání aktivit je pro žáky řešeno jiným způsobem než pro všechny ostatní přihlášené uživatele. Všichni uživatelé mají možnost si zvolit, zda chtějí zobrazovat systémové aktivity (vytvořené autorem aplikace) nebo aktivity vytvořené speciálními pedagogy. Pokud je uživatelem učitel či návštěvník, má zobrazené možnosti úrovně postižení, které může měnit. Je-li uživatelem žák, tyto filtry pro úrovně postižení mu nejsou zobrazeny, aby mu nepřipomínaly jeho nedostatky a nedemotivovaly jej. Učitel nastavuje žákovi úrovně postižení při registraci žáka. Autor práce plánuje možnost změnit toto nastavení u žáka, ale v praxi se téměř vůbec nestává, že by se žák výrazně mentálně posunul.

Pokud je uživatelem žák, pro vyhledávání se používá API serveru „B:/api/get-activities-of-subject-for-pupil?“, kde za otazníkem jsou poslány následující dodatečná data:

- pupilsId – id žáka, které se použije pro vyhledání jeho mentálních úrovní pro použití ve filtru
- subject – uchovává název předmětu
- category – obsahuje kategorii hledané aktivity
- bySovicka – uchovává pravdivostní hodnotu. Pro aktivity vytvořené sovičkou nabývá hodnoty true, pro aktivity speciálních pedagogů false

Pokud se jakákoliv z výše zmíněných hodnot změní, je automaticky zaslán nový požadavek pro získání aktuálních aktivit.
Ostatním uživatelům je dostupné API rozhraní „B:/api/get-activities-of-subject?“, kde za otazníkem se nacházejí stejné informace, jako pro předchozí API, pouze je zde vynechán pupilsId, a zároveň jsou dostupné tyto 3 proměnné: lmp, smp, tmp. Všechny tři proměnné uchovávají logickou hodnotu a představují úrovně mentálního postižení.
Samotný filtr je definovaný objektem, který má požadované vlastnosti. K filtru se přidá poslední vlastnost **isShared**, která musí nabývat hodnoty true. Filtr je připraven a nyní se v try a catch syntaxi může server pokusit načíst data z databáze. Při úspěšném nalezení se data zasílají ve formátu JSON ke klientovi. Při neúspěchu je zaslána chybová hláška s odpovídajícím chybovým kódem.

6.3.11 Hraní aktivity

Hraní aktivit probíhá v případě, že se prohlížeč nachází na adrese „F:/cviceni/:activityId“. Následně se vykreslí komponenta src\shared\pages\ ActivityPlayer**ActivityPlayer.js**. Tento soubor po vykreslení zašle požadavek na adresu „B:/api/get-activity-type/:activityId“, aby zjistil, o kterou typovou aktivitu se jedná. Server tedy nalezne podle zaslaného ID aktivity v databázi konkrétní aktivitu a zašle frontendu JSON s vlastnostmi:

- **activityType** – aby bylo jednoznačné, ve které kolekci se má samotná typová aktivita hledat
- **activityTypeId** – když už je kolekce známa, je zapotřebí znát id hledané typové aktivity.

Nyní je na řadě frontend, aby z přijatých dat ze serveru, na základě hodnoty vlastnosti **activityType** vykreslil odpovídající komponentu, určenou pro danou typovou aktivitu a předal jí **activityTypeId**.
Následující kapitoly jsou o implementaci jednotlivých typových aktivit.

Načtení Aktivity – Třídění

Po vykreslení komponenty **GroupSorting.js**, se zašle požadavek na „B:/api/get-activity-data-group-sorting?activityTypeId“. Server, pomocí zaslaného activityTypeId, v databázi najde data aktivity. Po načtení těchto dat, je potřeba některá data upravit a přidat i některé nové informace. Data, která jsou uložená v databázi, pro správný průběh hraní nestačí. Je potřeba provést následující změny. Každé možnosti je přidána vlastnost **isMoved**, které je přiřazena hodnota false. Tato hodnota představuje, zda byla možnost již správně rozřazena do skupiny. Frontendu se zasílá JSON, který obsahuje tyto vlastnosti:

