МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВОЛИНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ЛЕСІ УКРАЇНКИ
Кафедра комп’ютерних наук та кібербезпеки

На правах рукопису

БОНДАРЧУК ВАЛЕРІЯ ОЛЕКСАНДРІВНА

ПРОЄКТУВАННЯ ТА РОЗРОБКА ІНТЕРФЕСУ КОРИСТУВАЧА LMS СИСТЕМИ ІЗ ВИКОРИСТАННЯМ ФРЕЙМВОРКУ NEXT.JS

Спеціальність: 122 Комп’ютерні науки
Освітньо-професійна програма: Комп’ютерні науки та інформаційні технології
Кваліфікаційна робота на здобуття освітнього ступеня «бакалавр»

Науковий керівник:
ГРИШАНОВИЧ ТЕТЯНА ОЛЕКСАНДРІВНА
кандидат фіз.-мат наук, доцент кафедри
комп’ютерних наук та кібербезпеки

РЕКОМЕНДОВАНО ДО ЗАХИСТУ
Протокол №__
засідання кафедри комп’ютерних наук
та кібербезпеки від _____ 2025 р.
Завідувач кафедри
(_____) Гришанович Т. О.

Луцьк 2025

ЗМІСТ

ВСТУП

Сучасні технології стрімко розвиваються і разом із цим дистанційна освіта набуває все більшої популярності. Онлайн-школи, інтерактивні курси та навчальні платформи надають можливість отримувати знання без прив’язки до місця чи часу, що є надзвичайно зручним для учнів, студентів та фахівців, які прагнуть підвищити свою кваліфікацію. Однак ефективність таких платформ значною мірою залежить від зручності та функціональності їхнього інтерфейсу користувача.

Якісний інтерфейс є ключовим фактором, що впливає на взаємодію користувачів із системою. Він має бути інтуїтивно зрозумілим, естетично привабливим, адаптивним до різних пристроїв та відповідати сучасним стандартам UI/UX-дизайну. Особливо важливо враховувати особливості цільової аудиторії: школярів, студентів, викладачів та старших здобувачів знань, які можуть мати різний рівень цифрової грамотності. Крім того, важливим аспектом є забезпечення доступності для людей з особливими освітніми потребами, що дозволяє зробити навчання інклюзивним.

Актуальність цієї теми обумовлена зростаючою потребою в якісних освітніх платформах, які можуть бути інтегровані в навчальний процес без порушення його структури. Оскільки кількість користувачів онлайн-освіти постійно збільшується, виникає необхідність у створенні інтерфейсів, які сприяють покращенню навчального досвіду, підвищенню рівня мотивації та ефективності засвоєння матеріалу. Інтуїтивний дизайн, продумана структура контенту та інтерактивні елементи можуть суттєво впливати на рівень залученості учасників освітнього процесу.

Метою даної роботи є розробка вебінтерфейсу онлайн-школи, що забезпечить зручну та ефективну взаємодію користувачів із системою,

сприятиме підвищенню якості дистанційного навчання та автоматизації основних організаційних процесів.

Для досягнення поставленої мети потрібно виконати наступні **завдання**:

- розглянути основні принципи побудови інфтерфейсу користувача:
- ознайомитись із сучасними підходами до розробки інтерфейсу користувача:
- здійснити огляд актуальних фреймворків для розробки клієнтської частини вебзастосунків;
- розглянути основні концепції та архітектуру фреймворку Next.js;
- описати призначення програмного засобу та сформулювати вимоги до нього;
- здійснити вибір моделі розробки програмного засобу;
- здійснити загальний опис проєкту;
- описати особливості програмної реалізації;
- здійснити тестування та налагодження;
- сформулювати рекомендації щодо використання та впровадження.

Об’єктом дослідження виступають методи і засоби проєктування та реалізації функціональної архітектури програмного забезпечення, орієнтованої на автоматизацію ключових процесів навчальної діяльності.

Предметом дослідження є процес розробки вебплатформи для організації та підтримки навчального процесу в школі іноземних мов.

РОЗДІЛ 1
АНАЛІЗ ТЕХНОЛОГІЙ І ПІДХОДІВ ДО РОЗРОБКИ ІНТЕРФЕЙСІВ КОРИСТУВАЧА

1.1. Основи проєктування інтерфейсів користувача

Базовою частиною вебсторінки є HTML. Мова гіпертекстової розмітки HTML є стандартною мовою розмітки документів у браузері. Браузери отримують HTML-документи з вебсервера або з локального сховища і перетворюють їх у мультимедійні вебсторінки. HTML описує структуру вебсторінки семантично і спочатку включає підказки для зовнішнього вигляду документа.

Елементи HTML — це будівельні блоки HTML-сторінок. За допомогою конструкцій HTML зображення та інші об’єкти, такі як інтерактивні форми, можуть бути вбудовані у відображену сторінку. HTML надає засоби для створення структурованих документів, визначаючи структурну семантику для тексту, такого як заголовки, абзаци, списки, посилання, цитати та інші елементи. HTML-елементи виділяються тегами, записаними з використанням кутових дужок. Такі теги, як <img /> і <input />, безпосередньо вводять контент на сторінку. Інші теги, зокрема <p>, виконують функцію логічного структурування вмісту HTML-документа, визначаючи окремі текстові блоки та надаючи їм семантичного значення. Такі елементи можуть містити вкладені теги, які уточнюють або доповнюють зміст основного елементу, формуючи ієрархічну структуру розмітки документа. Браузери не відображають HTML-теги, але використовують їх для інтерпретації змісту сторінки.

HTML може вбудовувати програми, написані мовою сценаріїв, такою як JavaScript, яка впливає на поведінку і зміст вебсторінок. Використання CSS

визначає зовнішній вигляд і компонування контенту. Консорціум World Wide Web (W3C), колишній розробник HTML і поточний розробник стандартів CSS, з 1997 року заохочує використання CSS замість явного презентаційного HTML. [14]

Каскадні таблиці стилів (CSS) – це мова таблиць стилів, яка використовується для опису представлення документа, написаного мовою розмітки, такою як HTML. [3] CSS розроблений для відокремлення представлення та змісту, включаючи макет, кольори та шрифти. Це відокремлення здатне підвищити доступність контенту, запровадити гнучкість і контроль властивостей представлення, дозволяє низці вебсторінок спільно використовувати форматування шляхом вказівки відповідного CSS у окремому файлі .css, що зменшує складність і повторення структурного контенту, а також дозволяє створити файл .css, який кешується для покращення швидкості завантаження сторінок, що використовують спільне форматування. [3]

JavaScript, часто скорочено JS, є мовою програмування, яка відповідає специфікації ECMAScript. JavaScript – це високорівнева, інтерпретована та мультипарадигмальна мова програмування. JavaScript є C-подібною об’єктно-орієнтованою мовою з динамічною типізацією.[13]

Разом з HTML і CSS, JavaScript є однією з основних технологій всесвітньої павутини. JavaScript дозволяє створювати інтерактивні вебсторінки і є важливою частиною вебдодатків. Переважна більшість вебсайтів використовують його для керування поведінкою сторінки на клієнтському боці і всі основні веббраузери мають спеціальний механізм JavaScript для його виконання.

Як мова з кількома парадигмами, JavaScript підтримує керовані подіями, функціональні та імперативні стилі програмування. Він має інтерфейси прикладного програмування (API) для роботи з текстом, датами, регулярними

виразами, стандартними структурами даних та об’єктною моделлю документа (DOM).

Спочатку рушії JavaScript використовувалися лише у браузерах, але зараз вони вбудовані у деякі сервери, зазвичай через Node.js. Вони також вбудовані у різні додатки, створені за допомогою таких фреймворків, як Electron і Cordova.

На JavaScript написано багато додатків для різних сфер та потреб. Для полегшення розробки додатків існують різноманітні бібліотеки та фреймворки. Фреймворки дозволяють розробнику абстрагуватися від рутинних і тривалих операцій і зосередитися на розумінні та розробці логіки додатка. Однак вибір фреймворка, окрім очевидних переваг, накладає певні обмеження на розробку і функціонал. Тому важливим є вибір відповідного фреймворка для розроблюваного додатка.

1.2. Сучасні підходи фронтенд розробки

Більшість підприємств прийняли Інтернет як свою бізнес-платформу, і ця тенденція посилюється. Таким чином, існує потреба в розробці швидких, надійних і стійких ресурсів, які реалізують актуальні та перспективні технології.

1.2.1. Односторінкові застосунки

Із самого початку розвитку вебтехнологій розробники прагнули до створення вебзастосунків, які б за візуальними характеристиками та користувацьким досвідом наближалися до нативних настільних програм.

Упродовж цього процесу було випробувано низку рішень, спрямованих на забезпечення більш природної та інтерактивної взаємодії з користувачем, зокрема IFrames, Java-аплети, Adobe Flash та Microsoft Silverlight. Попри

технічні відмінності між цими підходами, усі вони переслідували спільну мету — надати вебзастосункам функціональні можливості, властиві повноцінним настільним програмам, водночас зберігаючи переваги кросплатформного браузерного середовища. Концепція односторінкового вебзастосунку (SPA) також реалізує цю мету, однак без необхідності використання сторонніх модулів браузера або вивчення додаткових мов програмування.

На початку 2000-х років у вебдизайні з’явився новий підхід, пов’язаний із розвитком технології AJAX. Спочатку це починалося з використання елемента ActiveX у браузері Microsoft Internet Explorer, який дозволяв асинхронно надсилати і отримувати дані. Хоча цей компонент спочатку був досить складним і незрозумілим, згодом його функціонал був стандартизований у вигляді API XMLHttpRequest (XHR), який підтримували основні браузери. Розробники почали поєднувати цей API з JavaScript, HTML і CSS, що дозволило створювати більш динамічні і швидкі вебсторінки. Такий підхід отримав назву AJAX — асинхронний JavaScript і XML. Використання AJAX дало змогу надсилати запити на сервер без перезавантаження сторінки, динамічно оновлювати її вміст через зміну структури документа (DOM) і змінювати стиль за допомогою CSS, що значно покращило користувацький досвід.

Подальший розвиток цієї ідеї привів до появи односторінкових застосунків (SPA), які розширили принципи AJAX на весь вебзастосунок. Використання SPA супроводжується певними шаблонами і методами розробки, що дозволяють підвищити ефективність створення, підтримки коду і скоротити час розробки програмного забезпечення.

Як і у випадку з більшістю нових технологій, дизайн односторінкових застосунків (SPA) охоплює різноманітні підходи. Різні погляди сучасних фахівців, а також велика кількість конкуруючих бібліотек і фреймворків ускладнюють вибір оптимального рішення для конкретного SPA-проєкту.

Односторінковий застосунок (SPA) — це вебзастосунок, в якому маршрутизацію обробляє не серверна частина, а клієнтський JavaScript. SPA швидко і легко використовувати, тому що користувачеві не потрібно чекати перезавантаження сторінок під час відвідування різних сторінок одного ресурсу і не потрібно завантажувати різні повторювані елементи сторінки, такі як шапка, сайдбар або підвал сайту. Застосунок завантажує стан на льоту. Це призводить до поліпшення взаємодії з користувачем, оскільки застосунок має високу продуктивність. Після початкового завантаження сторінки всі інструменти, необхідні для створення і відображення представлень, завантажуються і готові до використання. Якщо необхідне нове представлення, воно створюється локально в браузері й динамічно прикріплюється до DOM через JavaScript. Жодних оновлень браузера не потрібно. Оскільки наша логіка представлення в SPA в основному є клієнтською, завдання об’єднання HTML і даних переноситься з сервера в браузер. Як і на стороні сервера, вихідний HTML-код містить заповнювачі, в які мають бути вставлені дані (і, можливо, інші інструкції з рендерингу).

Браузер, як і раніше, залишається відмінним способом поширення програмного забезпечення через його повсюдне поширення і стандартизоване середовище. У кінцевих користувачів вже є браузер. Він також відмінно підходить для оновлень програмного забезпечення, оскільки оновлення відбуваються на сервері, а користувачам не потрібно турбуватися про процес встановлення. На жаль, перезавантаження сторінок, дублювання контенту під час кожного запиту і великі обсяги транзакцій зменшують переваги контенту, що доставляється браузером. Однак взаємодія з клієнтами через SPA дає змогу надати кінцевим користувачам найкращі можливості.

1.2.2. Прогресивні вебзастосунки

Прогресивний вебзастосунок (PWA) - це нове покоління вебзастосунків. Він пропонує схожий користувацький досвід, як і під час використання нативних застосунків під час роботи з вебсайтом, який використовує функції PWA. Зокрема, PWA забезпечує можливість перегляду вебсторінок без інтернету після попереднього разового завантаження, а також інтерактивні користувацькі сервіси, повністю використовуючи кеш, push-повідомлення і сервісну робочу силу. Сукупність цих нових функцій HTML5 стирає межу між нативними і вебзастосунками, особливо на мобільних пристроях, сприяючи використанню вебзастосунків з короткими затримками поставки даних і більш коротким шляхом публікації та доступу користувача.

