Technická univerzita v Košiciach
Fakulta elektrotechniky a informatiky

Metóda ikonicko-textovej formy výuky hendikepovaných detí na báze webových technológií

Bakalárska práca

2022 **Samuel Hambalek**

Technická univerzita v Košiciach
Fakulta elektrotechniky a informatiky

Metóda ikonicko-textovej formy výuky hendikepovaných detí na báze webových technológií

Bakalárska práca

Študijný program:	Informatika
Študijný odbor:	9.2.1. Informatika
Školiace pracovisko:	Katedra počítačov a informatiky (KPI)
Školiteľ:	Ing. Miriama Mattová
Konzultant:	Ing. Miriama Mattová

Košice 2022 **Samuel Hambalek**

Abstrakt v SJ

Bakalárska práca je zameraná na vytvorenie kolaboratívneho systému na báze webových technológií, ktorý obsahuje úlohy založené na metóde ikonicko-textovej formy výuky hendikepovaných detí. V práci sú analyzované spôsoby vytvorenia systému. Návrhová časť práce je venovaná webovému rozhraniu a serveru. Na základe návrhu bol systém implementovaný. V overení sú vyhodnotené záťažové testy a overená funkčnosť kolaborácie.

Kľúčové slová v SJ

Kolaborácia, Ikonicko-textová metóda výuky, Webové technológie, Vzdelávanie

Abstrakt v AJ

Bachelor thesis is focused on creating collaborative system via web technologies, which contains assignments based on ikonic-text method of teaching for handicapped children. The thesis contains analysis of possible ways to create the system. Design part of thesis is dedicated to web interface and server. The system was implemented based on design. Evaluation consist of evaluated load tests and proven collaboration functionality.

Kľúčové slová v AJ

Collaboration, Ikonic-text method form of teaching, Web technologies, Education

Bibliografická citácia

HAMBALEK, Samuel. *Metóda ikonicko-textovej formy výuky hendikepovaných detí na báze webových technológií*. Košice: Technická univerzita v Košiciach, Fakulta elektrotechniky a informatiky, 2022. 41s. Vedúci práce: Ing. Miriama Mattová

66939

TECHNICKÁ UNIVERZITA V KOŠICIACH

FAKULTA ELEKTROTECHNIKY A INFORMATIKY

Katedra počítačov a informatiky

ZADANIE BAKALÁRSKEJ PRÁCE

Študijný odbor: **Informatika**
Študijný program: **Informatika**

Názov práce:

Metóda ikonicko-textovej formy výuky hendikepovaných detí na báze webových technológií

Ikonic-text method form of teaching for handicapped childrens via web technologies

Študent: **Samuel Hambalek**
Školiteľ: **Ing. Miriama Mattová**
Školiace pracovisko: **Katedra počítačov a informatiky**
Konzultant práce: **Ing. Miriama Mattová**
Pracovisko konzultanta: **Katedra počítačov a informatiky**

Pokyny na vypracovanie bakalárskej práce:

1. Analyzovať možnosti kolaboratívneho prostredia pre metódu ikonicko-textovej formy výuky.
2. Na základe analýzy navrhnúť a implementovať kolaboratívny webový systém pre danú metódu.
3. Implementovať kolaboratívne riešenie úloh na základe metódy ikonicko-textovej formy výuky.
4. Overiť a vyhodnotiť funkčnosť implementovaného systému.
5. Vypracovať dokumentáciu podľa pokynov školiteľa.

Jazyk, v ktorom sa práca vypracuje: slovenský
Termín pre odovzdanie práce: 27.05.2022
Dátum zadania bakalárskej práce: 29.10.2021

TECHNICKÁ UNIVERZITA V KOŠICIACH FAKULTA ELEKTROTECHNIKY A INFORMATIKY -10-

prof. Ing. Liberios Vokorokos, PhD.
dekan fakulty

Čestné vyhlásenie

Vyhlasujem, že som záverečnú prácu vypracoval(a) samostatne s použitím uvedenej odbornej literatúry.

Košice, 27.5.2022 .....

Vlastnoručný podpis

Poďakovanie

Najväčšia vďaka patrí vedúcej a konzultantke práce Ing. Miriame Mattovej za jej čas, odborné vedenie a neoceniteľnú pomoc počas riešenia mojej záverečnej práce.

Rovnako by som sa rád poďakoval svojej rodine a priateľom za ich podporu a povzbudzovanie počas celého môjho štúdia.

Obsah

Zoznam obrázkov

Úvod

Vývoj digitálnych technológií bol za posledné roky enormný a stále napreduje. Vďaka tomu je možné povedať, že technológie pomáhajú a uľahčujú každodenné činnosti miliónov ľudí. V medicíne urýchľujú, zefektívňujú a zvýšujú úspešnosť zákrokov. Umožňujú zariadiť úradné úkony, posielať peniaze v priebehu pár sekúnd, či komunikovať s ľudmi z celého sveta. Zlepšenie a napredovanie nastalo aj v oblasti vzdelávania, keďže pribudlo množstvo vzdelávacích portálov, ktoré umožňujú ľuďom vzdelávať sa a posunúť svoje vedomostí na vyššiu úroveň. Avšak nie všetci majú túto možnosť. Pre niektorých ľudí je obmedzením prostredie, v ktorom žijú, pre iných zase finančná nedostupnosť technológií. Existuje však skupina ľudí, ktorí majú fyzické obmedzenie, kvôli ktorému je pre nich bežné vzdelávanie, v podobe, v akom ho bežne poznáme nedostupné.

Vzdelávaniu tejto skupiny ľudí sa venuje Spojená škola Pavla Sabadoša internátna v Prešove. Na tejto škole prišli s riešením zjednodušenia vzdelávania pomocou ikonicko-textovej metódy výuky. Aj táto metóda však má svoje limitácie, jednou z nich je nutnosť prezenčnej výučby, keďže pri výuke sú používané kartičky. Problém sa zhoršil aj kvôli aktuálnej situácii s pandémiou korona vírusu, kedy bolo nevyhnutné presunuť výučbu do online priestoru.

Táto práca je zameraná na odstránenie tohto obmedzenia. Riešením je vytvorenie systému na báze webových technológií, ktorý žiakom tejto školy umožňuje výuku prostredníctvom počítača. Práca je zameraná na vytvorenie kolaboratívneho sytému, ktorý umožňuje spoluprácu medzi používateľmi počas vypracovávania úloh. Úlohy sú dvoch typov, v prvom type majú používatelia k dispozícii znenie vo forme textu, ich úlohou je priradiť každému slovu obrázok. Druhým typom úlohy je opačný formát, používatelia vidia obrázky, ktorým majú priradiť text.

Práca je implementovaná pomocou webových technológií, čo pre používateľov znamená možnosť používania aplikácie z akéhokoľvek zariadenia, ktoré má nainštalovaný webový prehliadač a prístup k internetu.

Formulácia úlohy

Prvou úlohou v tejto práci bude naštudovanie ikonicko-textovej metódy, webových technológií, pomocou ktorých je možné systém vytvoriť a analyzovanie možností vytvorenia kolaboratívneho systému.

Druhou úlohou tejto bakalárskej práce je na základe analýzy navrhnúť kolaboratívny systém na báze webových technológií pre danú metódu, ktorý bude obsahovať dva typy úloh.

Po navrhnutí je potrebné implementovať systém do navrhovanej podoby. Implementovaný systém má obsahovať kolaboráciu, riešenie úloh ikonicko-textovej metódy a byť vytvorený vo webovom prostredí.

Ďalším bodom práce je overiť a vyhodnotiť funkčnosť implemenovaného kolaboratívneho systému.

Posledným bodom práce je vytvoriť dokumentáciu navrhnutého a implementovaného systému.

1 Analytická časť

Mnohé z hendikepovaných detí majú problém s porozumením významu slov v texte, za účelom vyriešenia tohto problému vznikla ikonicko-textová metóda výučby, ktorá zjednodušuje výučbu hendikepovaných detí pomocou piktogramov.[1]

Ikonicko-textová metóda výučby je aktuálne aktívne využívaná na Spojenej škole Pavla Sabadoša internátnej v Prešove. Zatiaľ však chýba riešenie v online priestore, ktoré by bolo optimalizované pre účely výučby pomocou tejto metódy. Chýbajúce riešenie by malo pomôcť odstrániť niektoré z obmedzení, ktoré so sebou prináša aktuálny spôsob výučby.

Riešenie, ktoré bude v tejto kapitole analyzované je kolaboratívny systém na báze webových technológií, v ktorom by sa hendikepované deti mohli vzdelávať. Tento systém by mal uľahčiť výučbovú činnosť tým, že by výučba ikonicko -textovej metódy bola presunutá do virtuálneho priestoru. Vytvorenie tohto systému by taktiež umožnilo zabezpečiť, aby výuka mohla byť vykonávaná na diaľku, čím by bola uľahčená najmä aktuálna situácia s výučbou, keďže kvôli pandémii, ktorá ľudí obmedzuje už dva roky, sa väčšina škôl musela presunúť do online priestoru.

1.1 Súvisiace riešenia

Táto kapitola je zameraná na existujúce riešenia, ktoré sa venujú vytvoreniu kolaboratívneho systému na báze webových technológií.

- **Devising a model of electronic School Management System based on web services for secondary schools in Macedonia**

  Pokročilé počítačové technológie, ktoré sú aktuálne dostupné môžu zohrať hlavnú rolu v reorganizácii manažmentu školských aktivít s cieľom podporiť jednotu a solidárnosť medzi študentmi, rodičmi a učiteľmi. Počítače sa však nepoužívajú na analýzu údajov a výstupných informácií. Preto bol

vyvinutý e-School Management System (e-SMS), aby uľahčil pedagogickým a administratívnym zamestnancom riadiť školské aktivity na stredných školách a rodičom/ učiteľom mať včasné a spoľahlivé informácie o výkone svojho dieťaťa / študenta. V tomto dokumente navrhujeme koncepčný model systému riadenia e-školy a vysvetľujeme funkčnosť e-SMS a výhody elektronického riadenia. [2]

- **Eduvid, web video to support digital learning in rural primary schools**

  Jeden z problémov vo vidieckych školách je nedostatok vzdelávacích materiálov. Učenie v triede je preto takmer jediný zdroj na vzdelávanie. Implementácia produktov určených na digitálne vzdelávanie môže pomôcť prekonať tieto problémy. Služby určené na digitálne vzdelávanie, ktoré už boli implementované v Keerom sú založené na kolaborácii učiteľov a virtuálnej triedy prostredníctvom webu. Študenti základných škôl potrebujú zaujímavý vzdelávací materiál, ktorý by zvýšil ich záujem o učenie. Predpokladá sa, že učenie pomocou videa je pre študentov viac pochopiteľné, ako pomocou textu. Tento výskum navrhol systém na prezeranie videí prostredníctvom webu. Výskum bol vykonávaný tak, aby produkt čo najviac spĺňal požiadavky koncových používateľov. Systém je navrhnutý formou web stránky, ktorá je pre používateľa ľahko dostupná a interaktívna. [3]

- **Transformative education Web 2.0 systems for enriching high school STEM education**

  Napriek tomu, že existuje množstvo online STEM vzdelávacích web stránok dostupných pre študentov, neexistuje žiadny vedecký prístup k riešeniu výziev vzdelávania STEM v mimoškolskom prostredí. Štúdia stručne odhaľuje prístup k transformatívnemu riešeniu toho, ako by učenie matematiky mimo formálneho prostredia strednej školy mohlo prispieť k matematickým schopnostiam študentov. Nasadený bol webový systém, ktorý sa zameriava na podporu kolaboratívneho učenia študentov. Na adresovanie objemných údajov sa používa architektúra veľkých údajov, ktorá zaisťuje, že neštrukturované dáta webu sú vždy pripravené na analýzu v reálnom čase. [4]

- **Esprit-social network: An internal collaborative platform for the students of ESPRIT**

  Cieľom tejto štúdie je prezentovať kolaboratívnu sociálnu platformu vyvinutú za účelom výskumného projektu na zlepšenie serióznej spolupráce

medzi študentmi Inžinierskej Školy v Tunisku. Platforma sa sústreďuje na automatické vytváranie spoluprác medzi študentmi na základe typu vykonávaných úloh, požadovaných zručností, minulých aktivít a vzájomnom hodnotení používateľov. Hlavnými motiváciami rozvoja internej sociálnej siete je rastúca tendencia používania sociálnych sietí učiteľmi a študentmi na komunikáciu v oblasti vzdelávania, negatívny predpoklad o vplyve verejnej sociálnej siete na produktivitu a na súkromie osobných údajov, menší záujem o koncept spoločného riešenia problémov prostredníctvom sociálnych sietí. Aplikácia bola vyvinutá pomocou rámca symfony2 na vytváranie webových aplikácií. [5]

1.2 Analýza platforiem riešenia problematiky

Systém môže byť vytvorený rôznymi spôsobmi. Taktiež môže byť určený pre rôzne platformy. Nasledovná časť analýzy je rozdelená na viacero častí. Venuje pozornosť niektorým vybraným spôsobom, ktorými je možné vytvoriť systém a zároveň opisuje výhody a nevýhody vybraných spôsobov tvorby systému.

Aplikácia pre webové prostredie

Výhody:

- Rýchly prístup prostredníctvom webovej adresy niekoľkými klikmi,
- dostupnosť prostredníctvom webových prehliadačov,
- nie je potrebné aplikáciu inštalovať do zariadenia,
- aplikácia nezaberá miesto v pamäti zariadenia,
- môže byť používaná na zariadení akejkoľvek platformy.

Nevýhody:

- Je potrebné mať prístup k internetu,
- rýchlosť aplikácie závisí od rýchlosti pripojenia k internetu,
- je potrebné optimalizovať aplikáciu na rôzne veľkostí obrazoviek a rôzne webové prehliadače.

Aplikácia pre mobilné zariadenie

Výhody:

- Lepši používateľský zážitok, [6]
- rýchlosť behu aplikácie nie je viazaná na rýchlosť internetového pripojenia,
- prístup k softvéru a hardvéru zariadenia.

Nevýhody:

- Aplikácia zaberá miesto v pamäti, keďže je nutné ju nainštalovať do zariadenia používateľa,
- je potrebné vlastniť mobilné zariadenie,
- je nevyhnutné optimalizovať aplikáciu na rôzne zariadenia a rôzne verzie operačných systémov.

Desktop aplikácia

Výhody:

- Nie je potrebné pripojenie k internetu,
- vyššia bezpečnosť dát,
- lepšia výkonnosť aplikácie.

Nevýhody:

- Horšia prenosnosť zo zariadenia na zariadenie,
- potrebná inštalácia,
- zaberá miesto v pamäti zariadenia.

Po zvážení všetkých výhod a nevýhod porovnávaných platforiem bude systém vytvorený na báze webových technológií.

1.3 Analýza technológií riešenia problematiky

Analyzovaný systém bude takzvaná full-stack webová aplikácia, ktorú je možné rozdeliť na dve časti. Prvou časťou bude backend, teda serverová časť webovej aplikácie, ktorej prácu používateľ nemá možnosť vidieť. Úlohou serverovej časti bude starať sa o spoluprácu servera s databázou. Pomocou serverovej časti bude aplikácia schopná čerpať dáta z databázy, ktoré je potrebné zobraziť. Predtým, než ich pošle na klientskú časť aplikácie, sú dáta pretvorené do formy, s ktorou je klientská časť aplikácie schopná pracovať. Pre túto časť aplikácie môžu byť zvažované programovacie jazyky ako Python, Java, Node.js, či PHP.

Druhou časťou bude frontend, teda už spomínaná klientská časť aplikácie, ktorá bude poskytovať používateľské rozhranie zobrazené prehliadačom, prostredníctvom ktorého bude používateľovi umožnené interagovať s aplikáciou. [7] Táto časť aplikácie môže byť vytvorená pomocou technológií, ako napríklad JavaScript, React, Angular alebo Vue.

V tejto kapitole budú analyzované spôsoby vytvorenia systému, čo by mohol systém obsahovať, pomocou akých technológií by mohol byť vytvorený a ako by mohli jednotlivé časti systému fungovať.

Vytváraný systém môže byť vyhotovený rôznymi spôsobmi, aplikácia by mala obsahovať kolaboráciu medzi používateľmi, ktorí by si mohli navzájom pomáhať, súťažiť alebo zdieľať svoje pokroky, prípadne kontrolovať pokroky a výsledky iných, napríklad učitelia by mohli sledovať žiakov v triede. Pomocou aplikácie by požívatelia mali mať možnosť riešiť, vytvárať, alebo generovať úlohy na základe metódy ikonicko-textovej formy výuky.