- **isNextAllowed** – hodnota, která uchovává informaci o tom, zda je dokončena zrovna hraná úloha a je tudíž povolena spustit další úkol. Je zaslána s hodnotou false. Tlačítko „Další“, které spustí další úlohu, je dovoleno stisknout právě tehdy, když tato proměnná nabývá hodnoty true.
- **currentTask** – obsahuje číselnou hodnotu, která představuje, v kolikátém úkolu se uživatel nyní nachází. Zasílá se s hodnotou 1.
- **lastTask** – obsahuje konstantu. Představuje pouze počet úkolů. Spolu s proměnnou **currentTask** je zobrazovaná uživateli, aby bylo vidět, v kolikáté úloze z kolika se nachází.
- **isDone** – hodnota, která nabývá hodnoty true, když je celá aktivita dokončená. Jinými slovy, že byl dokončen poslední úkol aktivity. Dává na vědomí vyskakovacímu oknu, aby se vykreslilo. V dalších verzích aplikace se

zde bude vyskytovat i animace. Je zaslána s hodnotou false. Změna hodnoty na true nastává v případě, že se rovnají hodnoty **currentTask** a **lastTask**, a zároveň proměnná **isNextAllow** je true.

- **mistakes** – proměnná uchovávající počet chyb. Výchozí hodnota nastavena na 0.
- **currentMistakeId –** obsahuje id možnosti, které mělo být špatně zařazeno. Pomáha k zobrazování červeného stínu kolem možnosti. Výchozí hodnota je null.
- **selectedGroupId** – obsahuje id skupiny, která je nyní vybraná. Jinými slovy, do které skupiny, se možnost, po jejím kliknutí, zařadí. Výchozí hodnota je 1.
- **tasks –** samotná herní data s upravenými možnostmi.

Načtení aktivity – Řazení

Načtení aktivity řazení probíhá podobným způsobem jako načtení aktivita předchozí. Komponenta **ImgSorting.js** zašle http požadavek na „B:/api/get-activity-data-img-sorting?activityTypeId“, kde server pomocí zaslaného parametru najde konkrétní typovou aktivitu. Opět se před zasláním dat frontendu provádí úpravy. Jak již bylo zmíněno v modelu, každá úloha typové aktivity – řazení, má vlastnost **sizeComparison**. Pokud je tato vlastnost true, server vezme první možnost úlohy a pomocí for cyklu vytvoří zbývající 4 možnosti. V případě hodnoty false se očekává, že již všechny možnosti jsou vytvořené a existují v databázi. Každá možnost obsahuje obrázek **img**, vlastnost **isMoved**, která uchovává informaci o již správném zařazení, je vytvořená pro všechny možnosti a nastavena na hodnotu false. Kazdá možnost obsahuje **value**, které nabývá hodnot od 1 až do 5. Na základě vlastnosti **value** je určena hodnota možnosti, a zároveň se na frontendu mění velikost obrázků v případě, zda se jedná o porovnávání velikostí obrázků. U úloh této aktivity se nachází ještě jedna dodatečná vlastnost. Úlohám je přidána vlastnost **neededValue**, která uchovává celočíselnou hodnotu, která určuje, jakou následující možnost je nutné zvolit pro bezchybné dohrání. Data úkolů jsou upravena, následuje úprava dat celé aktivity. Vlastnosti **isDone**, **isNextAllowed**, **currentTask**, **lastTask**, **mistakes**, **currentMistakeId**, **tasks** jsou zde přítomny, ale popsané jsou již

v předchozí kapitole. Data jsou připravena k odeslání frontendu. Poté, co frontend data přijme, vykreslí aktivitu pomocí kódu zobrazeného níže na obrázku.

```
return (
    <div className="img-sorting__activity-wrapper">
        {gameState && (
            <Fragment>
                <div className="img-sorting__task">
                    <h1>{gameState.tasks[gameState.currentTask].text}</h1>
                </div>
                <ImgSortingOptions
                    clickHandler={clickHandler}
                    options={gameState.tasks[gameState.currentTask].options}
                    neededValue={gameState.tasks[gameState.currentTask].neededValue}
                    sizeComparison={gameState.tasks[gameState.currentTask].sizeComparison}
                    currentMistakeId={gameState.currentMistakeId}
                />
                <ImgSortingResults
                    lowestToHighest={gameState.tasks[gameState.currentTask].lowestToHighest}
                    options={gameState.tasks[gameState.currentTask].options}
                    neededValue={gameState.tasks[gameState.currentTask].neededValue}
                    sizeComparison={gameState.tasks[gameState.currentTask].sizeComparison}
                />
                <div
                    onClick={nextTaskHandler}
                    className={`img-sorting__controls ${
                        gameState.isNextAllowed ? 'img-sorting__controls-next-allowed' : ''
                    }`}
                >
                    Další
                </div>
                {gameState.isDone && <FinishedModal mistakes={gameState.mistakes} activityId={activityId} />}
            </Fragment>
        )}
    </div>
);
```