Насправді це не нова технологія чи фреймворк, а набір кращих практик, які широко застосовуються в Інтернеті для створення користувацького досвіду, близького до нативних застосунків. Такі складові, як Service Worker, AppShell, маніфест вебзастосунку та push-повідомлення, у взаємодії забезпечують унікальні властивості прогресивних вебзастосунків (PWA). PWA можна встановити на головний екран пристрою безпосередньо з вебсайту одним дотиком. Користувач отримує можливість працювати у повноекранному режимі без зайвих затримок, характерних для завантаження. Завдяки кешуванню та іншим механізмам PWA забезпечує швидке завантаження навіть у нестабільних мережах та підтримує автономний режим роботи. Крім того, PWA має можливість надсилати релевантні push-повідомлення, що сприяє підвищенню залученості користувачів.

Популярність PWA зростає дуже швидкими темпами у сфері електронної комерції, бізнесу, новинних онлайн-порталів та інших областях. Ключовою перевагою даної технології є те, що PWA працює для всіх користувачів у всіх браузерах і має можливість своєчасного оновлення, надаючи користувачам

завжди актуальний функціонал і релевантну інформацію. Новий опублікований контент приходить користувачеві у вигляді оновлення, щойно користувач під’єднується до Інтернету. Оболонка і вміст застосунку після кешування завжди завантажуються з локального сховища.

Також, крім системи оновлень, є Push-повідомлення, які працюють за рахунок інтерфейсів Notifications Push API і Notifications API. Дані інтерфейси підвищують ймовірність повторного відвідування PWA користувачем. Користувач може отримувати повідомлення, відправлені з сервера, які можуть відображатися як повідомлення. Сповіщення прогресивних вебзастосунків повністю природні й схожі на сповіщення власних застосунків.

Крім цього, в плані доступності PWA варто відзначити структуру вебсайтів, що збереглася, яка підходить для всіх форм, факторів і розмірів пристроїв: мобільних, настільних і планшетних. Адаптивна функція досягається за рахунок дизайну, концепції гнучкої сітки та медіа-запитів CSS 3. Доступність даних застосунків також обумовлюється маніфестом вебзастосунку W3C, за рахунок чого пошукові системи ідентифікують PWA як застосунок. Це збільшує ймовірність відображення в пошукових системах порівняно з нативними застосунками, доступними у відповідних магазинах. Впровадження сервіс-воркерів дає змогу PWA працювати в автономному режимі та забезпечувати хорошу продуктивність навіть у локальній мережі. Застосунок розглядає втрату підключення не як помилку, а як випадок, який можна запланувати й обробити без зволікання.

Важливим моментом є обов’язкове застосування HTTPS-з’єднання та SSL-сертифіката для обслуговування сторінки, що дає змогу запобігти атакам типу «зловмисник посередині», злому пароля та маніпуляціям із контентом.

PWA є продуктом Інтернету, тобто швидше і дешевше для розробки, знижуючи витрати на залучення користувачів і завжди мають актуальну версію функціоналу та контенту. Проте PWA стикається з деякими проблемами.

З проблем технології варто відзначити проблеми з кросбраузерністю, внаслідок чого деякі браузери мають обмежену функціональність, оскільки вони не можуть отримати доступ до всього апаратного забезпечення для конкретного пристрою. Також відсутня підтримка входу в систему між застосунками.

1.2.3. Тіньова і віртуальна об’єктна модель документа

DOM (об’єктна модель документа) - це фундаментальна концепція в клієнтській розробці.[12] Маніпуляції з DOM не такі прості й зручні, і, що найважливіше, вони створюють безліч проблем із продуктивністю. Наразі існують дві основні концепції DOM: Shadow DOM і Virtual DOM, які з’явилися з прогресивними вебфреймворками, такими як Angular, React.js і Vue.js.

DOM - це API для документів HTML або XML, і він створює логічну структуру, до якої можна отримати доступ і якою можна керувати. Іншими словами, Javascript може отримувати доступ і вносити зміни в об’єктну модель документа. Причина реалізації об’єктної моделі документа полягала в тому, щоб надати стандартний інтерфейс програмування, який можна було б використовувати з будь-якою мовою програмування в різних середовищах.

Під модифікацією DOM ми можемо розуміти додавання, видалення або зміну елементів вебсайту, призначення їм різної поведінки і т. Д. Кожен браузер має свій глобальний об’єкт, який називається window. Всередині window є різні властивості та методи. Одна з властивостей об’єкта window - це документ, в якому ми можемо знайти безліч властивостей і методів, які можна використовувати для доступу до елементів DOM для взаємодії з ними.

Shadow DOM - це інструмент, який використовується для створення застосунків і вебсайтів на основі компонентів. Shadow DOM складається з невеликих частин і не представляє всю об’єктну модель документа. Ми можемо

розглядати його як піддерево або як окрему DOM для елемента. Тіньовий DOM можна представити як цеглини, з яких створюється DOM. Основна відмінність між DOM і Shadow DOM полягає в тому, як він створюється і як поводиться. Зазвичай вузли DOM, які ми створюємо, розміщуються всередині інших елементів, як у дереві, яке ми бачили раніше. У випадку Shadow DOM ми створюємо дерево з областю видимості, яке пов’язане з елементом, але відокремлене від дочірніх елементів. Воно називається тіньовим деревом, а елемент, до якого воно прикріплене, називається тіньовим хостом. І тут ми підходимо до великої переваги Shadow DOM: все, що ми додамо в Shadow DOM, є локальним, навіть стилі.

Перш за все, він ізолює DOM, тому DOM компонента є окремим елементом, який не відображається в глобальній DOM. Ще одна проблема, з якою він допомагає, визначення обсягу CSS, що означає, що стилі, створені всередині одного елемента Shadow DOM, ізольовані та залишаються в рамках цього Shadow DOM. Це значно спрощує стилізацію, оскільки нам не потрібно сильно турбуватися про іменування простору, і ми можемо використовувати прості селектори та імена класів. Крім того, ми можемо думати про застосунок, як про те, що він побудований з блоків (насправді він заснований на компонентах), а не як про один масивний глобальний об’єкт. Shadow DOM може вплинути на продуктивність застосунку. У випадку з Shadow DOM браузер знає, яку частину потрібно оновити.

Віртуальний DOM - це концепція DOM, яка використовується React.js і Vue.js. У концепції віртуального DOM копія DOM зберігається в пам’яті, і, хоча будь-які зміни виконуються в DOM, вона порівнюється, щоб знайти відмінності. Потім браузер знає, які елементи були змінені, і може оновлювати тільки цю частину застосунку, щоб уникнути повторного рендерингу всієї DOM. Це зроблено для підвищення продуктивності бібліотек користувацького інтерфейсу. Як ми знаємо, з попереднього абзацу в DOM кожен елемент

перемальовується, незалежно від того, чи був він змінений, чи ні. У концепції Virtual DOM можна застосувати більше однієї зміни одночасно, щоб уникнути повторного рендерингу для кожної окремої зміни елемента. Найбільша проблема, яку вирішує Virtual DOM, - це підвищення продуктивності під час маніпулювання DOM.

Єдине, що їх об’єднує, це те, що вони допомагають із проблемами продуктивності. Обидві створюють окремий екземпляр об’єктної моделі документа, крім цього, обидві концепції різні. Virtual DOM створює копію всього об’єкта DOM, а Shadow DOM створює невеликі частини об’єкта DOM, які мають свою власну ізольовану область дії для елемента, який вони представляють.

1.3. Огляд актуальних фреймворків для розробки клієнтської частини вебзастосунків

У сфері програмування немає чіткого визначення, яким критеріям має відповідати програмне забезпечення, щоб достовірно вважатися фреймворком. Іноді фреймворки плутають із бібліотеками.

Бібліотека зазвичай містить заздалегідь написаний код, класи, процедури, скрипти, конфігурації тощо. В основному її можна легко інтегрувати в існуючий проєкт для скорочення часу розробки. Це пояснюється тим, що багато питань, пов’язаних із основними алгоритмами та функціями, вже були вирішені іншими досвідченими програмістами. Бібліотека економить час розробників, дозволяючи їм зосередитися на проблемах, пов’язаних із логікою. Водночас використання сторонньої бібліотеки може нести потенційні ризики для застосунку.

На відміну від бібліотек, фреймворки пропонують повний стек корисних функцій і беруть на себе відповідальність за архітектурні рішення під час

розробки. Вони можуть містити механізми маршрутизації URL-адрес у застосунку, управління станом, об’єднання компонентів тощо. Крім того, фреймворки забезпечують покращення робочого процесу, зокрема найкращі практики для ключових аспектів розробки, таких як загальна структура застосунку або створення шаблонів.[4]

1.3.1. Огляд фреймворку Angular

Angular — це повнофункціональний фреймворк, що охоплює більшість потреб розробників. З іншого боку, Angular може здатися складним для вивчення та використання, оскільки потребує освоєння великого набору взаємодіючих елементів (сервіси, впровадження залежностей тощо).

Angular надає повний набір інструментів для розробки масштабованих інтерфейсів. На відміну від React, який часто потребує використання сторонніх бібліотек для таких функцій, як маршрутизація, Angular має вбудовані рішення. Це створює єдине середовище розробки з визначеними правилами та принципами. Чітка та сувора архітектура Angular є перевагою для великих команд і корпоративних проєктів, оскільки сприяє підтримуваності та масштабованості коду.

1.3.2. Огляд фреймворку React

Сильна сторона React полягає в тому, що він органічно виріс завдяки широкому використанню у світі та продовжує розвиватися у відповідь на інтенсивне використання. Він зазнав значного зростання, але його основні переваги залишаються незмінними: React є фреймворком, який використовується Facebook для власних застосунків.

React використовує незмінні стани об’єкта, доступні тільки через setState(), для представлення стану компонента. Це відрізняє його від Vue та Angular, які застосовують більш інтегрований підхід до керування станом даних у JavaScript.[5]

React використовує JSX для шаблонів представлення. JSX — цікавий підхід, оскільки він нагадує HTML із розширеними можливостями JavaScript (або JavaScript із розширеними можливостями HTML). JSX може спочатку здатися незвичним, але в довгостроковій перспективі він добре працює та його не складно вивчити. Централізоване управління даними в React за замовчуванням здійснюється через Redux.

1.3.3. Огляд фреймворку Vue

У певному сенсі Vue займає проміжне місце між Angular і React, будучи компромісом між ієрархічним дизайном Angular і органічним зростанням React. Попри те, що Vue є відносно новим фреймворком і не має підтримки великої корпорації, він не відстає від сучасних розробок і надає повноцінну інфраструктуру. Також існує низка якісних плагінів і бібліотек для Vue (наприклад, Quasar і Vuetify).

Vue має репутацію найбільш простого у вивченні фреймворку. Це, ймовірно, пов’язано з тим, що він використовує модель даних JSON і декларативні шаблони (на відміну від JSX у React). Шаблони Vue також можуть містити вбудовані функції JavaScript, чого не дозволяє JSX. Для централізованого управління станом Vue пропонує Vuex як вбудоване рішення.

1.4. Основні концепції та архітектура Next.js

Next.js є масштабованою та високопродуктивною платформою на базі React.js, що відкриває широкі можливості для сучасної веброзробки. Завдяки розширеному функціоналу, зокрема гібридному рендерингу, автоматичній оптимізації зображень, попередньому завантаженню маршрутів та підтримці інтернаціоналізації, цей фреймворк забезпечує ефективну роботу вебдодатків будь-якої складності.

Завдяки своїй універсальності Next.js може бути використаний для широкого спектру проєктів. Він є ідеальним вибором для розробки електронних комерційних платформ, блогів, корпоративних сайтів, а також складних вебдодатків. Використовуючи цей фреймворк, розробники можуть створювати продуктивні, зручні для користувачів рішення без шкоди для якості взаємодії чи задоволення від процесу розробки. Починаючи з базових концепцій, Next.js дозволяє поступово освоїти всі його можливості, створюючи реальні проєкти з детальними покроковими поясненнями. Окрім вибору оптимальної методології візуалізації, цей фреймворк забезпечує безпеку, гнучкість у розгортанні на різних платформах та високу продуктивність.

1.4.1. Вступ до Next.js

Next.js – це сучасний JavaScript-фреймворк з відкритим вихідним кодом для React, який надає потужний набір функцій, включаючи серверний рендеринг, статичну генерацію сайту та поступову статичну регенерацію. Ці можливості роблять Next.js універсальним інструментом як для корпоративних додатків, так і для невеликих вебпроєктів.

