Міністерство освіти і науки України
Харківський національний університет імені В. Н. Каразіна
Факультет комп'ютерних наук
Кафедра теоретичної та прикладної системотехніки

«Затверджую»
Зав. кафедри теоретичної та
прикладної системотехніки
_____ д.т.н., проф. С. І. Шматков
«___» _____ 2024 р

Пояснювальна записка

до кваліфікаційної роботи
бакалавра

на тему: «**Веб-платформа для підготовки до співбесіди на посаду в IT-компанії**»

Захищено на засіданні
Атестаційної комісії № 42
протокол № __ від __.06.2024 р.
Оцінка _____ / _____
Голова Атестаційної комісії
_____ **СКОБ Ю. О.**

Виконав:
студент 4 курсу, групи КІ–41
Галузь знань: 12 – Інформаційні технології
Спеціальність: 123 – Комп'ютерна інженерія.
КУНЦЕВИЧ Максим Павлович _____

Керівник:
PhD, ст. викладач ЗВО
кафедри ТПС
МОРОЗ Ольга Юріївна _____

Рецензент: кандидат технічних наук, доцент, доцент ЗВО кафедри моделювання систем і технологій факультету комп'ютерних наук
ГАМЗАЄВ Рустам Олександрович___

Харків – 2024

АНОТАЦІЯ

Пояснювальна записка до кваліфікаційної роботи бакалавра складається зі вступу, трьох розділів, висновків, списку використаних джерел і двох додатків. Загальний обсяг роботи складає 65 сторінок, із яких 40 сторінок основної частини з 30 рисунками, 17 найменуваннями списку використаних джерел та чотирма додатками.

Метою дослідження є покращення підготовки до технічної співбесіди на посаду Junior Java Developer в IT компанії.

Об'єкт дослідження – процес підготовки проведення технічної співбесіди на посаду Junior Java Developer в IT компанії.

Методи дослідження*:* методи системного аналізу та імітаційного моделювання.

Предмет дослідження – функціональні можливості та інтерфейс веб-платформи, що забезпечують проведення технічної співбесіди на посаду в IT-компанії

Розробка програмної моделі веб-додатку для підготовки до співбесіди на посаду Junior Java Developer має значну практичну цінність для студентів факультету комп'ютерних наук. Вона не тільки допомагає студентам у підготовці до співбесід, але й сприяє розвитку їхніх практичних навичок, ознайомленню з сучасними технологіями та підвищенню їхньої конкурентоспроможності на ринку праці.

Ключові слова: *SPA (Single Page Applications), веб-платформа, клієнтська частина, серверна частина, Java, Docker, Angular, HTML, CSS, MySQL*

ABSTRACT

The explanatory note to the bachelor's thesis consists of an introduction, three chapters, conclusions, a list of references and four appendices. The total volume of the work is 65 pages, including 40 pages of the main part with 30 figures, 17 references and four appendices.

The purpose of the study is to improve preparation for a technical interview for the position of Junior Java Developer in an IT company.

The object of the study is the process of preparing a technical interview for the position of Junior Java Developer in an IT company.

Research methods: methods of system analysis and simulation modeling.

The subject of the study is the functionality and interface of the web platform, which ensure a technical interview for a position in an IT company

The development of a software model of a web application to prepare for an interview for the position of Junior Java Developer has significant practical value for students of the Faculty of Computer Science. It not only helps students prepare for interviews but also promotes the development of their practical skills, and familiarization with modern technologies and increases their competitiveness in the labor market.

Keywords: *SPA (Single Page Applications), web platform, client part, server part, Java, Docker, Angular, HTML, CSS, MySQL*

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ І УМОВНИХ ПОЗНАЧЕНЬ

API – (англ. Application Programming Interface) інтерфейс прикладного програмування;

HTML – (англ. HyperText Markup Language) мова розмітки гіпертексту;

CSS – (англ. Cascading Style Sheets) каскадні таблиці стилів;

SPA – (англ. Single-Page Application) односторінковий додаток;

HTTP – (англ. HyperText Transfer Protocol) протокол передачі гіпертексту;

IDE – (англ. Integrated Development Environment) інтегроване середовище розробки;

SQL – (англ. Structured Query Language) мова структурованих запитів;

UML – (англ. Unified Modeling Language) уніфікована мова моделювання.

DTO – (англ. Data Transfer Object) об’єкти для передачі даних між підсистемами застосунку.

IoC – (англ. Inversion of Control) принцип в об'єктно-орієнтованому програмуванні.

ORM – (англ. Object-Relational Mapping) це технологія, яка дозволяє зв'язувати об'єктно-орієнтовану модель програми з реляційною базою даних.

ВСТУП

В сучасному світі інформаційних технологій конкуренція на ринку праці в галузі ІТ стає все більш напруженою. Захоплюючі велику частку економіки, ця галузь вимагає від кандидатів не лише технічних знань, але й високих навичок комунікації та вміння ефективно висловлювати свої ідеї під час співбесіди. Велике значення надається не тільки знанням програмування, але і здатності ефективно взаємодіяти з командою, розв'язувати завдання високого рівня складності та адаптуватися до швидкозмінюваних технологічних вимог.

Актуальність роботи. Спроби займання посад в сфері інформаційних технологій в сучасному світі вимагають від кандидатів не лише технічних знань, але й широкого спектру навичок та вмінь. Одним із ключових етапів в процесі кар'єрного зростання є успішна співбесіда, під час якої важливо продемонструвати не тільки технічні компетенції, але й здатність до комунікації, розв'язання проблем та адаптації до нових завдань. Специфіка вимог індустрії та її постійна еволюція вносять низку викликів для кандидатів, які шукають роботу в сфері ІТ. Актуальність даного дослідження зумовлена тим, що відсутність ефективних інструментів для підготовки до співбесід у сфері інформаційних технологій може призвести до втрати потенційно обіцяючих талантів та ускладнити процес відбору кваліфікованих спеціалістів для компаній.

Завдяки широкому доступу до Інтернету та поширенню дистанційного навчання, створення веб-платформи для підготовки до співбесід у сфері ІТ стає настільки важливим, що може вирішити проблеми, пов'язані зі створенням адаптивних програм, спрямованих на покращення технічних та міжособистісних навичок майбутніх кандидатів. Така платформа може стати ефективним інструментом як для індивідуальної, так і для групової підготовки, розширюючи можливості кандидатів та забезпечуючи їхню готовність до викликів сучасного ринку праці в сфері інформаційних

технологій. Тому актуальність обраної теми не викликає сумнівів, оскільки зростаюча конкуренція на ринку праці вимагає від кандидатів високого рівня підготовки. Веб-додаток, який забезпечує інтерактивне навчання та тестування, може значно підвищити шанси кандидатів на успішне проходження співбесіди.

Основна задача дослідження – розробити програмну модель веб-додатку для підготовки до співбесіди на посаду Junior Java Developer в IT компанії, що сприятиме ефективній підготовці до співбесіди, охоплюючи теоретично-технічні аспекти.

Метою дослідження є покращення підготовки до технічної співбесіди на посаду Junior Java Developer в IT компанії.

Об'єкт дослідження – процес підготовки проведення технічної співбесіди на посаду Junior Java Developer в IT компанії.

Методи дослідження: методи системного аналізу та імітаційного моделювання для проведення комплексного огляду існуючих аналогів і визначення оптимальних програмних інструментів для реалізації функціоналу платформи. Методи імітаційного моделювання використовувалися для тестування програмної моделі та перевірки її функціональності.

Предмет дослідження – функціональні можливості та інтерфейс веб-платформи, що забезпечують проведення технічної співбесіди на посаду в IT-компанії

Задачі дослідження:

1. Аналіз існуючих аналогів веб-додатків для підготовки до співбесіди на посаду Junior Java Developer в IT компанії.
2. Аналіз програмних інструментів та технологій для реалізації функціоналу та дизайну платформи.
3. Аналіз принципів роботи клієнт-серверних моделей веб-додатків.
4. Вибір платформи для розробки додатку проходження співбесіди на посаду в IT.

5. Розробка програмної моделі веб-додатку для підготовки до співбесіди на посаду Junior Java Developer в IT компанії.
6. Тестування програмної моделі для перевірки функціональності веб-платформи.

Тематика роботи є надзвичайно актуальною в сучасному IT-середовищі, де постійно зростає потреба у висококваліфікованих фахівцях. Веб-додаток для підготовки до співбесіди на посаду Junior Java Developer, безумовно, є корисним і своєчасним інструментом, який має значний потенціал для практичного використання.

Розробка програмної моделі веб-додатку для підготовки до співбесіди на посаду Junior Java Developer в IT-компанії має значну практичну цінність для студентів факультету комп'ютерних наук. Веб-додаток надає студентам можливість систематично готуватися до співбесід на посаду Junior Java Developer, охоплюючи як теоретичні, так і практичні аспекти. Студенти матимуть можливість ознайомитися з різними технологіями та інструментами, що використовуються у сучасній розробці веб-додатків. Завдяки систематичній підготовці та відточенню навичок вони будуть більш підготовленими до реальних співбесід, що збільшить їх шанси на успішне працевлаштування.

РОЗДІЛ 1
ДОСЛІДЖЕННЯ ТА АНАЛІЗ ІСНУЮЧИХ АНАЛОГІВ ВЕБ-ДОДАТКІВ ДЛЯ ПІДГОТОВКИ ДО СПІВБЕСІДИ

Підготовка до співбесід є критичним етапом у процесі пошуку роботи, особливо у сфері інформаційних технологій. Успішне проходження технічної співбесіди вимагає не тільки ґрунтовних знань з алгоритмів і структур даних, але й навичок їх практичного застосування. Сучасні веб-додатки пропонують широкий спектр інструментів для підготовки, включаючи інтерактивні задачі, мок-інтерв'ю та курси. У цьому розділі розглянуто найпопулярніші веб-додатки, що спрямовані на підготовку до технічних співбесід, оцінюючи їх функціональність, переваги та недоліки. Це допоможе визначити найбільш ефективні платформи для різних аспектів підготовки, забезпечуючи всебічний підхід до підготовки кандидатів.

1.1 W3Schools для веб-розробників

W3Schools – це сайт для веб розробників, де представлені навчальні посібники і довідкові матеріали по таким мовам веб-розробки та програмування, як HTML, CSS, JavaScript, PHP, SQL, MySQL, Python, Java, Kotlin, Go, C, C++, C#, TypeScript, ASP, Node.js, Django, W3.CSS, Bootstrap, React, jQuery, Vue, AngularJS, JSON, Ajax, AppMl, Sass та інше, які охоплюють більшість аспектів веб-розробки та програмування.

Розглядаючи W3Schools Online Web Tutorials у контексті підготовки до технічних співбесід, важливо звернути увагу на доступність викладеного матеріалу, якість навчального контенту, можливість практичного застосування і перевірку знань, а також на наявність активної спільноти для отримання підтримки та допомоги.

W3Schools славиться своєю широкою навчальною базою, що охоплює різноманітні технології веб-розробки від HTML та CSS до JavaScript, Python,

PHP та багатьох інших. Це робить його привабливим варіантом для тих, хто шукає універсальний ресурс для підготовки до співбесіди в різних областях програмування.

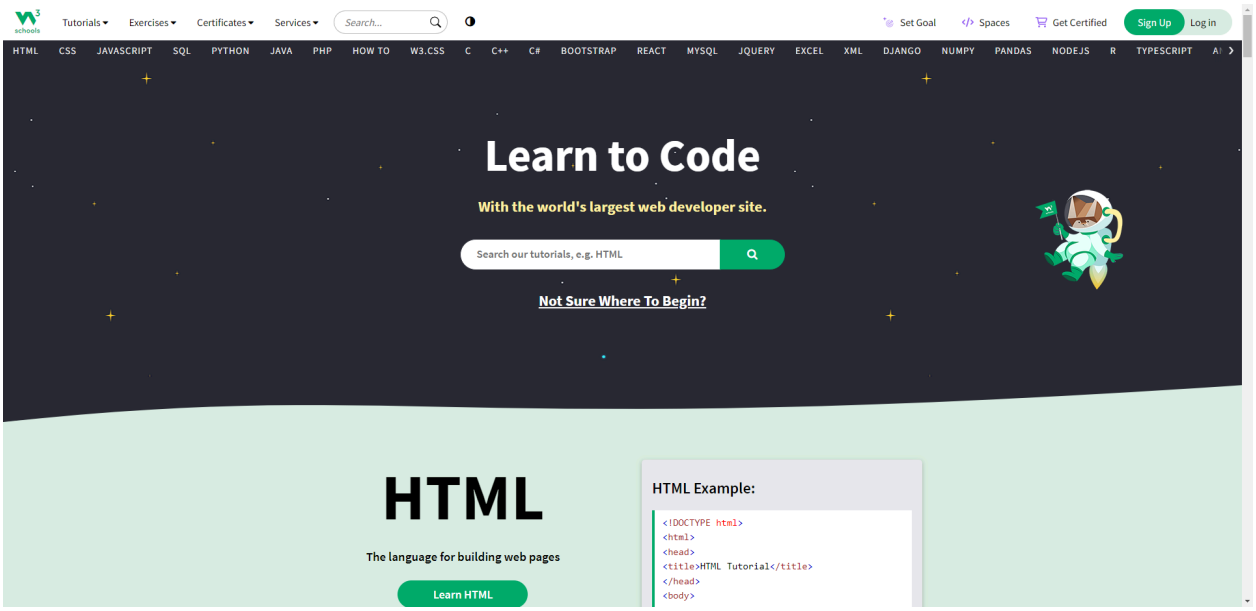

Рисунок 1.1 – Початкова сторінка веб-ресурсу W3Schools.

Якість навчального матеріалу на W3Schools висока, немає зайвої інформації. Матеріал добре структурований та легко зрозумілий, інтерфейс платформи дає можливість швидкої та зручної навігації для пошуку потрібного матеріалу або теми. Є багато прикладів використання, що допомагають уявити та закріпити засвоєні знання, а також є можливість негайної практичної реалізації отриманої інформації завдяки вбудованому онлайн-редактору та розділу "Try it Yourself", де користувачі можуть експериментувати з кодом та переглядати результати в реальному часі. Це сприяє кращому засвоєнню матеріалу та підготовці до практичних завдань під час співбесіди.

Але платформа не надає можливості виконувати більш складні завдання, та немає автоматичної перевірки цих завдань.

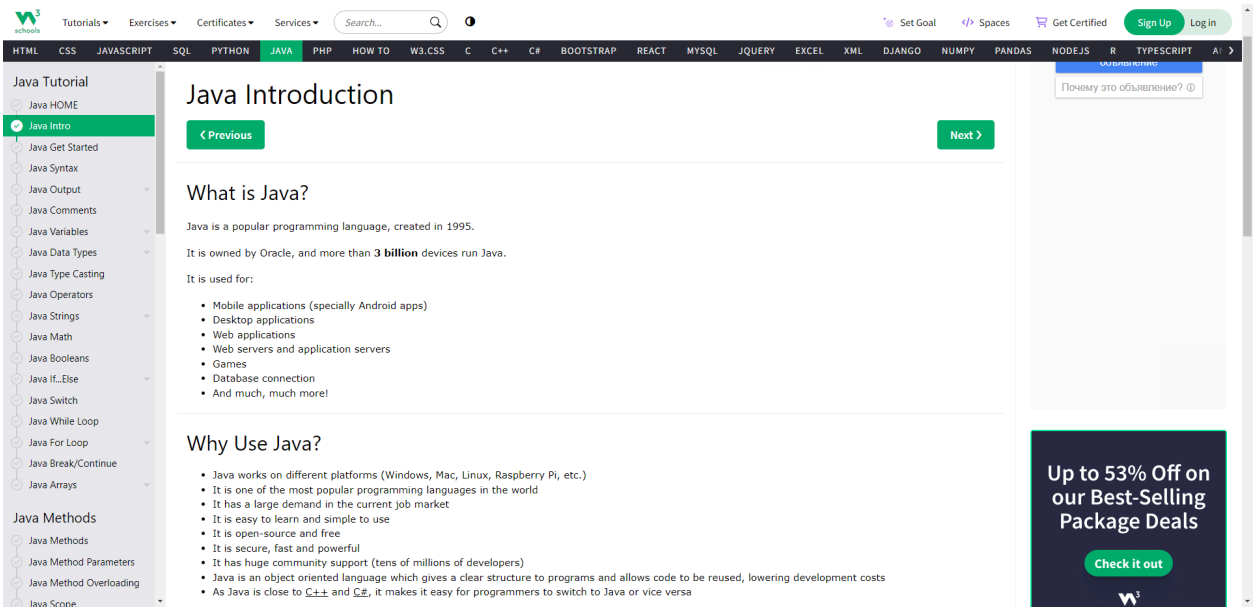

Рисунок 1.2 – Приклад наведеного матеріалу у W3Schools.

