КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА
Факультет комп'ютерних наук та кібернетики
Кафедра інтелектуальних програмних систем

Кваліфікаційна робота
на здобуття освітнього рівня бакалавра
за спеціальністю 121 Інженерія програмного забезпечення
На тему:

Розробка інтерфейсу користувача адміністративного модуля сервісу розкладу занять студентів з використанням бібліотеки Angular

Виконав студент 4-го курсу
Владислав ТОЧАНЕНКО

_____
(підпис)

Науковий керівник:
доцент, кандидат технічних наук
Євген ДЕМКІВСЬКИЙ

_____
(підпис)

Засвідчую, що в цій роботі немає запозичень з праць інших авторів без відповідних посилань.

Студент _____
(підпис)

Роботу розглянуто й допущено до захисту
На засіданні кафедри інтелектуальних програмних систем
«28» травня 2021р.,
протокол № 14
Завідувач кафедри
Олександр ПРОВОТАР

_____
(підпис)

Київ – 2021

РЕФЕРАТ

Обсяг роботи 41 сторінка, 5 зображень, 1 таблиця, 13 джерел посилань.

КОМПОНЕНТ, КОРИСТУВАЦЬКИЙ ІНТЕРФЕЙС, ОДНОСТОРІНКОВІ ДОДАТКИ, СЕРВЕР ДАНИХ, ТАБЛИЦІ СТИЛІВ, ANGULAR, HTTP, REST, SPA.

Об’єкт роботи: дослідження процесу створення односторінкових додатків та його частин у команді.

Мета роботи: створення та покращення компонентів адміністративного модулю сервісу Mytimetable.

Інструменти, які використовувались під час виконання роботи: мова програмування TypeScript, мова розмітки гіпертексту HTML, мова таблиці стилів CSS, інтегроване середовище розробки WebStorm, сервіс для організації виробничого процесу Trello.

Результат роботи: надано теоретичну інформацію про процес розробки односторінкових додатків, а також проаналізовані альтернативні технології для створення веб сайтів чи додатків. Розроблено компонент для адміністративного модулю сервісу розкладу занять студентів.

Інформація, яка описана в даній роботі, може допомогти розробникам швидше організувати робочий процес на нових чи вже існуючих проектах. Завдяки цьому зменшиться час на дослідження концепції односторінкових додатків та на порівняння сучасних технологій для організації процесу розробки у команді.

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАЧЕННЯ

CERN	Conseil Européen pour la Recherche Nucléaire
CHSS	Cascading HTML Style Sheets
CSS	Cascading Style Sheets
DOM	Document Object Model
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
JSON	JavaScript Object Notation
MPA	Multipage application
PWA	Progressive web application
REST	Representational state transfer
SDK	Software development kit
SPA	Single Page Application
URI	Uniform Resource Identifier
XML	Extensible Markup Language

ВСТУП

Оцінка сучасного стану об’єкта розробки. На сьогодні веб додатки є одною із найбільш швидко зростаючих галузей розробки користувацького програмного забезпечення. Із розвитком веб технологій з’явились нові види користувацьких продуктів. Одним з них є односторінкові додатки (SPA).

Концепція односторінкових додатків наближує веб сайти за зовнішнім видом до додатків, що були написані спеціально для платформи користувача. Односторінкові додатки, на відміну від звичайних сайтів, завантажують необхідний для роботи HTML, CSS та JavaScript код одразу для всього додатку, чи поступово у міру потреби. Це забезпечує гнучкість користувацького інтерфейсу та високу швидкість роботи. Також технологія односторінкових додатків значно спрощує розробку та підтримку додатків.

Мета роботи й завдання роботи. Метою роботи є надання теоретичної інформації про процес розробки односторінкових додатків, а також аналіз альтернативних технології для створення веб сайтів чи додатків; розробка компоненту для адміністративного модулю сервісу розкладу занять студентів.

Об’єкт, методи та засоби розроблення. Об’єктом розроблення є теоретичні відомості, що стосуються розробки односторінкових веб додатків у команді; компоненти адміністративного модулю сервісу розкладу занять студентів.

Можливі сфери застосування. Інформація, яка описана в даній роботі, може допомогти розробникам швидше організувати робочий процес на нових чи вже існуючих проектах. Завдяки цьому зменшиться час на дослідження концепції односторінкових додатків та на порівняння сучасних технологій для організації процесу розробки у команді.

РОЗДІЛ 1. ТЕХНОЛОГІЯ ОДНОСТОРІНКОВИХ ДОДАТКІВ

1.1 Концепція односторінкових додатків

1.1.1 Загальний опис концепції односторінкових додатків

Single Page Applications – односторінкові веб додатки, котрі надають користувачу лише одну HTML сторінку [1]. За допомогою JavaScript дані у додатку оновлюється динамічно, виключаючи необхідність постійного перезавантаження сторінки. Під час використання користувачем додатку, дані поступово завантажуються з серверу, або змінюються новими, що може зменшити кількість трафіку та потужності пристрою, які необхідні для нормального функціонування додатку. Статичні дані, такі як стилі, шапка та навігаційне меню сторінки, завантажуються лише один раз, із першим запуском додатку.

1.1.2 Переваги односторінкових додатків

Основними перевагами односторінкових додатків є:

- швидкість роботи: всі необхідні ресурси завантажуються під час першої сесії, при чому дані, котрі змінюються із часом, динамічно оновлюються із серверу;
- гнучкість користувацького інтерфейсу: за рахунок того, що увесь додаток фактично є однією сторінкою, зберігання даних сесії, керування елементами сторінки та збереженими даними потребують доволі простої імплементації від розробників;
- завантаження даних з серверу для доступу без підключення до мережі Internet: завдяки тому, що сторінка завантажується повністю один раз, з’являється можливість частково користуватись додатком при випадковому відключенні від мережі Internet, або навіть спеціальна розробка версій додатку, котрі працюватимуть без під’єднання до мережі Internet.

- зручність налагодження додатку: за рахунок того, що всі дії фактично виконуються на одній сторінці, налагодження програмного коду додатку стає значно простішим.

1.1.3 Недоліки односторінкових додатків

Основними недоліками односторінкових додатків є:

- оптимізація для пошукових сервісів потребує відмальовування сторінки на сервері;
- навантаження на браузер користувача: через те, що усі дії виконуються за допомогою JavaScript, односторінкові додатки можуть бути надто повільними на старих користувацьких пристроях, та зменшувати час автономної роботи як на старих, так і на нових користувацьких пристроях;
- всі дії виконуються за допомогою JavaScript, тому необхідно, щоб пристрій користувача підтримував запуск JavaScript у браузері;
- досить слабкий захист від зовнішнього втручання, адже дії виконуються на стороні користувача.

1.2 Опис технології HTTP

HTTP – протокол для передачі текстових документів між браузером та сервером. Протокол побудований на моделі клієнта – сервера. Клієнт створює запит на сервер, сервер відправляє відповідь на запит до клієнта. При передачі запитів та відповідей, проміжний стан не зберігається, тому збереження проміжного стану, якщо воно є необхідним, покладається на клієнта.

1.3 Опис технології AJAX

AJAX є основною технологією для розробки клієнтських веб додатків. Вона дозволяє оновлювати дані у DOM структурі сторінки без виконання операції оновлення усієї сторінки [2]. На звичайних багатосторінкових сайтах користувач

повинен чекати, поки сервер наддасть відповідь із певними даними. За допомогою AJAX користувачу дозволяється користуватись додатком вже під час запиту необхідних даних до серверу, що надає відчуття швидкості та чутливості додатку.

Технологія AJAX передбачає використання HTML та CSS для відображення елементів на сторінці додатку, DOM для роботи із частинами сторінки, запити на сервер для отримання необхідних даних, а JavaScript як проміжний рівень для з'єднання DOM з запитами на сервер (див. рисунок 1.1). AJAX запити є асинхронними, що дозволяє виконуватись одразу декільком запитам одночасно [3]. Завдяки цьому користувач може виконувати декілька дій одночасно, не чекаючи відповіді на запит від серверу.