Значною перевагою Next.js є його зручна екосистема, яка дозволяє розробникам уникати зайвих конфігурацій і зосередитися на створенні якісних

цифрових продуктів. Використання цього фреймворку сприяє покращенню продуктивності, зниженню навантаження на сервери та прискоренню відображення контенту для кінцевих користувачів.[6]

1.4.2. Основні технічні вимоги для роботи з Next.js

Для початку роботи з Next.js необхідно встановити відповідні інструменти. Основними компонентами, без яких неможливий запуск проєкту, є Node.js та npm. Детальні інструкції щодо встановлення цих інструментів можна знайти на офіційному вебсайті або у відповідних технічних ресурсах.

Також існує можливість працювати з Next.js у хмарних середовищах без необхідності локального встановлення. Онлайн-платформи, такі як CodeSandbox та Repl.it, дозволяють розробникам експериментувати з кодом у реальному часі.

1.4.3. Еволюція веброзробки та поява Next.js

Розвиток веброзробки пройшов довгий шлях, і з появою сучасних JavaScript-фреймворків, таких як React, Vue та Angular, створення інтерактивних вебдодатків стало значно простішим. React, як один із найпопулярніших фреймворків, був розроблений у Facebook і надав розробникам можливість створювати багаторазові компоненти з високою продуктивністю.

Однак класична модель роботи React має певні недоліки, зокрема проблему пошукової оптимізації (SEO) та тривалий час завантаження контенту. Щоб подолати ці обмеження, компанія Vercel представила Next.js, який відкрив можливості гібридного рендерингу, автоматичного поділу коду та

передзавантаження маршрутів. Це зробило Next.js важливим інструментом для розробки високоефективних вебрішень.[8]

Сьогодні Next.js активно використовується провідними світовими компаніями, серед яких Netflix, TikTok, Uber, Nike та багато інших. Завдяки своїй продуктивності та гнучкості, цей фреймворк став стандартом у розробці сучасних вебдодатків.

1.4.4. Сфери застосування Next.JS

Технологія Next.JS є досить універсальною, завдяки чому її можна використовувати для створення широкого спектра вебзастосунків — від простих статичних сайтів до складних односторінкових застосунків (SPA). Однак давайте розглянемо більш конкретні приклади використання цього фреймворку.

Next.JS ідеально підходить для створення електронних торгових платформ завдяки здатності швидко завантажувати сторінки, оптимізувати час завантаження та забезпечувати високу продуктивність. Його легко інтегрувати з різними системами управління контентом (CMS) та платіжними шлюзами, що робить Next.JS чудовим рішенням для електронної комерції.

Звісно, якщо йдеться про невеликі онлайн-магазини, використання Next.JS не є обов’язковим, оскільки сторінки при належній оптимізації й так завантажуються швидко. Проте для великих інтернет-магазинів або маркетплейсів із сотнями чи тисячами товарів Next.JS може стати найкращим варіантом, забезпечуючи чудову продуктивність та швидкість завантаження сторінок.

Корпоративні вебсайти зазвичай вимагають високої продуктивності, безпеки та SEO-оптимізації. Next.JS забезпечує всі ці параметри завдяки можливості статичного та серверного рендерингу, а також вбудованій SEO-

оптимізації, що робить його чудовим вибором для створення корпоративних вебсайтів.

Next.JS дозволяє створювати потужні й масштабовані сайти, використовуючи статичний рендеринг (SSG) для статичних сторінок (наприклад, сторінок про компанію чи контактної інформації) та серверний рендеринг (SSR) для динамічних розділів, таких як новини чи блоги. Це забезпечує високу продуктивність і легку підтримку, що дозволяє користувачам швидко знаходити необхідну інформацію.[10]

SPA-застосунки потребують швидкого завантаження та плавної навігації без перезавантаження сторінки. Next.JS забезпечує чудову підтримку для створення SPA завдяки автоматичному розподілу коду та використанню React для побудови інтерфейсу.

Завдяки Next.JS можна легко створювати інтерактивні односторінкові застосунки, що швидко реагують на дії користувачів і забезпечують високий рівень продуктивності. Крім того, використання SSR для певних сторінок покращує SEO. Next.JS також підтримує інтеграцію з різними бібліотеками й інструментами, що розширює можливості розробників. Це особливо важливо для застосунків із складними формами, інтерактивними картами чи навіть іграми, де користувацький досвід є ключовим.

Портфоліо та блоги часто містять великий обсяг контенту, зокрема графічного, що може впливати на швидкість завантаження й загальний користувацький досвід. Next.JS дозволяє створювати статичні сайти за допомогою SSG, що забезпечує швидке завантаження та високу продуктивність.

Крім того, можливість динамічного завантаження контенту та інтеграції з CMS дозволяє легко оновлювати й керувати матеріалами. Це робить Next.JS ідеальним вибором для створення персональних портфоліо, блогів та сайтів для спеціалістів різних професій. Вбудовані інструменти оптимізації зображень і

автоматичний розподіл коду забезпечують швидку роботу сайту навіть при великій кількості медіафайлів.

1.4.5. Ключові переваги Next.JS

Якщо говорити коротко, Next.JS має низку дійсно важливих переваг:

- швидке завантаження вебзастосунків завдяки вбудованому серверному рендерингу;
- підтримка експорту статичних сайтів;
- автоматичний розподіл коду для сторінок;
- будована підтримка маршрутизації сторінок, CSS, JSX і TypeScript;
- швидке додавання плагінів для налаштування Next.JS відповідно до вимог конкретного проєкту.

Проте варто детальніше розглянути ключові переваги Next.js, які забезпечують його популярність серед розробників.

Next.JS забезпечує автоматичну оптимізацію продуктивності, що дозволяє створювати швидкі та ефективні вебзастосунки. Це досягається завдяки автоматичному розподілу коду та завантаженню лише необхідних ресурсів. Така оптимізація зменшує час завантаження сторінок, що позитивно впливає на користувацький досвід.

Швидке виконання запитів і завантаження сторінок також є важливим аспектом для SEO. Якщо сайт завантажується повільно, це негативно позначається на ранжуванні в пошукових системах і підвищує рівень відмов. Натомість швидке завантаження покращує SEO-показники та сприяє зменшенню кількості відмов на сайті.

Завдяки серверному рендерингу (SSR) Next.JS допомагає покращити індексацію сайту пошуковими системами. Це особливо важливо для бізнесів, які залежать від органічного трафіку. Коли пошукові роботи отримують готовий

HTML-код, вони легше аналізують і індексують контент, що може підвищити рейтинг сайту в пошуку.

Водночас слід розуміти, що використання Next.JS — це лише один із кроків до ефективної SEO-оптимізації. Він значно покращує показники, проте не скасовує необхідність виконання додаткових заходів внутрішньої та зовнішньої оптимізації.[11]

Next.JS підтримує безшовну інтеграцію з різними сторонніми сервісами й API, що дозволяє легко розширювати функціонал вебзастосунку. Це включає інтеграцію з аналітичними інструментами, платіжними системами, системами авторизації тощо.

Звісно, й інші популярні фреймворки підтримують інтеграцію зі сторонніми сервісами, проте в Next.JS цей процес є значно простішим і швидшим. Це забезпечує більшу гнучкість у розробці та допомагає досягти очікуваного результату без значного впливу на продуктивність сайту.

Фреймворк має вбудовану підтримку TypeScript, що дозволяє використовувати всі переваги статичної типізації. Це допомагає зменшити кількість помилок у коді та спрощує його підтримку, що особливо важливо для великих проєктів.

TypeScript покращує читабельність коду та спрощує співпрацю між розробниками, оскільки всі типи й інтерфейси чітко визначені. Це також підвищує надійність та якість програмного забезпечення, що особливо корисно для проєктів із залученням різних команд на різних етапах розробки.

РОЗДІЛ 2
ПРОЄКТУВАННЯ ТА РОЗРОБКА ІНТЕРФЕЙСУ LMS СИСТЕМИ З ВИКОРИСТАННЯМ ФРЕЙМВОРКУ NEXT.JS

2.1. Призначення та вимоги до програмного засобу

Розробка програмного засобу для онлайн-навчання іноземних мов передбачає реалізацію комплексної вебплатформи, яка має забезпечити повноцінний цикл дистанційного освітнього процесу. Основним завданням є створення зручного середовища для трьох категорій користувачів: адміністраторів, викладачів і студентів. Платформа повинна забезпечувати ефективну комунікацію між усіма учасниками навчального процесу, підтримувати гнучке управління навчальними групами, автоматизоване створення розкладу занять, інтерактивну організацію уроків та зручний механізм перевірки знань.

З боку адміністратора необхідно передбачити функціональність для реєстрації користувачів, підтвердження оплати за навчання, призначення студентів до груп, створення навчального розкладу та призначення викладачів на відповідні курси. Крім того, адміністратор має мати змогу переглядати загальну аналітику, що стосується перебігу навчання, ефективності груп та прогресу студентів.

Зі сторони викладача потрібно реалізувати особистий кабінет, у якому буде доступ до розкладу занять, навчальних груп, результатів студентів та домашніх завдань. Під час уроку викладач повинен мати змогу демонструвати матеріали, взаємодіяти зі студентами в реальному часі, надавати усні або письмові коментарі, а також призначати домашні завдання. Після заняття викладач повинен мати доступ до надісланих робіт, виставляти оцінки,

залишати індивідуальні відгуки та формувати загальну картину успішності студента.

Для студентів передбачається реалізація функціональності перегляду розкладу, приєднання до занять безпосередньо через платформу, доступу до навчальних матеріалів, виконання завдань та спілкування з викладачем. Особистий кабінет студента має містити інформацію про відвідуваність, успішність, пройдені уроки та заплановані іспити. Іспити повинні створюватися автоматично наприкінці курсу та містити різні типи завдань, зокрема тестові питання, відкриті відповіді та усну частину, яка проводиться через відеозв’язок. Результати іспиту мають зберігатися в системі та бути доступними для перегляду.

Окрему увагу необхідно приділити адаптивності платформи, щоб усі функції були доступними як на комп’ютерах, так і на мобільних пристроях. Інтерфейс має бути інтуїтивно зрозумілим, мінімалістичним і орієнтованим на швидку взаємодію з основними функціями системи. Платформа має забезпечити швидкий доступ до матеріалів, гнучке планування занять, оперативну перевірку завдань та зручну систему зворотного зв’язку. Загалом програмний засіб має інтегрувати в собі всі ключові компоненти сучасного онлайн-навчання: реєстрацію, оплату, розклад, проведення занять, перевірку знань, комунікацію та аналітику. Реалізація таких функцій дозволить забезпечити безперервний освітній процес, підвищити залученість студентів та ефективність викладання.

2.2. Вибір моделі розробки програмного засобу

Розробку онлайн-платформи для школи іноземних мов доцільно виконувати із застосуванням методології Agile, яка забезпечує поетапне впровадження функціональності, гнучке реагування на зміни та врахування

зворотного зв’язку користувачів. На першому етапі проводиться збір і аналіз вимог трьох основних ролей — адміністратора, викладача і студента. Далі проєктується архітектура системи, визначається структура бази даних та логіка взаємодії між компонентами.

У процесі реалізації платформи поступово впроваджуються основні модулі: реєстрація користувачів, створення навчальних груп, інтеграція з відеосервісом Zoom, автоматизація перевірки завдань, ведення таблиці успішності, проведення фінального оцінювання тощо. Після завершення розробки виконується тестування функціональності, безпеки та продуктивності, після чого система впроваджується в експлуатацію.

Agile-підхід забезпечує гнучкість і адаптивність розробки, що дозволяє своєчасно враховувати нові вимоги та забезпечувати зручний і ефективний освітній процес.

2.3. Загальний опис проєкту

Онлайн-платформа школи іноземних мов створюється для організації навчального процесу в зручному цифровому середовищі. Вона поєднує всі необхідні функції для адміністраторів, викладачів і студентів, забезпечуючи просту комунікацію, ефективне управління навчальними групами та інтеграцію сучасних технологій для онлайн-освіти.

Головна ідея платформи – створення гнучкого та інтерактивного освітнього простору, де навчання відбувається безпосередньо на сайті, а всі процеси – від запису студента на курс до фінального іспиту – автоматизовані. Важливими компонентами платформи є онлайн-уроки, інтерактивні завдання, система управління групами, розклад занять і моніторинг успішності.