Obr. 2 Implementace vykreslování aktivity

Po vykreslení a již následném zařazení tří možností, vypadá obraz aktivity následovně

Obr. 3 Průběh typové aktivity - Řazení

Načtení aktivity – Hledej

Komponenta, která obstarává načítání a průběh hraní je **ClickCorrectAnswers.js**. Po jejím vykreslení je zaslán požadavek na adresu „B:/api/get-activity-data-clickCA?activityTypeId“. Díky ID typové aktivity, které je součástí adresy, se v databázi nalezne konkrétní aktivita s jejími herními daty. Před odesláním herních dat frontendu jsou tato data upravena. Zaprvé je všem možnostem přidán atribut **isTouched** a nastaven na hodnotu false. Tento atribut je použit pro zjištění, zda se uživatel pokusil označit možnost jako správnou. Poslední úpravou je pouhá změna pořadí prvků v poli **tasks**, aby při každém spuštění aktivity nebyla pokaždé totožná a žák si již nezafixoval pozice správných možností. Společně s herními daty jsou opět zaslány základní vlastnosti: **isNextAllowed**, **isDone, lastTask**, **mistakes**, **tasks, currentMistakeId, currentTask.** Jejich význam je popsán v kapitole Načtení aktivity - Třídění. Data jsou nyní připravena k odeslání frontendu.

Ukládání výsledků žáka

Pokud je aktivita dohrána žákem na účtu učitele, vykreslí se okno **FinishedModal.js**, do kterého je možné napsat poznámku. Toto okno obsahuje tlačítko uložit. Po jeho stisknutí je zaslán http požadavek na adresu

„B:/api/teacher/save-result/:pupilsId“ společně s daty **note** a **mistakes**. Server ověří, zda požadavek zaslal učitel. Následně se prověří, zda žák, kterému se má výsledek uložit, patří učiteli, který se o uložení výsledku pokouší. Nakonec se výsledek uloží do databáze. V případě autorizační či jakékoliv jiné chyby je zaslána frontendu chybová hláška.

6.3.12 Výběr aktivního žáka

Učiteli je dostupná komponenta **ActivePupil.js**, ve které si může zvolit, jestli bude spouštět aktivity jen sobě, tzn. že na konci cvičení nebude možné výsledek uložit. Nebo si vybere jednoho ze svých žáků a po dokončení aktivity je ukládání výsledků umožněno. Tato komponenta potřebuje ze serveru získat seznam žáků, na které se učitel může přepnout. Z tohoto důvodu zasílá http požadavek na adresu „B:/api/teacher/list-of-pupils/:teachersId“. V hlavičce je zaslán také autorizační token, který slouží pro ověření, zda se zadané ID učitele z adresy shoduje s ID uživatele v tokenu. Pokud ano, je povoleno v databázi najít seznam žáků a následně data vrátit frontendu.

Po přesměrování kurzoru na výše zmíněnou komponentu se vykreslí seznam žáků. Každá možnost v seznamu obsahuje jméno žáka. Po kliknutí na kterékoliv jméno žáka se zasílá požadavek na adresu „B:/api/teacher/set-active-pupils-id/:pupilsId“. Zkontroluje se, zda žáka vytvořil učitel, který si jej chce zvolit jako aktivní. Následně se upraví data učitele a do atributu **activePupilsId** se vloží ID zaslané v adrese požadavku.

Pokud je zrovna nějaký žák zvolen jako aktivním, je v seznamu vykreslena možnost „UČITEL“, která po klinutí nastaví atribut **activePupilsId** na hodnotu null. Tato změna je realizovaná požadavkem na „B:/api/teacher/set-active-pupils-id-off“. Pomocí zaslaného tokenu se zjistí, kterému učiteli se má atribut upravit.