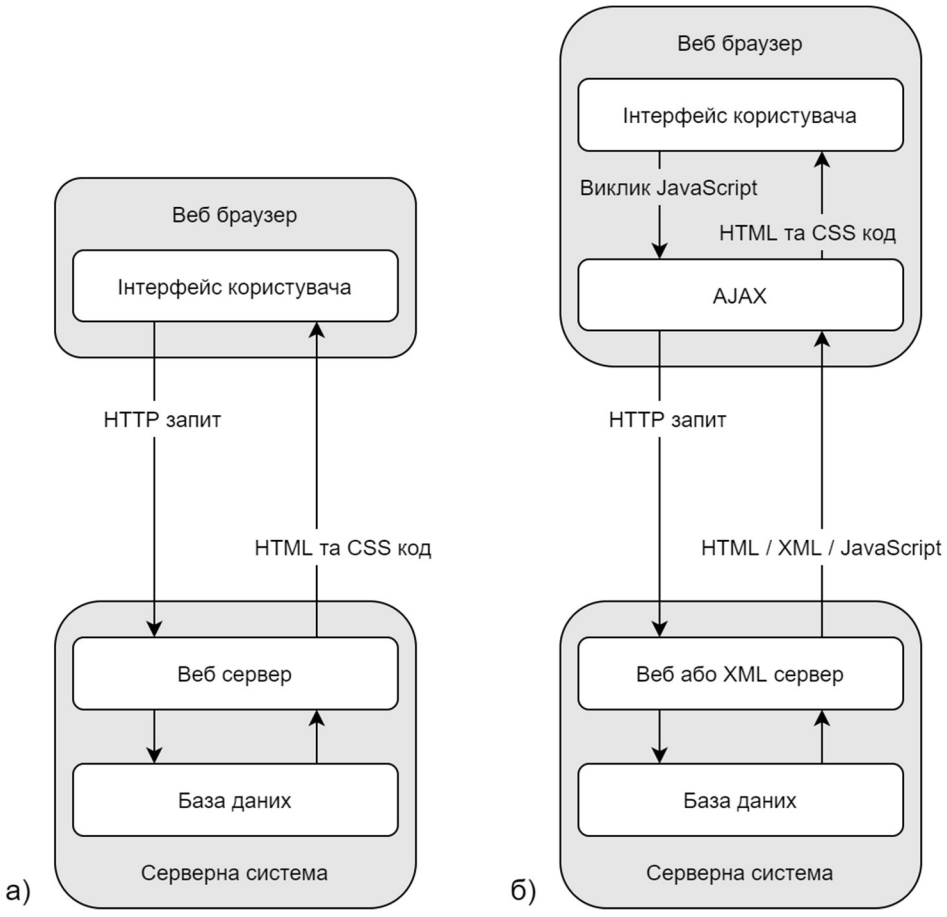

Рис. 1.1: а) запити звичайної веб сторінки; б) запити веб додатку, що використовує технологію AJAX (праворуч)

1.4 Опис технології REST

Технологія REST – підхід до дизайну мережевих протоколів, які забезпечують отримання чи зміну даних на сервері [4]. REST протоколи повинні задовольняти наступні вимоги:

- усі ресурси (зображення, відеоматеріали, документи та ін.) повинні передаватись у вигляді URI;
- ресурси повинні бути доступними через уніфікований інтерфейс, який зазвичай є HTTP;
- дані подаються в уніфікованих форматах XML, JSON або HTML;
- клієнти повинні використовувати посилання на ресурси, котрі отримує у відповіді сервера;
- протокол повинен бути без запам’ятовування стану, тобто кожен окремий запит ніяк не може бути пов’язаний із попереднім.

Веб додатки спілкуються з REST серверами за допомогою запитів HTTP. Такі запити можна називати безпечними, якщо після їх виконання дані на сервері не можуть змінитись. Також такі запити називають ідемпотентними, якщо при багаторазовому запуску одного і того ж самого запиту з однаковими даними, відповідь не зміниться. Існуючі HTTP запити приведені в таблиці 1.1.

Однією з головних переваг є кешування даних. Кожні дані повинні містити інформацію чи можливо їх записати у кеш, і, якщо так, скільки за часом вони можуть знаходитись у кеші. Це дозволяє зменшити кількість запитів на сервер, бо дані, які часто не змінюються, можуть знаходитись у кеші користувача. Перевагою цього також є зменшення часу завантаження сторінки на стороні користувача.

Таблиця 1.1: опис стандартних HTTP запитів

Назва	Опис	Безпечний	Ідемпотентний
GET	Отримати ресурс	Так	Так
POST	Додати новий ресурс	Ні	Ні
PUT	Оновити існуючий ресурс	Ні	Так
DELETE	Видалити існуючий ресурс	Ні	Так
PATCH	Оновити частину існуючого ресурсу	Так	Так
OPTIONS	Отримати список можливих HTTP запитів для ресурсу	Так	Так
HEAD	Отримати інформацію про ресурс	Так	Так

Сервери REST запитів використовуються не тільки для отримання даних у односторінкових додатках. За рахунок того, що спілкування між клієнтом та сервером відбувається через HTTP запити, клієнтами REST серверів можуть бути додатки на будь-яких платформах, які мають можливість доступу до мережі Internet.

1.5 Розподілення відповідальності та архітектурний шаблон MVC

Розподілення відповідальності є принципом розробки програмного забезпечення, при якому додаток розділяється на окремі модулі. Модулі складаються з взаємопов’язаних логічних елементів програми чи застосунку. Принцип розподілення відповідальності знижує складність розробки застосунку, дозволяє легко розширювати функціонал існуючих модулів.

В односторінкових додатках принцип розподілення відповідальності є архітектурним шаблоном MVC. Додатки, які розроблені за допомогою шаблону MVC, складаються за трьох окремих модулів:

- модель, що надає доступ до даних та змінює їх, в залежності від команд контролеру;

- представлення, яке відображає дані користувачу та реагує на їх зміну у моделі;
- контролер, який від залежності від дій користувача, надає певні команди для зміни даних до моделі.

На рисунку 1.2 наведена схема архітектурного шаблону MVC. На схемі наявний користувач, котрий взаємодіє з модулями представлення та контролеру.

Метою архітектурному шаблону MVC є розділення бізнес логіки від її візуалізації. Це дозволяє використовувати окремі частини візуалізації в різних частинах додатку. Завдяки цьому зменшується складність та час розробки додатку, полегшується розширення функціоналу додатку та підтримка існуючого додатку. Модулі можуть бути надані розробникам різних кваліфікацій, тобто розробники бізнес логіки можуть не брати учать в розробці інтерфейсу користувача, і навпаки, розробники користувацького інтерфейсу можуть лише використовувати готову бізнес логіку, котра розробляється іншими розробниками.