Nasledujúce častí sú venované spôsobom vytvorenia aplikácie a taktiež technológiam, ktorými je možné aplikáciu vybudovať.

Systém vytvorený pomocou CSS, Angularu, Javy a MySQL

- Jedným z možných variantov je vytvoriť základnú štruktúru klientskej časti aplikácie pomocou HTML.
- Na vytvorenie dizajnu aplikácie je možné uvažovať o využití CSS. Výhoda použitia CSS bez akéhokoľvek rámca je vytvorenie jedinečného dizajnu a prispôsobenie dizajnu aplikácie podľa vlastných predstáv. Nevýhodou je zvýšená náročnosť tvorby dizajnu, najmä verzie dizajnu určenej pre zariadenia s menšími displejmi.

- Logická stránka tvorenej aplikácie by mohla byť vytvorená rámcom Angular. [8] Výhodou tohto rámca je možnosť generovať html kód podľa stanovených podmienok, čo by uľahčilo dynamické zobrazovanie rôznych úloh a zadaní, ktoré má žiak vykonať. Jednou z ďalších výhod tohto rámca je tzv. Two-way binding [9], ktorý by mohol zabezpečiť zobrazovanie žiakom vybraných odpovedí pri riešení úloh v reálnom čase. Angular je určený skôr pre väčšie aplikáncie, z tohto hľadiska by bol pre vytváraný systém zbytočne robustný, taktiež kapacitne náročnejší.

- Pre serverovú časť aplikácie môže byť zvažovaná Java. Pomocou JDBC API, ktorú je možné v Jave využívať, by mohla aplikácia komunikovať s databázou. Webové aplikácie vytvorené v Jave sú schopné spracovať požiadavky od viacerých používateľov v rovnakom čase, táto vlastnosť je známa ako Multi-threading. V aplikácii by sa to prejavilo menším množstvom porúch, rýchlejšou odozvou a lepšou výkonnosťou.

- Za účelom ukladania dát by mohol byť využitý relačný databázový systém MySQL.

- Prihlásenie do aplikácie by mohlo byť riešené pomocou Google Sign-In. Výhodou tohto riešenia by bolo pomerne jednoduchá implementácia, taktiež by nebolo potrebné ukladať do databázy všetky údaje spojené s autentifikáciou používateľa. Nevýhodou riešenia pre používateľa aplikácie by mohla byť nutnosť mať vytvorený Google účet, alebo prípadná nutnosť vytvorenia si takéhoto účtu, čo by viedlo k zbytočným komplikáciám a k sťaženému prístupu k použitiu aplikácie. V tomto prípade by mohlo byť uvažované o vytvorení systému, v ktorom by mohli používatelia proti sebe súťažiť a to formou vykonávania úloh, ktoré by im boli generované systémom automaticky, prípadne by používatelia mohli byť rozdelení na učiteľov a žiakov, kde by učiteľ zadal žiakom výzvy, ktoré by mali splniť a tí by medzi sebou súťažili. Výhodou tohto systému, ktorý by bol založený na súťažení, by mohlo byť vytvorenie motivácie, ktorá by žiakov hnala vpred. Nevýhodou by mohlo byť demotivácia z prehry.

Systém vytvorený pomocou JavaScriptu, Pythonu a MongoDB

- V tomto variante by sa o dizajn HTML prvkov postaral jazyk CSS, ktorého funkčnosť by bola obohatená o pre-procesor Sass, ktorý by zjednodušil a zrýchlil písanie kódu a teda dizajnovanie prvkov.

- Vanilla JavaScript by sa postaral o funkčnosť na klientskej strane aplikácie. Výhoda Vanilla JavaScriptu oproti rámcom spočíva najmä v rýchlosti behu aplikácie, nevýhoda zase v tvorbe vlastnej štruktúry aplikácie, o čo by sa mohol postarať rámec.

- Chod serverovej časti aplikácie by zabezpečil jazyk Python, ktorého jedna z výhod je tá, že obsahuje knižnicu JMESPath, pomocou ktorej by bola uľahčená práca pri zápise a čítaní súborov JSON.

- Spomínané JSON súbory by boli uložené v NoSQL dokumentovej databáze MongoDB. Vďaka tomu, že MongoDB je nerelačná databáza, nie je potrebné vytvárať tabuľky.

- Alternatívou zabezpečenia prístupu používateľa ku svojmu kontu by mohlo byť prostredníctvom ukladania používateľkého mena v cookie. Výhodou tohto riešenia by bola jednoduchá implementácia, taktiež jednoduchý prístup používateľa ku kontu. Toto riešenie by však mohlo spôsobiť bezpečnostné problémy, jedným z nich by mohol byť prístup akéhokoľvek používateľa ku kontu iného používateľa len nastavením rovnakého používateľského mena, keďže konto by nebolo zabezpečené žiadnym heslom. Po prihlasení by mohli mať používatelia prístup k vypracovaniu úloh, na ktorých by mohli spolupracovať s inými používateľmi. Toto riešenie by prinieslo prehĺbenie spolupráce medzi používateľmi, ktorými budú žiaci. Taktiež by sa mohli navzájom dopĺňať a posúvať vpred. Problém by mohol nastať, ak by napríklad výučba prebiehala dištančnou formou, kedy by bola komunikácia medzi žiakmi obmedzená a výrazne sťažená.

Systém vytvorený v Reacte, Bootstrape, Node.js a PostgreSQL

- Základnú štruktúru HTML by obohatil o dizajn rámec jazyka CSS, Bootstrap. [10] Vďaka nemu by bolo vytvorenie dizajnu aplikácie zjednodušené a zrýchlené tým, že by boli použité už vytvorené CSS predlohy, tým pádom by nebolo potrebné pracne vytvárať responzívny dizajn, ktorý by vyhovoval všetkým veľkostiam obrazoviek. Nevýhodou použitia predvytvorených šablón je, že výsledný dizajn by nebol jedinečný.

- Funkčnú časť aplikácie na klientskej strane by zabezpečovala knižnica React. [11] Pomocou tejto knižnice by bolo uľahčené renderovanie rôznych komponentov na obrazovku. Napríklad za účelom zobrazenia úloh, ktoré má

žiak vykonať by bolo vhodné použiť takzvané props, pomocou ktorých by kompomenty mohli byť zobrazované dynamicky s rôznym obsahom.

- Výhodou použitia Node.js[12] na serverovej strane aplikácie by bolo písanie kódu v rovnakom jazyku na klientskej aj serverovej časti, čo by uľahčilo situáciu eliminovaním potreby učiť sa ďalší programovací jazyk. Node.js taktiež umožňuje integráciu modulu node-postgres, ktorý by uľahčil prepojenie serverovej časti aplikácie s databázou PostgreSQL.

- Ako už bolo naznačené databáza by bola vytvorená pomocou PostgreSQL. [13]

- Prihlásenie do aplikácie by mohlo byť riešené pomocou mena a prihlasovacieho hesla, ktoré by boli uložené v databáze. Výhodou takéhoto spôsobu prihlasovania by bolo zvýšenie bezpečnosti prihlasovania, taktiež by každý používateľ mal k dispozícii vlastný učet, vďaka čomu by bolo jednoduché používateľom priradiť roly. V tomto prípade by sa mohlo jednať o roly učiteľa a žiaka. Učiteľ by zadával žiakom úlohy, ktorých výsledky by mohol po ich vypracovaní skontrolovať, prípadne ohodnotiť. Taktiež by si žiak mohol vygenerovať vlastné testy, na ktorých by si otestoval svoje znalosti. Nevýhodou by však bolo nutnosť zapamätať si prihlásovacie údaje od svojho účtu.

1.4 Analytický záver

Na základe zhodnotenia všetkých výhod a nevýhod technológií a spôsobov vytvorenia systému, ktoré boli opísané v analýze, bude systém na klientskej strane vytvorený v Bootstrape a Reacte, na serverovej strane v Node.js, pretože je výhodné použiť rovnaký jazyk na obe časti aplikácie. Databáza bude vytvorená pomocou PostgreSQL.

2 Návrh kolaboratívneho systému

Navrhnúť systém, ktorý má implementovanú kolaboráciu je možné rozličnými spôsobmi. Spôsobov je veľa najmä kvôli pestrosti použitia rozličných rol a možností rôznych kolaborácií medzi nimi. Tento systém bude mať implementovaný spôsob kolaborácie, kde budú môcť študenti medzi sebou navzájom spolupracovať vo virtuálnej miestnosti, v ktorej budú mať na starostí vypracovanie rôznych úloh. V jednoduchosti je možné povedať, že úlohy budú dvoch typov.

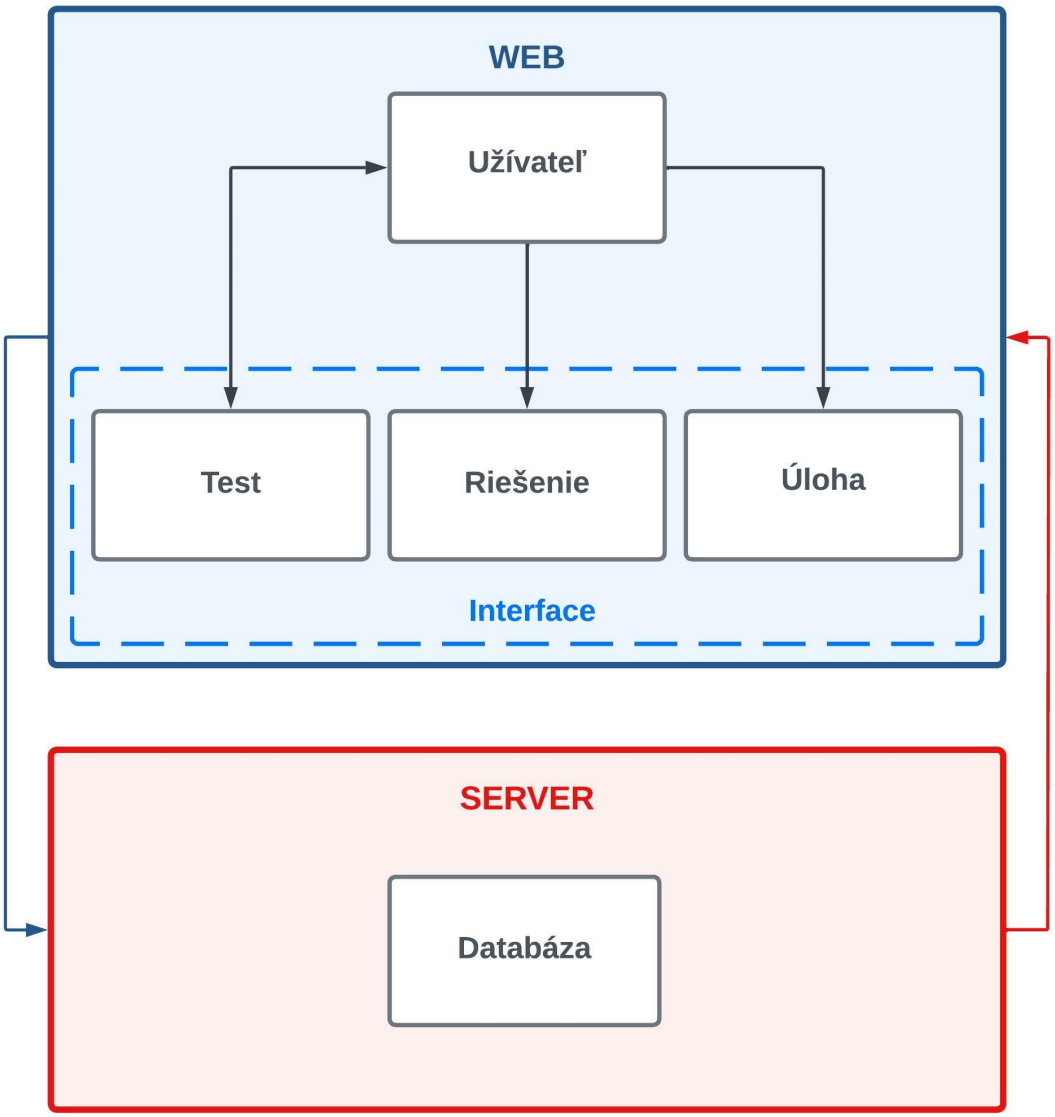
V prvom type úlohy budú mať študenti zadanú úlohu vo forme textu, v ktorom každému slovu budú priraďovať obrázok.

Druhý typ úlohy je opačného charakteru, jedná sa o úlohy, ktoré budú zadefinované pomocou obrázkov. Obrázky budú znázorňovať slová, taktiež budú nasledovať za sebou takým spôsobom, aby vytvorili vetu. Úlohou študentov bude správne priradiť obrázky k rôznym slovám, ktoré budú môcť vybrať z ponuky.

2.1 Všeobecná architektúra systému

Pri vytváraní vhodného návrhu bol najväčší ohľad braný na vytvorenie kolaborácie medzi používateľmi. Keďže používatelia vytváraného systému budú deti s hendikepom, špeciálnu pozornosť dostala pri vytváraní návrhu jednoduchosť, no zároveň so zámerom, aby študenti mohli medzi sebou spolupracovať a navzájom si pomáhať, či dopĺňať sa.

Architektúra systému bude pozostávať z dvoch hlavných častí. Prvá časť je Web, ktorý je možné chápať ako webové rozhranie, v ktorom budú môcť študenti riešiť zadané úlohy. Druhou časťou bude Server, ten bude obsahovať databázu, ktorá bude spolupracovať s webovým rozhraním systému. Vizuálne zobrazenie architektúry systému je možné vidieť v diagrame na nasledujúcej strane.



Obr. 2.1: Všeobecná architektúra kolaboratívneho systému

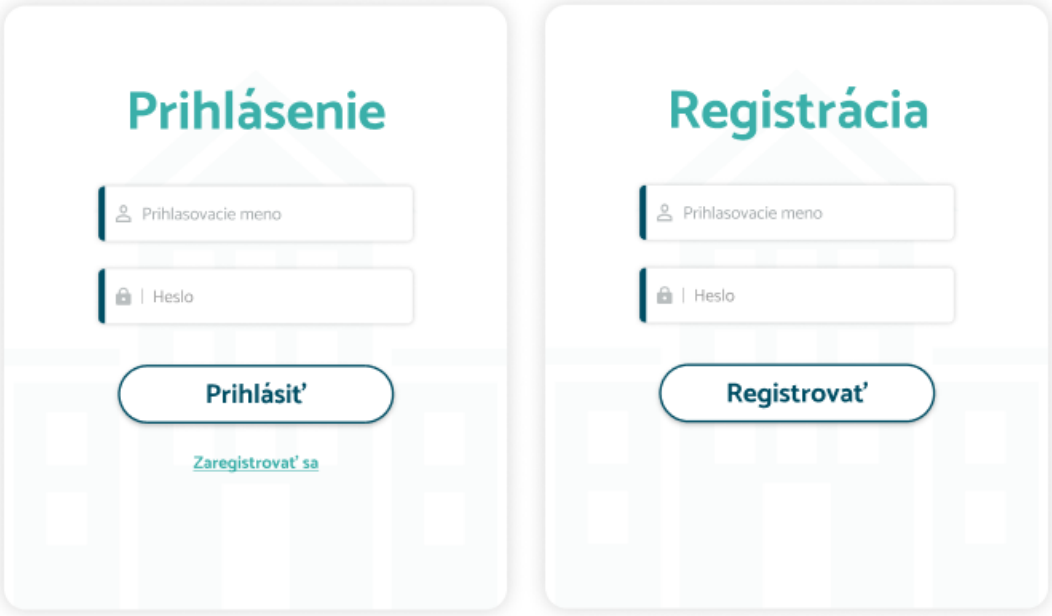
Na obrázku je možné vidieť, ako sú jednotlivé častí komponentov prepojené. Užívateľ dokáže komunikovať s každým z komponentov vo webovom rozhraní. Ostané komponenty vo webovom rozhraní nie sú medzi sebou prepojené, sú len zobrazované užívateľovi. Medzi hlavnými komponentmi systému, teda webom a serverom prebieha obojstranná komunikácia. Komponenty systému zobrazené na predchádzajúcom obrázku sú detailnejšie opísané v nasledujúcich kapitolách.

2.2 Webové rozhranie systému

Webové rozhranie je časť systému, s ktorou príde do kontaktu každý používateľ. Jeho úlohou bude načítavať dáta zo Serverovej časti sytému a v požadovanom formáte zobrazovať používateľovi na obrazovku. Taktiež bude môcť načítané dáta aktualizovať alebo pridať nové dáta do databázy, v prípade potreby úschovy daných dát. Webové rozhranie je možné rozdeliť do štyroch logických komponentov:

- Užívateľ,
- test,
- riešenie,
- úloha.