6.3.13 Tvorba aktivity

Každý učitel může začít tvořit své aktivity na adrese „F:/tvorba-aktivity“, kde si zvolí typovou aktivitu a bude přesměrován na „F:/tvorba-aktivity/*název typové aktivity*“. V této verzi aplikace k tvorbě dostupné typové aktivity řazení a třídění.

Všechny stránky pro tvorbu aktivit obsahují komponentu **BaseActivityOptions.js**, kde se nacházejí vstupní prvky pro změnu názvu, popisu, předmětu, kategorie a možnost určit stupně mentálního postižení. Jednotlivé úkoly se tvoří pro každou typovou aktivitu jiným způsobem.

6.3.14 Přiřazení obrázků možnostem

Obrázky lze možnostem přiřadit dvěma způsoby. Buď je obrázek převzat odjinud z internetu a pomocí vložení URL adresy obrázku přiřazen. Nebo lze na server nahrát vlastní obrázek a cesta k tomuto nově nahranému obrázku bude vytvořena a dosezena automaticky. K nahrávání obrázku byl použit modul **multer**, který na straně serveru dokáže pracovat s přijatými soubory.

Tvorba úkolů - Řazení

Na adrese „F:/tvorba-aktivity/razeni“ se nachází formulář pro tvorbu aktivity - Řazení. Pod komponentou pro základní informace o aktivitě se nachází komponenta **ImgSortingCreatingTool.js**, která umožňuje tvorbu jednotlivých úkolů aktivity. Počet úkolů není nijak omezen. U každého úkolu lze upravit zadání.

Tvorba úkolů – Třídění

Tvorba úloh aktivity třídění je dostupná díky komponentě **GroupSortingCreatingTools.js**. Úkolům lze měnit zadání, přidávat možnosti a skupiny.

Tvorba úkolů – Hledej

Úlohy aktivity hledej se dají tvořit pomocí formuláře, který se nachází v komponentě **ClickCorrectAnswersCreatingTool.js.** Úkolům lze měnit zadání a přidávat možnosti. Každé možnosti lze nastavit obrázek a pravdivostní hodnotu **isCorrect** pro identifikaci správných možností.

6.3.15 Aktivity učitele a jejich správa

Aktivity učitele reprezentuje stránka „F:/aktivity-ucitele/:teachersId“. Komponenta **TeachersActivities.js** zasílá požadavek na “B:/api/get-activities-of-

teacher/:teachersId“, kde se vrátí všechna cvičení, které učitel vytvořil. Učitel vidí všechny své aktivity, ať jsou sdílené či nikoliv. Zobrazují se mu tlačítka upravit a odstranit. Tlačítka a všechny nesdílené aktivity nejsou ostatním uživatelům zobrazeny.

Pokud se učitel rozhodne smazat některou ze svých aktivit, může tak učinit stisknutím tlačítka s ikonou koše. Tlačítko je obslouženo metodou, která zasílá požadavek na adresu „B:/api/teacher/delete-activity/:activityId“. Server se podle zaslaného ID aktivity a načteného ID učitele z tokenu pokusí smazat aktivitu, která se shoduje se zaslanými údaji. Smazání je prováděno metodou **deleteOne**, která je dostupná na všech mongoose modelech.

Úprava aktivity je možná stiskem tlačítka „upravit“. Klient bude přesměrován na adresu „F:/tvorba-aktivity/*typ-aktivity*/:idAktivity“. Tato adresa je stejná jako pro vytvoření nové aktivity, ale v adrese pro úpravu se nachází i ID aktivity. Pokud se v adrese nachází ID aktivity, klient zašle požadavek na „B:/api/teacher/get-creating-activity-data/:activityId“, a pokusí se načíst data již existující aktivity. Formulář pro úpravu aktivity je téměř totožný s formulářem pro jeho tvorbu. Jediný rozdíl je v tlačítku uložit. Pokud se tvoří nová aktivita, tlačítko uložit slouží k vytvoření nového dokumentu v kolekci aktivit. Po prvním uložení již klient od serveru získá ID Aktivity, aby mohl při dalším stisknutím tlačítka „uložit“ aktualizovat již vytvořený dokument, který nalezne pomocí ID aktivity.