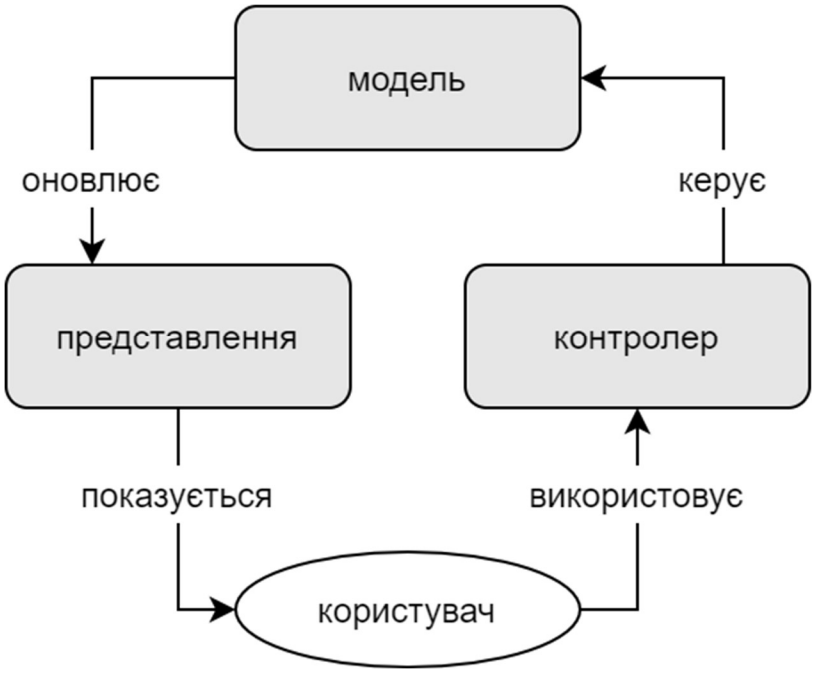

Рис. 1.2: архітектурний шаблон MVC

1.6 Процес прив’язки даних

Прив’язка даних є процесом, який встановлює зв’язок між користувацьким інтерфейсом та бізнес логікою. Цей процес є ключовим в додатках, які мають архітектуру MVC. Існує два види прив’язки даних: одностороння та двостороння.

Односторонній процес прив’язки даних передбачає передачу даних з моделі до представлення. Наприклад, модель може змінювати представлення, у зв’язку зі зміною даних у моделі. І навпаки, дані можуть передаватись з представлення до моделі. Наприклад, елемент на сторінці може викликати метод зміни даних.

Двосторонній процес прив’язки даних з’явився із появою сучасних фреймворків створення односторінкових додатків. Він передбачає можливість одночасної передачі даних з модуля до представлення, і навпаки. Необхідність використання такого процесу прив’язки даних з’являється із розширенням кількості компонентів веб додатку, адже одні і ті самі дані можуть відображатись на одному компоненті, а користувач може їх змінювати на іншому існуючому компоненті. Завдяки двосторонньому процесу прив’язки даних зникає необхідність створення окремих обробників подій для оновлення даних на представленні користувацького інтерфейсу.

1.7 Роутинг

Роутинг, або маршрутизація – модуль, який передає запити за завчасно заданими URI у відповідні методи програми. Наприклад, у додатках, що побудовані на архітектурі MVC, роутинг представлений у вигляді словника, який містить відповідні пари URI та параметрів, які передаються у відповідний метод програми. Завдяки модулю роутингу можлива навігація у односторінковому додатку. Зазвичай, URI вказує на окремий файл чи документ у мережі. У випадку односторінкових додатків, до статичного URI додатку додається хеш частини додатку, до котрого необхідно перейти. Завдяки цьому забезпечується зміна активних компонентів на користувацькому інтерфейсі без перезавантаження сторінки.

1.8 Порівняння концепції односторінкових додатків з іншими концепціями розробки веб сайтів та додатків

1.8.1 Порівняння концепції створення односторінкових додатків із концепцією створення багатосторінкових додатків

Multipage application – класична концепція розробки веб сайтів. Сайти складаються з окремих сторінок. Якщо дані оновлюються на сервері або на стороні користувача, сторінка повинна оновитись для оновлення даних на представленні. Популярними сайтами, що є багатосторінковими, були розроблені як статичні веб ресурси, котрі не матимуть великої кількості даних, що можуть змінюватись. Прикладом можуть бути веб сайти «візитки», котрі мають лише постійну інформацію про людину чи компанію. Такі веб сайти не є інтерактивними і мають на меті лише донести певну інформацію до користувача. Багатосторінковими сайтами можуть бути і інтерактивні веб ресурси, але їх розробка та підтримка вимагає багато зусиль зі сторони розробників, тому більшість розробників надають перевагу односторінковим додаткам.

Основними перевагами багатосторінкових сайтів є:

- проста оптимізація для пошукових сервісів, завдяки тому, що необхідні дані знаходяться на сторінці і змінюватись не можуть;
- просте користування сайту користувачем за рахунок класичної навігації посиланнями;
- можливість перегляду сайту у браузерах, котрі не мають підтримку мови програмування JavaScript;
- не потребують знань додаткових фреймворків, адже є можливість розробки веб сайтів лише за допомогою стандартних засобів: HTML, CSS та JavaScript.

Основними недоліками багатосторінкових сайтів є:

- сильна зв’язаність користувацького інтерфейсу та серверу, що не дає змогу їх окремої розробки;
- складна підтримка сайту, адже для додавання або зміни будь-якої частини сайту необхідно змінювати як клієнтську, так і серверну частину сайту.

У порівнянні з односторінковими веб додатками, багатосторінкові сайти:

- потребують знань лише базових технологій для розробки: HTML, CSS та JavaScript;
- займають набагато більше часу та ресурсів для розробки комплексних систем, на кшталт інтернет магазину або соціальної мережі;
- низька можливість розширення функціоналу готових продуктів;

1.8.2 Порівняння концепції створення односторінкових додатків із концепцією прогресивних веб додатків

Progressive Web Application – прогресивні веб додатки, які можуть встановлюватись на пристрій користувача як звичайний [5]. Вони мають можливість відправляти повідомлення користувачу, а також працювати без під’єднання до мережі Internet. Звичайний користувач може не помітити різниці між нативним додатком та прогресивним веб додатком.

Основними перевагами прогресивних веб додатків є:

- можливість розробки додатку одразу під найбільш популярні користувацькі платформи;
- можливість повністю функціонувати в режимі без підключення до мережі Internet;
- простота встановлення та видалення додатку;
- можливість створення прогресивного веб додатку з існуючого веб сайту чи веб додатку.

Основними недоліками прогресивних веб додатків є:

- швидкість роботи, адже у порівнянні із нативними додатками, прогресивний веб додаток фактично є браузером із завантаженим сайтом.

У порівнянні з односторінковими веб сайтами, прогресивні веб додатки:

- встановлюються як звичайні додатки та мають розширений функціонал, аніж подібні веб додатки.

1.8.3 Порівняння концепції створення односторінкових додатків із нативними додатками

Нативні додатки є додатками, які написані виключно для певної користувацької платформи. Такі додатки розробляються на певному визначеному наборі технологій. Наприклад, для розробки додатків для мобільної операційної системи Android, можна використовувати такі технології, як мова програмування Kotlin, дизайн додатку у вигляді XML файлів, а також бібліотеку Room для використання бази даних на мобільному пристрої [6]. Через строго визначений набір технологій, для розробки нативних додатків необхідні розробники вузьких спеціалізацій.

Основними перевагами нативних додатків є:

- надшвидка робота на пристроях, адже додатки розроблені виключно для поточної операційної системи;
- схожий зовнішній вигляд із іншими нативними додатками, адже для багатьох операційних систем існують бібліотеки, які мають у собі усі базові елементи користувацького інтерфейсу для додатків;
- можливість використання додаткових особливостей операційних систем;
- можливість використання додаткових особливостей апаратного забезпечення.

Основними недоліками нативних додатків є:

- складність розробки, адже для кожної операційної системи необхідно розробляти окрему версію додатку;
- оновлення операційної системи можуть викликати помилки при роботі додатку, або потребувати переробки частини додатку відповідно до нових орієнтирів у дизайні користувацького інтерфейсу нативного додатку певної операційної системи.

У порівнянні із односторінковими веб додатками, нативні додатки:

- мають набагато більш високу швидкість роботи;
- можуть використовувати особливості операційної системи та апаратного забезпечення;
- потребують набагато більше зусиль для розробки додатку;
- можуть запускатись лише на конкретних операційних системах.

