ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Кафедра інженерії програмного забезпечення

ЛАБОРАТОРНА РОБОТА №6

з дисципліни «Веб-технології»

.(назва дисципліни)

на тему: «Створення Frontend. Login/Register»

ЛРІПЗс.2401075.06.07.ЛР

(шифр, назва)

Рівень вищої освіти                     Перший (бакалаврський)

Галузь знань                      12 «Інформаційні технології»

Спеціальність     121 «Інженерія програмного забезпечення»

Освітня програма   «Інженерія програмного забезпечення»

Студента. І  курсу, група . ІПЗс-24. \_\_\_\_\_\_\_\_\_\_\_ М.О. Лапко.

            (шифр)                         (підпис)     (ініціали, прізвище)

Викладач канд. техн. наук, доцент  .   \_\_\_\_\_\_\_\_\_\_\_ О.М. Яшина

              (посада, вчене звання, науковий ступінь)         (підпис)   (ініціали, прізвище)

Кількість балів \_\_\_\_\_\_\_

Оцінка за шкалою:

національною\_\_\_\_\_\_\_\_\_/ЄКТС\_\_\_\_

Хмельницький, 2024

ЗВІТ

**Варіант 7**

**Мета:** Розробити та впровадити Frontend для функціоналу авторизації (login) та реєстрації (register) користувачів, забезпечивши інтеграцію з backend через REST API.

**ХІД РОБОТИ**

**Завдання №1.** Підготовка проекту

Налаштування структури проекту.

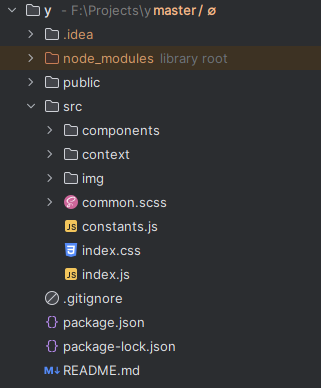


Рисунок 1 — Структура проекту

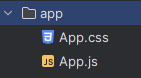


Рисунок 2 — Структура папки app

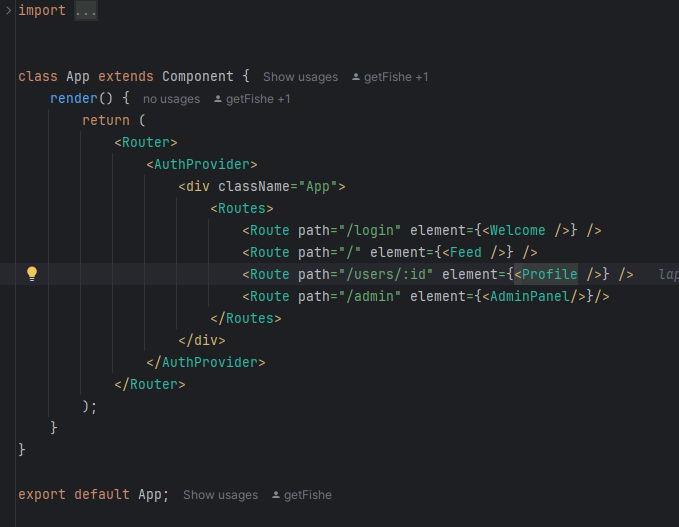


Рисунок 3 — Код компоненту App



Рисунок 4 — Структура папки welcome

Код welcome.jsx:

|  |
| --- |
| import "./welcome.scss"; import Popup from "../popup/popup"; import logo from "../../img/logo.svg"; import { useState } from "react"; import { PopupAuth } from "./popupAuth"; import { PopupLogin } from "./popupLogin";  export default function Welcome() {  const [isPopupOpen, setIsPopupOpen] = useState(false);  const [popupContent, setPopupContent] = useState(null);   const openPopup = (content) => {  setPopupContent(() => content);  setIsPopupOpen(true);  };   const closePopup = () => {  setIsPopupOpen(false);  };   return (  <section className="welcome">  <div className="container">  <div className="welcome-container">  <div className="welcome-icon">  <img src={logo} alt="" />  </div>  <div className="welcome-content">  <div className="title-2"><h2>Join today.</h2></div>  <div className="welcome-google\_\_links">  </div>  <button  onClick={() => openPopup(<PopupAuth />)}  id="signUp" className="btn"><span>Create account</span></button>  <div className="welcome-or">  <span>or</span>  </div>  <div className="welcome-account">  <span>Already have an account?</span>  <button  onClick={() => openPopup(<PopupLogin />)}  id="signIn" className="btn"><span>Sign in</span></button>  </div>  </div>  </div>  </div>  <Popup isOpen={isPopupOpen} onClose={closePopup}>  {popupContent}  </Popup>  </section>  ); } |