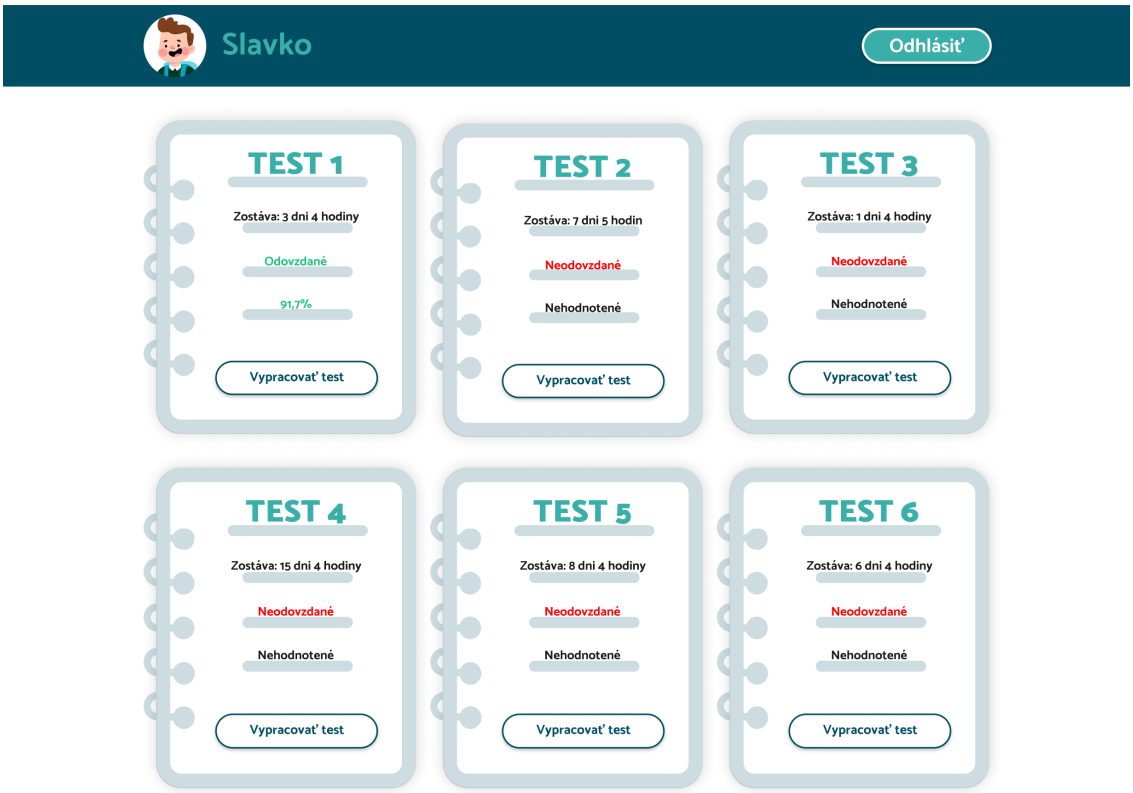
Po načítaní stránky sa používateľovi zobrazí formulár pre prihlásenie. Ak je návštevník stránky zaregistrovaný, môže tento formulár vyplniť a tým vstúpiť do systému. Ak nie je zaregistrovaný, môže využiť formulu na registráciu, na ktorú môže kliknúť vo formulári na prihlásenie. Ak klikne na možnosť zaregistrovať sa, zobrazí sa mu formulár, po vyplnení ktorého bude úspešne zaregistrovaný a môže sa prihlásiť. Formulár na prihlásenie a registráciu je možné vidieť na obrázku nižšie.



Obr. 2.2: Vľavo formulár prihlásenia, vpravo formulár registrácie

Po úspešnom prihlásení sa používateľovi zobrazí obrazovka, v ktorej bude mať možnosť sa odhlásiť alebo si bude môcť zvoliť test, ktorý bude chcieť vypracovať, ak takýto test bude k dispozícii.

V karte Test bude mať používateľ k dispozícii dátum, ktorý bude vyjadrovať kedy bude daný test aktívny. V tom čase bude môcť používateľ test vypracovať. Príklad úvodnej obrazovky s kartami testov je možné vidieť na nasledujúcom obrázku.



Obr. 2.3: Príklad úvodnej obrazovky po prihlásení

Užívateľ

Tento komponent predstavuje používateľa systému. Návštevník stránky sa stane užívateľom v momente, keď sa úspešne zaregistruje do systému. Po úspešnej registrácii sa užívateľ môže pohybovať v systéme. Má právo vstúpiť do virtuálnej miestnosti, kde bude vypracovávať úlohy spolu s ostatnými používateľmi, ktorí sa budú nachádzať v tej istej virtuálnej miestnosti. Vo virtuálnej miestnosti môže čítať znenie úlohy, spolupracovať na vypĺňovaní riešenia úlohy, taktiež môže vidieť ostatných používateľov, ktorí sú aktívni v rovnakej miestnosti a podieľajú sa na vypracovávaní úlohy.

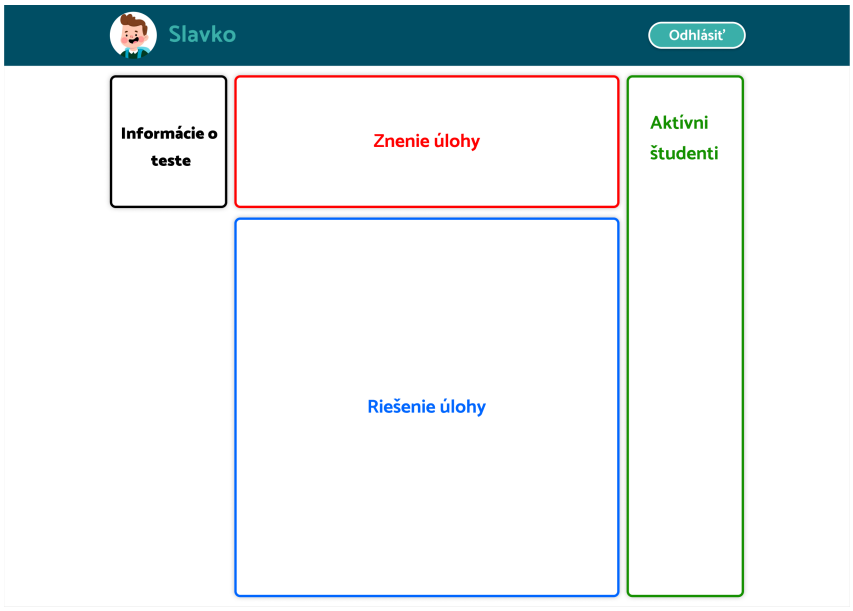
Test

Ďalším komponentom vo Webovom rozhraní bude virtuálna miestnosť, označovaná ako Test. Do takejto miestnosti môže používateľ vojsť z úvodnej obrazovky, ak je miestnosť v danej chvíli aktívna. Po vstupe do miestnosti bude môcť používateľ vidieť na ľavej strane obrazovky informácie o teste, ako napríklad poradie úloh, ktoré sú vypracovávané.

Na pravej strane miestnosti budú zobrazení používatelia, ktorí sú aktuálne v miestnosti. Ak niektorý z používateľov odíde z miestnosti, jeho meno bude automaticky zmazané, keďže už nebude považovaný za aktívneho člena miestnosti.

Hlavná časť miestnosti sa bude nachádzať v strede obrazovky. Táto časť bude pozostávať z dvoch komponentov. V hornej časti bude komponent, ktorý bude mať za úlohu zobraziť znenie úlohy. Pod ním sa bude nachádzať komponent zobrazujúci riešenie. V ňom budú používatelia, ktorí sa nachádzajú v miestnosti spolupracovať na vypracovávaní úlohy. Na vypracovanie každej úlohy bude stanovený limit, za ktorý budú študenti nutení úlohu vypracovať. Ak budú študenti riešiaci úlohu so svojím riešením spokojní ešte pred ukončením časového limitu, budú môcť použiť tlačidlo, pomocou ktorého bude možné úlohu predčasne skontrolovať.

Ak bude riešenie správne, zobrazí sa ďalšia otázka. Ak správne nebude, študenti budú môcť opraviť svoje riešenie do ukončenia časového limitu. Rozdelenie komponetov vo virtuálnej miestnosti označovanej ako Test je možné vidieť na nasledujúcom obrázku.



Obr. 2.4: Obrazovka po vstupe do Testu

Úloha

V poradí tretím kompomentom je Úloha, v ktorom sa nachádza znenie úlohy. Znenie úlohy môže byť buď textové alebo obrázkové. Pri prvej možností je úlohou študenta vypracovať zadanie tak, že z textu, ktorého znenie mu je zadané, má vytvoriť rovnakú vetu s rozdielom, že vytvorená veta má byť vo forme obrázkov. Obrázky majú za sebou následovať tak, aby tvorili vetu s rovnakým významom. Druhým variantom znenia úlohy je formou obrázkov. V tomto variante majú študenti spraviť opak prvej možnosti. Z vety, ktorá je vyjadrená obrázkami je potrebné vytvoriť textový ekvivalent. To znamená, že musia každému obrázku priradiť korešpondujúce slovo.

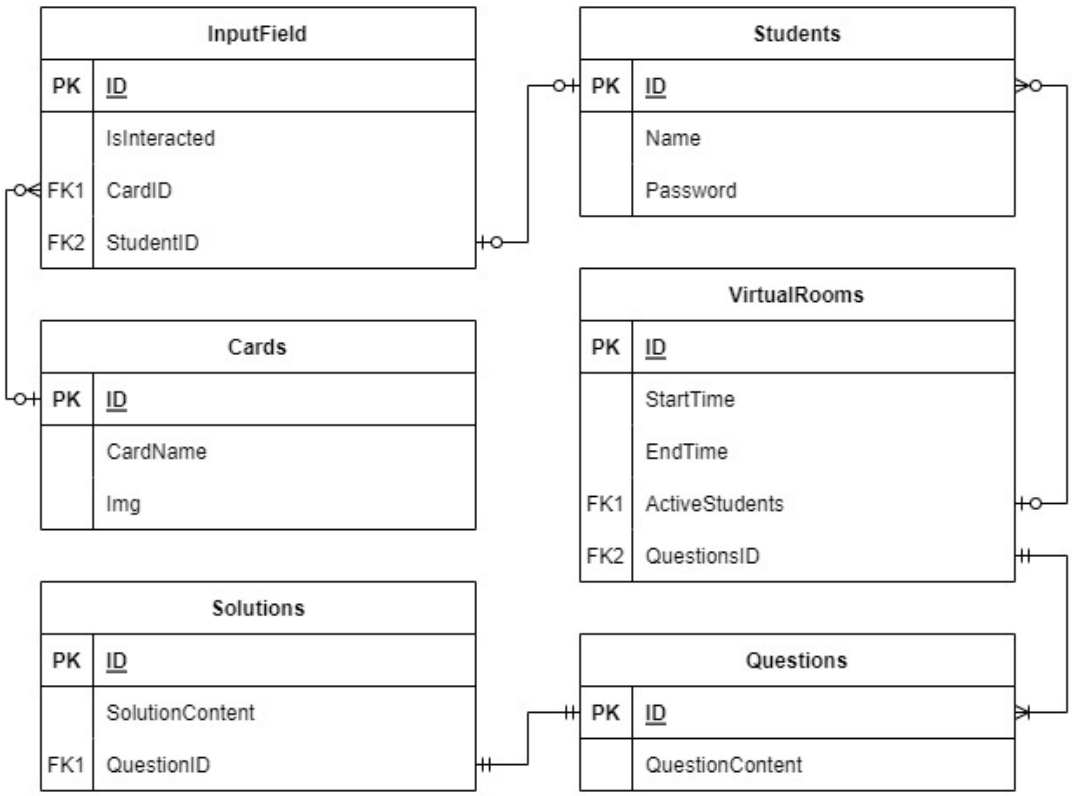
Riešenie

Posledným kompomentom Webového rozhrania je Riešenie. Riešenie sa bude nachádzať pod kompomentom Úloha. V tomto komponente budú študenti, ktorí budú aktívni vo virtuálnej miestnosti, vypracovávať úlohu podľa jej znenia. Ak bude znenie úlohy vo forme textu, študent bude v tomto komponente vyberať z ponuky obrázkov a naopak, ak bude zadanie vo forme obrázkov, študent bude priraďovať z ponuky slov. Ponuka bude obsahovať všetky obrázky, prípadne slová, ktoré sa budú aktuálne v databáze nachádzať.

Pre každé slovo bude vytvorený input field, v ktorom si bude môcť študent vybrať slovo alebo obrázok, podľa toho o aký typ úlohy sa bude jednať. Počas toho, kým bude študent vyberať slovo alebo obrázok v konkrétnom inpute, bude tento input pre ostatných aktívnych študentov nedostupný. Po tom, čo si študent vyberie slovo alebo obrázok z ponuky, bude tento input znova dostupný pre ostatných študentov. Medzi tým môžu ostatní študentí vyberať a priraďovať slová alebo obrázky v iných inputoch, prípadne môžu opraviť odpoveď niekoho iného.

2.3 Databáza

Úlohou databázy bude uchovávať dáta spojené s používaním systému. Databáza sa bude skladať zo siedmich tabuliek. Dáta budú uchované v tabuľkách takým spôsobom, aby prístup k ním bol jednoduchý, dáta boli logicky rozdelené a vzťahy medzi dátami uchované. Diagram tabuliek databázy vytvaraného systému je zobrazený na obrázku pod týmto textom.



Obr. 2.5: Schéma databázy vo forme diagramov

Students

Tabuľka Students bude uchovávať dáta spojené s uživateľom, teda študentom. Bude sa skladať z troch stĺpcov. Prvý stĺpec s názvom ID bude na identifikáciu záznamu. Druhý stĺpec bude uchovávať meno študenta a tretí heslo študenta, pomocou ktorého bude študent pristupovať k svojmu účtu.

VirtualRooms

Tabuľka VirtualRooms bude mať uložené dáta spojené s komponentom virtuálnej miestnosti, ktorá bude označovaná ako Test. Tabuľka sa bude skladať z piatich stĺpcov. Prvý bude ID, druhý bude čas a dátum od kedy bude miestnosť aktívna, tretí s názvom EndTime bude obsahovať čas a dátum ukončenia aktívnosti miestnosti. Stĺpec s názvom ActiveStudents bude obsahovať pole aktívnych študentov vo virtuálnej miestnosti. Posledným stĺpcom s názvom QuestionsID bude taktiež cudzí kľúč, ktorý bude uchovávať pole otázok, ktoré bude virtuálna miestnosť obsahovať.

Questions

Treťou tabuľkou je Questions, ktorej úlohou bude uchovávať znenie otázok, respektíve zadaní. Tabuľka Questions bude mať dva stĺpce, jeden pre ID otázky a druhý pre obsah otázky. Obsah otázky bude JSON formátu a obsahovať bude typ otázky a znenie úlohy. Ak bude znenie úlohy vo formáte textu, v JSON-e bude string, v ktorom bude znenie úlohy. Ak bude znenie úlohy vo forme obrázkov, v JSON-e budú napísané názvy kartičiek s obrázkami.

Solutions

Tabuľka Solutions bude mať tri stĺpce, prvý pre ID riešenia, druhý pre obsah riešenia vo formáte JSON súboru a tretí pre ID otázky, ku ktorej je konkrétna odpoveď priradená. SolutionContent alebo teda obsah riešenia bude koncipovaný podobne ako QuestionContent v tabuľke Questions. Teda, ak bude znenie úlohy vo forme obrázkov, v JSON-e tejto tabuľky bude odpoveď vo forme textu, ktorý bude korešpondovať s obrázkami zadania a ak bude znenie úlohy vo forme textu, v JSON-e budú názvy kartičiek s obrázkami.

Cards

Táto tabuľka bude uchovávať obrázky kartičiek. V prvom stĺpci bude identifikačné číslo, v druhom názov obrázka a v treťom samotný obrázok.

InputField

Posledná tabuľka bude obsahovať štyri stĺpce, prvý bude slúžiť na identifikáciu záznamu. Úlohou druhého stĺpca bude uchovávať informáciu o tom, či niektorý z aktívnych študentov v danom momente konkrétny input používa. Táto informácia bude slúžiť na zablokovanie prístupu ostatných študentov k danému inputu, počas používania inputu niektorým zo študentov. Tretí stĺpec bude uchovávať informáciu o tom, aká karta, prípadne slovo, bolo v danom inpute zvolené. Posledný stĺpec bude uchovávať meno študenta, ktorý s inputom pracoval ako posledný.

3 Implementácia návrhu kolaboratívneho systému

Táto časť práce bude venovaná implementácii systému podľa návrhu spracovania kolaboratívneho systému opísaného v kapitole 2, taktiež bude poukazovať na zmeny, ktoré boli vykonané voči návrhovej časti.

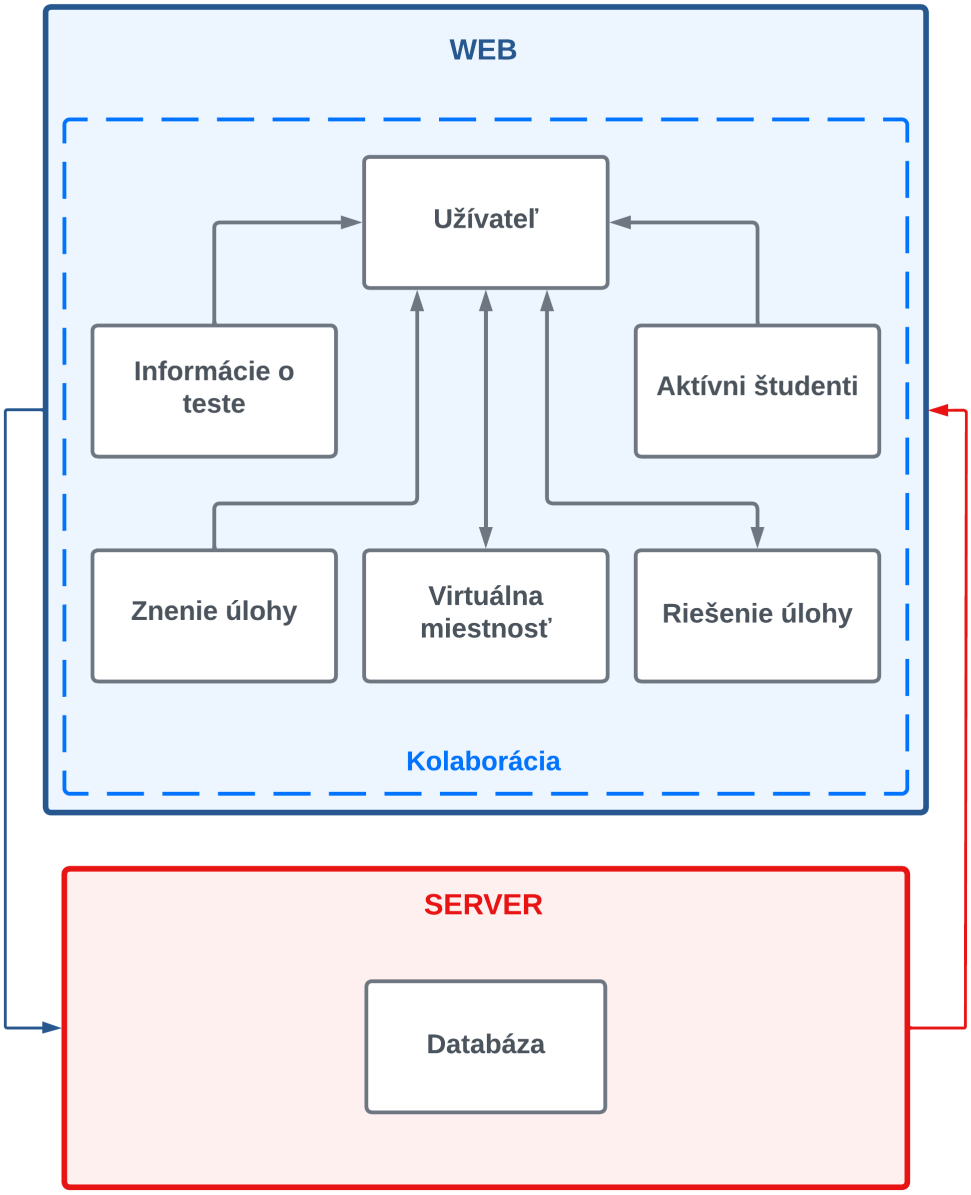
Na začiatku kapitoly bude opísaná všeobecná architektúra systému, následne budú dôkladne rozobrané všetky obrazovky aplikácie a opísané, čo môže používateľ na konkrétnych obrazovkách vykonávať. Nasledovať bude opis implementácie kolaborácie medzi študentmi, ako posledné v kapitole venovanej implementácii budú ukázané zmeny, ktoré nastali v databáze systému.

3.1 Všeobecná architektúra v implementovanom systéme

V návrhu všeobecnej architektúry systému v kapitole 2.1 bol systém rozdelený na webový kompoment a komponent servera. Toto rozdelenie zostalo zachované, zmenil sa ale obsah týchto komponentov.

Vo webovej časti aplikácie pribudli komponenty Informácie o teste a Aktívni študenti, zmenili sa komponenty Riešenie na Riešenie úlohy a Úloha na Znenie úlohy. Taktiež sa zmenili vzťahy medzi nimi, keďže sa zmenil typ komunikácie medzi jednotlivými komponentmi.

V serverovom komponente, ktorý v návrhu obsahoval len funkcionalitu potrebnú na komunikáciu s databázou, nastala jedna zmena, pribudla komunikácia pomocou WebSocketu, ktorá zabezpečuje opitmalizovanejšiu komunikáciu medzi serverom a klientom. Všeobecná architektúra po implementácii je zobrazená na nasledujúcom obrázku.



Obr. 3.1: Všeobecná systémová architektúra po implementovaní systému

Na obrázku je možné vidieť komponenty, ktoré pribudli alebo v nich nastala zmena. Komunikácia medzi webom a serverom zostala zachovaná. Pribudnuté komponenty informácie o teste a aktívni študenti majú s užívateľom jednosmernú komunikáciu, keďže ich užívateľ nemá možnosť meniť. Rovnako jednosmerná komunikácia je medzi komponentom znenie úlohy a užívateľom. Prepojenie medzi užívateľom a riešením úlohy je obojsmerné.

3.2 Implementácia webového rozhrania

V tejto sekcii bude opísané, ako bolo implementované webové rozhranie. Oproti návrhu došlo vo webovom rozhraní k niekoľkým zmenám. Tieto zmeny budú podrobnejšie opísané v nasledujúcich častiach tejto sekcie.

Prvá zmena, ktorá nastala je pridanie a zmena komponentov, ako bolo napísané v predchádzajúcej sekcii. Po implementácii je možné rozdeliť komponent webového rozhrania na šesť častí:

- Užívateľ,
- virtuálna miestnosť,
- informácie o teste,
- aktívni študenti,
- znenie úlohy,
- riešenie úlohy.

Vo webovom rozhraní sú štyri URL adresy, ktoré sú užívateľovi prístupne. Url adresy, ktoré sa v systéme nachádzajú je možné rozdeliť na dve častí.

Prvá sú nezabezpečené url adresy pre prihlásenie a registráciu, ktoré sú používateľovi prístupne aj bez overenia.

Druhá časť sú zabezpečené url adresy pre domovskú časť systému, kde sa nachádzajú virtuálne miestností a url adresy pre konkrétne virtuálne miestností. Zabezpečené url adresy nie sú dostupné užívateľovi, ktorý nebol overený prihlásením.

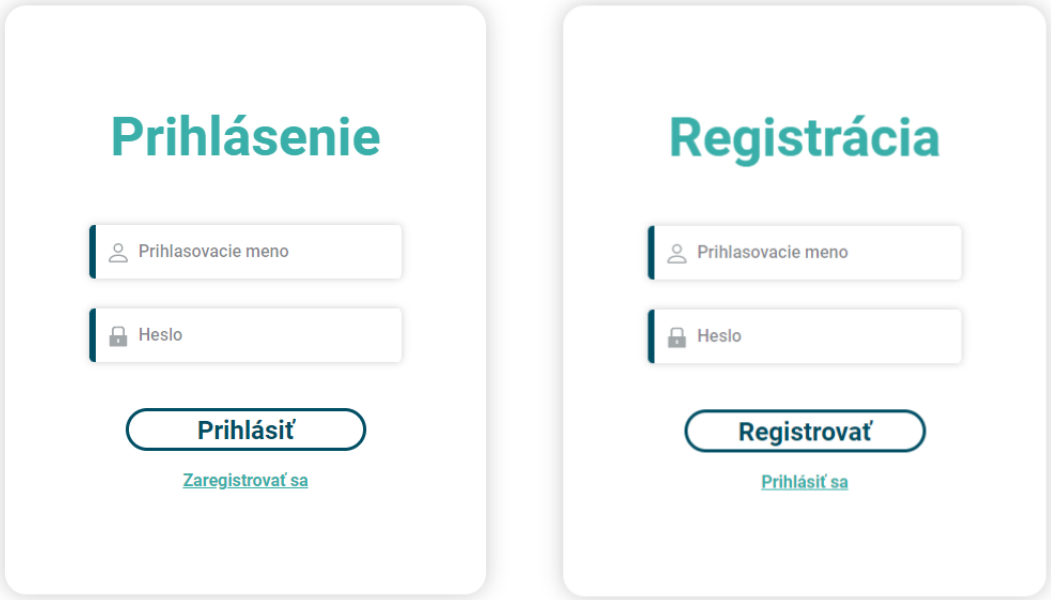
Ak používateľ aplikácie zadá url adresu, ktorá v systéme nie je evidovaná, zobrazí sa mu obrazovka, kde mu je oznámené, že zadaná url adresa neexistuje. Po uplynutí troch sekúnd je používateľ automaticky presmerovaný na url adresu, kde sa môže prihlásiť.

Na prvej obrazovke po načítaní stránky užívateľ bude môcť vidieť formulár prihlásenia. Formulár bude označený nadpisom Prihlásenie, pod ním sa nachádzajú polia pre vstup od používateľa, kde môže zadať prihlasovacie meno a heslo. Pod nimi sa nachádza tlačidlo, ktoré zadané údaje odošle na server, kde sa údaje porovnajú s údajmi existujúcimi v databáze. Ak sa údaje nezhodujú, používateľovi sa vo formulári zobrazí hláška, ktorá oznámi, že zadané údaje nie sú správne. Používateľ, ktorý ešte nie je v systéme evidovaný sa nemôže prihlásiť.

Ak by chcel svoje údaje pridať do databázy, môže využiť odkaz s názvom registrovať sa, ktorý sa nachádza v spodnej časti formulára na prihlásenie.

Po kliknutí na odkaz je používateľ presmerovaný na url adresu určenú pre registráciu, kde je používateľovi zobrazený formulár na registráciu, ktorý sa od formulára na prihlásenie líši len v nápisoch. Po vyplnení formulára a kliknutí na tlačidlo registrovať sa, sú údaje odoslané na server, kde sú porovnané s údajmi v databáze. Ak databáza obsahuje zadané meno, užívateľovi je vypísané upozornenie o tom, že zadané meno je obsadené, v tomto prípade je nutné zvoliť iné meno. Ak sa meno v databáze nenachádza, užívateľ vidí hlášku o tom, že registrácia bola úspešná a zároveň je vyzvaný na prihlásenie do systému, čo môže spraviť kliknutím na odkaz s nápisom prihlásiť sa, po ktorom bude presmerovaný na url adresu pre prihlásenie.

Po úspešnom prihlásení používateľovi v prehliadači pribudne cookie s názvom a jedinečnou hodnotou. Tým sa vytvorí pre prihlaseného používateľa jedinečná relácia, ktorá zabezpečí, aby používateľ zostal prihlasený pokiaľ sa sám neodhlási. Formuláre na prihlásenie aj registráciu sú zobrazené na obrazku nižšie.



Obr. 3.2: Vľavo formulár na prihlásenie, vpravo formulár na registráciu

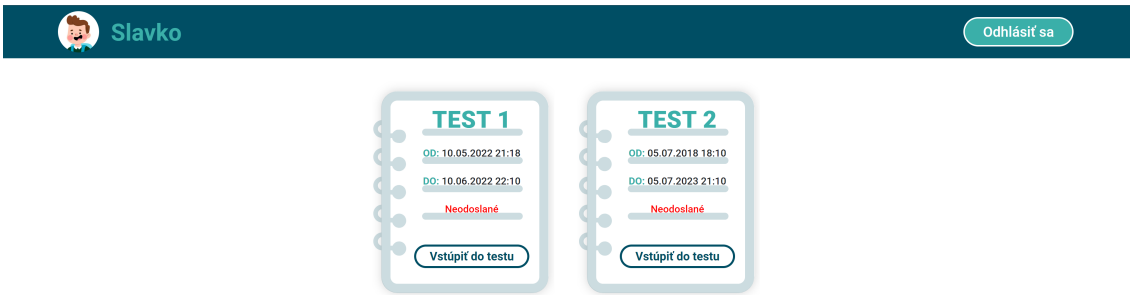
Po prihlásení do systému je užívateľ presmerovaný na domovskú obrazovku, v ktorej vidí karty. Každá karta predstavuje test z databázy s istou logikou, ktorá otvorí novú kolaboratívnu virtuálnu miestnosť.

V hornej časti obrazovky sa nachádza horná lišta. V ľavej časti lišty je zobrazený avatar a meno používateľa. Na opačnej strane lišty sa nachádza tlačidlo odhlásiť sa, ktoré po kliknutí zruší vytvorenú reláciu a presmeruje používateľa na obrazovku prihlásenia.

V strede domovskej obrazovky sa nachádza zoznam jednotlivých virtuálnych miestností. V zázname konkrétnej virtuálnej miestnosti sa nachádza jej názov, pod názvom sú umiestnené dva časy a dátumy, prvý vyjadruje od kedy je miestnosť aktívna a druhý do kedy je miestnosť aktívna.

Pod nimi sa nachádza informácia o tom, či už boli otázky v danej miestnosti vypracované. Ak otázky boli vypracované, používateľ môže prejsť kurzorom na túto kolónku, kedy sa mu zobrazí modal so zoznamom žiakov, ktorí sa na vypracovaní testu podieľali. Táto časť bola pozmenená oproti návrhu, keďže v návrhu bol zobrazený len čas, ktorý vyjadroval, ako dlho bude konkrétna miestnosť aktívna, bolo potrebné pridať aj čas začiatku aktivity. Taktiež bolo odstránené hodnotenie testov jednotlivých užívateľov, keďže užívatelia budú môcť odoslať svoje riešenie zadania až po tom, čo bude vyhodnotené ako správne.

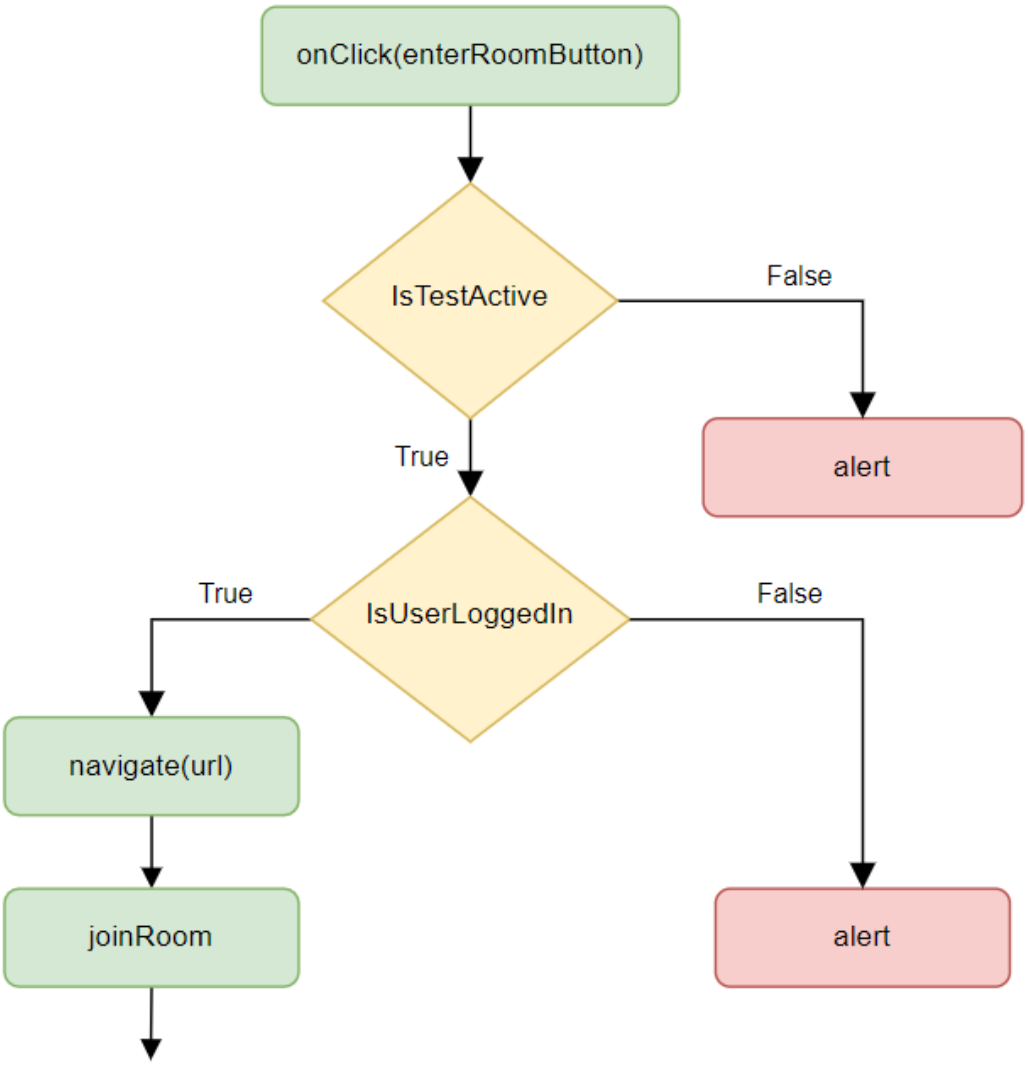
Posledný prvok, ktorý sa nachádza v zázname virtuálnej miestnosti je tlačidlo, ktoré umožňuje vstúpiť do miestnosti. Vstup je povolený len vtedy, ak je miestnosť aktívna, alebo ak ešte neboli v miestnosti vypracované otázky. Domovská obrazovka je zobrazená na obrázku nižšie.



Obr. 3.3: Príklad domovskej obrazovky

Pripojenie do virtuálnej miestnosti

Ak sa chce používateľ pripojiť do miestnosti, má možnosť tak urobiť kliknutím na tlačidlo miestnosti, do ktorej chce vstúpiť. Po kliknutí na tlačidlo je spustený algoritmus, ktorého diagram je zobrazený na nasledujúcom obrázku.

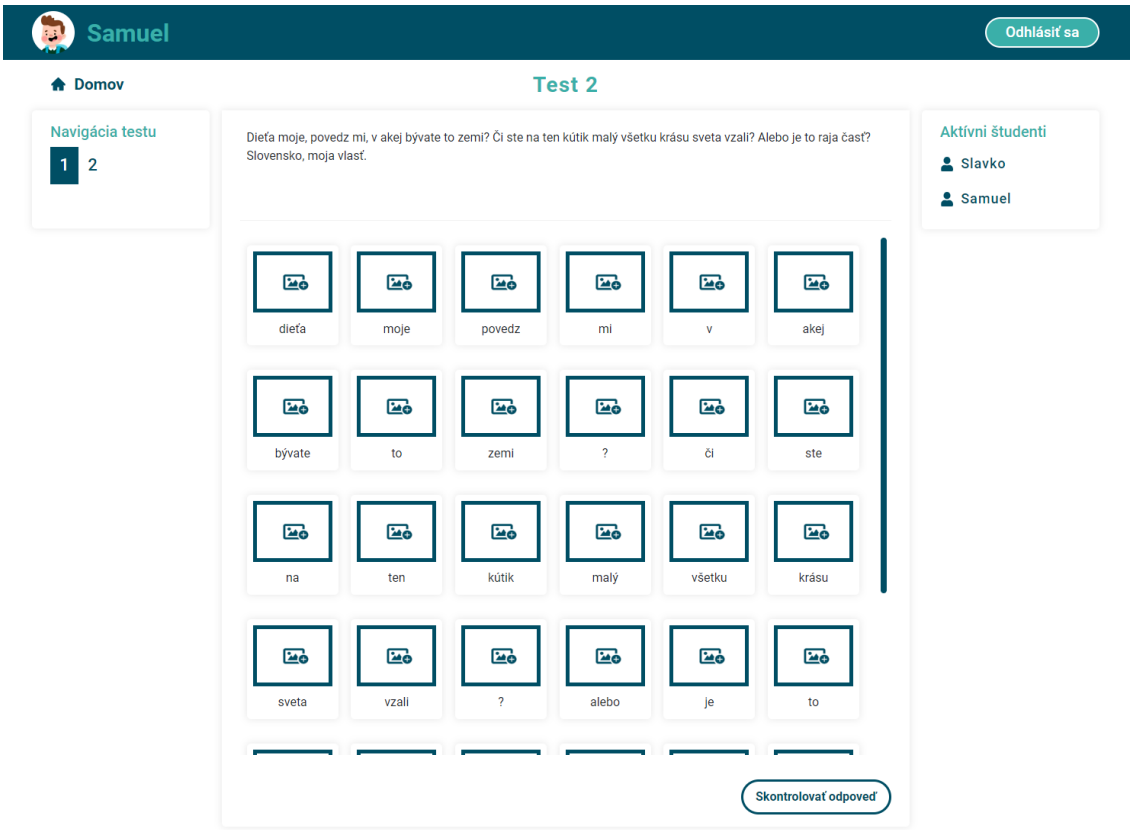


Obr. 3.4: Algoritmus pripojenia používateľa do miestnosti

Diagram algoritmu pripojenia do miestnosti začína kliknutím, po ktorom nasleduje podmienka, v ktorej je zisťované, či je aktuálny čas v rozmedzi času, kedy aktivita miestnosti začína a času, kedy je aktivita miestnosti ukončená. Ak je táto podmienka vyhodnotená ako nepravdivá, teda aktuálny čas nie je v rozmedzi týchto časov, používateľovi je oznámené, že miestnosť, do ktorej sa snaží vstúpiť je aktuálne neaktívna. Ak je podmienka splnená, program zisťuje, či je používateľ prihlásený, keďže platnosť jeho prihlásenia mohla vypršať. Ak používateľ prihlásený nie je, zobrazí sa mu upozornenie, v ktorom je vyzvaný, aby sa prihlásil.

Ak je používateľ prihlásený, podmienka je vyhodnotená kladne. Po kladnom vyhodnotení podmienky je používateľ presmerovaný na url adresu konkrétnej virtuálnej miestnosti. Keďže miestnosti môžu pribúdať, url adresa je v tomto prípade dynamicky doplňovaná tak, že za url adresu virtuálnej miestnosti je doplnené jej identifikačné číslo, pod ktorým je uložená v databáze. Posledná funkcia, ktorá je spustená po presmerovaní používateľa do miestnosti zabezpečí prostredníctvom websocketov odoslanie informácie ostatným používateľom nachádzajúcim sa v miestnosti o tom, že pribudol nový používateľ.

Za prepokladu, že všetky podmienky boli splnené, používateľ úspešne vstúpil do virtuálnej miestnosti. Ak sa už v miestnosti nachádzajú aktivní používatelia, ktorí vykonali nejaké zmeny v miestnosti, tieto zmeny sa zobrazia novopripojenému používateľovi. Teda, ak už stihli používatelia prejsť na ďalšiu otázku a v nej vyplniť odpovede, novopripojenému užívateľovi sa zobrazí aktuálna otázka, na ktorej sa ostatní používatelia nachádzajú a taktiež používateľmi vyplnené odpovede. Príklad obrazovky, ktorú má používateľ možnosť vidieť po pripojení do virtuálnej miestnosti je zobrazený na nasledujúcom obrázku.



Obr. 3.5: Príklad obrazovky virtuálnej miestnosti

Po vstupe do virtuálnej miestnosti používateľ vidí vo vrchnej časti obrazovky hornú lištu, na ktorej sú zobrazené rovnaké prvky, ako to bolo na domovskej obrazovke, teda avatar, meno používateľa a tlačidlo odhlásiť sa.

Pod hornou lištou sa nachádzajú dva prvky. Vľavo tlačidlo, ktoré po kliknutí vyvolá akciu presmerovania používateľa na domovskú obrazovku. V strede je umiestnený nadpis, ktorý používateľa informuje o tom, v ktorej miestnosti sa aktuálne nachádza.

Pod týmto nadpisom sú umiestnené štyri hlavné komponenty webového rozhrania, informácie o teste, aktívni študenti, znenie úlohy a riešenie úlohy. Keďže je virtuálna miestnosť vytvorená tak, aby bolo umožnené používateľom spolupracovať na vypracovaní úloh, tieto štyri komponenty sa zobrazujú všetkým používateľom rovnako.

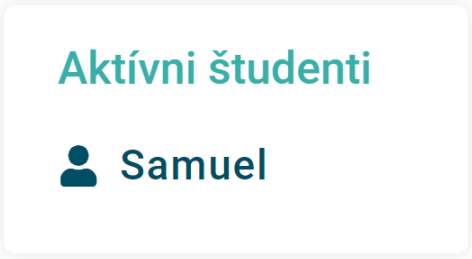
Informácie o teste

V komponente informácie o teste sú umiestnené dva prvky. Prvý je nadpis, ktorý tento komponent označuje ako navigáciu testu a druhý je zoznam. Zoznam je naplnený číslami otázok, ktoré virtuálna miestnosť obsahuje. Číslo otázky, ktorá je aktuálne zobrazovaná používateľom je zvýraznené odtieňom modrej farby. Ak sa používatelia rozhodnú prejsť na nasledujúcu otázku, zmení sa stav komponentu, ktorý označuje aktuálnu otázku, čím vyznačí modrou farbou ďalšie číslo otázky. Na nasledujúcom obrázku je možné vidieť príklad komponentu informácie o teste.

Obr. 3.6: Príklad komponentu informácie o teste

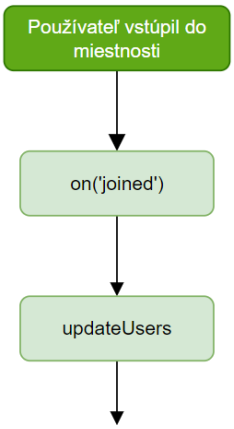
Aktívni študenti

Tento komponent sa skladá z nadpisu a zoznamu mien. Zoznam obsahuje mená prihlásených ľudí, ktorí sú zároveň pripojení v danej miestnosti. V momente, keď sa do miestnosti pripojí nový používateľ, v zozname pribudne nový prvok s jeho menom. Príklad komponentu so zoznamom aktívnych študentov je zobrazený na nadchádzajúcom obrázku.



Obr. 3.7: Príklad komponentu aktívni študenti

Po pripojení nového používateľa do virtuálnej miestnosti sa vykoná algoritmus znázornený na nasledujúcom diagrame.



Obr. 3.8: Diagram algoritmu pridania nového používateľa

Pripojením nového používateľa do miestnosti je spustená funkcia websocketu, ktorá pošle na server meno používateľa, ktorý sa pripojil spolu s identifikačným číslom miestnosti. Server po prijatí tejto informácie spustí funkciu, ktorá zabezpečí pridanie novopripojeného používateľa do zoznamu k ostatným, k menu pridá aj číslo miestnosti. Zmenený zoznam mien pošle späť na klientskú časť aplikácie tým používateľom, ktorí sa nachádzajú v rovnakej miestnosti. Aktualizovaný zoznam je pridaný do komponentu aktívni študenti.

Znenie úlohy

Tento komponent v sebe uchováva znenie úlohy vo forme textu. Komponent sa zobrazuje len v prípade, ak je typ otázky z textu na piktogramy. V tomto prípade bolo úmyslom užívateľovi ukázať plné znenie úlohy. V opačnom prípade, čiže ak je typ otázky z piktogramov na text, komponent zobrazovaný nie je, keďže by jeho zobrazenie nedávalo zmysel a bolo by viac chaotické, ako užitočné. Príklad vizuálneho zobrazenia znenia úlohy je možné vidieť v nasledujúcom obrázku.

Obr. 3.9: Príklad komponentu znenia úlohy

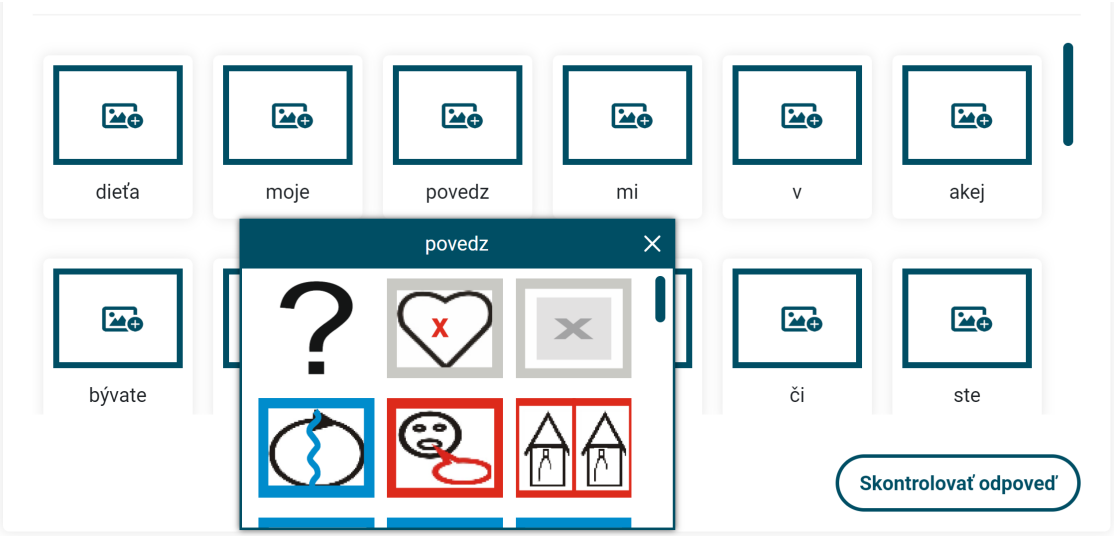
3.3 Riešenie úlohy

Riešenie úlohy je hlavným komponentom, v ktorom prebieha najpodstatnejšia kolaborácia medzi aktívnymi študentmi v miestnosti. Komponent sa skladá z blokov. Počet blokov je definovaný počtom slov v znení úlohy. Riešenie úlohy môže mať dve podoby.

Preklad úlohy z obrázkov na text

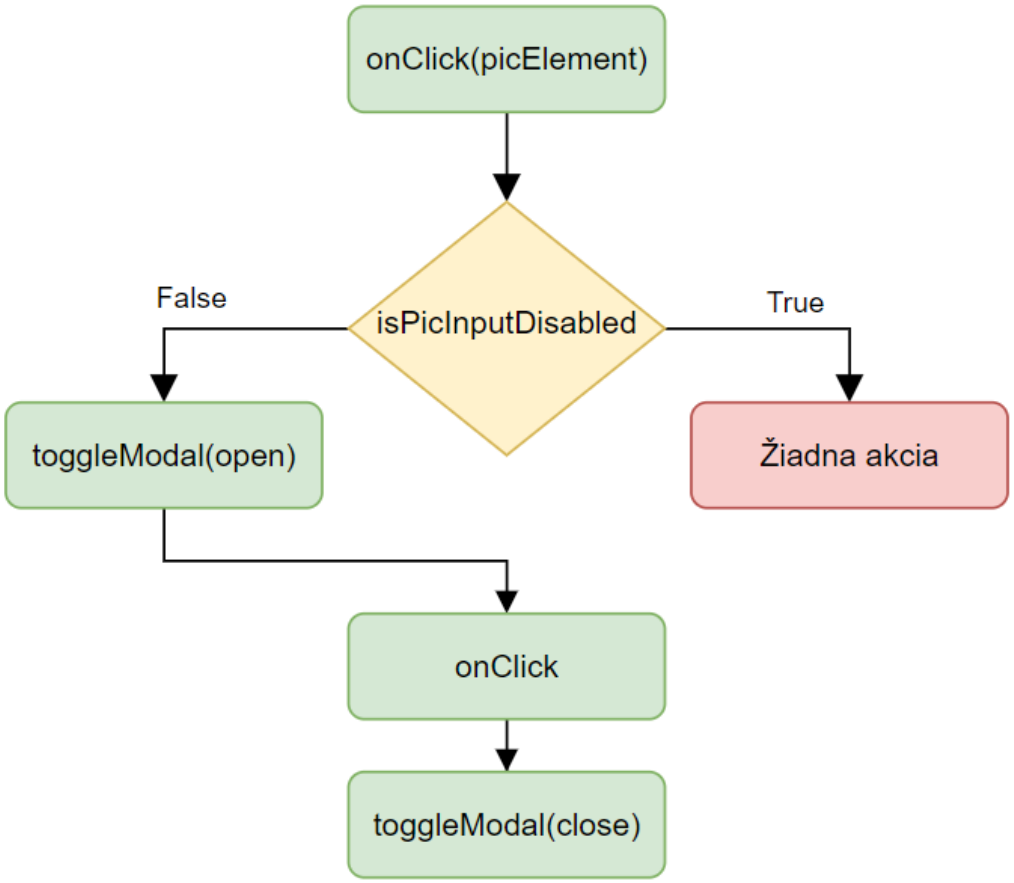
Prvá možnosť je tá, že v spodnej časti bloku sa nachádza text. Vo vrchnej časti bloku sa nachádza obrázok, ktorý indikuje študentovi, aby naňho klikol. Po kliknutí na obrázok je zobrazený modal, ktorý obsahuje všetky obrázky, ktoré sa v danej úlohe nachádzajú. Študent môže na jeden z týchto obrázkov kliknúť a tým ho priradiť bloku, z ktorého bol modal otvorený. Vybraný obrázok je následne zobrazený v bloku. Ak ho niektorý zo študentov chce zmeniť, môže tak urobiť opätovným kliknutím na obrázok a vybratím iného obrázka z modalu. [14]

Ak jeden z používateľov klikol na obrázok a má zobrazený modal, ostaným študentom je tento obrázok zobrazený sivou farbou a kurzor na obrázku sa zmení tak, aby používateľovi dal najavo, že obrázok, na ktorý sa snaží kliknúť používa niekto iný. Príklad komponentu riešenia úlohy z textu na piktogramy s otvoreným modalom je zobrazený na nasledujúcom obrázku.



Obr. 3.10: Príklad komponentu riešenia úlohy z textu na piktogramy

Pre pridávanie piktogramov do blokov platí nasledujúca postupnosť krokov:



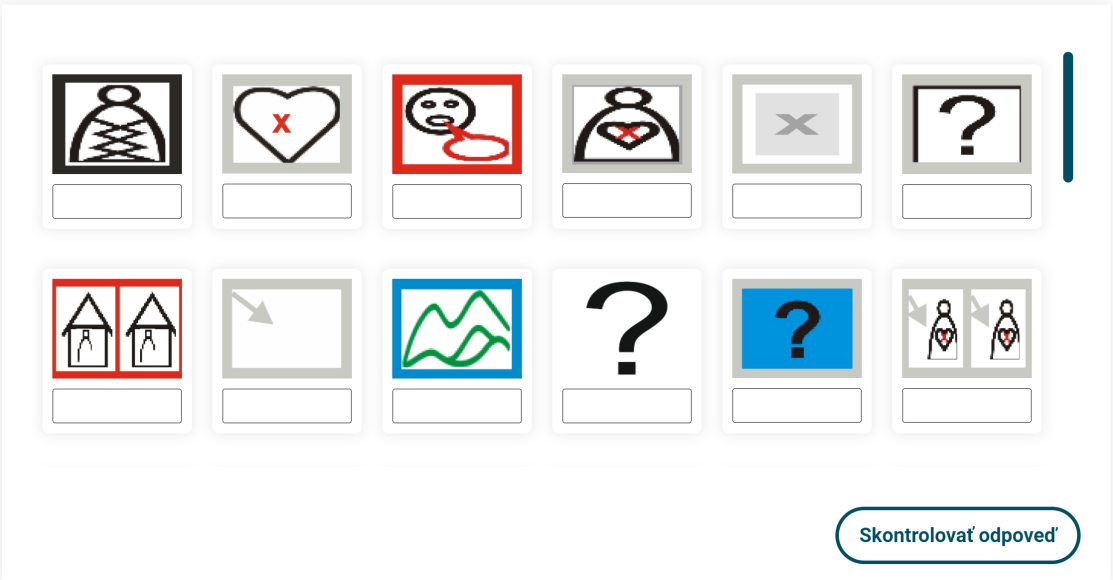
Obr. 3.11: Diagram algoritmu pridávania obrázkov do blokov

V diagrame začína algoritmus kliknutím používateľa na obrázok. Ak je obrázok v znefunkčnenom stave, znamená to, že ho aktuálne používa iný používateľ v miestnosti. Kliknutie naň v tomto prípade nevyvolá žiadnú akciu. Ak je obrázok v dostupnom stave, kliknutím je spustená funkcia toggleModal, ktorá zabezpečí otvorenie modalu. Ihneď po otvorení modalu je na server odoslaná informácia o tom, že používateľ klikol na daný obrázok. Server túto informáciu prepošle všetkým aktívnym študentom v danej miestnosti. Po prijatí tejto informácie sa ostatným používateľom obrázok znefunkční.

Algoritmus pokračuje po kliknutí používateľa na jeden z piktogramov v modali. Vybraný piktogram je pridaný do zoznamu piktogramov, ktoré sú zobrazované v blokoch. Zároveň je zatvorený modal a opätovne odoslaná informácia ostatným aktívnym študentom o tom, že je modal opäť k dispozícii po kliknutí na obrázok, taktiež je odoslaný nový zoznam piktogramov, aby prebehla zmena obrázka v bloku aj ostatným študentom.

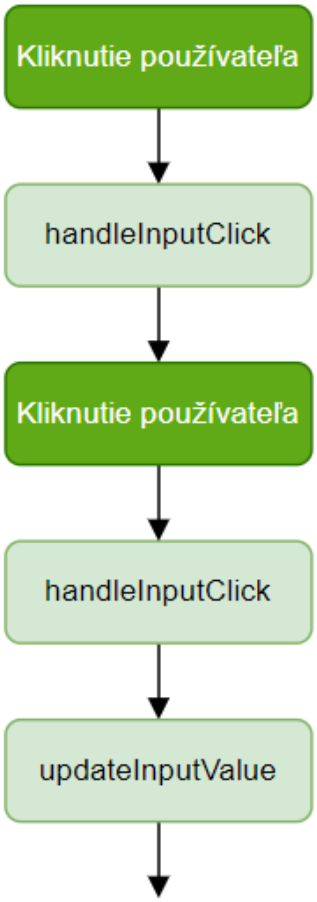
Preklad úlohy z obrázkov na text

Druhý typ úlohy je priradenie slov piktogramom. V tomto prípade sa tiež v riešení úlohy nachádza blok. Na rozdiel od prvého typu úlohy sa vo vrchnej časti bloku nachádza napevno obrázok zo znenia úlohy a v spodnej časti bloku je očakávaný vstup používateľa, kde má napísať slovo, ktoré bude priradené piktogramu. [14] Príklad komponentu riešenia úlohy z piktogramu na text je možné vidieť na nasledujúcom obrázku.



Obr. 3.12: Príklad komponentu riešenia úlohy z piktogramu na text

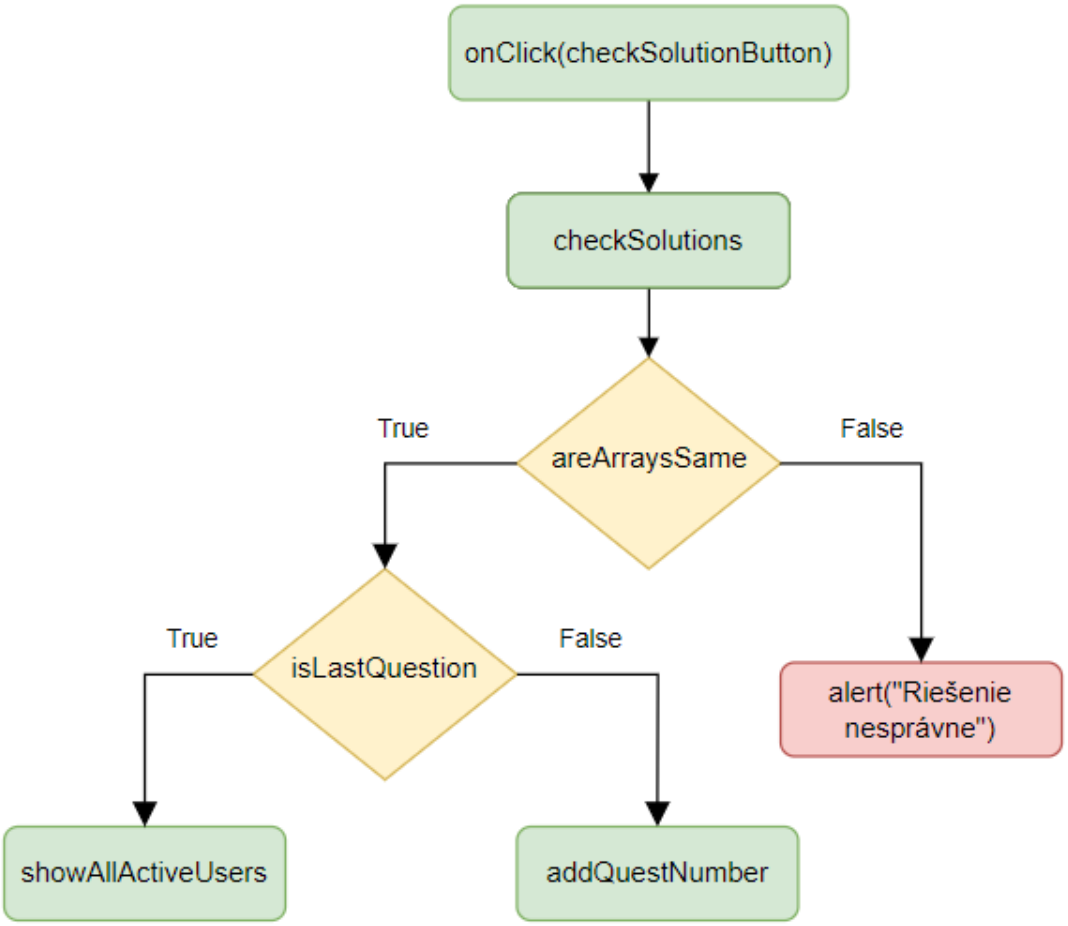
Priradenie textu piktogramom je zabezpečené touto postupnosťou krokov:



Obr. 3.13: Diagram algoritmu aktualizovania obsahu inputov

V diagrame je možné vidieť, že algoritmus začína kliknutím používateľa na input, do ktorého chce písať. Po kliknutí sú pomocou websocketu odoslané informácie o tom, že input bol niektorým zo študentov zakliknutý. V odoslaných informáciách je zahrnuté id inputu, id používateľa, ktorý klikol na input a miestnosť, v ktorej sa nachádzajú. Server po prijatí týchto informácií odošle správu ostatným aktívnym študentom v miestnosti. Študentom, ktorí správu prijali je input zablokovaný, aby nedošlo k vzájomnému prepisovaniu hodnôt inputov v rovnakom čase. Po každej zmene hodnoty v inpute sa prepíše pole, v ktorom sú hodnoty inputov uchované. Po ukončení interakcie používateľa s inputom je opätovným kliknutím používateľa, tentokrát mimo inputu opäť zavolaná funkcia handleInputClick, ktorá naspäť sprístupní input ostaným používateľom. Po nej je spustená funkcia updateInputValue, ktorá odošle na server zmenené pole s hodnotami v inputoch. Server túto informáciu odošle ostatným študentom v miestnosti, ktorým sa hodnoty v inputoch aktualizujú.

Ak sú študenti spokojní s riešením zadania, môžu si ho skontrolovať pomocou tlačidla umiestneného v spodnej časti komponentu riešenia úlohy. Kliknutím na tlačidlo je spustená kontrola riešenia, ktorá prebieha v tejto postupnosti krokov:



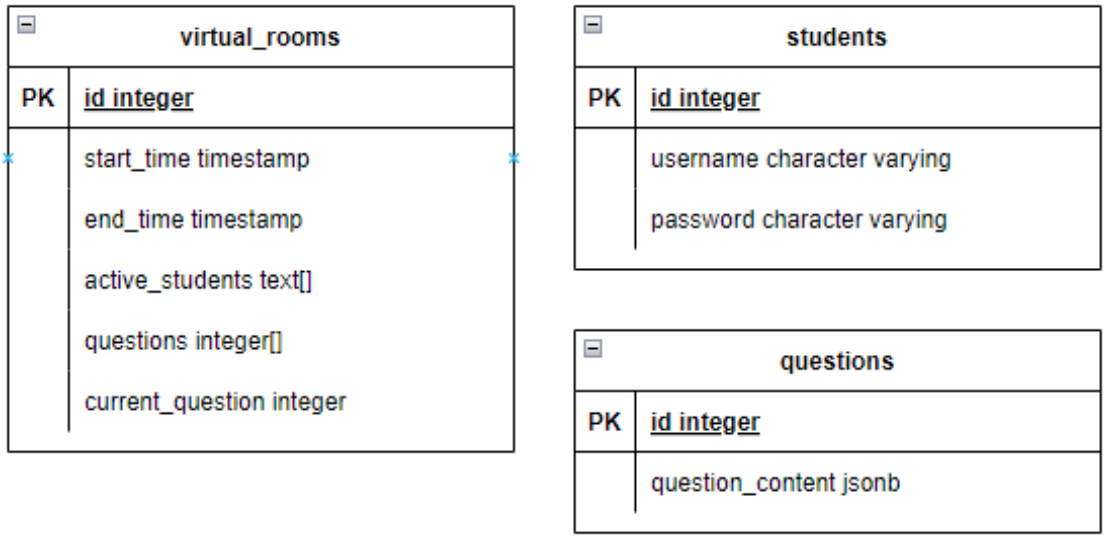
Obr. 3.14: Diagram algoritmu kontroly riešenia

Algoritmus začína kliknutím používateľa na tlačidlo skontrolovať riešenie. Kliknutie spustí funkciu checkSolutions, ktorá kontroluje, či je riešenie v inputoch zhodné so správnym riešením úlohy. Ak je riešenie vyhodnotené ako nesprávne, študenti vidia upozornenie, ktoré im oznámi, aby svoje riešenie opravili. Po správnom vyhodnotení riešenia nasleduje podmienka, ktorá porovná, či je kontrolovaná otázka posledná. Ak nie je, študentom sa zobrazí oznam o správnosti riešenia, po ktorom nasleduje zobrazenie nasledujúcej otázky. Ak je kontrolovaná otázka posledná, študentom je zobrazený modal, v ktorom je zoznam všetkých študentov podieľajúcich sa na riešení testu vo virtuálnej miestnosti. Po zavretí modalu sú študenti presmerovaní na domovskú obrazovku.

3.4 Databáza

Databáza je oproti návrhu o tri tabuľky menšia. Keďže boli použité websockety, nebolo potrebné vytvárať tabuľku InputField, ako bolo pôvodne plánované v návrhu. Toto riešenie prinieslo optimálnejšiu komunikáciu medzi klientom a serverom, keďže aktuálne prebieha komunikácia len vtedy, keď nastane zmena v inpute.

Ďalšou zmenou je odstránenie tabuľky Cards. K tejto zmene došlo najmä kvôli malej veľkosti obrázkov, ktoré sú aktuálne uložené lokálne v projekte. Tretia tabuľka Solutions bola odstránená kvôli rozšíreniu tabuľky Questions o správnu odpoveď na otázku. Aktuálna schéma databázy je zobrazená na nasledujúcom obrázku.



Obr. 3.15: Schéma databázy vo forme diagramov po implementácií

Virtual rooms

V tabuľke, ktorá slúži na uchovávanie dát spojených s virtuálnymi miestnosťami pribudol stĺpec s názvom `current_question`, ktorý uchováva aktuálnu otázku zobrazovanú vo virtuálnej miestnosti. Pridanie tohto stĺpca bolo vyžadované potrebou zobrazovania rovnakej otázky všetkým používateľom v miestnosti. Stĺpce `start_time` a `end_time` sa oproti návrhu nezmenili, obsahujú časy, kedy začína a končí aktivita miestnosti. Takisto sa nezmenil stĺpec questions, ktorého záznamy vyjadrujú čísla otázok nachádzajúcich sa v miestnosti.

Stĺpec `active_students` bol zachovaný, ale jeho význam bol zmenený, keďže už neuchováva zoznam aktuálne aktívnych študentov, ale zoznam všetkých študentov, ktorí boli aktívni v miestnosti.

Students

V tejto tabuľke nenastali žiadne zmeny. Tabuľka obsahuje id, meno používateľa v stĺpci username a heslo používateľa v zahešovanej forme v stĺpci password.

Questions

Táto tabuľka obsahuje dva stĺpce, jeden pre identifikáciu záznamu a druhý na uloženie údajov potrebných pre znenie a správne riešenie úlohy vo formáte JSON. Spôsob umiestnenia dát v zázname typu JSON:

```
{
  "type": true,
  "question": "povedz mi",
  "content": [
    {
        "pic": "povedz",
        "text": "povedz"
    },
    {
        "pic": "mi",
        "text": "mi"
    }
  ]
}
```

Prvé kľúčové slovo *type* označuje typ otázky, ak je v ňom priradená hodnota true, úlohou študenta je priradiť textu piktogramy. Naopak, ak je false, úlohou je priradiť piktogramom správne slová. Záznam *question* obsahuje celé znenie úlohy uložené v stringu. Následuje pole s kľučovým označením *content*, ktoré uchováva priradené správne hodnoty textu a názvy obrázkov.

4 Vyhodnotenie

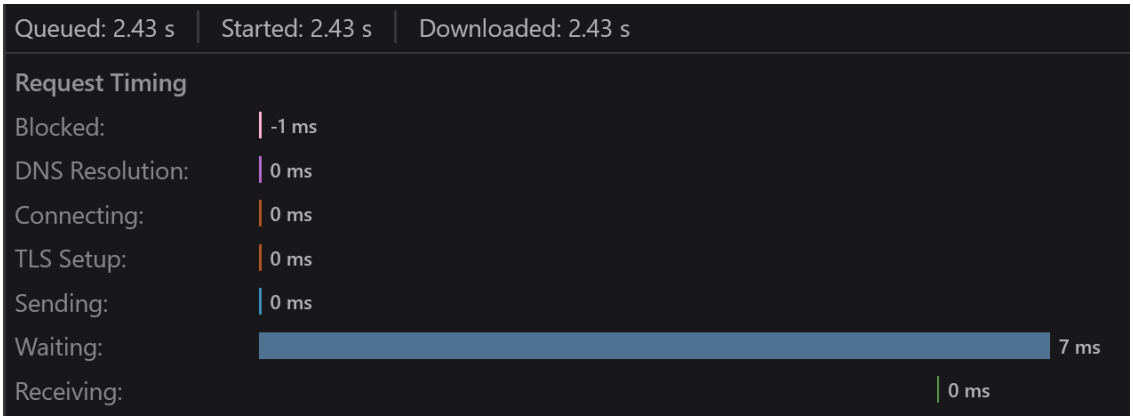
Táto kapitola sa bude venovať niekoľkým záťažovým testom, ktoré umožnia zistiť prípadné slabiny aplikácie. V ďalšej sekcii tejto kapitoly bude overenie funkčnosti kolaborácie medzi používateľmi.

4.1 Záťažové testy

Systém je vytvorený na báze webových technológií. Jedna zo slabín webových aplikácií je to, že čas, za ktorý sú načítané je úzko spätý s rýchlosťou pripojenia na internet používateľa. Z tohto dôvodu je potrebné otestovať aplikáciu takým spôsobom, ktorý odhalí nedostatky aplikácie z hľadiska rýchlostí načítavania dát, ktoré je potrebné zobraziť.

Načítavanie väčšieho množstva kariet

Pre simuláciu reálneho používania aplikácie bolo vytvorených niekoľko dalších virtuálnych miestností. Keďže pre každú miestnosť je dynamický vytvorená karta, ktorá predstavuje konkrétnu virtuálnu miestnosť, na obrazovke sa zobrazuje 22 kariet. Každá z týchto kariet obsahuje obrázok, ktorý je potrebné načítať. Na obrázku nižšie je možné vidieť odozvu načítavania týchto obrázkov.



Obr. 4.1: Znázornenie času načítavania obrázkov

Hodnota zobrazená na obrázku predstavuje čas, za ktorý bol obrázok načítaný. Čas načítania je zanedbateľný, avšak hodnota času, v akom sa začal načítavať je vyššia. Táto hodnota sa prejavuje vizuálne, keďže dáta, ktoré sú umiestnené v karte sú už zobrazené a postupne načítavajú obrázky, ktoré sú pozadím karty. Je teda predpokladom, že väčší počet kariet spôsobí dĺhšie načítavanie a teda zhorší používateľský zážitok z aplikácie.

Načítanie väčšieho množstva piktogramov

Keďže sú obrázky použité aj vo virtuálnych miestnostiach, boli testované rovnakým princípom, teda bola vytvorená úloha, ktorá obsahovala veľký počet slov. Pri väčšom počte slov je nutné zobraziť viac obrázkov, ktorých načítanie môže trvať dlhšie. Trvanie načítania obrázkov je možné vidieť na nasledujúcom obrázku.

boy.36d7c52a8005c49281bd.jpg	jpeg	cached	32.13 kB	0 ms
moj.4f37f4c64b918ef6ed9f.png	png	10.16 kB	9.83 kB	1 ms
zemi.9322d20113c5995cbe0d.png	png	10.93 kB	10.60 kB	2 ms
maly.7623c2db826c3892fb48.png	png	17.74 kB	17.41 kB	2 ms
krasu.6331a5cf0ecd97d17e8f.png	png	12.53 kB	12.20 kB	2 ms
sveta.49ad6d317fce99a9fd5a.png	png	12.87 kB	12.54 kB	2 ms
alebo.6d32ebc97baf4dd7acb6.png	png	13.22 kB	12.89 kB	2 ms
cast.c9a65b812268e94e347e.png	png	12.44 kB	12.11 kB	1 ms
slovensko.7001edcc0547f0eac67c.png	png	10.81 kB	10.48 kB	1 ms
vlast.d81a6bd4499f7b0392db.png	png	10.86 kB	10.53 kB	2 ms
david.6daf3258175813251ca3.png	png	19.81 kB	19.48 kB	2 ms
dedkovia.ba365b0b909e6ee4869d.png	png	11.91 kB	11.58 kB	3 ms
den.425d1516b809c9e3180f.png	png	18.51 kB	18.18 kB	3 ms
hanbit.8dbb03d30309afdfba57.png	png	19.25 kB	18.92 kB	5 ms
hladat.c1ad7e9f138bd0b347f5.png	png	32.44 kB	32.11 kB	8 ms
cely.011ca6aa0001ace252c4.png	png	16.31 kB	15.98 kB	8 ms
daleko.66454ab3991c9c517678.png	png	16.52 kB	16.19 kB	14 ms
dievca.db8c035916241d4cb487.png	png	19.01 kB	18.68 kB	10 ms
alzbeta.5bb41f91fd0f08e5f063.png	png	19.40 kB	19.07 kB	7 ms
dostat.de8a6a9f4663d7bf4db8.png	png	22.54 kB	22.21 kB	17 ms
dobroprajny.ed839232ed672ba805ea.png	png	19.82 kB	19.49 kB	17 ms
dlh.62a5714bebd2edefa136.png	png	20.28 kB	19.95 kB	17 ms
dnes.09bb02999f6e1c7ec508.png	png	12.90 kB	12.57 kB	16 ms
burka.e9a4ec195d29d81b5e8e.png	png	28.88 kB	28.54 kB	14 ms
celkom.54ea22b0311554055fbb.png	png	16.41 kB	16.08 kB	10 ms
hladni.e284a07c68c890394e83.png	png	15.26 kB	14.93 kB	9 ms
auto.25f21cd990f25062d81f.png	png	16.96 kB	16.63 kB	2 ms
brat.cdac724402dc8d6d8f72.png	png	21.80 kB	21.47 kB	3 ms
bohaty.b20ba35978c089bf2f6c.png	png	29.60 kB	29.27 kB	4 ms
cita.48cf1c2769d958bae777.png	png	33.45 kB	33.12 kB	5 ms

Obr. 4.2: Znázornenie času načítavania obrázkov

Na obrázku sú zobrazené hodnoty času, za ktoré sú obrázky načítané. Hodnoty sú v rozmedzí od 1 ms do 17 ms, čo je stále možné považovať za postačujúce hodnoty. V tomto prípade nie je ani z používateľského pohľadu zaznamenané dĺhšie načítavanie, ako tomu bolo v predchádzajúcom teste.

4.2 Testovanie kolaborácie medzi používateľmi

Testovanie spolupráce medzi používateľmi bolo vykonávané prihlasením sa do systému prostredníctvom dvoch rôznych prehliadačov. Testované boli tri hlavné prvky kolaborácie. Prvým bolo kliknutie na input, respektíve obrázok, keďže v tomto prípade je potrebné znemožniť prístup ostatným používateľom do inputu alebo obrázku. Druhým testovaným prvkom bolo obdržanie informácie o doplnení textu do inputu a tretím bolo overenie získania zvolených obrázkov ostatnými používateľmi.

Oznámenie kliknutia na obrázok alebo input ostaným používateľom

Keď používatelia, ktorí sa nachádzajú vo virtuálnej miestnosti kliknú na input element alebo obrázok bloku, je potrebné poslať dáta ostatným používateľom, ktorí sa v miestnosti nachádzajú. Nasledujúci obrázok zobrazuje príklad dát, ktoré sú posielané ostatným používateľom v miestnosti.

Obr. 4.3: Príklad dát, ktoré sú posielané po kliknutí na input alebo obrázok

Na obrázku je možné vidieť, že ostatným používateľom je posielané pole. Pole obsahuje objekty, ktorých indexy sú zhodné s indexmi inputov, respektíve obrázkov. Objekty obsahujú meno používateľa, ktorý klikol na element ako posledný, identifikačné číslo používateľa, miestnosť, v ktorej sa používateľ nachádza a boolean hodnotu, ktorá vyjadruje, či je element aktuálne používaný. Rovnaký obsah je posielaný v oboch prípadoch, či sa jedná o input alebo o obrázok v prípade druhého typu úlohy.

Obdržanie informácie o doplnení textu do inputu

Ak niektorý z používateľov po vpísani textu do inputu klikne mimo inputu, do ktorého vpisoval, odošle sa informácia o doplnení textu ostatným používateľom. Formát informácie, ktorá je odoslaná je zobrazený na nasledujúcom obrázku.

Obr. 4.4: Obdržanie informácie o zmene hodnoty v inpute

Z obrázka je možné vyčítať, že informácia o zmene hodnoty v inpute je ostatným používateľom poslaná vo formáte poľa, v ktorom indexy sú zhodné s indexami inputov. Každý index poľa obsahuje reťazec textu, ktorý sa v inpute s daným indexom nachádza.

Obdržanie informácie o zmene obrázka

V druhom type úlohy sa dáta o zmene obrázkov posielajú v momente, keď používateľ klikne na vybraný obrázok v modali. V tomto prípade sa posielajú dáta v rovnakom formáte ako to bolo pri zmene hodnoty v inpute, teda jedná sa o pole, ktoré obsahuje reťazce vyjadrujúce názvy obrázkov.

5 Záver

Najprv bolo potrebné analyzovať možnosti kolaboratívneho prostredia pre metódu ikonicko-textovej formy výuky. Z tohto dôvodu je vypracovaná analýza, v ktorej je riešená platforma, na ktorú je systém vytvorený, kde sú zvážené výhody a nevýhody každej platformy. Analyzované sú taktiež technológie, ktoré mohli byť zvažované na vytvorenie systému. Pri analyzovaní technológií boli do úvahy brané rôzné kombinácie technológií, ktoré vyhovovali požiadavkám. Na konci analýzy je vyhodnotené, ktorá kombinácia technológií a spôsobu vytvorenia je pre vytváraný systém najvýhodnejšia.

Na základe výberu technológií a spôsobu vytvorenia systému je vytvorený návrh aplikácie, kde je navrhnutá všeobecná architektúra systému, webové rozhranie, ktoré sa skladá zo štyroch komponentov a databázy, ktorá obsahuje šesť tabuliek.

Po vypracovaní návrhu bolo potrebné systém implementovať. Pri implementácii systému bolo zistené, že v aplikácii nastali oproti návrhu zmeny. Prvé zmeny nastali vo všeobecnej architektúre systému. Tieto zmeny sú opísané v kapitole 3.1. V kapitole 3.2 je opísané ako bol systém implementovaný, rozoberané sú aj samotné algoritmy, ktoré sú v systéme implementované. Implementácia najpodstatnejšej časti kolaborácie v systéme je opísaná v kapitole 3.3. Posledná kapitola implementácie je venovaná databáze aplikácie, v ktorej oproti návrhu nastali zmeny, vďaka ktorým je činnosť aplikácie zjednodušená a zrýchlená.

Posledným krokom v tejto bakalárskej práci bolo overiť efektívnosť aplikácie pomocou zaťažových testov a funkčnosť koláborácie medzi používateľmi, ktorá je overená pomocou zobrazenia posielaných dát používateľom v miestnosti, ak jeden z používateľov použije niektorý z kolaboratívnych komponentov aplikácie.

Jedno z vylepšení pre tento systém by mohlo byť rozšírenie systému o používateľské roly. Mohlo by sa jednať o učiteľa, ktorý by mohol pridávať a upravovať testy a žiaka, ktorý by mohol testy vypracovať. Taktiež by systém mohol byť rozšírený o hodnotený spôsob vykonávania testov, keďže aktuálne je možné prejsť na ďalšiu otázku, len po správnom vypracovaní aktuálnej otázky.

Literatúra

1. PETRÍKOVÁ DARINA, Sobota Branislav a. *Virtuálno-realitné technológie súčasnosti a vzdelávanie postihnutých ľudí* [elektronický zdroj]. 2019. ISBN 978-80-553-3389-2.
2. FETAJI, Majlinda; FETAJI, Bekim; AJREDINI, Armend; EBIBI, Mirlinda. Devising a model of electronic School Management System based on web services for secondary schools in Macedonia. In: *Proceedings of the ITI 2013 35th International Conference on Information Technology Interfaces*. 2013, s. 187–192. Dostupné z DOI: 10.2498/iti.2013.0573.
3. ATMOJO, Kuncoro Dwi; BANDUNG, Yoanes. Eduvid, web video to support digital learning in rural primary schools. In: *2012 International Conference on Cloud Computing and Social Networking (ICCCSN)*. 2012, s. 1–4. Dostupné z DOI: 10.1109/ICCCSN.2012.6215756.
4. QIU, Robin G.; LEE, Doris. Transformative education Web 2.0 systems for enriching high school STEM education. In: *2013 IEEE 4th International Conference on Software Engineering and Service Science*. 2013, s. 352–356. Dostupné z DOI: 10.1109/ICSESS.2013.6615322.
5. SIFI, Sami; ALOUANE, Rym. Esprit-social network: An internal collaborative platform for the students of ESPRIT. In: *2016 IEEE Global Engineering Education Conference (EDUCON)*. 2016, s. 563–567. Dostupné z DOI: 10.1109/EDUCON.2016.7474607.
6. PATIDAR, Anil; SUMAN, Ugrasen. Towards Analyzing Mobile App Characteristics for Mobile Software Development. In: *2021 8th International Conference on Computing for Sustainable Global Development (INDIACom)*. 2021, s. 786–790. Dostupné z DOI: 10.1109/INDIACom51348.2021.00141.
7. MCFEDRIES, Paul. *Web Coding & Development All-in-One For Dummies*. 1. vyd. 111 River Street, Hoboken, NJ 07030: John Wiley & Sons, Inc., 2018. ISBN 978-1119473923.

8. ANGULAR, angular.io. *Introdunction to the Andular Docs*. Dostupné tiež z: `https://angular.io/docs`.
9. ANGULAR, angular.io. *Two-way binding*. Dostupné tiež z: `https://angular.io/guide/two-way-binding`.
10. BOOTSTRAP, getbootstrap.com. *About*. Dostupné tiež z: `https://getbootstrap.com/docs/5.1/about/overview/`.
11. REACT, reactjs.org. *Getting Started*. Dostupné tiež z: `https://reactjs.org/docs/getting-started.html`.
12. NODEJS.ORG, Node.js. *About documentation*. Dostupné tiež z: `https://nodejs.org/en/docs/`.
13. POSTGRESQL, postgresql.org. *About*. Dostupné tiež z: `https://www.postgresql.org/about/`.
14. HARNIČÁR, František. *Metóda ikonicko-textovej formy výuky hendikepovaných detí na báze webových technológií*. 2022. Bakalárska práca. 3.3 - Rola študenta.

Zoznam skratiek

API Application Programming Interface.

CSS Cascading Style Sheets.

HTML HyperText Markup Language.

JDBC Java Database Connectivity.

JSON JavaScript Object Notation.

Sass Syntactically Awesome Style Sheets.

URL Universal Resources Locator.

Zoznam príloh

A Systémová príručka

A.1 Funkcia programu

Systém vytvorený vrámci bakalárskej práce je určený na riešenie úloh metódy ikonicko-textovej formy výuky hendikepovaných detí pomocou webových technológií. Systém je vytvorený na báze kolaborácie medzi používateľmi počas riešenia úloh vo virtuálnej miestnosti. Správna funkčnosť aplikácie bola dosiahnutá vytvorením klientskej a serverovej časti aplikácie. Klientská časť aplikácie je určená na interakciu s používateľom, ktorý svojími vstupmi zabezpečuje odosielanie dát na serverovú časť, ktorej úlohou je prijímať a spracovávať požiadavky tak, aby boli v správnom tvare odoslané späť na klientskú časť aplikácie. Po tomto procese sa výsledok zobrazí používateľovi.

A.2 Funkcie programov

A.2.1 Technické požiadavky na spustenie webovej aplikácie

Používanie webovej aplikácie je podmienené tým, že používateľ má zariadenie s webovým prehliadačom a pripojením na internet. Fungovanie aplikácie je zabezpečené na zariadeniach s operačným systémom od Windows 7, na Apple Mac zariadeniach aj na zariadeniach s operačným systémom Linux. Aplikácia bola optimalizovaná na webový prehliadač Google Chrome vo verzii 101. Bezproblémové fungovanie je zabezpečené vo webových prehliadačoch Google Chrome od verzie 57, Firefox od verzie 52, Safari od 10.1, Edge od verzie 16. Používanie aplikácie v prehliadači Internet Explorer nie je odporúčané.

A.3 Preklad programu

Klientská časť aplikácie bola vyvíjaná v Reacte, HTML, CSS a Bootstrap. Tieto jazyky a rámce boli rozšírené o potrebné knižnice.

A.3.1 Spustenie aplikácie

Aplikácia je komprimovanom stave na úložnom médiu. Pre jej spustenie je potrebné vykonať nasledujúce kroky:

1. Skopírovať z uložného média na interné úložisko zariadenia.
2. Rozbaliť skomprimovaný súbor aplikácie.
3. Otvoriť umiestnenie súboru *client* v termináli.
4. Nainštalovať potrebné knižnice pomocou príkazu:
   npm *i axios bootstrap node-sass query-string react-bootstrap react-dom react-icons react-router-dom sass socket.io-client*
5. Otvoriť umiestnenie súboru *server* v termináli.
6. Nainštalovať potrebné knižnice pomocou príkazu:
   npm *i bcrypt body-parser cookie-parser cors express express-fileupload express-session knex nodemon pg socket.io*
7. Spustiť sever pomocou príkazu:
   node *index*
8. Otvoriť umiestnenie súboru *client* v termináli.
9. Spustiť klienta aplikácie príkazom:
   npm *start*

Klientská časť aplikácie bola vyvíjaná v Reacte, HTML, CSS a Bootstrap.

A.3.2 Zoznam zdrojových súborov klientskej časti aplikácie

Zdrojový kód klientskej časti aplikácie je rozdelený do priečinkov s názvami components, pages a styles. Sú v nich uložené zdrojové súbory. V priečinku pages sa nachádzajú zdrojové kódy obrazoviek. Priečinok components obsahuje zdrojové kódy jednotlivých komponentov a priečinok styles je využívaný na uchovávanie súborov so zdrojovým kódom na vytvorenie štýlu komponentov.

Priečinok pages obsahuje:

- LoginPage.js
- MainPage.js
- NotFoundPage.js
- RegistrationPage.js
- VirtualRoomPage.js

Priečinok styles obsahuje:

- ActiveUsers.css
- App.css
- custom.scss
- Input.css
- Login.css
- Navbar.css
- Notebook.css
- Question.css
- Solution.css
- TestInfo.css
- UsersModal.css
- VirtualRoomPage.css

Priečinok components obsahuje:

- AcitveUsers.js
- Input.js
- InputModal.js
- Login.js
- Navbar.js
- Notebook.js
- ProtectedRoutes.jsx
- Question.js
- Register.js
- Solution.js
- TestInfo.js
- UniversalModal.js
- UsersModal.js
- views.jsx
- VRoomContent.js

A.3.3 Zoznam zdrojových súborov klientskej časti aplikácie

Zdrojový kód serverovej časti aplikácie sa nachádza v jednom hlavnom zdrojovom súbore a viacerých vedľajších.
Zdrojové súbory serverovej časti:

- index.js
- activeUsers.js
- allVRooms.js
- login.js
- question.js
- register.js
- VRoom.js

A.3.4 Popis programu klientskej časti aplikácie

Klientská časť aplikácie pozostáva z nasledujúcich činnosti:

- Zobrazenie stránky na registráciu - RegisterPage,
- Zobrazenie stránky prihlásenia - LoginPage,
- Zobrazenie domovskej stránky - MainPage,
- Zobrazenie neexistujúcej stránky - NotFoundPage,
- Zobrazenie virtuálnej miestnosti - VirtualRoomPage,
- Prihlásenie používateľa - Login,
- Registrácia používateľa - Register,
- Zobrazenie kariet predstavujúcich virtuálne miestnosti - Notebook,
- Zobrazenie hornej lišty - Navbar,
- Zobrazenie obsahu virtuálnej miestnosti - VRoomContent,
- Zobrazenie komponentu riešenia úlohy - Solution,
- Zobrazenie komponentu znenia úlohy - Question,
- Zobrazenie aktívnych používateľov - ActiveUsers,
- Zobrazenie informácii o teste - TestInfo,
- Zobrazenie blokov v riešení úlohy - Input,
- Zobrazenie modalu obrázkov - InputModal.

Popis riešenia

LoginPage zobrazuje stránku prihlásenia, obsahuje jeden komponent Login. RegisterPage zobrazuje stránku registrácia, obsahuje komponent Register. MainPage zobrazuje domovskú stránku, pozostáva z komponentu Navbar a dynamicky generovných kompomentov Notebook. NotFoundPage je zobrazený používateľovi pri zadaní neexistujúcej url adresy systému. VirtualRoomPage zobrazuje stránku virtuálnej miestnosti, ktorá obsahuje komponent Navbar a VRoomContent. VRoomContent zobrazuje komponenty TestInfo, ActiveUsers, Question a Solution, ktorý zobrazuje komponenty Input.

A.3.5 Popis programu serverovej časti aplikácie

Serverová časť aplikácie pozostáva z nasledujúcich činnosti:

- Hlavná činnosť servera, komunikácia pomocou Websocketov - index.js,
- Činnosti po požiadavke na url adresu servera /login - login.js
- Činnosti po požiadavke na url adresu servera /register - register.js
- Činnosti po požiadavke na url adresu servera /virtualrooms - allVRooms.js
- Činnosti po požiadavke na url adresu servera /question - question.js
- Činnosti po požiadavke na url adresu servera /vroom - VRoom.js

Popis riešenia

Hlavná činnosť servera a komunikácia prostredníctvom websocketov prebieha v súbore index.js. Požiadavky na server sú spracované v konkrétnych súboroch. V login.js sú požiadavky týkajúce sa autentifikácie. V register.js sú požiadavky na registráciu používateľa. Súbor allVRooms.js vratí zoznam všetkých miestností nachádzajúcich sa v databáze. Požiadavky týkajúce sa úloh sú spracovávané v question.js. Požiadavky týkajúce sa virtuálnych miestností sú vybavené v VRoom.js.

A.4 Vytvorenie databázy

Najprv je nutné vytvoriť PostgreSQL databázu s názvom *Database* a heslom *password*. Pri zvolení iného mena alebo hesla databázy, je potrebné tieto údaje zmeniť v serverovej časti aplikácie v súbore index.js. Pre vytvorenie tabuliek je potrebné zadať tento skript do vytvorenej databázy:

```
CREATE TABLE virtual_rooms (
    id SERIAL PRIMARY KEY,
    start_time TIMESTAMP,
    end_time TIMESTAMP,
    active_students TEXT[],
    questions INTEGER[],
    current_question INTEGER,
    is_sent BOOLEAN
);
```

```
CREATE TABLE students (
    id SERIAL PRIMARY KEY,
    username CHARACTER VARYING,
    password CHARACTER VARYING,
    UNIQUE(username)
);

CREATE TABLE questions (
    id SERIAL PRIMARY KEY,
    question_content JSONB
);
```

B Používateľská príručka

B.1 Funkcie programov

B.1.1 Technické požiadavky na spustenie webovej aplikácie

Používateľ aplikácie musí mať zariadenie s funkčným pripojením na internet a nainštalovaným webovým prehliadačom.

B.1.2 Inštalácia

Aplikácia je komprimovanom stave na úložnom médiu. Pre jej spustenie je potrebné vykonať nasledujúce kroky:

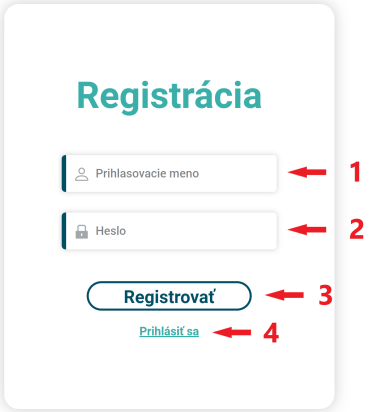
1. Skopírovať z uložného média na interné úložisko zariadenia.
2. Rozbaliť skomprimovaný súbor aplikácie.
3. Otvoriť umiestnenie súboru *client* v termináli.
4. Nainštalovať potrebné knižnice pomocou príkazu:
   npm *i axios bootstrap node-sass query-string react-bootstrap react-dom react-icons react-router-dom sass socket.io-client*
5. Otvoriť umiestnenie súboru *server* v termináli.
6. Nainštalovať potrebné knižnice pomocou príkazu:
   npm *i bcrypt body-parser cookie-parser cors express express-fileupload express-session knex nodemon pg socket.io*
7. Spustiť sever pomocou príkazu:
   node *index*
8. Otvoriť umiestnenie súboru *client* v termináli.
9. Spustiť klienta aplikácie príkazom:
   npm *start*

B.2 Použitie programu

Aplikácia je určená na riešenie úloh metódy ikonicko-textovej výuky hendikepovaných detí. Aplikácia má implementovanú kolaboráciu, ktorá umožňuje používateľom spolupracovať vo virtuálnej miestnosti.

B.2.1 Registrácia

Používateľovi je pri registrácii zobrazená táto obrazovka:

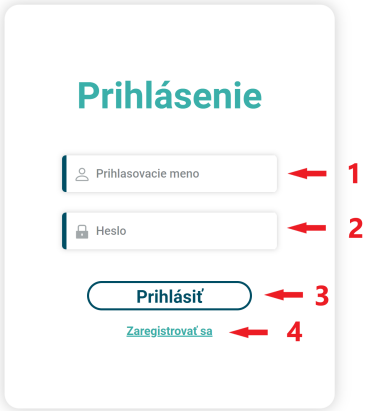


Obr. B.1: Obrazovka registrácie

1. Textové pole pre vytvorené meno používateľa.
2. Textové pole pre vytvorené heslo používateľa.
3. Tlačidlo na odoslanie údajov vložených v textových poliach.
4. Odkaz na formulár prihlásenia.

B.2.2 Prihlásenie

Používateľovi je pri prihlasovaní zobrazená táto obrazovka:



Obr. B.2: Obrazovka prihlásenia

1. Textové pole pre meno používateľa.
2. Textové pole pre heslo používateľa.
3. Tlačidlo na odoslanie údajov vložených v textových poliach.
4. Odkaz na formulár registrácie.

B.2.3 Domovská obrazovka

Po prihlásení do aplikácie je zobrazená obrazovka, ktorú je možné vidieť na nasledujúcom obrázku.

Obr. B.3: Domovská obrazovka

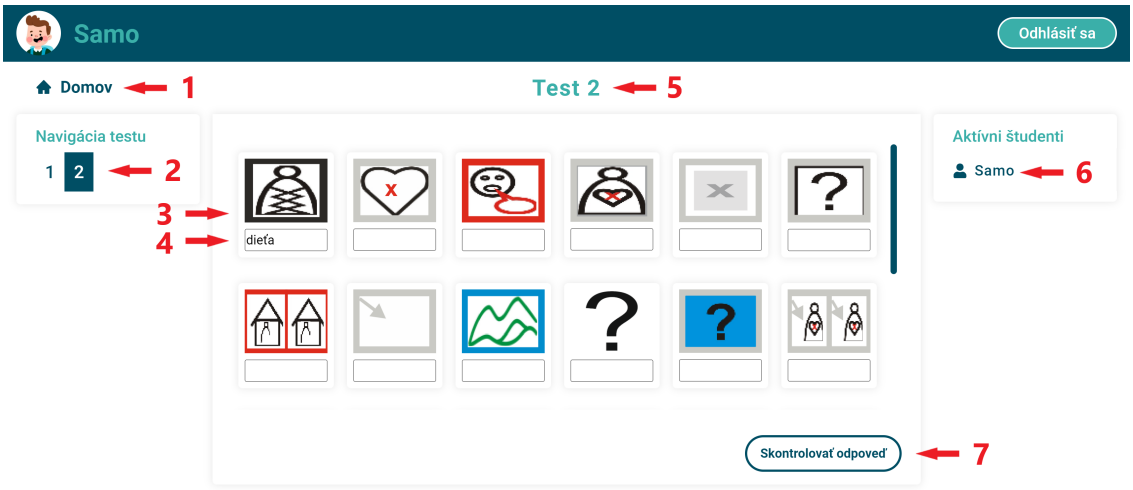
1. Meno používateľa.
2. Tlačidlo na odhlásenie používateľa.
3. Názov virtuálnej miestnosti, ktorú karta predstavuje.
4. Začiatok aktivity miestnosti.
5. Koniec aktivity miestnosti.
6. Stav miestnosti - Odoslaná/Neodoslaná.
7. Tlačidlo pre vstup do miestnosti.

B.2.4 Virtuálna miestnosť

Virtuálna miestnosť je zobrazovaná odlišne vzhľadom na typ úlohy.

Typ úlohy z obrázkov na text

Úlohou používateľov je napísať význam zadaných obrázkov.



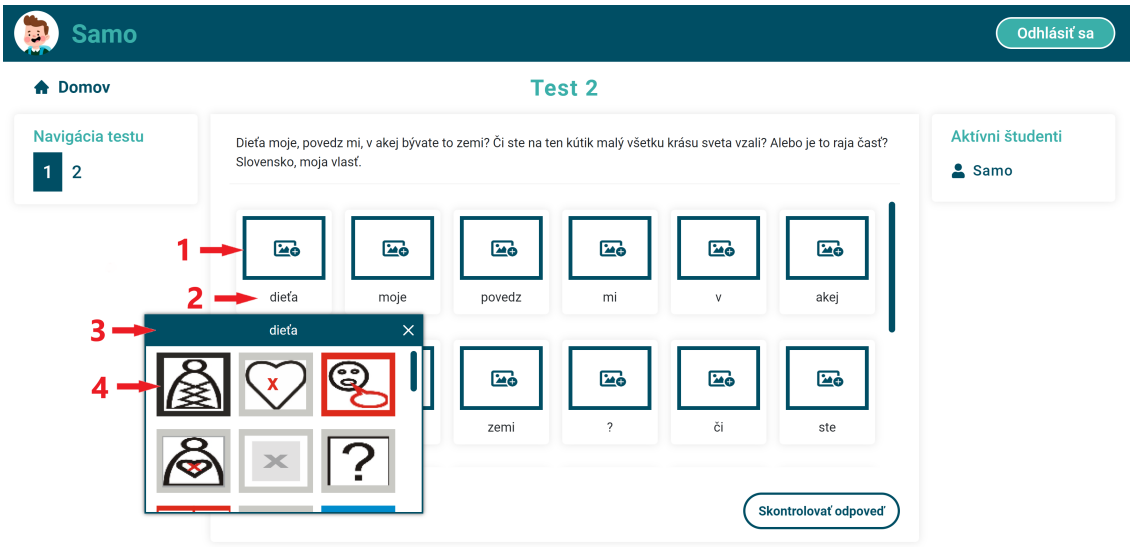
Obr. B.4: Virtuálna miestnosť s úlohou z obrázkov na text

1. Tlačidlo návratu na domovskú obrazovku.
2. Aktuálna otázka.
3. Obrázok, ktorého názov je potrebné určiť.
4. Textové pole, do ktorého je potrebné vložiť názov obrázka.

5. Názov virtuálnej miestnosti.
6. Zoznam aktívnych používateľov v miestnosti.
7. Tlačidlo na kontrolu odpovede.

Typ úlohy z textu na obrázky

Úlohou používateľov je priradiť obrázky zadanému textu.



Obr. B.5: Virtuálna miestnosť s úlohou z textu na obrázky

1. Obrázok, ktorý vyjadruje prázdnu odpoveď. Po kliknutí sa otvorí modal.
2. Text, ktorému je potrebné priradiť obrázok.
3. Modal so slovom v hlavičke z bloku, z ktorého bol otvorený.
4. Zoznam obrázkov, ktorý môže používateľ priradiť slovu v hlavičke modalu. Vybraný obrázok je po kliknutí pridaný do bloku.