Změnit stav sdílení aktivity lze kliknutím na současný stav sdílení. Pokud se aktivita nesdílí, tak se po kliknutí zašle požadavek o sdílení na adresu „B:/share-activity/:activityId“. Po autentizaci a autorizaci se pomocí metody **updateOne,** zavolané na modelu aktivity, aktualizuje hodnota vlastnosti **isShared** na hodnotu true. Opačná možnost, že pokud se aktivita sdílí, tak se zašle požadavek na adresu „B:/stop-sharing-activity/:activityId“. Server postupuje analogicky, jako při sdílení aktivity. Rozdíl je pouze v tom, že vlastnost aktivity **isShared** se nastaví na hodnotu false.

6.3.16 Neimplementované funkcionality

V rámci dosavadní verze aplikace nejsou implementovány funkcionality pro vyhledávání aktivit. Nebylo zhotovené prostředí pro změnu údajů žáků. Dále nebyla

implementována funkce, která měla zajišťovat možnost zadání domácích úkolů. Systém prozatím nezaznamenává počet odehrání dané aktivity, a tudíž jej ani nezobrazuje. Funkce přímého vyhledávání podle názvu aktivity není prozatím implementována. Aktivity lze tedy vyhledávat pouze kliknutím na předmět a vybráním kategorie. Aktivita pexeso není plně implementovaná. Pro pexeso je zatím vytvořeno pouze herní prostředí. Načítání a ukládání herních dat pexesa z databáze není umožněno. Na straně serveru prozatím není ověřováno:

- zda žáka, kterému učitel ukládá výsledek, vytvořil stejný učitel,
- jestli změnu stavu sdílení aktivity mění učitel, který ji vytvořil,
- zda učitel upravuje herní data jen těch aktivit, které vytvořil

Posledním požadavkem školy, který nebyl implementován, je přehrání animace za úspěšné dokončení aktivity.

7 Shrnutí výsledků

Hlavním výsledkem práce je vytvořená webová aplikace, která byla vyvíjena s ohledem na návrhový vzor MVC, podporující výuku žáků se znevýhodněním.

Na samém začátku došlo k analýze požadavků. Požadavky byly sbírány formou volného rozhovoru. Pedagogičtí pracovníci školy tímto způsobem sdělili funkční i nefunkční požadavky. Na základě těchto požadavků byl vytvořen návrh. Ke všem požadavkům návrh nebyl zrealizován. Nebyly navrhnuty funkcionality pro zadání domácích úkolů, možnost měnit údaje žáků, vyhledávání mezi aktivitami pomocí názvu aj. Návrh byl podkladem pro implementaci webové aplikace. Vzdělávacím prvkem systému jsou aktivity pro využití u žáků s lehkým, středním nebo těžkým mentálním postižením.

V rozhovorech s pedagogy byly rozebrány způsoby používání aktivit. Některé způsoby, které si byly velmi blízké vzhledem k ovládání, se sjednotily a vznikly typové aktivity. Byly navrhnuty následující typové aktivity: Řazení, Hledej, Třídění a Pexeso. Z vyjmenovaných typových aktivit byly plně implementovány všechny kromě Pexesa. Typová aktivita Pexeso se nachází v rozpracované fázi, jejíž frontendová část je již implementovaná, tudíž je funkční, prozatím s lokálními daty. Je dostupná k vyzkoušení na adrese „F:/pexeso“. Po budoucím vytvoření modelu herních dat Pexesa a dokončení implementace backendové části s tím spojené, bude možné tuto typovou aktivitu využívat stejným způsobem, jako ostatní typové aktivity.

Aplikace zadané požadavky ze strany školy z větší části splňuje. K implementaci všech požadavků nedošlo. Do systému je možno vstoupit v roli hosta, žáka a učitele. Každá role má své funkcionality i omezení. Pro učitele byly implementovány funkce: registrace žáka, uchovávání seznamu žáků, zvolení aktivního žáka, ukládání výsledků žáka, tvoření vlastních aktivit z výběru dostupných typových aktivit, správa vlastních aktivit. Všem rolím je umožněno filtrovat aktivity dle předmětů, kategorií a úrovní postižení. Datumy ve výsledcích žáka prozatím nejsou zobrazeny v přirozeném formátu.

Webová aplikace byla vyvinuta za použití hlavních technologií React.js a Node.js. Pro tvorbu uživatelského rozhraní byl využit React.js. Node.js byl použit pro tvorbu webového serveru.