Код popupAuth.jsx:

|  |
| --- |
| import { handleInputFocus, handleInputBlur } from "../popup/popup"; import { useRef, useState } from "react"; import { useNavigate } from "react-router-dom"; import {useAuth} from "../../context/authContext"; import {*constants*} from "../../constants"; import {wait} from "@testing-library/user-event/dist/utils";   export function PopupAuth() {  const [email, setEmail] = useState('');  const [pass, setPass] = useState('');  const [confirmPass, setConfirmPass] = useState('');  const [username, setUsername] = useState('');   const {setAccessToken} = useAuth();  const [error, setError] = useState('');  const navigate = useNavigate();    const [isBtnBlocked,setBtnBlocking] = useState(false);   async function dataPickUp(event) {  event.preventDefault();  setError('');  setBtnBlocking(true);   try {  if (!*constants*.loginRegex.test(username)) {  setError('Username Error');  return;  }  if (!*constants*.passwordRegex.test(pass)) {  setError('Password Error');  return;  }  if (pass !== confirmPass) {  setError('Confirm Password Error');  return;  }   let user = {  username: username,  email: email,  password: pass,  };   let response = await fetch(`${*constants*.authApiV1}/register`, {  method: 'POST',  headers: {  'Content-Type': 'application/json;charset=utf-8',  },  body: *JSON*.stringify(user),  });   let json = await response.json();  if (!response.ok) {  setError(`Error request: ${json.message}`);  return;  }   let activate = await fetch(`${*constants*.authApiV1}/activate/${json["activation\_token"]}`, {  credentials: 'include',  });  let responseActivationToken = await activate.json();   if (!activate.ok) {  setError(`Error activation: ${responseActivationToken.message}`);  return;  }   setAccessToken(responseActivationToken.token);  navigate("/");  } catch (error) {  setError(`Unexpected error: ${error.message}`);  } finally {  setBtnBlocking(false);  }  }     const maxLength = 52;   function passwordOnChange(event) {  setPass(event.target.value);   const popupInput = event.currentTarget.closest(".popup-input");   if (username.length > maxLength) {  popupInput.classList.add("error");  } else {  popupInput.classList.remove("error");  }  }      return (  <form onSubmit={dataPickUp} className="popup-tab">  <div className="popup-title"><h3>Create your account</h3></div>  <div onClick={handleInputFocus} className={`popup-input ${username.length > maxLength && 'error'}`}>  <div className="popup-input\_\_text">  <span className='popup-input\_\_text-name'>Name</span>  <span className='popup-input\_\_text-symbols'>{username.length}/{maxLength}</span>  </div>  <input required onBlur={handleInputBlur} type="text" onChange = {(e) => {setUsername(e.target.value)}} />  </div>  <div onClick={handleInputFocus} className="popup-input">  <div className="popup-input\_\_text">  <span className='popup-input\_\_text-name'>Email</span>  </div>  <input required onBlur={handleInputBlur} type="email" onChange = {(event) => {setEmail(event.target.value)}} />  </div>  <div onClick={handleInputFocus} className="popup-input">  <div className="popup-input\_\_text">  <span className='popup-input\_\_text-name'>Password</span>  <span className='popup-input\_\_text-symbols'>{pass.length}/{maxLength}</span>  </div>  <input required onBlur={handleInputBlur} type="password" onChange = {passwordOnChange} />  </div>  <div onClick={handleInputFocus} className="popup-input">  <div className="popup-input\_\_text">  <span className='popup-input\_\_text-name'>Confirm password</span>  </div>  <input required onBlur={handleInputBlur} type="password" onChange = {(event) => {setConfirmPass(event.target.value)}} />  </div>  <button disabled={isBtnBlocked} type='submit' className="popup-next">  <span>Confirm</span>  </button>  {error && <div className="popup-error">{error}</div>}  </form>  ); } |

Код popupLogin.jsx:

|  |
| --- |
| import {handleInputFocus,handleInputBlur} from "../popup/popup"; import {useState} from "react"; import {useAuth} from "../../context/authContext"; import {useNavigate} from "react-router-dom"; import {*constants*} from "../../constants";   export function PopupLogin() {   const [pass, setPass] = useState('');  const [username, setUsername] = useState('');   const {setAccessToken, setRefreshToken} = useAuth();  const [error, setError] = useState('');  const navigate = useNavigate();   let [isBtnBlocked,setBtnBlocking] = useState(false);   async function dataPickUp(event) {  event.preventDefault();  setBtnBlocking(true)   setError('');   try {  let user = {  login: username,  password: pass,  };   let response = await fetch(`${*constants*.authApiV1}/login`, {  method: 'POST',  credentials: 'include',  headers: {  'Content-Type': 'application/json;charset=utf-8',  },  body: *JSON*.stringify(user),  });   let json = await response.json();  if (!response.ok) {  setError(`Error request: ${json.message}`);  return;  }   setAccessToken(json.token);  navigate("/");  } catch (error) {  setError(`Unexpected error: ${error.message}`);  } finally {  setBtnBlocking(false)  }  }    const maxLength = 52;   function loginOnChange(event) {  setUsername(event.target.value);   const popupInput = event.currentTarget.closest(".popup-input");   if (username.length > maxLength) {  popupInput.classList.add("error");  } else {  popupInput.classList.remove("error");  }   }  function passwordOnChange(event) {  setPass(event.target.value);   const popupInput = event.currentTarget.closest(".popup-input");   if (username.length > maxLength) {  popupInput.classList.add("error");  } else {  popupInput.classList.remove("error");  }  }    return (  <form onSubmit={dataPickUp} className="popup-tab">  <div className="popup-title"><h3>Sign in</h3></div>  <div onClick={handleInputFocus} className="popup-input">  <div className="popup-input\_\_text">  <span className='popup-input\_\_text-name'>Login</span>  </div>  <input required onBlur={handleInputBlur} onChange={loginOnChange} type="text"/>  </div>  <div onClick={handleInputFocus} className="popup-input">  <div className="popup-input\_\_text">  <span className='popup-input\_\_text-name'>Password</span>  <span className='popup-input\_\_text-symbols'>{pass.length}/{maxLength}</span>  </div>  <input required onBlur={handleInputBlur} onChange={passwordOnChange} type="password"/>  </div>  <button disabled={isBtnBlocked} type='submit' className="popup-next">  <span>Confirm</span>  </button>  </form>  ) } |

Код welcome.scss:

|  |
| --- |
| @import "../mixins"; @import "../vars"; @import "../fonts";   .welcome {  &-container {  display: flex;  align-items: center;  justify-content: center;  flex-direction: row-reverse;  height: 100vh;  width: 100%;  >\* {  width: 50%;  }   }  &-icon {  width: rem(300);  height: rem(300);  fill:white;  img {  height: 100%;  object-fit: contain;  }  }  &-content {  text-align: left;  h1 {  margin-bottom: rem(50);  }  h2 {  margin-bottom: rem(20);  }  }  &-buttons {  margin-bottom: rem(9);  }  &-description {  width: rem(300);  font-size: remF(11);  line-height: remF(14);  color: #3e3e3e;  margin-bottom: rem(20);  a {  color: var(--blue);  }  }  &-account {  margin-top: rem(10);  >\* {  display: block;  }  >:not(:last-child) {  margin-bottom: rem(20);  }  }  &-or {  text-align: center;  width: rem(300);  position: relative;  margin:rem(8) 0;  &::before, &::after {  content: '';  position: absolute;  height: .5px;  background-color: gray;  border-radius: 1px;  width: 45%;  top: 50%;  }  &::before {  left: 0;  }  &::after {  right:0;  }  } } |

Для підтримки аутентифікації був використаний authContext.jsx

Код authContext.jsx:

|  |
| --- |
| import React , {useContext , useEffect , useState} from 'react'; import {useNavigate} from "react-router-dom"; import {constants} from "../constants";   const AuthContext = React.createContext(null);  export function useAuth() {  return useContext(AuthContext); }  export function AuthProvider({ children }) {  const [loading, setLoading] = useState(true);  const [user, setUser] = useState(null);  const navigate = useNavigate();   let accessToken = "init";  function setAccessToken(token) {  accessToken = token;   //console.log(`[${new Date().toISOString()}] Access Token Changed: `, accessToken)  if (accessToken === "") {  navigate("/login");  setLoading(false)  } else {  updateUser()  }  }   async function sendRequest(requestOptions) {  async function makeRequest() {  if(!requestOptions.headers) {  requestOptions.headers = {};  }  requestOptions.headers['Authorization'] = `Bearer ${accessToken}`;  return await fetch(requestOptions.url, requestOptions).catch((err) => {*console*.error("Network error:", err)});  }   const response = await makeRequest();   if (response.status === 401) {  await refreshToken();  return await makeRequest();  }   return response;  }   useEffect(() => {  refreshToken()  }, [navigate]);   useEffect(() => {  if (user) {  setLoading(false)  }  }, [user])   async function updateUser() {  try {  const firstResp = await sendRequest({url: `${*constants*.authApiV1}/authenticate`});  if (!firstResp.ok) throw new Error(`Failed to fetch user id: ${firstResp.statusText}`);  const firstRespData = await firstResp.json();   const secondResp = await sendRequest({url: `${constants.userApiV1}/users/${firstRespData.user\_id}/full`});  if (!secondResp.ok) throw new Error(`Failed to fetch user details: ${secondResp.statusText}`);   const userData = await secondResp.json();  setUser(userData);  return userData;  } catch (error) {  console.error('Error fetching current user:', error);  throw error;  }  }   async function refreshToken() {  let newAccessToken = "";  try {  const refresh = await fetch(`${constants.authApiV1}/refresh`, {  method: 'POST',  credentials: 'include',  });   if(!refresh.ok) {  return refresh;  }   const refreshResponse = await refresh.json();  newAccessToken = refreshResponse.token;  } catch (e) {  console.error(e);  return e;  } finally {  setAccessToken(newAccessToken);  }  }   async function getUser(id) {  try {  const resp = await sendRequest({url: `${constants.userApiV1}/users/${id}`});  if (!resp.ok) throw new Error(`Failed to fetch user details: ${resp.statusText}`);  return await resp.json();  } catch (error) {  console.error('Error fetching user:', error);  throw error;  }  }   async function changePass(newPassword) {  const response = await sendRequest({  url: `${constants.authApiV1}/change-password`,  method: 'PATCH',  headers: {'Content-Type': 'application/json'},  body: JSON.stringify({password: newPassword}),  })  if (!response.ok) throw new Error(`Failed to change password: ${response.statusText}`);  }   const value = {  setAccessToken,  sendRequest,  user,  updateUser,  getUser,  changePass  }   return (  <AuthContext.Provider value={value}>  {!loading && children}  </AuthContext.Provider>  ); } |

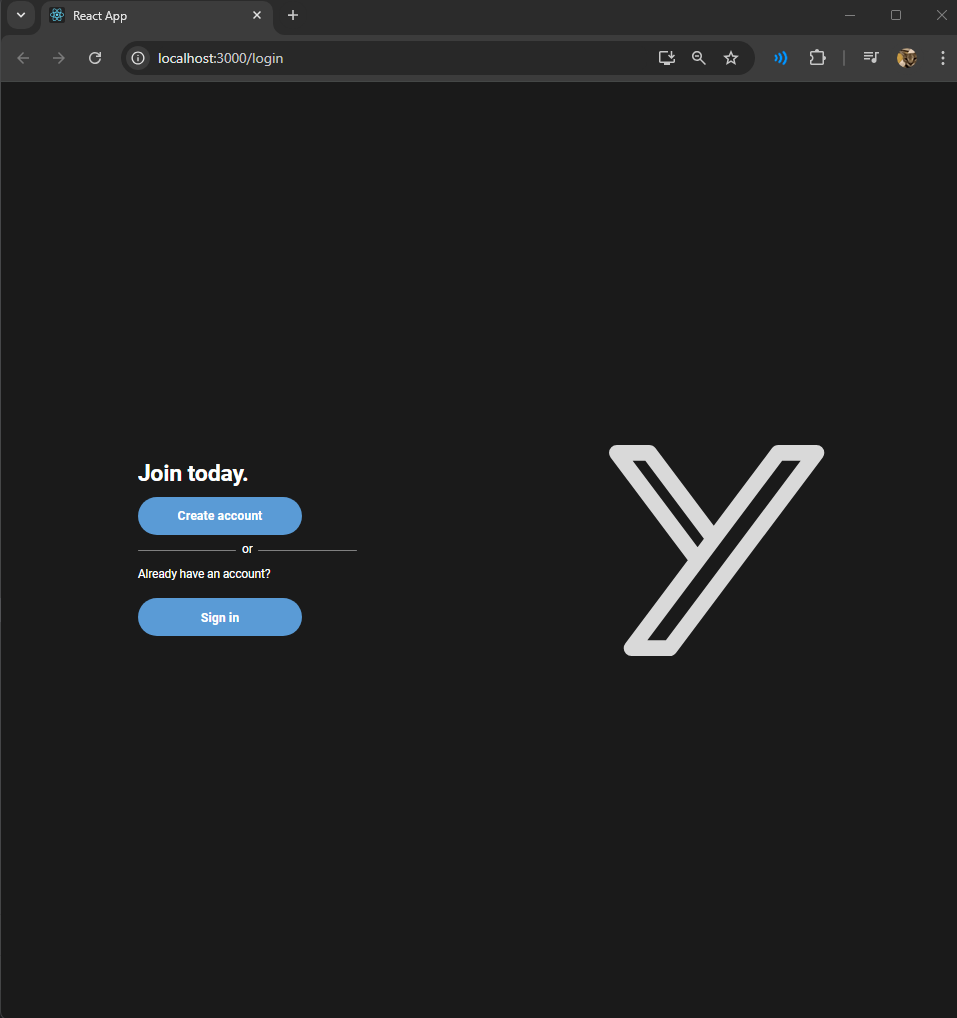


Рисунок 5 — Рисунок сторінки /login

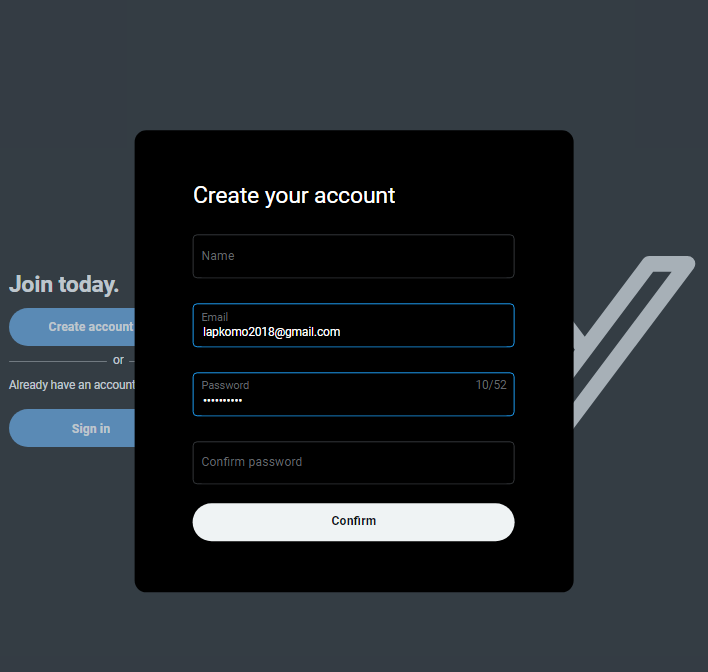


Рисунок 6 — Рисунок діалогового вікна Create Account

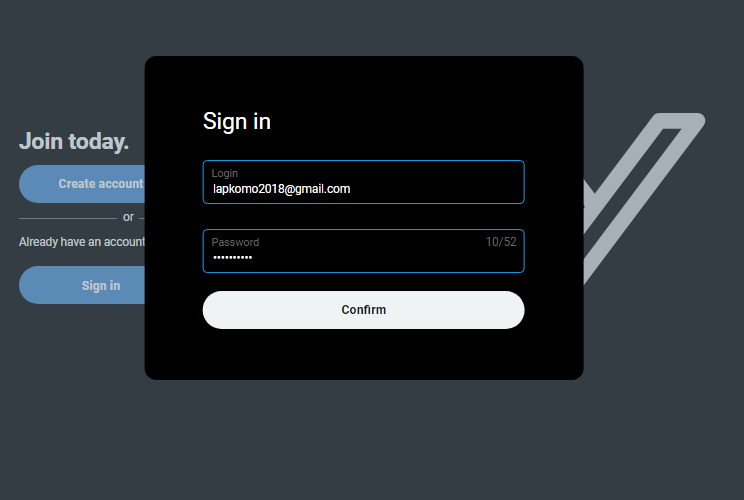


Рисунок 7 — Рисунок діалогового вікна Sign In