1.9 Приклади існуючих односторінкових додатків

Сервіс BuiltWith [7] зберігає інформацію про велику кількість веб сайтів та додатків у мережі Internet. Прикладами односторінкових додатків є:

- сайт авіаліній Канади aircanada.com;
- сайт виробника товарів Adidas adidas.com;
- більшість сайтів компанії Alphabet, такі як YouTube, Google Maps та Gmail;
- сайти соціальних мереж таких, як Medium, Facebook та Twitter;
- сайт для розміщення репозиторіїв GitHub.

Велика кількість найпопулярніших веб сайтів є односторінковими додатками, або знаходяться у процесі розробки нових версій у вигляді односторінкового додатку.

РОЗДІЛ 2. ФРЕЙМВОРК ANGULAR

2.1 Загальний опис

Angular – фреймворк, розроблений у 2009 році, для створення швидких у розробці і використанні односторінкових додатків. Особливостями фреймворку є:

- структура всього додатку складається з окремих модулів та компонентів, що надає змогу простої їх підтримки та розробки нових компонентів додатку;
- колекція стандартних та користувацьких бібліотек як для розробки бізнес логіки, так і користувацького інтерфейсу;
- використання мови програмування TypeScript, що є версією JavaScript з покращеним механізмом слідкування за типами даних;
- офіціальне розширення для браузерів, що побудовані на основі Chromium, для більш зручної розробки та налагодження додатку.

2.2 Основи фреймворку

2.2.1 Компоненти

Додаток складається із багатьох користувацьких компонентів. Кожен компонент складається із окремих файлів бізнес логіки, користувацького інтерфейсу та таблиці стилів. Кожен компонент має HTML тег, який фреймворк Angular буде асоціювати з ним. Таким чином можна використовувати окремі компоненти всередині інших, за допомогою лише одного тегу, який зв’язаний з поточним компонентом.

В Angular існує можливість запису бізнес логіки, користувацького інтерфейсу та таблиці стилів лише в одному файлі, але наразі такий метод розробки інтерфейсу не рекомендований для використання, адже він погіршує можливість для легкої розробки та підтримки компоненту.

Компоненти можна групувати в окремі модулі. Вони фактично є незалежними частинами додатку із своїми стилями, роутингом та іншим.

Як і в багатьох інших веб фреймворках, в Angular є можливість вставки TypeScript коду у файли HTML за допомогою спеціальної послідовності символів. У поточному випадку використовується послідовність із двох відкриваючих фігурних дужок та двох закриваючих фігурних дужок.

Також існує можливість прив’язати дані в одну чи обидві сторони. При односторонній прив’язці даних використовується одна з послідовностей символів, в залежності від напрямку, в якому необхідно передати дані. Наприклад, при передачі даних з файлу бізнес логіки до файлу користувацького інтерфейсу, використовується послідовність з квадратних дужок, і послідовність із круглих дужок при передачі даних з файлу користувацького інтерфейсу до файлу бізнес логіки. Для використання двосторонньої прив’язки даних використовується послідовність з квадратних та круглих дужок одночасно.

Відмінною рисою фреймворку Angular є окремі керуючи атрибути у тегах HTML. За їх допомогою можна керувати які теги будуть відображатись на сторінці, відображати масив даних за допомогою атрибутів циклів або позначати стан тегів спеціальними атрибутами.

2.2.2 Інтерфейс командного рядку

Частиною фреймворку Angular є спеціальний інтерфейс командного рядку, який має назву «Angular CLI». За його допомогою можна створювати нові проекти Angular, додавати компоненти та модулі, запускати проект у режимі розробника чи компілювати його, запускати користувацькі тести чи тестувати проект на наявність помилок.

2.2.3 Основні бібліотеки

Angular включає кілька бібліотек, які вирішують більшість базових проблем, із якими зустрічаються розробники під час розробки веб сайтів чи односторінкових додатків. Бібліотеки, що наведені у списку нижче, значно спрощують процес розробки додатку, а також сприяють більш стабільній роботі додатку:

- Angular Router є бібліотекою для використання та регулювання роутингу у додатку;
- Angular Forms є спеціальною бібліотекою для перевірки та керування формами;
- Angular HttpClient є надійним HTTP клієнтом для встановлення зв'язку та отримання даних з REST серверів даних;
- Angular Animations є бібліотекою для керування анімацією;
- Angular PWA є бібліотекою для створення прогресивних веб додатків, які працюють на основі фреймворку Angular;
- Angular Schematics є потужним засобом для підтримки та розширення додатку на проектах великих масштабах.

2.3 Мова програмування TypeScript

Мова програмування TypeScript поєднує в собі простий синтаксис мови JavaScript, а також строгу типізацію, яка відсутня в стандартах ECMAScript. Код, що написаний на мові програмування TypeScript, перетворюється на код мовою JavaScript завдяки процесу транспіляції. Транспіляція – процес перетворення програмного коду з однієї мови програмування у іншу. Важливо зазначити, що код, написаний на сучасних стандартах JavaScript транспілюється у більш старий стандарт ECMAScript 5. Код TypeScript транспілюється у код стандарту ECMAScript 3, що надає змогу його виконанню у більш старих браузерах.

Як зазначалось раніше, перевагою TypeScript є строга типізація. Завдяки цьому з’являється велика кількість обмежень на можливості роботи зі змінними. Наприклад, у JavaScript змінна може зберігати дані будь-якого типу, включаючи той, що не був наданий цій змінній у момент її ініціалізації. Це може спричинити низку проблем, які зв’язані із неправильною інтерпретацію типу. Проблеми із інтерпретацією типу є одними із найпоширеніших при розробці програм на мові JavaScript. TypeScript вирішує більшість з цих проблем, надаючи конкретний тип змінній при її ініціалізації. Таким чином повідомлення про помилку з типами розробник отримає уже під час розробки додатку, а не під час його виконання у браузері.

У TypeScript зберіглась можливість зробити будь-яку змінну динамічного типу. Для цього існує спеціальний тип «any», котрий дозволяє записувати дані різних типів до цієї змінної. Такий функціонал є дуже зручним під час розробки додатків, які спілкуються із невідомим REST сервером, адже завчасно може не бути відомо у якій саме формі REST сервер надаватиме дані. Якщо ж завчасно відомо структура даних, можна задати спеціальний користувацький тип даних змінній. Таким чином додаток завжди отримуватиме дані у правильному вигляді, чи надаватиме сповіщення про помилку типу даних, які були отримані з REST серверу.

У порівнянні з JavaScript, у TypeScript набагато краще розвинена концепція ООП. Щоб оголосити поля класу у JavaScript, необхідно їх описати в конструкторі класу, а модифікаторів доступу не існує, адже усі поля та методи класу є публічними. У TypeScript, на відміну від JavaScript, існує можливість оголошення полів у класі поза його конструктору, а також є три загальновідомі модифікатори доступу: публічний, приватний за захищений. За замовченням усі поля та методи класу є публічні, як і у JavaScript, але є можливість звуження їх області видимості до приватних, що робить їх видимими тільки в поточному класі. Також є можливість використання захищеного модифікатору, який дозволяє отримати доступ до поля чи методу лише в поточному класі, чи класах, які його успадковують.

Не менш важливою особливістю мови TypeScript є наявність інтерфейсів. Концепція інтерфейсу надає можливість описати лише структуру класу, без описання його логічної частини. Створити змінну типу інтерфейсу неможливо, проте є можливість написання класів, до яких застосовується інтерфейс. Такий функціонал дозволяє використовувати змінні, що є екземплярами різних класів, до яких застосовується один і той самий інтерфейс.

2.4 Фреймворк AngularJS

У 2009 році вийшов перший реліз фреймворку AngularJS. Його назва складається з загальної назви фреймворку та мови програмування, на котрій пишуть код – JavaScript. Ця версія фреймворку не була ідеальна, адже вона займала досить великий об’єм пам’яті та мала доволі низьку швидкість роботи. Розробка веб додатків була доволі складною, адже основні концепти могли змінюватись із кожною версією фреймворку. Не дивлячись на це, AngularJS був інноваційним у сфері HTML файлів компонентів. Завдяки спеціальним атрибутам, які писались у HTML тегах, розробники мали простий та швидкий спосіб змінювати зовнішній вигляд сторінки за певними умовами.

Через низьку стабільність роботи та постійну зміну основних концептів, розробники з Google вирішили розробити нову версію фреймворку Angular, котра мала б усі зручні та зрозумілі концепти з першої версії, але була б набагато простішою та стабільною. Версія фреймворку Angular 2 була написана без використання коду минулої, адже змінились основні концепти, але вони стали стандартними.

Нова версія, на відміну від попередньої, має за основу чотири потужні бібліотеки. Завдяки їх використанню зменшилась кількість загального коду та збільшилось розширюваність додатку.

2.4.1 Мова програмування TypeScript

Як було описано раніше, мова програмування TypeScript була обрана як основна для розробки додатків з використанням фреймворку Angular. Завдяки строгій типізації зменшилась загальна кількість помилок під час виконання коду. Завдяки покращеним можливостям ООП, збільшився функціонал та синтаксис став більш зрозумілим для розробників.

2.4.2 Бібліотека Observable

Бібліотека Observable була обрана як основна для отримання даних з асинхронного потоку. Це дозволяє отримувати данні поступово, не зберігаючи їх у величезних масивах. Таким чином зменшується навантаження на пам’ять та полегшується розробка компонентів, що отримують дані з REST серверів.

2.4.3 Бібліотека RxJS

Бібліотека RxJS використовує Observable у поєднанні зі спеціальними операторами огляду, обробки, фільтрування або навіть поєднування потоків даних, щоб створювати більш потужні потоки даних [8]. Існують декілька основних операторів для об’єднання потоків даних, при використанні котрих змінюється послідовність отримання даних.

2.4.4 Бібліотека Zone.js

Zone.js є розширенням для виконання асинхронних операцій. Ця бібліотека запускає прослуховувачі подій, такі як зміна DOM елементів або обробка HTTP запитів, в окремій віртуальній зоні, що надає змогу надання контролю над асинхронними операціями. Наприклад, існують можливості отримання поточного стану, відміни операцій та обробку помилок.

РОЗДІЛ 3. ОГЛЯД СЕРВІСУ MYTIMETABLE

3.1 Технології що використовуються на проекті

3.1.1 SCSS

SCSS є скриптовою мовою, яка інтерпретується у класичні таблиці стилів CSS. У порівнянні із CSS, SCSS має більш потужний інструмент змінних, можливість виконання математичних обчислень із змінними сумісних типів без використання методу калькуляції. Також особливостями є міксини, які слугують як окремі стилі, які є спільними для декількох інших. До міксинів можна передавати параметри, а також використовувати більше, ніж два міксина одночасно. Існує два стандарти написання стилів за допомогою SCSS. Перший має назву SASS і він є оригінальним, оскільки не потребує фігурних дужок. Другий називається SCSS, що означає, що він має звичайний синтаксис CSS із усіма можливостями скриптової мови.

Потужним інструментом для створення великої кількості подібних стилів є цикли. За їх допомогою спрощується створення стилів, що мають властивості, які залежать від певних змінних. Наприклад, цикли можна використовувати для створення стилів сітки таблиці, у котрій від того, котру частину сторінки займає блок, залежить його ширина.

Не менш важливим інструментом є успадкування. Завдяки йому можна створювати вкладені стилі, які надають змогу набагато зменшити кількість коду вкладених класів, а також надати можливість легкого перенесення стилів із файлу стилів одного компоненту, до файлу стилів іншого.

3.1.2 Angular Material

Angular Material та Bootstrap є двома найпопулярнішими бібліотеками користувацького інтерфейсу для сучасних версій фреймворку Angular. Перша стабільна версія бібліотеки Bootstrap вийшла ще в 2012 році. Angular Material, який

вийшов в 2014 році, за 2 роки обійшов за кількістю пошукових запитів у Google, а у 2020 році займає вдвічі більше пошукових запитів, ніж бібліотека Bootstrap [9].

Бібліотека Angular Material надає низку елементів користувацького інтерфейсу, які розроблені за допомогою документації Material Design, що є філософією дизайну, яку Google представив на конференції розробників Google I/O 2014 [10]. Саме завдяки цьому у розробників веб додатків з'явилась можливість легкого створення програмних продуктів, які б мали сучасний та простий дизайн. Виходячи з того, що додатки, що написані за допомогою фреймворку Angular є односторінковими, у поєднанні із дизайном, який є популярним на мобільній операційний системі Android, розробники можуть створювати односторінкові додатки, що є схожими за зовнішнім виглядом до нативних мобільних додатків.

Набір стандартних компонентів включає в себе, але не обмежується кнопками, формами, картками, навігаційними меню, сіткою розташування елементів тощо. Кожен з цих елементів має різні версії стилювання, щоб надати можливість персоналізувати зовнішній вигляд окремих компонентів, чого розробники не можуть зробити під час використання бібліотеки Bootstrap. Крім різних стилів елементів, у бібліотеці Angular Material є можливість задання головного та другорядного кольорів додатку. Фактично вони є звичайними змінними, за допомогою яких можна швидко змінювати кольорове оформлення всього сайту.

Разом із готовими елементами користувацького інтерфейсу, Angular Material має CDK, який дозволяє розробникам розробляти схожі за функціоналом та зовнішнім виглядом елементи, але із більш широкою можливістю із персоналізації.

3.1.3 Бібліотека тестування Karma

Karma є спеціальною бібліотекою для тестування JavaScript коду [11]. Розробником цієї бібліотеки є команда Angular, яка називає її застосунком для автоматизації запуску тестів. Основними функціями бібліотеки Karma є:

- запуск окремого веб серверу, на який завантажуються файли JavaScript коду та тестів;
- запуск окремого вікна одного чи декількох веб браузерів для подальшої роботи;
- запуск файлів JavaScript та відповідних тестів у правильній послідовності для уникнення помилок при тестуванні коду;
- генерування звіту, виконання тестів;
- автоматичний перезапуск тестів при модифікації файлів JavaScript коду чи файлів тестів.

Спеціально для фреймворку Angular було розроблено інструмент Karma Generator, який допомагає розробникам налаштувати проект для використання бібліотеки Karma.

3.2 Опис функціоналу сервісу Mytimetable

Сервіс Mytimetable складається із двох модулів: користувацький та адміністративний. Користувацький модуль складається з функціоналу, який надає можливість студентам та викладачам краще орієнтуватись в розкладі навчального процесу та у розкладі залікового та екзаменаційних контролів.

На головній сторінці надається можливість обирання розкладу для студентів чи викладачів. Кожен користувач має можливість перегляду обидвох частин не залежно від того чи є користувач студентом чи викладачем.

3.2.1 Частина сервісу із розкладом для студентів

Розглянемо частину сервісу, який присвячений розкладу для студентів. Після натискання на кнопку «Розклад для студентів», користувачу надається можливість обирання одного з чотирьох університетів: Київський національний університет імені Тараса Шевченка, Київський національний університет технологій та дизайну, Національний університет «Києво-Могилянська академія» та Національний

технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського». Як тільки користувач обрав необхідний університет, йому надається можливість обрати необхідний для нього факультет. Наразі у системі існують факультети Київського національного університету імені Тараса Шевченка такі, як:

- історичний факультет;
- факультет комп'ютерних наук та кібернетики;
- факультет радіофізики, електроніки та комп'ютерних систем;
- економічний факультет.