Výsledkem analýzy, návrhu a implementace typových aktivit a ostatních požadavků je systém umožňující přidávat aktivity, tedy systém pro podporu vzdělávání.

Hlavní cíl, analyzovat, navrhnout a implementovat systém podporující vzdělávání žáků s vybranými speciálními potřebami, byl splněn.

8 Závěry a doporučení

Předmětem bakalářské práce byl vývoj systému pro podporu vzdělávání žáků se znevýhodněními pro Základní školu speciální a praktickou školu DČCE Vrchlabí. Hlavním cílem bylo analyzovat, navrhnout a implementovat systém pro podporu vzdělávání žáků se speciálními vzdělávacími potřebami. Na základě školou stanovených požadavků došlo k vypracování návrhu, podle kterého byl systém v konečné fázi implementován. V bakalářské práci byly vymezeny jednotlivé stupně mentálního postižení s jejich charakteristikami. Dále byly uvedeny a popsány technické prostředky, kterými škola disponuje. Byly předloženy předměty s kategoriemi, které jsou v systému použity. Došlo k seznámení s architekturou aplikace, způsobem komunikace mezi částmi aplikace a použitým návrhovým vzorem. Následující kapitola doplnila, jakým způsobem se typová cvičení vykonávají a jak jsou strukturovány. Následně byly vytvořeny modely, ve kterých se detailně popsaly informace, které se u nich budou uchovávat. Poslední hlavní kapitola se zaměřila na implementaci aplikace, kde byly charakterizovány jednotlivé použité technologie, jejímž prostřednictvím byla aplikace vyvinuta. Nakonec byla popsána struktura projektu s jejími dílčími částmi a implementace jednotlivých funkcí.

Jako doporučení pro budoucí vývoj aplikace by mohla být přidána funkcionalita ověřování emailu při registraci, aby se zabránilo neoprávněnému přiřazení adresy. Model učitele je pro implementaci této funkce předpřipravený a všechny potřebné vlastnosti již obsahuje. Následně by pro učitele systém mohl umožnit funkci pro změnu svého hesla i hesla svých žáků. Dalším vylepšením, které bude součástí aplikace, je stránkování při načítání aktivit, ať už při načítání aktivit ze sekce předmětů, či při vyhledávání ze všech aktivit. Bez této funkce by při zobrazení většího množství výsledků docházelo ke snížení výkonnosti aplikace. Funkci stránkování je vhodné přidat i k zobrazení výsledků daného žáka. Bude možné plně implementovat další typové úlohy, např. jako již rozpracované pexeso, které je dostupné na adrese „F:/pexeso“. Bylo by vhodné validovat herní data při ukládání aktivity, aby byl zajištěn bezproblémový průběh jejího hraní. V aplikaci také autor práce zamýšlí použít grafickou technologii THREE.js pro vývoj dalších aktivit, která by jim umožňovala přidat třetí rozměr. Aktivity vytvořené s pomocí této technologie

by mohly působit moderněji a poutavěji. Bonusem by mohla být funkce přehrávání zvukových stop, např. za účelem vylepšení odměnového a motivačního systému. V příští verzi aplikace by bylo vhodné upravit vzhled úvodní stránky a před vykreslováním aktivit počkat na stažení potřebných obrázků.

Aplikace má za úkol zefektivnit výchovně vzdělávací proces a motivovat žáky v přístupu k získání kompetence k učení. Vnést moderní přístup do výuky a naplnit požadavky současného vzdělávání.

Lze doporučit využívání této aplikace i ostatním speciálním školám.

9 Seznam použité literatury

[1] Front-end Developer: povinnosti [online]. Praha, 19. prosince 2019 [cit. 2021-04-06]. Dostupné z: https://www.welcometothejungle.com/cs/articles/front-end-developer-cz

[2] Backend. Co je backend [online]. Praha [cit. 2021-04-06]. Dostupné z: https://www.strafelda.cz/backend

[3] MÁCA, Jindřich. Úvod do React. Lekce 1 - Úvod do React [online]. [cit. 2021-04-07]. Dostupné z: https://www.itnetwork.cz/javascript/react/zaklady/uvod-do-react/

[4] SOON, Ming, CHAN, Jon a kol. LEARNING React: Getting started with React: What is ReactJS? [online]. [cit. 2021-04-07]. Dostupné z: https://riptutorial.com/Download/react.pdf

[5] DELBONO, Emanuele. *Node.js Succinctly*: What is Node.js? [online]. 2016 [cit. 2021-04-07]. ISBN 978-1-64200-116-7. Dostupné z: https://www.syncfusion.com/succinctly-free-ebooks/nodejs/introduction-to-node-js

[6] PIPEKOVÁ, Jarmila. *Kapitoly ze speciální pedagogiky*. 3., přeprac. a rozš. vyd. Brno: Paido, 2010. 401 s. ISBN 978-80-7315-198-0.