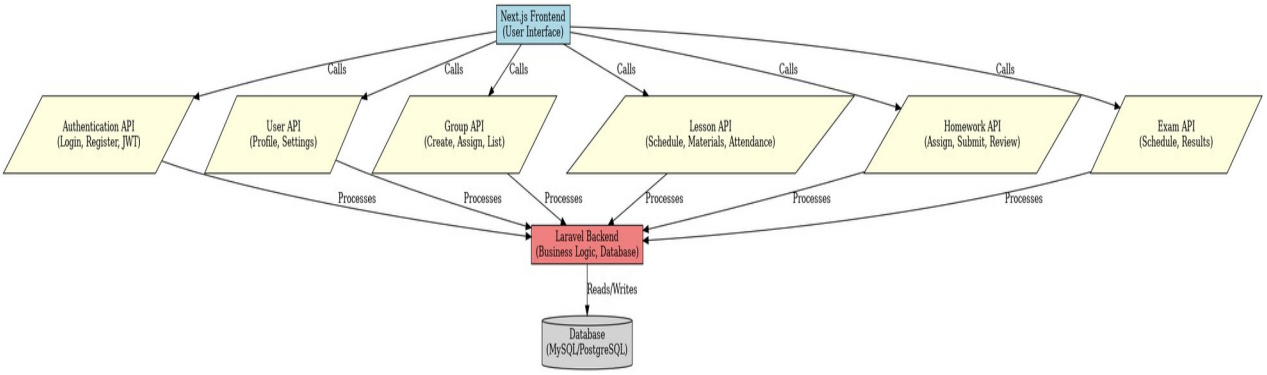

Рис. 2.1 - UML-діаграма структури проєкту

Адміністратор відіграє ключову роль у налаштуванні та управлінні системою. Він створює навчальні групи, додає викладачів, призначає студентів у відповідні групи після оплати курсу. Також адміністратор контролює загальний процес навчання, веде базу студентів і викладачів, налаштовує календар подій та організовує фінальні іспити.

Викладач керує навчальним процесом у межах своєї групи: проводить уроки через вбудовану Zoom-інтеграцію, публікує інтерактивні завдання, перевіряє домашні роботи, залишає коментарі до відповідей студентів, а також веде журнал успішності, де відображаються оцінки та відвідуваність.

Студент після реєстрації та оплати курсу потрапляє в навчальну групу, отримує доступ до розкладу занять, бере участь в онлайн-уроках, виконує домашні завдання та отримує оцінки. Платформа автоматично призначає студентам домашні завдання після кожного уроку, а також зберігає їхній прогрес у навчанні.

Окремо реалізовано можливість проведення фінального іспиту. Викладач або адміністратор призначає дату тестування, після чого студенти проходять його у визначений час. Результати відображаються в системі та впливають на загальну оцінку.

Платформа побудована таким чином, щоб забезпечити зручність використання як для студентів, так і для викладачів та адміністраторів. Вона має

адаптивний дизайн, що дозволяє комфортно працювати як з комп’ютерів, так і з мобільних пристроїв. Автоматизація ключових процесів знижує адміністративне навантаження та дає змогу зосередитися на навчальному процесі.

Таким чином, онлайн-школа іноземних мов стане ефективним інструментом для організації дистанційного навчання, забезпечуючи високу якість освіти, зручність управління та інтерактивність навчального процесу.

2.4. Особливості програмної реалізації

У процесі реалізації програмного забезпечення було впроваджено механізм автентифікації користувачів, що включає можливість входу до системи з використанням облікового запису Google. Такий спосіб забезпечує надійний та безпечний обмін даними між платформою та зовнішнім сервісом автентифікації. Такий підхід дозволив значно спростити процес доступу до платформи, зменшити час, необхідний для авторизації, а також мінімізувати ризики, пов’язані з обробкою та зберіганням паролів. Реалізація авторизації через Google сприяє підвищенню загальної зручності користування системою, а також відповідає сучасним вимогам до безпеки вебресурсів.

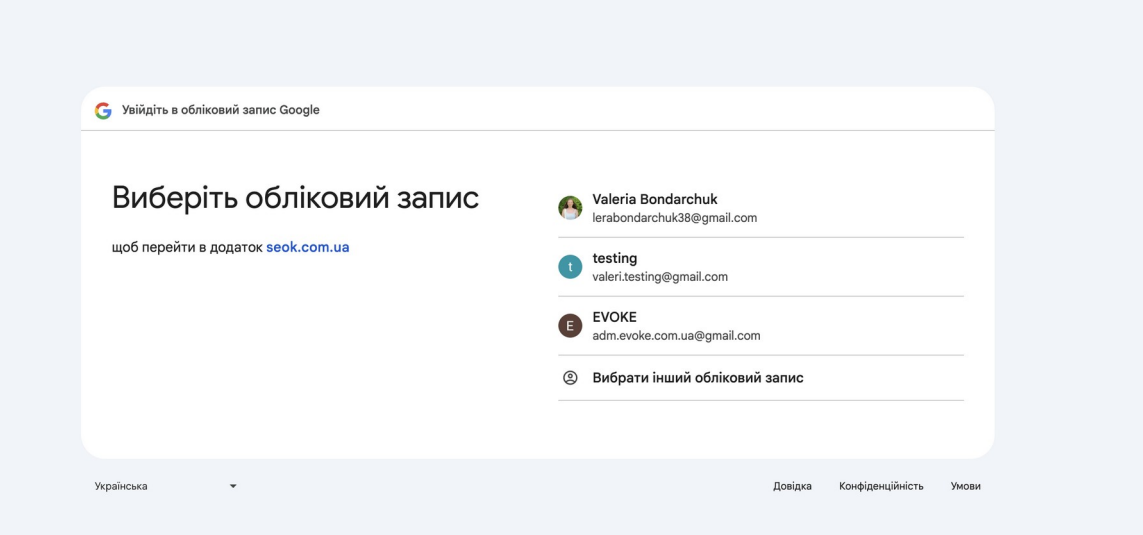

Рис. 2.2 - Авторизація через Google

Рис. 2.3 - Реалізація авторизації через Google

У рамках реалізації програмного забезпечення було впроваджено гнучку систему ролей і прав доступу, що забезпечує диференційований рівень доступу до функціональних компонентів платформи відповідно до призначених користувачеві повноважень. Система дозволяє адміністраторам створювати додаткові категорії користувачів, зокрема контент-менеджерів, бухгалтерів та інших фахівців, із чітко визначеними обмеженнями щодо їхніх дій у межах платформи.

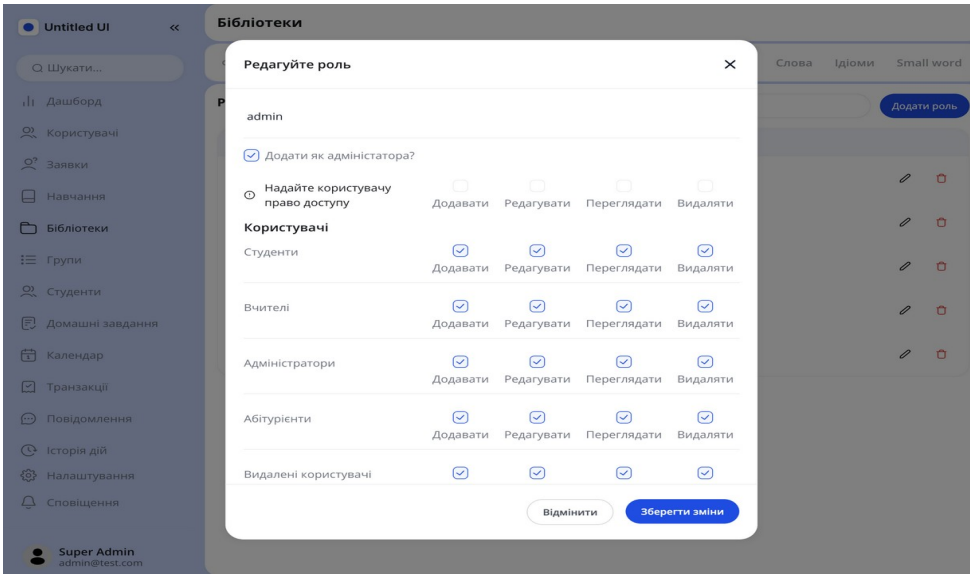

Рис. 2.4 - Модальне вікно регулювання прав

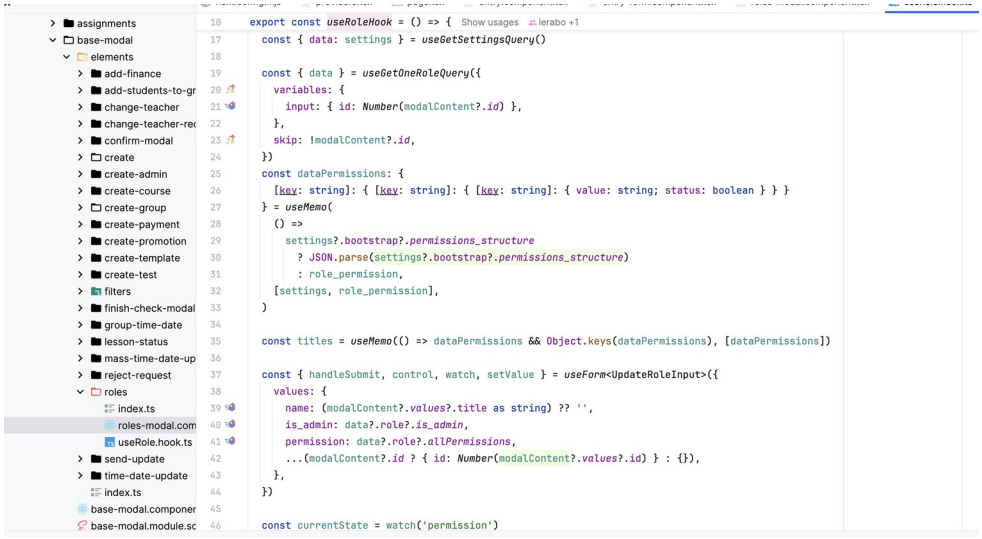

Рис. 2.5 - Регулювання прав різних ролей

З метою підвищення зручності користувачів у межах платформи було реалізовано функціональний календар подій, інтерфейс якого стилістично наближений до Google Calendar. Технічна реалізація даного компонента виконана з використанням бібліотеки react-big-calendar, що забезпечує інтерактивне відображення розкладу занять, підтримку подій у реальному часі, а також можливість динамічної взаємодії користувача з календарем. Такий

підхід сприяв покращенню візуальної організації навчального процесу та підвищенню зручності навігації в межах системи.[1]

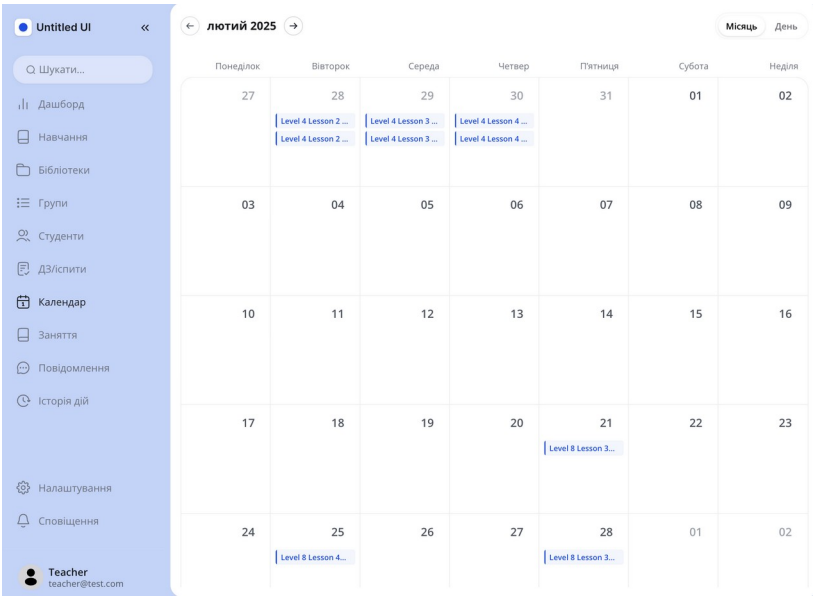

Рис. 2.6 - Вигляд календаря

Рис. 2.7 - Підключення бібліотеки react-big-calendar

В межах розробки платформи було впроваджено функціональну систему чатів, що забезпечує ефективну комунікацію між користувачами. Система включає як індивідуальні чати для двосторонньої взаємодії, зокрема між студентами та викладачами під час перевірки домашніх завдань, так і групові чати, які дозволяють учасникам обмінюватися інформацією в межах навчальних груп. Для реалізації миттєвого обміну повідомленнями застосовано технологію WebSocket, що гарантує швидкий та безперервний діалог без необхідності оновлення сторінки, підвищуючи тим самим зручність і оперативність комунікації в системі.

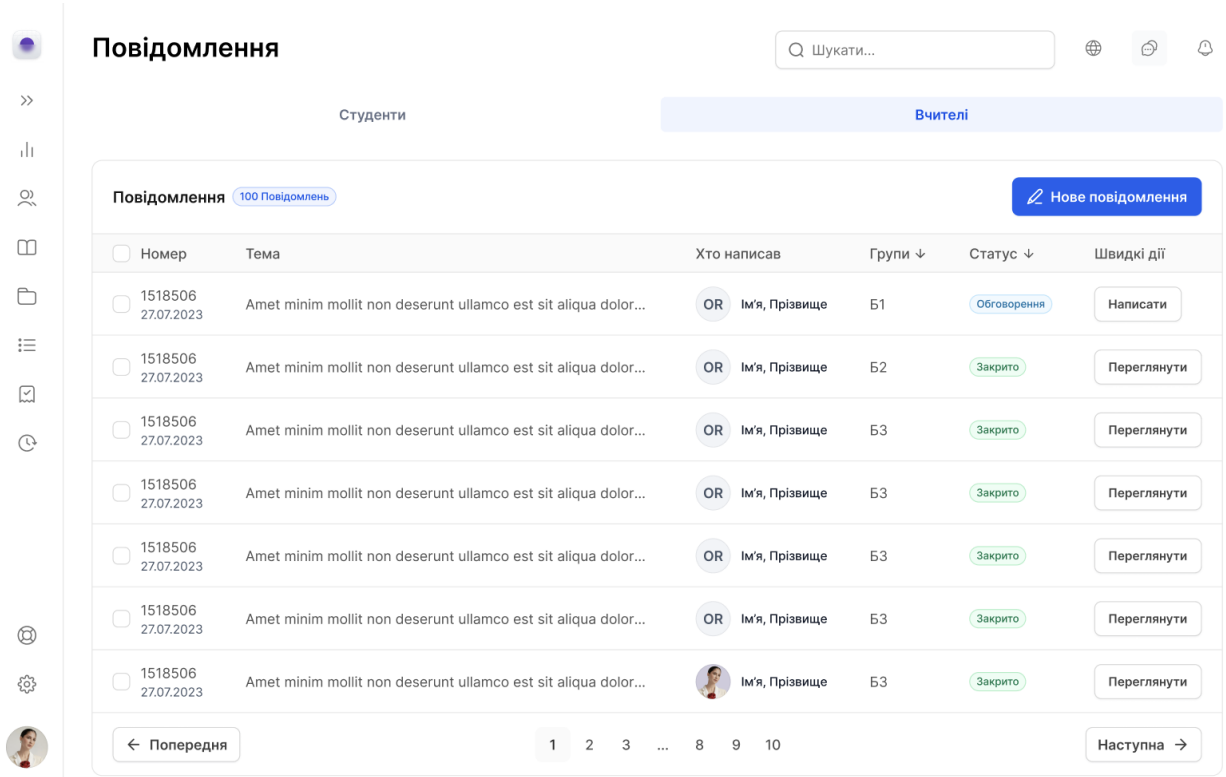

Рис. 2.8 - Mockup загального чату

Під час розробки користувацького інтерфейсу було використано бібліотеку NextUI, що забезпечило сучасний дизайн, адаптивність та високу продуктивність усіх UI-компонентів. Всі елементи інтерфейсу були

кастомізовані відповідно до затвердженого макету платформи, що сприяло формуванню цілісного, інтуїтивно зрозумілого та естетично привабливого користувацького середовища. [2]

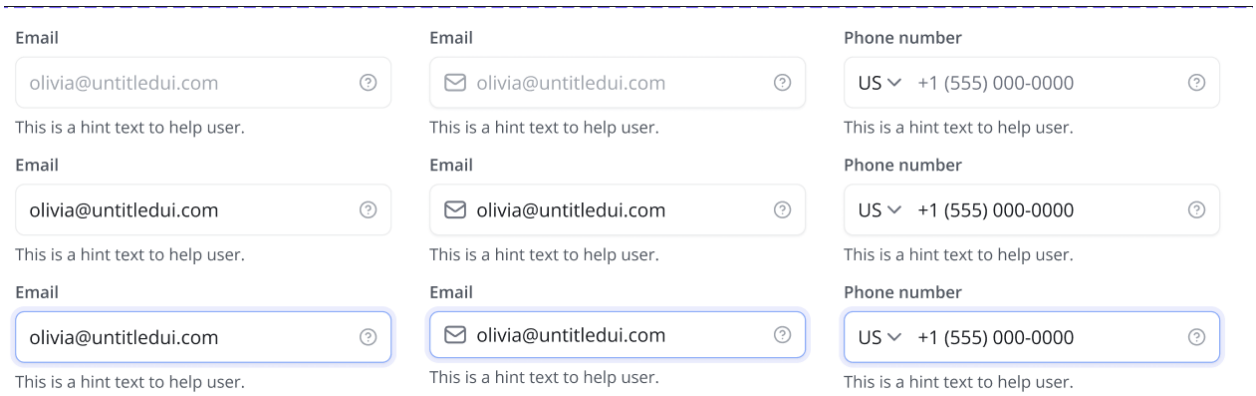

Рис. 2.9 - Частина ui-kit

Отже, розроблена платформа інтегрує високий рівень функціональності, адаптивності та зручності використання, що робить її ефективним засобом для організації дистанційного навчання. Використання сучасних бібліотек та технологічних рішень забезпечує стабільність роботи системи, її масштабованість і потенціал для подальшого розвитку з урахуванням змінних вимог користувачів.

2.5. Організація тестування та налагодження

У процесі розробки інтерфейсу онлайн-школи було використано автоматичне тестування та тестування через QA-спеціаліста для забезпечення стабільності, безпеки та зручності платформи. Автоматизоване тестування охоплювало кілька важливих аспектів. Юніт-тести, реалізовані за допомогою Jest і React Testing Library, перевіряли коректну роботу окремих компонентів інтерфейсу, таких як кнопки, форми та інші інтерактивні елементи. Інтеграційне тестування з використанням Cypress дозволило переконатися, що взаємодія між

модулями проходить без збоїв, зокрема між формою входу та API авторизації. Для перевірки повного користувацького сценарію, від реєстрації до проходження тестів і отримання результатів, застосовувалося E2E-тестування за допомогою Playwright і Cypress

Окрім автоматичного тестування, важливу роль відігравало тестування через QA-спеціаліста, яке дозволило оцінити функціональність та зручність інтерфейсу з погляду кінцевого користувача. Фахівець вручну перевіряв усі основні можливості платформи, зокрема реєстрацію, авторизацію, завантаження курсів і проходження тестів. Особливу увагу приділили тестуванню юзабіліті, яке допомогло оцінити інтуїтивність навігації, логічність розташування елементів і доступність платформи для користувачів різного віку. Також було проведено кросбраузерне та кросплатформене тестування, щоб переконатися у коректному відображенні сайту в браузерах Google Chrome, Mozilla Firefox, Safari та Microsoft Edge, а також на пристроях з Windows, macOS, iOS та Android. Додатково виконувалося навантажувальне тестування за допомогою JMeter для оцінки продуктивності системи при великій кількості одночасних користувачів. Після кожного етапу тестування QA-спеціаліст створював звіти про знайдені помилки у Jira, а після виправлення багів повторно перевіряв, чи були вони усунені. Завдяки поєднанню автоматизованого та ручного тестування вдалося досягти високої якості інтерфейсу, зробивши платформу стабільною, безпечною та зручною у використанні.

ВИСНОВКИ

У процесі розробки інтерфейсу онлайн-школи було досягнуто всіх поставлених цілей, основними з яких були створення зручної, інтуїтивно зрозумілої та функціональної платформи для дистанційного навчання. Для реалізації цих завдань використовувалися сучасні технології, що забезпечують високу продуктивність, адаптивність та доступність інтерфейсу. Важливим аспектом розробки стало впровадження гнучкої архітектури, оптимізація взаємодії з користувачем та інтеграція різних форматів навчальних матеріалів.

Мета проєкту була досягнута, оскільки платформа успішно реалізує можливості онлайн-навчання, надаючи користувачам доступ до відеолекцій, тестових завдань і конспектів. Функціональність авторизації, управління навчальними курсами та перевірки знань була впроваджена відповідно до сучасних стандартів безпеки та UX-дизайну. Завдяки автоматизованому та ручному тестуванню вдалося забезпечити стабільну роботу системи, кросбраузерну та кросплатформену сумісність, а також високу продуктивність навіть при великій кількості користувачів.

Розробка інтерфейсу онлайн-школи завершена, і система готова до впровадження у реальному середовищі. Проте, з урахуванням динамічного розвитку технологій та можливих змін у потребах користувачів, подальше вдосконалення функціоналу залишається актуальним завданням. Планується інтеграція нових методів аутентифікації, розширення можливостей персоналізації навчального процесу та покращення адаптації інтерфейсу під мобільні пристрої. Також передбачено зворотний зв’язок із користувачами для подальшої оптимізації системи та її відповідності сучасним тенденціям у сфері онлайн-освіти.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Документація React big calendar (4 травня, 2024) URL: https://www.scaler.com/topics/react/react-big-calendar/

2. Документація Next UI (16 січня, 2025) URL: https://www.heroui.com/

3. Ключові інструменти та технології Front-end, що використовуються для розробки веб-сайтів (5 жовтня, 2023) URL: https://webbookstudio.com/ua/articles/key-tools-and-technologies-used-in-front-end-website-development

4. Майбутнє веб-розробки: Тенденції та технології, на які слід звернути увагу. (20 квітня, 2023) URL: https://stfalcon.com/uk/blog/post/web-development-trends-part-2

5. 5 фреймворків та бібліотек для Front-end розробки (05 березня, 2023) URL: https://www.digest.pro/news/5-freimvorkiv-ta-bibliotek-dlia-front-end-rozrobky/

6. Що повинен знати FrontEnd розробник у 2024 році (13 лютого, 2024) URL: https://itvdn.com/ua/blog/article/frontend-developer-tech-skills

7. Всеосяжний посібник з фронтенд-розробки (13 травня, 2024) URL: https://javascript.org.ua/vseosyazhnij-posibnik-z-frontend-rozrobki/

8. Розкриття потужності React та Next.js для створення високопродуктивних веб-додатків (19 вересня, 2024) URL: https://javascript.org.ua/rozkrittya-potuzhnosti-react-ta-next-js-dlya-stvorennya-visokoproduktivnih-veb-dodatkiv/

9. Основні принципи веб-дизайну інтерфейсу користувача, яких слід дотримуватися у 2024 році (11 жовтня, 2023) URL: https://stfalcon.com/uk/blog/post/user-interface-web-design-principles

10. SSR vs. SSG in Next.js: Differences, Advantages, and Use Cases (30 жовтня, 2024) URL: https://strapi.io/blog/ssr-vs-ssg-in-nextjs-differences-advantages-and-use-cases

11. Tailwind and Next.js (15 січня, 2025) URL: https://dev.to/oliviarizona88/tailwind-and-nextjs-1733

12. What is the DOM? (22 червня, 2022) URL: https://www.freecodecamp.org/news/what-is-the-dom-explained-in-plain-english/

13. Чому важлива типізація в JS? (23 жовтня, 2023) URL: https://foxminded.ua/typizaciya-js/

14. The evolution of HTML:From Version 1.0 to 5.0 (5 липня, 2023) URL: https://medium.com/@techwisenow/the-evolution-of-html-from-version-1-0-to-5-0-fbb7a60121a5

ДОДАТКИ

Додаток А

Технічне завдання на розробку LMS системи

Мета та призначення документа:

Цей документ є технічним завданням (ТЗ) на розробку системи управління навчанням (LMS). Метою цього ТЗ є детальне описання функціональних та нефункціональних вимог до системи, що дозволить команді розробки ефективно спланувати та реалізувати проект. Система призначена для автоматизації процесів навчання, управління студентами, викладачами, групами, уроками, фінансами та іншими адміністративними задачами.

Огляд системи:

LMS система буде складатися з трьох основних ролей користувачів:

Адміністратор: Повний контроль над системою, управління користувачами, групами, уроками, фінансами, бібліотеками та налаштуваннями.

Викладач: Управління групами, уроками, домашніми завданнями, оцінювання студентів та перегляд їхнього прогресу.

Студент: Доступ до своїх уроків, домашніх завдань, тестів, історії оплат та можливості оплати навчання.

Абітурієнт: Можливість реєстрації, ознайомлення з курсами та рівнями, а також оплати навчання.

Загальні вимоги до системи:

Продуктивність

- Час завантаження сторінок: не більше 3 секунд.
- Час відгуку на дії користувача: не більше 1 секунди.

- Система повинна підтримувати одночасну роботу не менше 500 користувачів без зниження продуктивності.

Надійність

- Доступність системи: 99.5% часу.
- Резервне копіювання даних: щоденно, з можливістю відновлення.

Безпека

- Всі дані користувачів повинні бути захищені за допомогою шифрування.
- Система повинна відповідати стандартам захисту даних (наприклад, GDPR, якщо це актуально).
- Авторизація та аутентифікація користувачів з підтримкою двофакторної аутентифікації (за бажанням).
- Журналування всіх критичних дій користувачів.

Масштабованість

- Архітектура системи повинна дозволяти легке додавання нових функцій та розширення обсягів даних.
- Можливість горизонтального та вертикального масштабування.

Зручність Використання (Usability)

- Інтуїтивно зрозумілий інтерфейс для всіх ролей користувачів.
- Відповідність принципам UI/UX дизайну (Figma-файл буде надано як посилання).
- Підтримка адаптивного дизайну для коректного відображення на різних пристроях (десктоп, планшет, мобільний).

Технічні вимоги:

Технологічний Стек (Приклад, може бути змінений за згодою сторін)

- Backend: Python (Django/Flask), Node.js (Express) або PHP (Laravel).
- Frontend: React, Angular або Vue.js.
- База даних: PostgreSQL або MySQL.

- Хмарна інфраструктура: AWS, Google Cloud або Azure (залежить від вимог).
- Система контролю версій: Git (GitHub/GitLab/Bitbucket).

Інтеграції (Приклад, може бути змінений за згодою сторін)

- Платіжні системи: Stripe, LiqPay або інші місцеві платіжні провайдери.
- Відеоконференції: Zoom, Google Meet або інша платформа для онлайн-уроків.
- SMS/Email розсилки: Сервіси для сповіщень.

Функціональність для Адміністратора:

Пошук:

1.1 По персоналу, студентам: Пошук користувачів за різними критеріями (ім'я, email, телефон, роль).

1.2 По матеріалам: Пошук навчальних матеріалів (уроків, тестів, слів, ідіом).

1.3 По групам: Пошук груп за назвою, викладачем, рівнем, статусом.

Дашборд для Адміністратора:

2.1 Дашборд: Централізований екран з оглядом ключових показників.

2.2 Віджет студентів (40 годин): Кількість активних студентів, нові реєстрації, статистика відвідуваності.

2.3 Віджет вчителів: Кількість викладачів, їхній статус.

2.4 Віджет груп: Кількість активних груп, вільні місця, завантаженість.

2.5 Віджет курсів: Доступні курси, популярність.

2.6 Віджет заявок: Кількість нових заявок, їхній статус.

2.7 Заборгованості по оплатам: Список студентів з неоплаченими рахунками.

2.8 Календар: Огляд всіх запланованих уроків та подій.

2.9 Видалення уроку: Можливість видалення уроку безпосередньо з дашборду.

Управління фінансами:

3.1 Транзакції: Перегляд всіх фінансових транзакцій.

3.2 Список оплат студентів: Детальний список оплат кожного студента.

3.3 Перегляд: Деталізований перегляд кожної оплати (сума, дата, метод).

3.4 Зміна статусу оплат: Можливість зміни статусу оплати (оплачено, неоплачено, відмінено).

Профіль (Адміністратора):

4.1 Профіль: Перегляд та редагування власного профілю.

4.2 Персональна інформація: Ім'я, прізвище, контактні дані.

4.3 Інформація про викладання (для адмінів, які також можуть викладати): Курси, що викладаються.

4.4 Збереження змін: Функціональність збереження відредагованої інформації.

4.5 Родинні зв'язки (12 годин): Можливість додавання та перегляду родинних зв'язків між студентами (наприклад, брати/сестри для застосування сімейних знижок).

4.6 Вчитель привів на навчання: Відстеження студентів, які були залучені певним викладачем.

4.7 Мова, мови навчання: Вказання мов, якими володіє/викладає адміністратор.

4.8 Форма навчання: Дистанційна/очна/гібридна.

4.9 Філіал: Прив'язка до певного філіалу.

4.10 Тип, типи навчання: Індивідуальні/групові заняття.

4.11 Групи в яких викладає (якщо є): Список груп, де адміністратор виступає викладачем.

Календар:

5.1 Календар: Загальний календар занять.

5.2 Календар / місяць: Перегляд занять за місяць.

5.3 Календар / Сьогодні: Перегляд занять на поточний день.

5.4 Функція перетягування: Можливість перетягувати уроки для зміни часу/дати.

5.5 Надіслати оновлення: Запит на підтвердження розсилки сповіщень про зміни в розкладі.

5.6 Календар / уроки всіх груп дня: Огляд всіх уроків для всіх груп на певний день.

5.7 Зміна дати та часу навчання (початок, кінець): Редагування часу початку та завершення уроку.

5.8 Додати дні у виключення: Виключення окремих днів з розкладу (наприклад, свята).

5.9 Видалення уроку: Видалення окремого уроку.

Управління користувачами:

6.1 Користувачі + Додати / Редагувати / Видалити: Загальна функціональність для управління всіма типами користувачів.

3.6.1. Управління Студентами

6.2 Користувачі / Студенти: Перегляд списку студентів.

6.3 Користувачі / Студенти / Редагування таблиці: Можливість швидкого редагування даних без переходу в детальний профіль.

6.4 Користувачі / Студенти / Фільтри: Фільтрація за різними критеріями (група, викладач, статус оплати, рівень).

6.5 Користувачі / Студенти / Редагувати / Персональна інформація: Детальне редагування персональних даних.

6.6 Користувачі / Студенти / Редагувати / Інформація про навчання: Рівень, група, дата початку, історія навчання.

6.7 Користувачі / Студенти / Редагувати / Коментарі про студента: Додавання та редагування нотаток про студента.

6.8 Користувачі / Студенти / Редагувати / Персональна інформація / Видалити: Видалення студента.

6.9 Користувачі / Студенти / Видалено повідомлення: Підтвердження видалення.

6.9 Користувачі / Студенти / Додати нового студента / Персональна інформація: Форма додавання нового студента.

6.10 Користувачі / Студенти / Додати нового студента / Інформація про навчання: Заповнення інформації про навчання при додаванні.

6.11 Користувачі / Студенти / Профіль / Інформація: Перегляд детального профілю студента.

6.12 Користувачі / Студенти / Профіль / Інформація / Ховер на родинні зв’язки: Спливаюче вікно з інформацією про родинні зв'язки.

6.13 Користувачі / Студенти / Профіль / ДЗ: Перегляд історії домашніх завдань студента.

6.14 Користувачі / Студенти / Профіль / Коментарі: Перегляд та додавання коментарів про студента.

3.6.2. Управління Викладачами

6.15 Користувачі / Вчителі: Перегляд списку викладачів.

6.16 Користувачі / Вчителі / Дивитись групи: Перегляд груп, які веде викладач.

6.17 Користувачі / Вчителі / Редагувати / Персональна інформація: Редагування персональних даних викладача.

6.18 Користувачі / Вчителі / Редагувати / Інформація про викладання: Курси, рівні, мови викладання.

6.19 Користувачі / Вчителі / Редагувати / Коментарі про вчителя: Нотатки про викладача.

6.20 Користувачі / Вчителі / Обрано декілька / Видалити: Масове видалення викладачів.

Управління користувачами, які очікують реєстрації:

6.21 Користувачі / Очікують реєстрації / Обрано декілька / Видалити декількох: Масове видалення заявок на реєстрацію.

Управління адміністраторами:

6.22 Користувачі / Адміністратори: Перегляд списку адміністраторів.

6.23 Користувачі / Адміністратори / Редагувати / Персональна інформація: Редагування персональних даних адміністратора.

6.24 Користувачі / Адміністратори / Редагувати / Інформація про викладання (якщо є): Інформація про викладацьку діяльність.

6.25 Користувачі / Адміністратори / Редагувати / Коментарі про адміністратора: Нотатки про адміністратора.

6.26 Користувачі / Адміністратори / Додати адміністраторів: Форма додавання нового адміністратора.

6.27 Користувачі / Адміністратори / Додати адміністраторів / Роль адміністраторів: Призначення ролі та прав доступу.

6.28 Користувачі / Адміністратори / Додати адміністраторів / Роль адміністраторів / Більше про роль: Детальний опис прав для кожної ролі.

6.29 Користувачі / Адміністратори / Додати адміністраторів / Введіть емейли: Можливість додавання адміністраторів за допомогою email.

3.6.5. Управління Абітурієнтами

6.30 Користувачі / Абітурієнти: Перегляд списку абітурієнтів.

6.31 Користувачі / Абітурієнти / Персональна інформація: Перегляд персональних даних абітурієнта.

6.32 Користувачі / Абітурієнти / Редагувати / Інформація про навчання: Інформація про потенційне навчання.

6.33 Користувачі / Абітурієнти / Редагувати / Історія оплат / Пусто: Відображення відсутності оплат.

6.34 Користувачі / Абітурієнти / Створити платіж (26 годин): Можливість створення платежу для абітурієнта.

6.35 Користувачі / Абітурієнти / Створити платіж / Оберіть Рівні: Вибір рівня навчання для оплати.

6.36 Користувачі / Абітурієнти / Створити платіж / Сімейна знижка: Застосування сімейної знижки.

6.37 Користувачі / Абітурієнти / Створити платіж / Оберіть Тип оплати: Вибір типу оплати (готівка, картка, переказ).

6.38 Користувачі / Абітурієнти / Створити платіж / Тип оплати Готівка: Введення інформації про оплату готівкою.

6.39 Користувачі / Абітурієнти / Створити платіж / Оберіть Оберіть філіал: Прив'язка платежу до філіалу.

6.40 Користувачі / Абітурієнти / Редагувати / Історія оплат / Створено платіж: Відображення створеного платежу.

6.41 Користувачі / Абітурієнти / Редагувати / Історія оплат / Створено платіж / Більше інфи: Детальна інформація про платіж.

6.42 Користувачі / Абітурієнти / Редагувати / Історія оплат / Створено платіж / Більше інфи / Змінити: Редагування інформації про платіж.

6.43 Користувачі / Абітурієнти / Редагувати / Історія оплат / Створено платіж / Більше інфи / Змінити (дублюється): Перевірка функціоналу.

6.44 Користувачі / Абітурієнти / Редагувати / Історія оплат / Створено платіж / Підтверджено оплату: Підтвердження факту оплати.

6.45 Користувачі / Абітурієнти / Редагувати / Історія оплат / Створено платіж / Підтверджено оплату / (дублюється): Перевірка функціоналу.

6.46 Користувачі / Абітурієнти / Редагувати / Перемістити в групу: Переміщення абітурієнта в активну групу.

Управління Видаленими Користувачами:

6.47 Користувачі / Видалені користувачі: Перегляд списку видалених користувачів та можливість їх відновлення.

Управління заявками:

7.1 Заявки: Розділ для управління заявками на навчання.

7.2 Заявки (Основне): Загальний список заявок.

7.3 Заявки / Menu: Навігація по статусах заявок.

7.4 Заявки / Full view: Детальний перегляд окремої заявки.

7.5 Заявки / Редагувати / Персональна інформація: Редагування персональних даних у заявці.

7.6 Користувачі / Очікують реєстрації / Редагувати / Інформація про навчання: Додавання інформації про навчання до заявки.

7.7 Користувачі / Очікують реєстрації / Редагувати / Перемістити в абітурієнти: Переміщення заявки до списку абітурієнтів.

7.8 Заявки / Додати новий статус: Створення власних статусів для заявок.

7.9 Заявки / Додати новий статус / Колір статусу: Вибір кольору для візуалізації статусу.

7.10 Заявки / Додати новий статус / Колір статусу / Введено: Збереження нового статусу.

7.11 Заявки / Додати новий статус / Редагувати: Редагування існуючих статусів.

7.12 Заявки / Додати новий статус / Видалити: Видалення статусу.

Бібліотеки:

8.1 Libraries: Загальний розділ для управління довідниками.

Форми:

8.2 Бібліотеки / Форми: Управління формами.

8.3 Бібліотеки / Форми / Додати форму: Додавання нової форми.

8.4 Бібліотеки / Форми / Додати форму / Введено: Збереження нової форми.

8.5 Бібліотеки / Форми / Редагувати: Редагування існуючих форм.

8.6 Бібліотеки / Форми / Видалити: Видалення форм.

Типи Навчання:

8.7 Бібліотеки / Типи навчання: Управління типами навчання (індивідуальне, групове).

8.8 Бібліотеки / Типи навчання / Додати тип навчання: Додавання нового типу.

8.9 Бібліотеки / Типи навчання / Додати тип навчання / Введено: Збереження нового типу.

8.10 Типи навчання / Додати тип навчання / Редагувати: Редагування існуючих типів.

8.11 Типи навчання / Додати тип навчання / Видалити: Видалення типів.

3.8.3. Рівні Мови Навчання

8.12 Бібліотеки / Рівні мови навчання: Управління рівнями (A1, B2, C1 тощо).

8.13 Бібліотеки / Рівні мови навчання / Додати рівень мови: Додавання нового рівня.

8.14 Бібліотеки / Рівні мови навчання / Додати рівень мови / Введено: Збереження нового рівня.

8.15 Бібліотеки / Рівні мови навчання / Редагувати: Редагування існуючих рівнів.

8.16 Бібліотеки / Рівні мови навчання / Видалити: Видалення рівнів.

3.8.4. Філіали

8.17 Бібліотеки / Філіали: Управління філіалами.

8.18 Бібліотеки / Філіали / Додати філіал: Додавання нового філіалу.

8.19 Бібліотеки / Філіали / Додати філіал / Введено: Збереження нового філіалу.

8.20 Бібліотеки / Філіали / Редагувати: Редагування існуючих філіалів.

8.21 Бібліотеки / Філіали / Видалити: Видалення філіалів.

Статуси Д/З:

8.22 Бібліотеки / Статуси Д/З: Управління статусами домашніх завдань (перевірено, не перевірено).

8.23 Бібліотеки / Статуси Д/З / Додати статус: Додавання нового статусу.

8.24 Бібліотеки / Статуси Д/З / Додати статус / Колір статусу: Вибір кольору для статусу.

8.25 Бібліотеки / Статуси Д/З / Додати статус / Введено: Збереження нового статусу.

8.26 Бібліотеки / Статусу ДЗ / Редагувати: Редагування існуючих статусів.

8.27 Бібліотеки / Статусу ДЗ / Видалити: Видалення статусів.

3.8.6. Мови Навчання

8.28 Бібліотеки / Мови навчання: Управління мовами навчання (англійська, німецька).

8.29 Бібліотеки / Мови навчання / Додати мову: Додавання нової мови.

8.30 Бібліотеки / Мови навчання / Додати мову / Введено: Збереження нової мови.

8.31 Бібліотеки / Мови навчання / Редагувати: Редагування існуючих мов.

8.32 Бібліотеки / Мови навчання / Видалити: Видалення мов.

3.8.7. Ролі

8.33 Бібліотеки / Ролі: Управління ролями користувачів та їхніми правами доступу.

8.34 Бібліотеки / Ролі / Додати роль / Користувачі: Призначення користувачів до ролі.

8.35 Бібліотеки / Ролі / Додати роль / Користувачі / content: Детальний опис прав для кожної ролі.

8.36 Бібліотеки / Ролі / Додати роль / Користувачі обрані: Відображення обраних користувачів.

8.37 Бібліотеки / Ролі / Редагувати: Редагування існуючих ролей.

8.38 Бібліотеки / Ролі / Видалити: Видалення ролей.

3.8.8. Статуси Оплати

8.39 Бібліотеки / Статуси оплати: Управління статусами оплат (оплачено, очікується).

8.40 Бібліотеки / Статуси оплати / Додати статус: Додавання нового статусу.

8.41 Бібліотеки / Статуси оплати / Додати статус / Введено: Збереження нового статусу.

8.42 Бібліотеки / Статуси користувачів / Редагувати: Редагування статусів користувачів.

8.43 Бібліотеки / Статусу ДЗ / Редагувати (дублюється): Перевірка функціоналу.

8.44 Бібліотеки / Статуси користувачів / Видалити: Видалення статусів користувачів.

3.8.9. Бібліотека Слів

8.45 Бібліотеки / Слова: Управління словником.

8.46 Бібліотеки / Слова / Меню: Навігація по розділу слів.

8.47 Бібліотеки / Слова / Додати слово: Додавання нового слова.

8.48 Бібліотеки / Слова / Додати слово / Рівень мови та властивість (3 годин): Призначення рівня та інших властивостей слову.

8.49 Бібліотеки / Слова / Додати слово / Озвучування: Можливість додавання аудіофайлу для озвучування слова.

3.8.10. Бібліотека Small Words

8.50 Бібліотеки / Small word: Управління "малими" словами.

8.51 Бібліотеки / Small word / Меню: Навігація по розділу.

8.52 Бібліотеки / Small word / Додати слово: Додавання нового "малого" слова.

8.53 Бібліотеки / Small word / small word / Аудіо та фото (6 годин): Можливість додавання аудіо та фото для "малих" слів.

3.8.11. Бібліотека Ідіом

8.54 Бібліотеки / Ідіоми: Управління ідіомами.

8.55 Бібліотеки / Ідіоми / Меню: Навігація по розділу.

8.56 Бібліотеки / Слова / Додавання ідіоми: Додавання нової ідіоми.

3.8.12. Бібліотека Слів до Відео

8.57 Бібліотеки / Слова до відео: Управління словами, пов'язаними з відео.

8.58 Бібліотеки / Слова до відео / Меню: Навігація по розділу.

8.59 Бібліотеки / Слова до відео / Додавання: Додавання нових слів до відео.

8.60 Бібліотеки / Слова до відео / Відео додано / Слова обираються: Вибір слів для конкретного відео.

Бібліотека Sentence Patterns:

8.61 Бібліотеки / Sentence pattern: Управління зразками речень.

8.62 Бібліотеки / Слова / Додати Sentence pattern: Додавання нового зразка речення.

Управління навчанням:

9.1 Lessons Themes: Загальний розділ тем уроків.

Уроки:

9.2 Навчання / Уроки / Додати урок: Створення нового уроку.

9.3 Навчання / Уроки / Додати урок / Головна інформація: Назва, опис, рівень.

9.4 Навчання / Уроки / Додати урок / Читання теорії / Пусто: Інтерфейс для додавання теорії (текст).

9.5 Навчання / Уроки / Додати урок / Читання теорії / Заповнено: Відображення заповненої теорії.

9.6 Desktop / Слова до відео: Інтерфейс для додавання слів до відео в уроці.

9.7 Desktop / Слова до відео / спадне меню: Вибір слів зі словника.

9.8 Desktop / Ідіоми: Інтерфейс для додавання ідіом в уроці.

9.9 Desktop / Ідіоми / спадне меню: Вибір ідіом з бібліотеки.

9.10 Desktop / Ідіоми / Small word: Інтерфейс для додавання "малих" слів.

9.11 Desktop / Ідіоми / Small word / спадне меню: Вибір "малих" слів.

9.12 Desktop / Домашні завдання: Інтерфейс для додавання домашніх завдань.

9.13 Desktop / Домашні завдання / заповненні поля: Відображення заповнених полів ДЗ.

9.14 Desktop / Ідіоми / Sentence pattern: Інтерфейс для додавання зразків речень.

New Desktop / Conversation (10 годин): Можливість додавання інтерактивних діалогів/розмов.

New Desktop / Додаткові вправи: Можливість додавання інших типів вправ.

3.9.2. Шаблони Уроків

9.15 Навчання / Уроки / Створити шаблони: Створення шаблонів уроків для швидкого створення нових.

9.16 Навчання / Уроки / Створити шаблони / Рівень мови: Прив'язка шаблону до рівня мови.

9.17 Навчання / Уроки / Створити шаблони: Збереження шаблону.

3.9.3. Курси

9.18 Навчання / Курси: Управління курсами.

9.19 Навчання / Курси / Створити курс: Створення нового курсу.

9.20 Навчання / Курси / Створити курс / Уроки: Прив'язка уроків до курсу.

9.21 Навчання / Курси / Створити курс / Дата нової ціни (10 годин): Можливість планування зміни ціни курсу на певну дату.

Акції:

9.22 Навчання / Акції: Управління акційними пропозиціями.

9.23 Навчання / акції / Створіть акцію: Створення нової акції.

9.24 Навчання / акції / Створіть акцію / Введено: Збереження нової акції.

3.9.5. Тести

9.25 Навчання / Тести: Управління тестами.

9.26 Навчання / Тести / Весь текст: Можливість додавання великого обсягу тексту для тесту.

9.27 Навчання / Тести / Створити тест: Створення нового тесту.

9.28 Навчання / Тести / Створити тест / Введено: Збереження нового тесту.

Управління групами:

10.1 Groups: Загальний розділ для управління групами.

10.2 Група номер 1 / Немає студентів: Відображення групи без студентів.

10.3 Група номер 1 / Видалити групу: Видалення групи.

10.4 Групи: Перегляд списку груп.

10.5 Групи / Фільтри: Фільтрація груп за викладачем, рівнем, статусом.

10.6 Групи / Фільтри обрано: Відображення застосованих фільтрів.

3.10.1. Деталі Групи

10.7 Група номер 1 / Загальна інформація: Інформація про групу (назва, рівень, викладач, філіал).

10.8 Група номер 1 / Загальна інформація / Пошук студентів: Пошук студентів для додавання в групу.

10.9 Група номер 1 / Загальна інформація / Пошук студентів / Введено ім'я: Введення імені студента для пошуку.

10.10 Група номер 1 / Загальна інформація / Дати та час навчання: Редагування розкладу групи.

10.11 Групи / Загальна інформація / Зберегти зміни: Збереження змін у групі.

10.12 Групи / Загальна інформація / Зберегти зміни / Попап підтвердження: Підтвердження збереження.

Календар занять групи:

10.13 Група номер 1 / Календар занять / Місяць: Перегляд розкладу групи за місяць.

10.14 Група номер 1 / Календар занять / Сьогодні: Перегляд розкладу групи на сьогодні.

10.15 Група номер 1 / Календар занять / Місяць / Перетягуємо урок: Drag-and-drop функціональність для зміни розкладу.

10.16 Група номер 1 / Календар занять / Місяць / Надіслати оновлення?: Запит на відправку сповіщень про зміни.

10.17 Група номер 1 / Календар занять / Сьогодні / Перетягуємо урок: Drag-and-drop на сьогоднішньому розкладі.

10.18 Група номер 1 / Календар занять / Сьогодні / Надіслати оновлення?: Запит на відправку сповіщень.

10.19 Група номер 1 / Календар занять / Місяць / редагувати: Редагування деталей уроку.

10.20 Група номер 1 / Календар занять / Місяць / редагувати / дата навчання: Зміна дати уроку.

10.21 Група номер 1 / Календар занять / Місяць / редагувати / Час навчання: Зміна часу уроку.

10.22 Група номер 1 / Календар занять / Місяць / редагувати / Надіслати оновлення користувачам?: Запит на відправку сповіщень.

10.23 Група номер 1 / Календар занять / Місяць / Натиснули на урок / Видалити урок: Видалення конкретного уроку.

10.24 Групи / Календар занять / Відпустка / Вихідний / Заміна (8 годин): Можливість відмічати відпустки, вихідні, заміни викладача.

3.10.3. Створення Групи

10.25 Створити групу: Форма для створення нової групи.

10.26 Створити групу / Рівень мови: Вибір рівня мови.

10.27 Створити групу / Викладач: Призначення викладача.

10.28 Створити групу / Час навчання / Пн + Ср + Сб: Вибір днів та часу занять.

10.29 Створити групу / Дати навчання: Встановлення періоду навчання.

10.30 Створити групу / Дати навчання / Виключити дні: Виключення окремих днів з розкладу групи.

10.31 Створити групу / Час навчання / Створити групу / Відпустка: Можливість додавання відпусток викладача при створенні групи.

10.32 Створити групу / Час навчання / Створити групу: Збереження нової групи.

Повідомлення:

11.1 Messages (по тікетам, без чатів): Система тікетів для комунікації, без функції онлайн-чату.

11.2 Messages Default: Загальний вигляд повідомлень.

11.3 Messages Active: Активне повідомлення/тікет.

Історія Дій:

12.1 History: Загальний розділ для історії дій.

12.2 Історія дій / Пусто (16 годин): Відображення відсутності дій.

12.3 Історія дій: Список всіх дій, виконаних в системі.

12.4 Історія дій / більше інформації про дію: Деталізований опис кожної дії.

Сповіщення:

13.1 Notifications: Розділ для управління сповіщеннями.

13.2 Сповіщення / Пусто: Відображення відсутності сповіщень.

13.3 Сповіщення / Всі: Перегляд всіх сповіщень.

13.4 Сповіщення / Не прочитані: Перегляд непрочитаних сповіщень.

13.5 Сповіщення / Прочитані: Перегляд прочитаних сповіщень.

Налаштування:

14.1 Settings: Загальний розділ налаштувань.

14.2 Налаштування / Загальні: Загальні налаштування системи.

14.3 Налаштування / Повідомлення: Налаштування сповіщень (email, sms).

14.4 Налаштування / Віджети (16 годин): Налаштування відображення віджетів на дашборді.

14.5 Налаштування / Фінанси: Налаштування фінансових параметрів.

14.6 Налаштування / Фінанси / Додати ФОП: Додавання інформації про ФОП для фінансових операцій.

14.7 Налаштування / Загальні / Зберегти зміни: Збереження загальних налаштувань.

14.8 Налаштування / Загальні / Зберегти зміни / Попап підтвердження: Підтвердження збереження.

14.9 Налаштування / Фінанси / Видалити ФОП: Видалення інформації про ФОП.

Функціональність для викладача:

Профіль викладача:

15.1 Профіль: Перегляд та редагування власного профілю.

15.2 Профіль / Персональна інформація: Ім'я, прізвище, контактні дані.

15.3 Профіль / Інформація про викладання: Курси, рівні, мови викладання, філіал.

15.4 Профіль / Персональна інформація / Зберегти зміни: Збереження змін у профілі.

15.5 Профіль / Персональна інформація / Зберегти зміни / Попап підтвердження: Підтвердження збереження.

Дашборд викладача:

16.1 Dashboard: Огляд ключових показників для викладача.

16.2 Дашборд: Відображення майбутніх уроків, статистики груп.

16.3 Дашборд / Шукати: Пошук по групах, студентах, уроках.

16.4 Профіль / Персональна інформація: Швидкий доступ до профілю.

16.5 Дашборд / Група: Детальна інформація про групу.

16.6 Дашборд / Група / Календар / Місяць: Перегляд розкладу групи за місяць.

16.7 Дашборд / Група / Календар / Сьогодні: Перегляд розкладу групи на сьогодні.

16.8 Дашборд / Elementary курс: Швидкий доступ до матеріалів курсу.

16.9 Дашборд / Elementary курс / Дивитись урок / Читання теорії: Перегляд теоретичного матеріалу уроку.

16.10 Дашборд / Elementary курс / Дивитись урок / Ідіоми: Перегляд ідіом, пов'язаних з уроком.

16.11 Дашборд / Elementary курс / Дивитись урок / Small word: Перегляд "малих" слів.

16.12 Дашборд / Elementary курс / Дивитись урок / Домашні завдання: Перегляд ДЗ для уроку.

16.13 Дашборд / Elementary курс / Дивитись урок / Sentence pattern: Перегляд зразків речень.

16.14 Дашборд / Студент / Інформація: Швидкий доступ до інформації про студента.

16.15 Дашборд / Студент / Інформація / Ховер на родинні зв’язки: Спливаюче вікно з інформацією про родинні зв'язки.

16.16 Дашборд / Студент / ДЗ: Перегляд історії домашніх завдань студента.

16.17 Дашборд / Студент / Коментарі: Перегляд та додавання коментарів про студента.

Календар викладача:

17.1 Calendar: Персональний календар викладача.

17.2 Календар / місяць: Перегляд занять за місяць.

17.3 Календар / Сьогодні: Перегляд занять на поточний день.

Уроки:

18.1 Lessons (40 годин): Загальний розділ уроків.

18.2 Заняття / Заплановані: Список запланованих уроків.

18.3 Заняття / Заплановані / Превю уроку: Короткий огляд уроку.

18.4 Заняття / Заплановані / Превю уроку / Zoom: Можливість підключення до Zoom-конференції.

18.5 Урок / Початий урок: Інтерфейс активного уроку.

18.6 Урок / Згорнуті теми: Можливість згортання/розгортання тем уроку.

18.7 Урок / Відповіді: Можливість фіксації відповідей студентів під час уроку.

18.8 Урок / Правила: Можливість відображення правил уроку.

18.9 Урок / Студенти: Список студентів присутніх на уроці.

18.10 Урок / Нотатки: Можливість додавання нотаток під час уроку.

18.11 Заняття / Урок / Завершити урок: Функціональність для завершення уроку.

18.12 Заняття / Підсумок уроку: Формування підсумку уроку (присутність, оцінки, коментарі).

18.13 Заняття / Урок / Завершити урок: Підтвердження завершення.

18.14 Заняття / Відбулись: Список проведених уроків.

18.15 Заняття / Відбулись / Переглянути / Теми: Перегляд тем проведеного уроку.

18.16 Заняття / Відбулись / Теми: Перегляд тем.

18.17 Заняття / Відбулись / Переглянути / Студенти: Перегляд списку студентів, які були присутні на уроці.

18.18 Заняття / Відбулись / Переглянути / Коментарі: Перегляд коментарів до уроку.

Управління групами (Викладач):

19.1 Groups: Розділ груп викладача.

19.2 Групи: Перегляд груп, які веде викладач.

19.3 Групи / Фільтри: Фільтрація груп.

19.4 Групи / Фільтри обрано: Відображення застосованих фільтрів.

19.5 Групи / Група / Загальна інформація: Перегляд загальної інформації про групу.

19.6 Групи / Група / Календар занять / Місяць: Перегляд розкладу групи за місяць.

19.7 Дашборд / Група / Календар / Сьогодні: Перегляд розкладу групи на сьогодні.

Управління студентами (Викладач):

20.1 Students: Розділ студентів викладача.

20.2 Студенти: Перегляд студентів в групах викладача.

20.3 Студенти / Фільтри: Фільтрація студентів.

20.4 Студенти / Фільтри / Фільтри обрано: Відображення застосованих фільтрів.

20.5 Студенти / Студент / Інформація: Перегляд інформації про студента.

20.6 Студенти / Студент / Інформація / Ховер на родинні зв’язки: Спливаюче вікно з інформацією про родинні зв'язки.

20.7 Студенти / Студент / ДЗ: Перегляд домашніх завдань студента.

20.8 Студенти / Студент / Коментарі: Перегляд та додавання коментарів про студента.

20.9 Студенти / Студент / Коментарі (дублюється): Перевірка функціоналу.

Налаштування викладача:

21.1 Settings: Загальні налаштування викладача.

21.2 Налаштування / Загальні: Загальні налаштування профілю.

21.3 Налаштування / Повідомлення: Налаштування сповіщень.

21.4 Налаштування / Віджети (8 годин): Налаштування відображення віджетів на дашборді.

Сповіщення викладача:

22.1 Notifications: Розділ сповіщень для викладача.

22.2 Сповіщення / Всі: Перегляд всіх сповіщень.

22.3 Сповіщення / Не прочитані: Перегляд непрочитаних сповіщень.

22.4 Сповіщення / Прочитані: Перегляд прочитаних сповіщень.

Домашні завдання, тести, оцінювання (Викладач):

23.1 ДЗ,Тести, Оцінювання: Центральний розділ для управління навчальним процесом.

23.2 Розділ зі списком груп, в яких викладає вчитель: Перегляд всіх груп.

23.3 Сегментація по мові викладання, рівню, типу (ДЗ, урок, тест): Фільтрація ДЗ/тестів за критеріями.

23.4 Розділ зі списком рівнів в яких зберігається ДЗ: Організація ДЗ за рівнями.

23.5 Розділ рівня зі списком ДЗ, тестів, модулів: Перегляд ДЗ/тестів/модулів для певного рівня.

23.6 Список студентів: Перегляд студентів.

23.7 Автоматична відправка після "Завершення уроку" ДЗ, тестів, модулів: Автоматичне надсилання завдань після уроку.

23.8 Сторінка ДЗ, Тесту (слова, слова до відео, small words, SP...): Інтерфейс для створення та перегляду ДЗ/тестів.

23.9 Перевірка ДЗ, тесту та відправка студенту: Можливість перевірки робіт та надсилання зворотного зв'язку.

23.10 Статуси перевірено, не перевірено, на доопрацюванні, не здані: Управління статусами ДЗ.

23.11 Простановка оцінки за ДЗ, Тест з коментарями для студента: Система оцінювання з можливістю додавання коментарів.

Функціональність для студента:

Реєстрація та Авторизація:

24.1 Log in / Sign Up: Сторінка входу та реєстрації.

24.2 Логін: Функціональність входу до системи.

24.3 Реєстрація 1/3: Перший крок реєстрації.

24.4 Реєстрація 2/3: Другий крок реєстрації.

24.5 Реєстрація 3/3: Третій крок реєстрації.

24.6 Реєстрація з соцмережами 1/3: Реєстрація через соціальні мережі.

24.7 Реєстрація з соцмережами 2/3: Другий крок реєстрації через соцмережі.

24.8 Реєстрація з соцмережами 3/3: Третій крок реєстрації через соцмережі.

Дашборд студента:

25.1 Dashboard: Центральний екран для студента.

25.2 Дашборд / Menu: Навігаційне меню.

25.3 Дашборд / Календар ваших уроків: Перегляд запланованих уроків.

25.4 Дашборд / Акційні пропозиції: Відображення актуальних акційних пропозицій.

25.5 Дашборд: Загальний вигляд дашборду.

25.6 Дашборд / Дивитись всі: Перегляд всіх акцій.

25.7 Дашборд / Сплачено Акційні пропозиції: Відображення акцій, які вже оплачені студентом.

25.8 Дашборд / Купівля рівня / Оплата: Функціональність для оплати навчання.

25.9 Дашборд / Купівля рівня / Рівні навчання / 1 Рівень: Вибір одного рівня для оплати.

25.10 Дашборд / Купівля рівня / Рівні навчання / 2 Рівня: Вибір двох і більше рівнів.

25.11 Дашборд / Купівля рівня / Категорія оплати: Вибір категорії оплати.

25.12 Дашборд / Купівля рівня / Оплата / Кредитна картка / Ввести дані картки: Введення даних банківської картки.

25.13 Дашборд / Купівля рівня / Оплата / По реквізитам: Оплата за банківськими реквізитами.

25.14 Дашборд / Купівля рівня / Оплата / Кредитна картка / Введено: Підтвердження введення даних.

25.15 Дашборд / Купівля рівня / Оплата / Кредитна картка: Сторінка оплати карткою.

25.16 Дашборд / Купівля рівня / Оплата / Кредитна картка / Додати картку: Можливість прив'язки картки.

25.17 Дашборд / Купівля рівня / Оплата / Кредитна картка / Додати картку / Введено: Підтвердження прив'язки.

Профіль студента:

26.1 Profile: Перегляд та редагування власного профілю.

26.2 Профіль / Персональна інформація: Ім'я, прізвище, контактні дані.

26.3 Профіль / Інформація про навчання: Рівень, група, дата початку, викладач.

26.4 Профіль / Історія оплат: Перегляд історії всіх оплат.

26.5 Профіль / Історія оплат / Дивитись більше / Не оплачено: Деталі неоплачених рахунків.

26.6 Профіль / Історія оплат / Дивитись більше / Оплачено: Деталі оплачених рахунків.

26.7 Профіль / Персональна інформація / Зберегти зміни: Збереження змін у профілі.

26.8 Профіль / Персональна інформація / Зберегти зміни / Попап підтвердження: Підтвердження збереження.

Налаштування студента:

27.1 Settings: Загальні налаштування студента.

27.2 Налаштування / Загальні: Загальні налаштування профілю.

27.3 Налаштування / Повідомлення: Налаштування сповіщень (email, sms).

27.4 Налаштування / Віджети (12 годин): Налаштування відображення віджетів на дашборді.

Домашні завдання, тести, оцінювання (Студент):

28.1 ДЗ,Тести, Оцінювання: Розділ для управління навчальними завданнями.

28.2 Список ДЗ або тестів, модулів, які пройшов або отримав: Перегляд всіх призначених та виконаних завдань.

28.3 Сторінка ДЗ, Тесту (слова, слова до відео, small words, SP...): Інтерфейс для виконання домашніх завдань та тестів.

28.4 Процес написання ДЗ, Тесту та відправка на перевірку вчителю: Функціональність для виконання та надсилання робіт.

28.5 Отримання ДЗ, Тесту з перевірки з коментарями вчителя: Перегляд перевірених робіт та коментарів викладача.

28.6 Отримання оцінки за виконане ДЗ, Тест: Перегляд оцінок за виконані завдання.

Функціональність для абітурієнта:

Дашборд абітурієнта:

29.1 Dashboard: Центральний екран для абітурієнта.

29.2 Дашборд: Огляд доступних курсів та акційних пропозицій.

29.3 Дашборд / Купівля рівня / Акційні пропозиції (30 годин): Відображення та можливість придбання акційних пропозицій.

29.4 Дашборд / Купівля рівня / Рівні навчання / 1 Рівень: Можливість придбати один рівень навчання.

29.5 Дашборд / Купівля рівня / Рівні навчання / 2 Рівня: Можливість придбати два або більше рівнів навчання.

29.6 Дашборд / Купівля рівня / Категорія оплати: Вибір категорії оплати.

29.7 Дашборд / Купівля рівня / Оплата: Перехід до сторінки оплати.

29.8 Дашборд / Купівля рівня / Оплата / Кредитна картка / Ввести дані картки: Введення даних банківської картки для оплати.

29.9 Дашборд / Купівля рівня / Оплата / По реквізитам: Можливість оплати за банківськими реквізитами.

29.10 Дашборд / Купівля рівня / Оплата / Кредитна картка / Ввести дані картки (дублюється): Перевірка функціоналу.

29.11 Дашборд / Купівля рівня / Оплата / Кредитна картка / Введено: Підтвердження введення даних.

Купівля рівня зі сімейною знижкою:

29.12 Купівля рівня / Оплата двох рівнів зі сімейною знижкою: Спеціальна опція для сімейної знижки.

29.13 Купівля рівня / Оплата двох рівнів зі сімейною знижкою / Зв’яжіться з менеджером: Повідомлення про необхідність зв'язку з менеджером.

29.14 Купівля рівня / Оплата двох рівнів зі сімейною знижкою / Зв’яжіться з менеджером / Дані для зв’язку з вами: Форма для введення контактних даних.

29.15 Купівля рівня / Оплата двох рівнів зі сімейною знижкою / Зв’яжіться з менеджером / Дата здвінка: Вибір бажаної дати дзвінка.

29.16 Купівля рівня / Оплата двох рівнів зі сімейною знижкою / Зв’яжіться з менеджером / Час дзвінка: Вибір бажаного часу дзвінка.

29.17 Купівля рівня / Оплата двох рівнів зі сімейною знижкою / Зв’яжіться з менеджером / Дані для зв’язку з вами (дублюється): Перевірка функціоналу.