Після обирання потрібного користувачу факультету, користувач повинен буде обрати необхідну йому групу. Після того, як користувач обрав групу, йому одразу надається сторінка із розкладом за днями тижня. При чому ця сторінка має як версію для персональних комп'ютерів, так і для мобільних пристроїв. Розклад кожного дня складається із карток, котрі розташовані у відповідних до номеру заняття. Кожна картка включає в собі таку важливу інформацію, як ПІБ викладача, назву навчальної дисципліни, номер підгрупи, вид тижня, на котрому проводиться заняття: непарний чи парний, та тип заняття: лекція, практична робота чи семінар. Якщо користувач натисне на картку, йому надається більш детальна інформація, що стосується обраного заняття: номер тижнів, за якими воно відбувається, вказівка чи дане заняття буде проводитись в онлайн форматі, і посилання для під'єднання до заняття, якщо таке існує.

Якщо на мобільних пристроях обраний день, у котрий студент чи викладач не має пар, буде відображено повідомлення «Цього дня заняття відсутні».

3.2.2 Частина сервісу із розкладом для викладачів

Розглянемо другу частину сервісу Mytimetable, яка надає можливість переглядати розклад для необхідного викладача. Після натискання на кнопку «Розклад для викладачів», користувачу надається можливість обрати один з існуючих

в системі університетів: Київський національний університет імені Тараса Шевченка, Київський національний університет технологій та дизайну, Національний університет «Києво-Могилянська академія» та Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського». Як тільки користувач обрав необхідний для нього університет, йому надається можливість скористатись пошуком з випадаючого списку чи за допомогою введення його прізвища у відповідне поле на сторінці.

Як тільки користувач обрав ПІБ викладача, для котрого йому необхідно подивитись розклад, користувачу одразу надається сторінка із розкладом за днями тижня. При чому ця сторінка, так само як і з розкладом для студентів, має як версію для персональних комп'ютерів, так і для мобільних пристроїв. Розклад кожного дня складається із карток, котрі розташовані у відповідних до номеру заняття. Кожна картка включає в собі таку важливу інформацію, як назву навчальної дисципліни, назву групи, номер підгрупи, вид тижня: парний чи непарний, та тип заняття: лекція, практична робота чи семінар.

3.2.3 Сторінка для співпраці

За допомогою натискання на значок трьох крапок, який знаходиться у правому верхньому куті будь-якої сторінки, користувач може зв'язатись із відділом масштабування та просування. За допомогою наданої форми будь-який представник від університету, чи представник студентської організації університету, має можливість подати заявку на використання сервісу Mytimetable як адміністратор.

3.2.4 Додаткові функції

Для забезпечення більш зручного користувацького функціоналу, сервіс надає можливість «запам'ятовувати» головній сторінці розклад якої з груп чи викладачів користувач нещодавно переглядав. Таким чином користувачу не потрібно кожного

разу обирати потрібний для нього розклад, а достатньо обрати його зі списку, що розташований на головній сторінці сервісу.

Також сервіс надає можливість перегляду розкладу сесії за групою та викладачем. Розклад сесії надається у вигляді календаря на персональних комп'ютерах, та списку на мобільних пристроях. Кожен з цих видів складається із карток, що мають інформацію про час проведення контролю, його тип, назву навчальної дисципліни. Також надається інформація про викладачів, які будуть присутні на контролі, якщо переглядається розклад сесії для групи. Відповідно, надається інформація про номер групи, якщо переглядається розклад сесії для викладача.

3.2.5 Опис адміністративної системи

Допоміжним інструментом для сервісу Mytimetable є модуль адміністративної панелі. Користувач, який має адміністративні права, має можливість створювати, додавати та видаляти основні сутності, що відносяться до розкладу студентів чи викладачів. Кожному такому користувачу надається доступ лише до окремого університету, щоб зменшити вірогідність ризику ненавмисного шкідливого внесення змін у дані інших університетів.

Окрім статусу адміністратора, користувачі адміністративної моделі можуть мати статус персоналу. Таким користувачам надається повний доступ до всього сервісу. За допомоги цього надається можливість вносити необхідні зміни користувачам, котрі є частиною команди, що розробляє сервіс Mytimetable.

Також існують додаткові можливості, котрі спрощують формування файлу готового розкладу для друку, або завантаження додаткових дій для модифікації даних на сервері для певного семестру певної групи.

РОЗДІЛ 4. РОЗРОБКА КОМПОНЕНТУ ДЛЯ АДМІНІСТРАТИВНОЇ ПАНЕЛІ

Процес розробки додаткових компонентів та частин адміністративної панелі схожий на процес розробки будь-якого продукту у компанії. Існує команда розробників, котрі використовують спеціальні сервіси для налагодження роботи над проектом.

Станом на сьогоднішній день існує нова версія адміністративної панелі, котра знаходиться у статусі розробки.

4.1 Інструменти які використовуються на проекті

4.1.1 Система управління проектами Trello

Trello виконує роль системи організації процесу розробки програмного забезпечення. У процес розробки входять такі важливі складові, як складання розкладу роботи, надання завдань певним розробникам, можливість спілкування між розробники, документування роботи та її керування [12]. Trello є безкоштовним програмним забезпеченням, який використовує парадигму програмування канбан.

Канбан – метод організації робочого процесу за допомогою дошки. Робочі завдання надаються в графічному вигляді (шматочки паперу), що візуалізує прогрес роботи над завданнями. У класичному вигляді дошки із завданнями існує п’ять основних колонок, але їх можна змінювати, відповідно до робочого процесу:

- резерв завдань;
- завдання у процесі;
- завдання, що потребують перегляду;
- завдання, що потребують тестування;
- готові завдання.

Trello надає можливість самостійного створення колонок з будь-якими призначеннями. На проекті вони були налаштовані, як: резерв, завдання, в прогресі,

потреба відгуку, готові завдання. Завдання представлені у вигляді карток, які можна легко переміщувати з однієї колонки до іншої. До завдань можна надавати таку додаткову інформацію, як зображення, категорія, ім’я розробника та додаткові файли.

4.1.2 Система контролю версій Git

Системи контролю версій значно спрощують розробку і підтримку програмного продукту. Завдяки таким системам спрощується процес вирішення проблем при об’єднанні роботи різних розробників, а також забезпечується цілісність історії програмного коду.

Існує три найпопулярніші інструменти контролю версій: Git, SVN та Mercurial. Згідно статистики [13], система контролю версій Git є найбільш популярною із наданих трьох. Linus Torvalds розробив систему Git, коли працював над ядром операційної системи Linux.

Нова версія адміністративного модулю сервісу Mytimetable розробляється за допомогою використання системи контролю версій Git.

Концепція Git включає у себе три сутності: репозиторій, гілка та коміт:

- коміт: зафіксована інформація про внесені зміни, що включає у себе ім’я та електронну адресу автора внесених змін, їх дату та час, короткий опис змін, а також хеш, який вираховується, виходячи із минулого коміту;
- репозиторій – папка, у якій зберігаються дані про зміни у файлах проекту, при чому зберігаються дані як про поточні зміни, так і про минулі.
- гілка – окрема лінія розробки додатку, яка має у собі послідовність комітів.

Однією з головних переваг є надійність історії внесених змін. Завдяки тому, що при створенні коміту для нього вираховується хеш, виходячи з попереднього, є

швидка можливість перевірки на цілісність та оригінальність наявних комітів у поточному репозиторії.

4.1.3 Сервіс Bitbucket

Через популярність систем контролю версій, а зокрема системи Git, з’явилась необхідність полегшення її використання шляхом створення відповідних веб сервісів. Таким чином з’явились безліч веб сервісів для розміщення репозиторіїв. Найпопулярнішими з них є GitHub, Bitbucket та GitLab. Такі сервіси надають можливість безкоштовного або платного розміщення репозиторію на віддалених серверах. Окрім надання веб інтерфейсу для використання стандартних функцій системи контролю версій Git, веб ресурси надають додаткові можливості такі, як коментування коду, контроль над можливостями, до котрих розробники мають доступ та інше.

Виконана робота була розміщена у репозиторії проектну нової версії Адміністративної панелі на веб сервісі Bitbucket. Також код розроблених компонентів у рамках поточної кваліфікаційної роботи був розміщений у репозиторії на веб сервісі GitHub.

4.1.4 Інтегроване середовище розробки WebStorm

WebStorm є інтегрованим середовищем розробки. WebStorm розроблений компанією JetBrains і має за основу IntelliJ IDEA, що є інтегрованим середовищем розробки програм на мовах програмування Java та Kotlin. На відміну від IntelliJ IDEA, WebStorm розрахований на веб розробку, включаючи розробку веб додатків за допомогою фреймворку Angular.

На проекті використовується WebStorm, який значно спрощує розробку та підтримку веб додатків.

4.2 Підготовка до роботи над проектом

Початок роботи над проектом потребує попередніх дій. Одним з них є встановлення необхідного програмного забезпечення: додатку для організації процесу розробки Trello, системи контролю версій Git та інтегрованого середовища розробки WebStorm.

Наступною дією підготовки до роботи над проектом є налаштування репозиторію. Так як репозиторій для проекту уже існує, його треба завантажити на персональний комп’ютер. Цю дію можна виконати за допомогою заздалегідь встановленої системи Git та посилання на потрібний репозиторій. Дія копіювання репозиторію з сервера на персональний комп’ютер називається клонуванням. Якщо репозиторій є приватним, необхідно ввести особисті дані для проходження авторизації на серверах Bitbucket.

Після того, як репозиторій був отриманий на персональний комп’ютер, його необхідно відкрити за допомогою інтегрованого середовища розробки WebStorm, яке налаштує проект для запуску автоматично.

Щоб запустити проект, необхідно скористатись однією із конфігурацій запуску. Якщо WebStorm не надав їх автоматично, можна додати необхідну конфігурацію запуску власноруч.

4.3 Процес розробки компоненту для адміністративної панелі

Під час роботи над проектом було отримано низку завдань від одного з розробників сервісу Mytimetable. Завдання мали різний ступінь складності виконання та вимагали комплексних знань для їх виконання. Одним з найскладніших завдань було розібратись зі структурою компонентів та кодом існуючого проекту.

4.3.1 Винесення віджету контролю до окремого модулю

Першим завданням на розробку було винесення віджету контролю (сесії) до окремого модулю. Для виконання цього завдання розглянемо структуру додатку (рис. 4.1), який складається з окремих компонентів, котрі є звичайними компонентами Angular, тобто мають у собі файл бізнес логіки, представлення інтерфейсу користувача та таблиці стилів. Компоненти групуються у окремі модулі за призначенням, котрі групуються як частини окремих частин сервісу (частина авторизації, панелі керування, тощо). Самі частини сервісу належать до структури усього проекту.

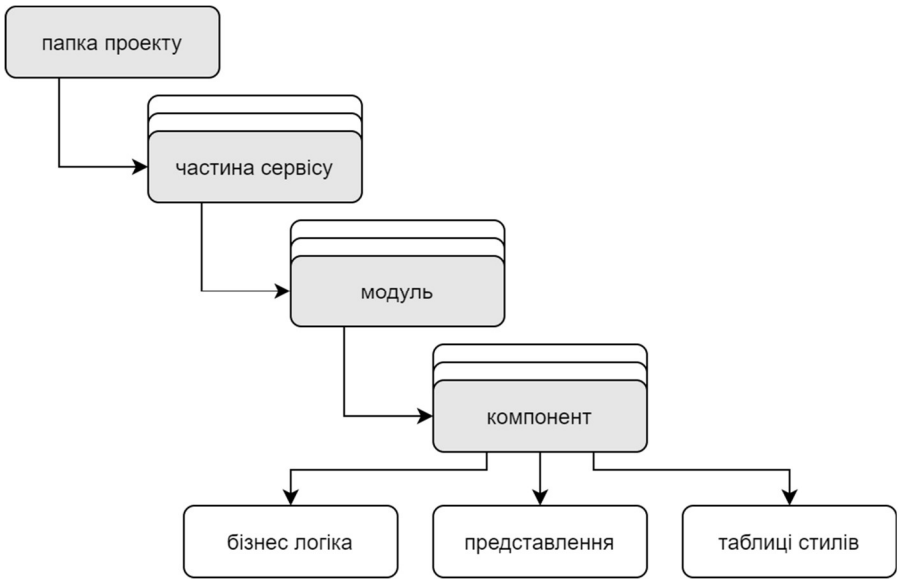

Рис. 4.1: структура проекту нової версії адміністративної панелі

Щоб компонент завантажувався при старті додатку як головна сторінка, необхідно змінити шлях роутингу до компоненту. На рисунку 4.2 (а) зазначена структура, за яким користувач мав можливість отримати доступ до компоненту SemesterControls. Сірими прямокутниками зазначений головний файл роутингу додатку, з якого починається просування за ієрархією додатку. Тобто при переході за

шляхом до головної сторінки сайту, саме файл роутингу app отримає керування над подальшими діями. Далі надається керування роутингу модулю dashboard, котрий в свою чергу передає керування модулю controls-schedule. Останній модуль відображає користувачу компонент StructureFilters, котрий використовує компонент SemesterControls. За завданням необхідно надати компонент SemesterControls роутингу app (рисунок 4.2 б).

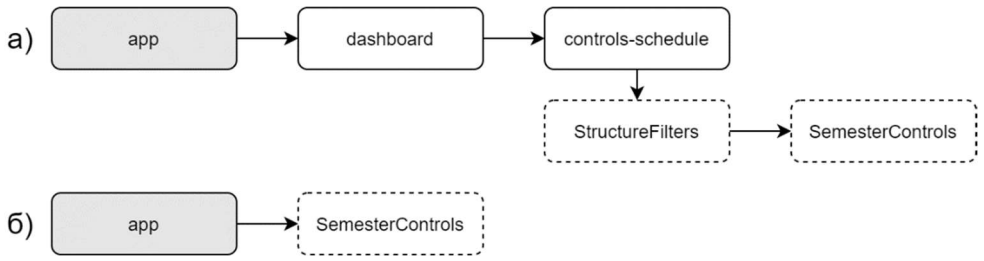

Рис. 4.2: а) початковий шлях до компоненту SemesterControls, б) потрібний шлях до компоненту SemesterControls.

Після зміни роутингу у додатку з’явилась низка проблем. Основною з них була неможливість отримання розкладу контролів для групи, адже компонент SemesterControls залежав від того, який університет, факультет, курс та групу буде обрано в компоненті StructureFilters. Для цього була розроблена можливість показу списку контролю на компоненті SemesterControls лише за кодом групи.

У результаті було отримано додаток, котрий надає користувачу сторінку із розкладом контролю за кодом групи. Такий компонент можна використовувати для пришвидшення розробки додатку у майбутніх версіях адміністративної панелі.

4.3.2 Розроблення компоненту отримання звіту

Наступним завданням було розробка нового компоненту, за допомогою якого можна отримати наступні типи звітів:

- розклад групи;
- розклад факультету;
- розклад аудиторій факультету;
- розклад сесії для групи;
- посилання на розклад занять факультету;
- посилання на розклад занять спеціальності;
- посилання на розклад сесії факультету;
- посилання на розклад сесії спеціальності;
- розклад викладача;
- розклад спеціальності;
- розклад курсу по факультету;
- посилання на розклад занять викладачів, що читають на факультеті;
- посилання на розклад сесії викладачів, що читають на факультеті;
- розклад аудиторій корпусу;
- розклад аудиторій корпусу (всі корпуси);
- розклад викладачів, що читають пари на спеціальності;
- розклад сесії викладача;
- розклад сесії для курсу.