[7] EMERSON, Eric. *Problémové chování u lidí s mentální retardací*. Praha: Portál, 2008. 166 s. ISBN 978-80-7367-390-1.

[8] ŠVARCOVÁ-SLABINOVÁ, Iva. *Mentální retardace: vzdělávání, výchova, sociální péče*. Vyd. 4., přeprac. Praha: Portál, 2011. Speciální pedagogika (Portál). 224 s. ISBN 978-80-7367-889-0.

[9] HOFFENBERG, Sara E. Mild Mental Retardation [online]. [cit. 2019-04-03]. Dostupné z: https://link.springer.com/referenceworkentry/10.1007%2F978-0-387-79061-9_1786

[10] BAZALOVÁ, Barbora. *Dítě s mentálním postižením a podpora jeho vývoje*. Praha: Portál, 2014. 184 s. ISBN 978-80-262-0693-4.

[11] BENDOVÁ, Petra a Pavel ZIKL. *Dítě s mentálním postižením ve škole*. Praha: Grada, 2011. Pedagogika (Grada). 144 s. ISBN 978-80-247-3854-3.

[12] VALENTA, Milan a Oldřich MÜLLER. *Psychopedie*. 2. vyd. Praha: Parta, 2004. 443 s. ISBN 80-7320-063-5.

[13] React. React – *A JavaScript library for building user interfaces* [online]. [cit. 2021-04-10]. Dostupné z: https://reactjs.org/

[14] Express. *Express - Node.js web application framework* [online]. [cit. 2021-04-10]. Dostupné z: https://expressjs.com/

10 Přílohy

Součástí bakalářské práce je i zdrojový kód webové aplikace, který je odevzdán v samostatném souboru. Nejaktuálnější verze zdrojových kódů jsou dostupné na adresách:

https://github.com/kumprji1/sovicka-fe-mongo

https://github.com/kumprji1/sovicka-be-mongo

Aplikace je pro vyzkoušení zveřejněna na adrese:

https://sovicka-app.herokuapp.com/

UNIVERZITA HRADEC KRÁLOVÉ
Fakulta informatiky a managementu
Akademický rok: 2019/2020

Studijní program: Aplikovaná informatika
Forma studia: Prezenční
Obor/kombinace: Aplikovaná informatika (ai3-p)

Podklad pro zadání BAKALÁŘSKÉ práce studenta

Jméno a příjmení:	**Jiří Kumprecht**
Osobní číslo:	**I1800191**
Adresa:	Bílá Třemešná 120, Bílá Třemešná, 54472 Bílá Třemešná, Česká republika
Téma práce:	Systém pro podporu vzdělávání dětí se znevýhodněními
Téma práce anglicky:	Education support system for children with disabilities
Vedoucí práce:	Mgr. Daniela Ponce, Ph.D. Katedra informačních technologií

Zásady pro vypracování:

Cílem je analyzovat, navrhnout a implementovat systém podporující vzdělávání dětí s vybranými speciálnými potřebami

Seznam doporučené literatury:

Východiska, podmínky a strategie ve vzdělávání žáků s těžkým postižením - Hana Ošlejšková, Marie Vítková
Integrované vzdělávání dětí se zrakovým postižením – Alena Keblová
Metody aktivního vyučování – Dagmar Sitná
Motivační tříminutovky – Kathy Paterson
Co funguje ve třídě? Most mezi výzkumem a praxí – Robin Macpherson, Hendrick Carl
Learning React: Functional Web Development with React and Redux – Alex Banks, Eve Porcello
Learning Node.js – Wandschneider, Marc

Podpis studenta: Kumprecht

Datum: 27. 4. 2021

Podpis vedoucího práce:

Datum: