УНИВЕРЗИТЕТ „СВ. КИРИЛ И МЕТОДИЈ“ – СКОПЈЕ

ФАКУЛТЕТ ЗА ИНФОРМАТИЧКИ НАУКИ И КОМПЈУТЕРСКО ИНЖЕНЕРСТВО

Напреден Веб Дизајн

Програма за споделување материјали меѓу студентите на ФИНКИ (FINKIShare)

Изработиле:

Владимир Грнчаровски, 211028

Андреј Влаховски, 211136

Ментор:

проф. д-р Бобан Јоксимоски

Датум: 27/08/2024

Вовед **3**
Цели на проектот **3**
Опис на архитектурата и технологиите **4**
1. Backend 4
1.1. Употреба на Java и Spring Boot во Backend-от 4
1.1.1. Модели во системот 5
1.1.2. Логика во сервисниот слој 5
1.1.3. Придобивки од Java и Spring Boot 5
1.2. Податоци за предметите: Web Scraping 6
1.2.1. Процес на scraping 6
1.2.2. Придобивки и предизвици 6
2. Frontend 6
2.1. Употреба на React во проектот 7
2.2. Зошто React? 7
2.3. Што научивме преку React? 7
2.4. Предизвици и постигнувања 7
2.5. Рутирање низ различните страни 8
2.6. Повици кон backend 9
2.6.1. Пример за POST Request за коментари 9
2.6.2. Пример за GET Request за upvote 10
2.7. Употреба на React Three.js за 3D Анимации 11
2.7.1. Компјутер на почетната страна 12
2.7.2. Предизвици и постигнувања 12
Опис на функционалностите **12**
Home Page 12
Дополнителна интерактивна компонента 13
Импорт на модули 14
Функцијата Model 15
useGLTF.preload('/computer.gltf'): 15
Функцијата ComputerAnimation: 15
Страница за предмети 15
Дополнителен извор на информација за подобрување на корисничкото искуство 16
Swal, SweetAlert2 16
Autocomplete од MaterialUI 16
Форум 17
Поставување на Материјали 18
Пример код од делот за поставување материјали 18
Регистрација и логирање: 20
Предизвици и решенија **20**
Тестирање и оптимизација **21**
Заклучок **21**

Вовед

Идејата на овој проект беше да се развие апликација која ќе им овозможи на студентите од ФИНКИ да споделуваат материјали за одредени предмети, со што ќе се создаде интегрирана платформа која ќе ги поддржи нивните академски активности. Апликацијата е внимателно дизајнирана со цел да ги олесни процесите на учење и размена на информации помеѓу студентите. Преку неа, студентите можат да пристапат до разни ресурси, како претходни испитни задачи, аудиториски вежби и други корисни документи, што ќе им помогне да се подготват подобро за испити и да ги надоградат нивните знаења.

За реализација на оваа платформа користевме современи технологии кои овозможуваат развој на високо интерактивна и кориснички пријателска апликација. **Java и Spring Boot** се користени за изградба на backend-от на системот, што ни овозможува да изградиме стабилен и ефективен серверски дел кој управува со податоците и логиката на апликацијата. На frontend-от, **JavaScript и React** се користени за креирање на динамичен и интуитивен кориснички интерфејс, кој овозможува лесно користење и пристап до сите функции на платформата. Дополнително, **Python** е искористен за web scraping, што овозможува автоматско собирање на податоци од интернет извори (во нашиот случај официјалната страна на ФИНКИ) и интегрирање на овие податоци во системот.

Фокусот на проектот е обезбедување на позитивно корисничко искуство и висока интерактивност. Секоја од овие технологии придонесува кон различни аспекти на развојот на апликацијата, осигурувајќи дека системот е стабилен, брз и лесен за користење. Во следните делови од извештајот, ќе се разгледа секоја од овие технологии детално, за да се разбере како секоја од нив игра клучна улога во остварувањето на целиите на проектот, како и главните функционалности на апликацијата, предизвици со кои сме се соочиле и заклучок, односно што сме научиле со овој проект.

Цели на проектот

Оваа платформа има за цел да ги олесни студентите во нивното образование, обезбедувајќи им пристап до разни ресурси кои ќе им помогнат во подготовките за испити, учење на нови концепти и подобро разбирање на материјата. Материјалите што студентите можат да ги најдат на платформата вклучуваат претходни испитни задачи, аудиториски вежби, предавања, книги, белешки, како и други корисни ресурси кои можат да им помогнат во нивните студии.

Освен овозможувањето на споделување и преземање на материјали, платформата исто така нуди и можност за дискусија и размена на идеи преку интегриран форум. Форумот е место каде што студентите можат да поставуваат прашања, да дискутираат за различни

аспекти на предметите, да споделуваат совети за учење и да добијат помош од нивните колеги. Секој пост на форумот може да биде коментиран, а секој коментар и пост може да биде оценет преку систем за гласање (upvote/downvote), што помага да се истакнат најкорисните и најточните информации.

Платформата исто така поддржува и поставување на материјали, каде што секој студент може да сподели свои материјали со другите, што овозможува создавање на богата колекција на ресурси која е од голема вредност за студентската заедница. Ова создава средина каде што знаењето се споделува и надградува, што е во согласност со принципите на колаборативно учење и заедничка поддршка меѓу студентите.

Со ова, платформата не само што обезбедува средства за академски успех, туку исто така промовира и заедничка работа и комуникација меѓу студентите, правејќи ја важен дел од студентскиот живот на ФИНКИ.

Опис на архитектурата и технологиите

Архитектура на системот

Архитектурата на нашиот систем е поделена на два главни дела: frontend и backend, кои комуницираат преку REST API повици. Со оваа поделба обезбедивме флексибилност во развојот, можност за паралелна работа и лесно одржување на кодот.

1. Backend

Backend-от е изграден со користење на Java и Spring Boot, што ни овозможи да обезбедиме стабилна и безбедна основа за апликацијата. Spring Boot беше клучен во менаџирањето на бизнис логиката, базата на податоци, како и обработката на REST API повиците.

Дополнително, за собирање на податоците за предметите и нивните атрибути користевме Python за web scraping. Со оваа техника, автоматски ги прибравме потребните информации од интернет извори, што значително го забрза процесот на внесување и ажурирање на податоците.

1.1. Употреба на Java и Spring Boot во Backend-от

Во backend-от на нашата апликација користевме Java заедно со Spring Boot за да обезбедиме стабилна и скалабилна инфраструктура. Оваа комбинација ни овозможи брз развој, лесно менаџирање на зависности, како и интеграција на различни технологии во еден координиран систем.

1.1.1. Модели во системот

Во системот дефинираме неколку основни модели кои ги претставуваат клучните ентитети во апликацијата:

- **Attachment**: Го претставува прилогот што студентите можат да го постават во системот (на пример, PDF документи, слики).
- **Comment**: Секој коментар што може да биде додаден на некој пост во форумот.
- **Post**: Секој пост што студентите го објавуваат на форумот, со можност за коментирање и гласање.
- **Role**: Ги дефинира улогите на корисниците (на пример, студент, администратор).
- **SubjectDetails**: Ги содржи деталите за секој предмет што студентите можат да го следат.
- **User**: Го претставува корисникот во системот, со сите негови информации и интеракции, кој што може да биде дел од една од горенаведените улоги.

1.1.2. Логика во сервисниот слој

Сервисниот слој на апликацијата беше местото каде што се имплементираа најголемиот дел од бизнис логиката. Некои од најзначајните методи вклучуваат:

- **takeSubject(String name, String username)**: Оваа метода дозволува корисникот да земе одреден предмет и да го додаде во неговите предмети.
- **findAllTakenSubjects(String username)**: Методата враќа листа на сите предмети што корисникот ги има земено.
- **findAllPostsById(Long id):** Оваа метода ги враќа сите постови поврзани со одреден предмет.
- **createPost(String title, String text, SubjectDetails subjectDetails, User author):** Оваа метода креира нов пост на форумот.
- **increaseScore(Long id) и decreaseScore(Long id):** Овие методи ја менуваат вредноста на гласањето за одреден пост.
- **saveAttachment(MultipartFile file, String description, Long subjectId, String username**): Оваа метода дозволува студентите да поставуваат прилози за одреден предмет.
- **getAttachment(String fileId) и getAllAttachmentsBySubject(long subjectId):** Овие методи ги враќаат деталите за еден или повеќе прилози, овозможувајќи пристап до материјалите што студентите ги споделиле.

1.1.3. Придобивки од Java и Spring Boot

Со користење на Java и Spring Boot, успеавме да создадеме стабилен и лесно проширлив систем. Java ни овозможи да изградиме силна објектно-ориентирана структура, додека Spring Boot ни понуди алатки за брзо создавање на REST API, менаџирање на безбедноста, и обработка на податоците.

1.2. Податоци за предметите: Web Scraping

Во процесот на развој на апликацијата, едно од клучните барања беше да се обезбедат точни и ажурирани податоци за предметите што се изучуваат на ФИНКИ. За да го постигнеме ова, користевме техника наречена **web scraping**, за која го користевме Python. Ова ни овозможи автоматски да собереме податоци од официјалните веб-страници на факултетот и да ги интегрираме во нашата база на податоци.

1.2.1. Процес на scraping

Процесот на scraping вклучуваше неколку чекори:

1. Идентификација на релевантни извори: Прво, ги идентификувавме веб-страниците кои ги содржат податоците што ни беа потребни, како што се предметните програми, описи на курсеви, листи на предавачи и литература.
2. Користење на Python за scraping: Развивме скрипти на Python кои автоматски ги преземаат податоците од овие страници. Python беше избран поради неговата едноставност и големата поддршка за библиотеки како BeautifulSoup и Scrapy, кои овозможуваат лесно манипулирање со HTML структурата на веб-страниците.
3. Екстракција и форматирање на податоците: Податоците што ги добивме преку scraping беа неструктурирани и беше потребно дополнително обработување за да бидат употребливи. Ги организиравме во структура која овозможува лесно филтрирање и пребарување во апликацијата. Пример за ова беше академската година, заедно со тоа дали станува збор за летен или зимски семестар
4. Интеграција со базата на податоци: Конечно, податоците беа внесени во базата на податоци на нашата апликација, каде што се складираат и се прикажуваат на корисниците преку интерфејсот на апликацијата.

1.2.2. Придобивки и предизвици

Користењето на Python за web scraping ни овозможи да добиеме голем волумен на податоци за краток период, што беше клучно за брзата имплементација на функционалностите поврзани со предметите. Сепак, процесот не беше без предизвици. Неопходно беше да се осигуриме дека скриптите ќе продолжат да функционираат и при промени на веб-страниците од каде што ги преземавме податоците. Дополнително, се соочивме со предизвикот на етичко scraping, при што водевме сметка да ги почитуваме правилата на веб-страниците од кои ги преземавме податоците.

2. Frontend

Во frontend-от на апликацијата користевме React, кој е современа и брзорастечка библиотека за изградба на кориснички интерфејси. Со React, успеавме да креираме интерактивни компоненти кои се лесно управливи и скалабилни.

Покрај основните функционалности, имплементиравме и **React Three.js** за креирање на 3D анимации. Ова ни овозможи да додадеме дополнителна визуелна динамика во

апликацијата, како што е 3D моделот на компјутерот на почетната страна, со кој корисниците можат интерактивно да комуницираат.

2.1. Употреба на React во проектот

За фронтендот на нашата апликација, избравме да го користиме React, технологија која е во постојан раст и за која верувавме дека е идеална за потребите на нашиот проект. Една од главните причини за оваа одлука беше нашата желба да ја научиме и усовршиме оваа технологија, што се покажа како вистински предизвик и можност за значително зголемување на нашето знаење и вештини.

2.2. Зошто React?

React е популарен избор за развој на модерни веб-апликации поради неговата едноставност и флексибилност. Ни овозможи да создадеме динамичен и интерактивен кориснички интерфејс со релативно мала количина на код. Покрај тоа, React се користи широко во индустријата, па затоа сакавме да ја научиме оваа технологија и да ја примениме во нашиот проект.

2.3. Што научивме преку React?

Во текот на развојот, научивме многу за React, особено во следните области:

- **Hooks**: Постојано користевме различни React hooks како useState, useEffect, и други, кои ни овозможија да управуваме со состојбата на апликацијата и да извршуваме странични ефекти.
- **States**: Управувањето со состојбите (states) во апликацијата беше клучно за одржување на интерактивноста и реактивноста на интерфејсот. Научивме како правилно да управуваме со состојбите и како тие влијаат врз компонентите.
- **Менаџирање со компоненти**: Развивме силни вештини за организирање и менаџирање на компоненти, со што осигуравме дека апликацијата остана модуларна и лесно одржлива.
- **Праќање параметри и функции во компоненти**: Научивме како да праќаме податоци и функции од една компонента на друга, што ни овозможи да изградиме сложени и функционални интерфејси.

2.4. Предизвици и постигнувања

Еден од најголемите предизвици со React беше правилното менаџирање на состојбите во посложените компоненти. Сепак, преку постојана практика, успеавме да совладаме многу од предизвиците и да создадеме апликација која е стабилна и одговара на потребите на корисниците.

2.5. Рутирање низ различните страни

Во проектот, навигацијата помеѓу различните страници на апликацијата беше реализирана со помош на **react-router-dom**. Оваа библиотека овозможи управување со рутите и ефикасна прелистање на страниците во апликацијата. Кодот за конфигурација на рутите изгледа вака:

```
 import { BrowserRouter as Router, Route, Routes } from 'react-router-dom';

<Router>

   <Routes>

      <Route path="/" element={<HomePage />} />

      <Route path="/homepage" element={<HomePage />} />

      <Route path="/materials" element={<MaterialsPage />} />

      <Route path="/subjects" element={<AllSubjects />} />

      <Route path="/register" element={<RegisterPageMaterialUI />} />

      <Route path="/login" element={<LoginPageMaterialUI />} />

      <Route path={"/materials/forum/:id"} element={<ForumPage />} />

      <Route path={"/materials/upload/:id"} element={<AttachmentPage />} />

      {/* TESTING PURPOSES */}

      <Route path="/animationComputer" element={<ComputerAnimation />} />

      <Route path={"/materials/upload"} element={<UploadMaterials />} />

   </Routes>

</Router>
```

Со оваа конфигурација, корисниците можат да навигираат низ сите главни страници на апликацијата, вклучувајќи ги Home Page, страница за предмети, страница за материјали, форум, и страница за регистрација и логирање. Исто така, се вклучени и тест страници за анимации и за поставување материјали. Оваа структура осигурува лесна и организирана навигација низ целата апликација.

2.6. Повици кон backend

2.6.1. Пример за POST Request за коментари

```
import { useState } from "react";
const useSetComments = () => {
    const [comment, setComment] = useState("");
    const handleSubmit = async (event, username, props, subjectId) => {
        event.preventDefault();
        try {
            const response = await fetch('http://localhost:8080/comments/add', {
                method: 'POST',
                headers: {
                    'Content-Type': 'application/json'
                },
                body: JSON.stringify({
                    comment,
                    postId: props.postId,
                    username
                })
            });
            console.log(response.text());
            if (response.ok) {
                console.log('Created post successfully');
                props.onHide();
                window.location.href = `/materials/forum/${subjectId}`;
            } else {
```

```
                console.error('Creating post failed');
            }
        } catch (error) {
            console.error('Error:', error);
        }
    };
    return { comment, setComment, handleSubmit };
}
export default useSetComments;
```

Опис:

- Оваа функција користи React hook и useState за управување со состојбата на коментарот.
- **handleSubmit** функцијата се активира при испраќање на форма (на пример, по кликање на копчето за испраќање на коментар).
- Со помош на **fetch**, се прави POST барање до серверот на http://localhost:8080/comments/add со JSON податоци кои содржат коментар, ID на постот, и корисничко име.
- По успешна испраќање на коментарот, се **пренасочува** на форум страницата за конкретниот предмет.

2.6.2. Пример за GET Request за upvote

```
const useSetUpVote = () => {
    const updateVote = (id, onSuccess) => {
        const url = `http://localhost:8080/posts/${id}/increase_score`;
        fetch(url)
            .then((response) => {
                if (!response.ok) {
                    throw new Error('Network response was not ok');
                }
```

```
            return response.json();
        })
        .then((data) => {
            console.log(data);
            if (onSuccess) {
                onSuccess(data); // Call the success callback with the new data
            }
        })
        .catch((error) => {
            console.error('Error fetching data:', error);
        });
    };
    return { updateVote };
};


export default useSetUpVote;
```

Опис:

- Оваа функција не користи useState, туку директно ја извршува логиката за upvote.
- updateVote функцијата прави GET барање на URL кој се користи за зголемување на резултатот на постот.
- Ако серверот врати успешен одговор, функцијата onSuccess (ако е обезбедена) се повикува со новите податоци.
- Во случај на грешка, се печати грешката во конзолата.

Овие примери покажуваат основни техники за комуникација со серверот во React и управување со асинхрони задачи.

2.7. Употреба на React Three.js за 3D Анимации

Освен основната употреба на React за креирање на корисничкиот интерфејс, во нашиот проект се одлучивме да додадеме и 3D елементи со цел да ја направиме апликацијата

поинтерактивна и привлечна за корисниците. За оваа намена го користевме React Three.js, поточно библиотеките **@react-three/fiber и @react-three/drei**.

2.7.1. Компјутер на почетната страна

На почетната страна на апликацијата вградивме 3D модел на компјутер, со кој корисниците можат интерактивно да си играат. Оваа функционалност беше дизајнирана да го привлече вниманието на корисниците и да ја зголеми интеракцијата со апликацијата.

Користењето на React Three.js ни овозможи да интегрираме сложени 3D анимации директно во React компонентите, што значеше дека можевме лесно да ги менаџираме и анимираме моделите во реално време, во зависност од интеракциите на корисниците.

2.7.2. Предизвици и постигнувања

Креирањето на 3D анимации и нивната интеграција со React беше еден од најголемите технички предизвици во проектот. Преку користење на @react-three/fiber за рендерирање на 3D сцената и @react-three/drei за додавање на различни ефекти и контроли, успеавме да изградиме динамичен и привлечен визуелен елемент кој го збогатува корисничкото искуство.

Оваа функционалност не само што го направи интерфејсот поатрактивен, туку и ни даде можност да научиме како да ги користиме напредните можности на React за создавање на современи и иновативни веб-апликации.

Опис на функционалностите

Апликацијата нуди неколку клучни функционалности кои се дизајнирани да ја подобрат студентската искуства и да ја олеснат размената на информации и ресурси.

Home Page

Почетната страница на апликацијата е дизајнирана да биде централно место за навигација и информации. На Home Page, корисниците можат да филтрираат курсеви по различни категории, како што се web development, AI и математика. Ова овозможува брз и лесен пристап до релевантни курсеви, со цел да се пронајдат тие што најдобро одговараат на нивните интереси и потреби. Дополнително, на почетната страница се прикажани картички со информации за студентски организации и компании кои нудат можности за пракса, што им помага на студентите да ги истражат можностите за професионален развој. Footer-от на страницата вклучува основни информации за апликацијата, како што се контакт податоци, линкови до политики на приватност и услови за користење.

Дополнителна интерактивна компонента

Дополнително, на почетната страница е интегрирана Three.js анимација со интерактивен 3D модел на ретро компјутер, кој корисниците можат да го ротираат и зумираат. Оваа анимација додава елемент на забава и интерактивност на Home Page, привлекувајќи го вниманието на корисниците и правејќи го нивното искуство уште попривлечно.

Анимацијата е реализирана со React Three.js и користи 3D модел на компјутер, кој е преземен од Sketchfab. Претходно опишавме со предизвиците кои шро се соочивме додека пробувавме да ја имплементираме оваа функционалнсот, а сега ќе го разгледаме следниот код кој е дел од имплементацијата на оваа анимација:

```
import React from 'react'

import { useGLTF } from '@react-three/drei'


export default function Model(props) {

  const { nodes, materials } = useGLTF('/computer.gltf')

  return (

    <group {...props} dispose={null}>

      <group rotation={[-Math.PI / 2, 0, 0]}>

        <mesh geometry={nodes.Object_2.geometry} material={materials.None} scale={5}/>

        <mesh geometry={nodes.Object_3.geometry} material={materials.None} scale={5}/>

      </group>

    </group>

  )

}


useGLTF.preload('/computer.gltf')
```

```
import {Canvas} from "@react-three/fiber";
import {ContactShadows, Environment, OrbitControls} from "@react-three/drei";
import {Suspense} from "react";
import Computer from "./Computer"


const ComputerAnimation = () => {
    return (
        <Canvas style={{height: "80vh", width: "100%"}}>
            <ambientLight intensity={2}/>
            <OrbitControls enableZoom={false}/>
            <Suspense fallback={null}>
                <Computer/>
            </Suspense>
            <Environment preset='sunset'/>
            <ContactShadows position={[0, -0.5, 0]} opacity={0.8} scale={80} blur={1} far={10}
resolution={256} color="#000000"/>
        </Canvas>
    );
}


export default ComputerAnimation;
```

Кодот овозможува прикажување на интерактивен 3D модел на ретро компјутер на веб-страница користејќи React Three.js. Еве кратко објаснување на неговите делови:

Импорт на модули

- useGLTF: Оваа хук помага да се вчита 3D модел во GLTF формат.

- Canvas, ContactShadows, Environment, OrbitControls: Овие компоненти се користат за поставување на 3D сцената, додавање на светлина, сенки, и контроли за интеракција.

Функцијата Model

- Оваа функција го вчитува и прикажува 3D моделот на компјутерот.
- Моделот се вчитува од датотеката computer.gltf и се поставува на сцената со одредена ротација и скалирање.

useGLTF.preload('/computer.gltf'):

- Оваа линија овозможува претходно вчитување на моделот за да се подобри перформансот кога ќе се прикаже.

Функцијата ComputerAnimation:

- Главната компонента која поставува 3D сцена во Canvas елемент.
- ambientLight: Додава амбиентално светло на сцената.
- OrbitControls: Овозможува корисниците да ротираат и движат околу 3D моделот, но зумирањето е оневозможено.
- Suspense: Се користи за да се покаже резервен приказ додека се вчитува моделот.
- Environment: Сетира амбиенталното осветлување, во овој случај поставено на „sunset“.
- ContactShadows: Додава сенка под 3D моделот, која ја прави сцената пореалистична.

Овој код создава 3D сцена каде што корисниците можат да го ротираат и истражуваат компјутерот, придонесувајќи за интерактивноста и визуелната привлечност на апликацијата.

Страница за предмети

Оваа страница е особено корисна за студентите кои сакаат да пронајдат материјали специфични за одредени предмети. Страната овозможува филтрирање по година, семестар и име на предмет, што помага во прецизно пребарување на информации. По изборот на конкретен предмет, студентите се пренасочуваат кон посебна страница каде што можат да ги видат сите материјали поврзани со тој предмет и да додаваат нови ресурси. Оваа функционалност им овозможува на студентите полесно организирање и управување со своите училишни материјали.

Покрај тоа, за секој предмет, корисникот има опција да прегледа дополнителни детали преку посебен модал прозорец, каде што се прикажуваат информации како професори, предуслови за одреден предмет, и други релевантни податоци. Оваа функционалност им помага на студентите да добијат поцелосна слика за предметот и да донесат информирани одлуки во врска со нивниот избор на курсеви.

Дополнителен извор на информација за подобрување на корисничкото искуство

Swal, SweetAlert2

Кога корисникот ќе избере или отстрани предмет од своите материјали, се активира анимација користејќи ја библиотеката SweetAlert2. Оваа анимација прикажува порака која симболизира дека предметот е успешно додаден или одземен. Анимацијата е имплементирана на следниов начин:

```
import Swal from "sweetalert2"


Swal.fire(
    nextClicked ? "Успешно додаден предмет во материјали" : "Успешно тргнат предмет од материјали",
    nextClicked ? "Со лесно :)" : "Редно беше",
    nextClicked ? "success" : "error"
).then(r => r.isConfirmed);
```

Во зависност од тоа дали предметот е додаден или тргнат, анимацијата прикажува соодветна порака и икона, како и позитивен или негативен фидбек. Ова додава уште еден слој на интерактивност и го подобрува корисничкото искуство, овозможувајќи на корисниците јасно да разберат дека нивната акција е успешно извршена.

Autocomplete од MaterialUI

Во текот на развојот на проектот, искористивме разни компоненти од Material-UI за да го подобриме корисничкото искуство и функционалноста на апликацијата. Една од поинтересните компоненти што ја интегриравме е Autocomplete, која овозможува лесно пребарување и избор на предмети од листата на достапни предмети. Оваа компонента е конфигурирана да прикажува сугестии додека корисникот пишува, со што се олеснува процесот на пронаоѓање на потребниот предмет.

Во нашиот проект, Autocomplete компонента беше особено корисна во ситуации каде што студентите требаа брзо да го најдат и изберат предметот на кој сакаат да ги постават своите прашања, коментари или материјали. Компонентата е конфигурирана со функции како getOptionLabel и getOptionKey, кои овозможуваат правилно поврзување на предметите со нивните уникатни имиња и идентификатори. Исто така, интегрирана е

функција handleTextField, која се активира при секоја промена на текстот во полето за пребарување, овозможувајќи динамично ажурирање на листата на опции.

Со оваа имплементација, студентите можеа брзо и ефикасно да пристапат до потребните информации, што значително го подобри нивното искуство при користењето на апликацијата. Компонентата Autocomplete беше одличен додаток кој придонесе кон интуитивниот и кориснички ориентиран дизајн на системот.

```
<Autocomplete
                    value={activeSubject}
                    onChange={(_event, newValue) => handleChange(newValue)}
                    getOptionLabel={(option) => option.name}
                    getOptionKey={(option) => option.id}
                    options={Object.values(subjects)}
                    size="small"
                    renderInput={(params) => (
                        <TextField
                         onChange={(event) => handleTextField(event)}
                        {...params}
                         label="Најди предмет"
                         InputProps={{
                            ...params.InputProps,
                            endAdornment: <div>{params.InputProps.endAdornment}</div>,
                        }}
                        />
                    )}
                />
```

Форум

Секој предмет има свој форум каде што студентите можат да поставуваат статуси, прашања или коментари и да учествуваат во дискусии. Форумот е дизајниран да поттикне активна размена на идеи и знаење.

Користењето на форумот вклучува:

- Опции за гласување: Секој статус и коментар имаат опции за upvote и downvote, што овозможува на корисниците да ја оценуваат вредноста на информациите и да ги истакнат најкорисните постови.
- Додавање на коментари: За секој пост постои можност за додавање коментари, што им овозможува на студентите да разгледаат различни перцепции и перспективи.
- Информации за авторот: Повеќе информации за авторот на постот, како што се време на објавување и број на коментари, се прикажани за да се подобри транспарентноста и интерактивноста на дискусиите.

На backend-от, оваа функционалност вклучува управување со податоци за постови, коментари и гласови. Секој коментар и пост се чуваат во базата на податоци, и се обработуваат на серверот за да се овозможи нивно правилно прикажување и интеракција на клиентската страна.

Поставување на Материјали

Студентите имаат можност да поставуваат различни типови на датотеки, вклучувајќи PDF, DOCX, JPG и други формати. Оваа функционалност им овозможува на студентите да споделуваат корисни документи како претходни испитни задачи, белешки од предавања и други образовни ресурси.

Клучни аспекти на оваа функционалност:

- Разновидност на формати: Поддржани се различни типови на датотеки, што ги олеснува студентите при споделување на различни ресурси.
- Чување и управување со фајлови: Секој поставен материјал се чува во базата на податоци, за што е потребна специјална обработка на backend-от.

Пример код од делот за поставување материјали

```
@Override

public Attachment saveAttachment(MultipartFile file, String description, Long subjectId, String username) throws Exception {

    String fileName = StringUtils.cleanPath(file.getOriginalFilename());

    try {
```

```
        if (fileName.contains("..")) {
            throw new Exception("Filename contains invalid path sequence " + fileName);
        }

        SubjectDetails subjectDetails = subjectDetailsRepository.findById(subjectId).orElse(null);
        User author = userRepository.findByUsername(username).orElse(null);
        Attachment attachment = new Attachment(fileName,
                file.getContentType(),
                file.getBytes(),
                description,
                subjectDetails,
                author);

        return attachmentRepository.save(attachment);

    } catch (Exception e) {
        throw new Exception("Could not save File: " + fileName);
    }
}
```

Клучни детали:

- Примена на MultipartFile: Овој клас е користен за обработка на фајлови кои се качуваат преку HTTP POST запроси.
- Проверка на патеката на фајлот: Се обезбедува дека името на фајлот не содржи недозволени патеки.
- Чување на фајлови во база: Фајловите се чуваат како бајти во базата на податоци заедно со мета податоците како што се име, тип и опис.

- Ракување со грешки: Се користи try-catch блок за обработка на сите можни грешки при чување на фајлови, со што се обезбедува стабилност и сигурност на системот.

Оваа функционалност не само што овозможува лесно споделување на ресурси, туку исто така ја подобрува доступноста на важни документи за студентската заедница, поддржувајќи ги нивните академски напори.

Регистрација и логирање:

Апликацијата нуди атрактивен и кориснички пријателски интерфејс за регистрација и логирање на корисниците, изграден со Material-UI. Процесот на регистрација е лесен и брз, овозможувајќи им на новите корисници да креираат свои профили. Секој корисник има можност да избере уникатен аватар што го претставува во системот, додавајќи персонализиран допир на корисничките профили. Оваа функционалност не само што ги олеснува корисниците при идентификацијата, туку исто така ја подобрува нивната интеракција со платформата.

Апликацијата нуди атрактивен и кориснички пријателски интерфејс за регистрација и логирање на корисниците. Процесот на регистрација е лесен и брз, овозможувајќи им на новите корисници да креираат свои профили. Секој корисник има можност да избере уникатен аватар што го претставува во системот, додавајќи персонализиран допир на корисничките профили. Оваа функционалност не само што ги олеснува корисниците при идентификацијата, туку исто така ја подобрува нивната интеракција со платформата.

Предизвици и решенија

Една од најголемите потешкотии беше поврзувањето на корисниците со предметите што ги имаат земено. Овој аспект на проектот бараше внимателно управување и интеграција на податоците за да се осигура дека секој корисник може точно да види кои предмети ги следи и кои ресурси се достапни за тие предмети. Имавме грешки на почетокот затоа што ако еден студент земе еден предмет, тоа значеше дека и секој друг студент би го имал избрано тој предмет, иако случајот не беше секогаш таков.

Понатаму, друг предизвик во проектот беше имплементацијата на анимациите со помош на React Three.js. Желбата беше да се создаде интерактивен интерфејс кој ќе биде атрактивен за студентите. Со употребата на три-димензионални модели и анимации, постигнавме интересен и динамичен дизајн кој ги привлекува корисниците.

Исто така, научивме како правилно да зачуваме PDF, слики и DOCX документи во база на податоци, што беше клучно за функционалноста на системот за споделување на материјали. Ова знаење го стекнавме преку рачно управување со бајт податоци и користење на Spring Boot за да ги зачуваме овие датотеки во база.

Тестирање и оптимизација

Тестирањето на апликацијата беше спроведено преку симулација на различни кориснички сценарија. Применети се тестови за функционалностите на форумот, поставувањето на материјали и навигацијата низ различните страници. Оптимизацијата на апликацијата вклучуваше намалување на времето за вчитување на страниците и подобрување на перформансите на анимациите.

Заклучок

Проектот претставува успешен обид за создавање на платформа која ги обединува студентите во споделување и размена на материјали. Преку овој проект, научивме дека за успешна имплементација на ваков систем е потребна тимска работа, трпение и добра комуникација. Покрај техничките аспекти како што се користење на React и Spring Boot за изградба на функционална и стабилна апликација, се соочивме и со предизвици како оптимизација на корисничкиот интерфејс и правилно складирање на датотеки. Ова искуство нè научи на важноста од детално планирање и тестирање за да се избегнат грешки кои можат да влијаат на корисничкото искуство.