Особливістю цього завдання було створення нового компоненту, який повністю повторює функціонал існуючого на першій версії адміністративної панелі. Іншими словами, було надано для використання сторінка із готовим функціоналом, котрий необхідно повторити.

Для розробки нового компоненту, був створений компонент додатку під назвою GetExcel, котрий повністю повторює назву сторінки з першої версії адміністративної панелі. Дизайн таких компонентів сторінки, як випадаючі списки та кнопки були зроблені за допомогою використання бібліотеки елементів користувацького інтерфейсу Angular Material.

Форма, яка розташована на сторінці, повинна бути динамічною, тобто змінюватись в залежності від того, який тип звіту був обраний користувачем. Для її створення була використана бібліотека Angular Forms, яка спрощує створення та перевірку форм, до яких користувач буде вносити дані. Перевагою цієї бібліотеки є можливість створення та модифікації елементів форми під час роботи додатку.

При обиранні типу звіту, сторінка отримає списки необхідних параметрів за допомогою запитів на сервер REST, котрі керуються спеціальним модулем для отримання даних. Цей модуль є універсальним для багатьох типів даних, котрі є стандартними для даного додатку: університети, групи, заняття, тощо.

Після цього, за допомогою визначено списку, відображається один чи декілька з восьми полів для обирання чи введення користувацьких даних. Такими полями є:

- факультет;
- спеціальність;
- курс;
- група;
- викладач;
- назва навчальної дисципліни;
- аудиторія;
- будівля.

Користувач повинен заповнити додаткові параметри. Після того, як користувач заповнив усі поля, користувач натискає на кнопку «Завантажити звіт», після чого починається процес завантаження потрібного користувачу звіту у форматі Excel.

Важливо звернути увагу, що форма добре вписується в загальний дизайн нової версії адміністративної панелі (рисунок 4.3), адже усі компоненти сторінки повторюють дизайн інших компонентів, які наявні у поточному проекті.

У результаті було отримано новий компонент, який додає новий функціонал на другу версію адміністративної панелі. Цей компонент слідкує дизайну адміністративного модулю та надає користувачам швидкий та зручний доступ до отримання необхідних звітів.

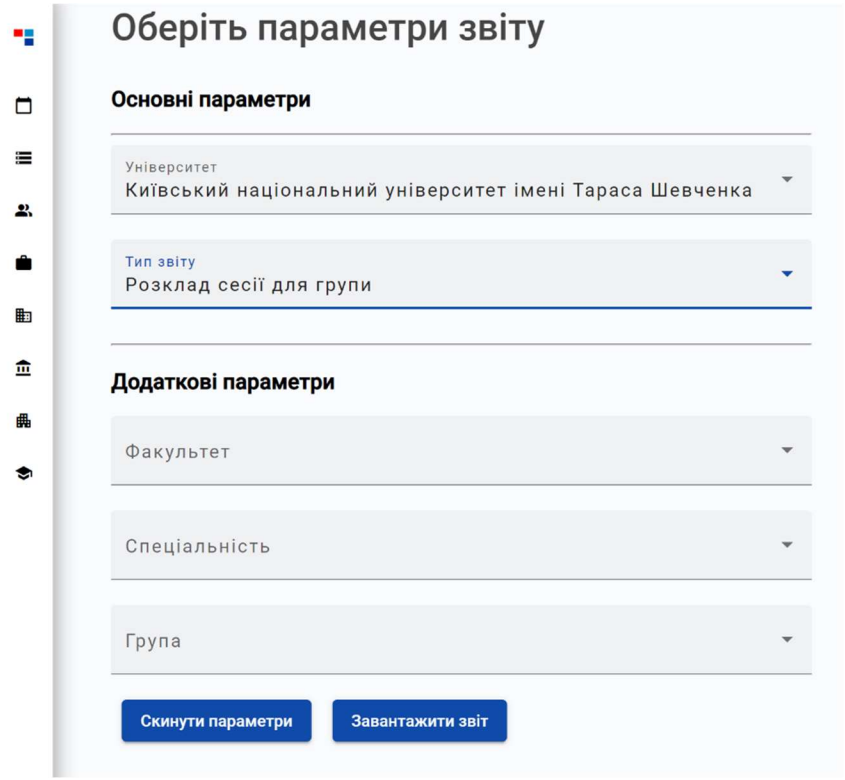

Рис. 4.3. Вигляд компоненту GetExcel

ВИСНОВКИ

Під час виконання дипломної роботи було:

- розглянуто розвиток технологій, які привели до появи сучасних інструментів для створення односторінкових додатків;
- приділено велику увагу до опису концепції односторінкових додатків;
- проаналізовані альтернативні концепції для розробки веб сайтів та додатків;
- описано підхід для створення односторінкових додатків з використанням фреймворку Angular ;
- розглянуто функціонал сервісу Mytimetable;
- проведено опис розробки частини адміністративної панелі;
- проведено робочий процес у команді, з використанням сучасних технологій для покращення та пришвидшення робочого процесу.

Під час розробки частини адміністративного модулю були використані сучасні інструменти Git, WebStorm та Trello, та також фреймворк Angular.

Дана робота має практичну значущість, адже описує концепцію та процес розробки односторінкових додатків у команді. Таким чином розробники зможуть швидше організувати робочий процес на нових чи вже існуючих проектах. Завдяки цьому зменшиться час на дослідження концепції односторінкових додатків та на порівняння сучасних технологій для організації процесу розробки у команді.

В подальшому може бути проведена робота для створення чи покращення інших компонентів на адміністративній панелі сервісу Mytimetable.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

[1] Klaus Nygård. Single page architecture as basis for web applications: Aalto University – 2015p. – 80c.

[2] Jesse James Garrett. Ajax: A New Approach to Web Applications – 2005p. – 5c.

[3] Michael Mahemoff. Ajax Design Patterns – 2006p. – 609c. – ISBN 987-0-596-10180-0

[4] Moriyoshi Ohara. Aggregating REST requests to accelerate Web 2.0 applications – 2010p. – 12c. - 10.1147/JRD.2009.2036974

[5] Andreas Biørn-Hansen, Tim A. Majchrzak, Tor-Morten Grønli. Progressive Web Apps: The Possible Web-native Unifier for Mobile Development – 2017p. – 351c. - ISBN: 978-989-758-246-2

[6] Marcin Moskala, Igor Wojda. Android development with Kotlin – 2017p. – 415c. – ISBN 978-1-78712368-7

[7] Built With: Web Technology Usage Trends [Електронний ресурс] – Режим доступу: https://trends.builtwith.com/

[8] Christoffer Noring. Architecting Angular Applications with Redux, RxJS, and NgRx – 2018p. – 364c. – ISBN 978-1-78712240-6

[9] Google Trends: Порівняння кількості запитів «angular material» та «angular bootstrap» з 2014 року [Електронний ресурс] – Режим доступу: https://trends.google.com/trends/explore?date=all&q=angular%20material,angular%20bootstrap

[10] Liam Spradlin, Android Police: Exclusive: Quantum Paper And Google's Upcoming Effort To Make Consistent UI Simple – 2014p. [Електронний ресурс] – Режим доступу: https://www.androidpolice.com/2014/06/11/exclusive-quantum-paper-and-googles-upcoming-effort-to-make-consistent-ui-simple/

[11] Karma: How It Works [Електронний ресурс] – Режим доступу: https://karma-runner.github.io/6.3/intro/how-it-works.html

[12] К.М. Лавріщева. Програмна інженерія : підруч. [для студ. вищ. навч. закл.] – 2008р. – 322с. – 300 пр. – ISBN 978–966–02–5052–9

[13] Google Trends: Порівняння кількості запитів «git», «svn» та «mercurial» з 2014 року [Електронний ресурс] – Режим доступу: https://trends.google.com/trends/explore?date=all&q=git,svn,mercurial