Активна спільнота користувачів на W3Schools надає можливість отримати підтримку, поради та допомогу у вирішенні будь-яких питань чи проблем. Форуми створюють сприятливе середовище для обміну досвідом та взаємної допомоги, що є великою перевагою для тих, хто вивчає веб-розробку.

Підводячи підсумки, перейдемо до критеріїв. W3Schools

Функціональність:

- Платформа має гарно структурований матеріал зрозумілий для новачків.
- Великий вибір мов програмування.
- Можливість використовувати набуті знання в тестах або “Try it yourself”.

Переваги:

- Велика кількість задач для початківців.
- Розділи за темами для цільової підготовки.
- Велика кількість теоретичного матеріалу для різних мов програмування.

- Інтерактивні інструменти для написання та перевірки коду.

Недоліки:

- Мала кількість практичних завдань високої або середньої складності..
- Відсутність персоналізованих рекомендацій.
- Різнонаправлений контент.

1.2 Ресурс для розробників LeetCode

LeetCode – сайт для розробників, де представлені практичні завдання і довідкові матеріали з різних існуючих мов програмування, що охоплюють більшість аспектів розробки та програмування.

Розглядаючи LeetCode у контексті підготовки до технічних співбесід, відзначається його спеціалізація на алгоритмічному програмуванні та практичному вирішенні завдань з алгоритмів та структур даних. Цей ресурс є невід'ємною частиною підготовки кандидатів до співбесід у сфері програмування, де алгоритмічні навички є ключовим фактором успіху з точки зору LeetCode.

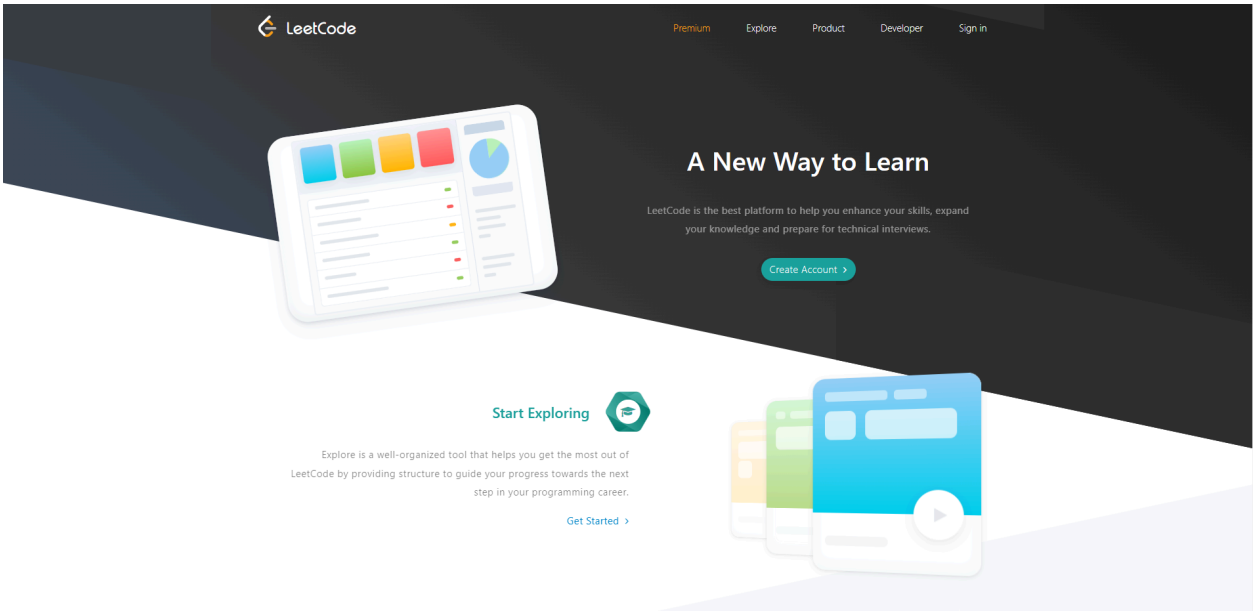

Рисунок 1.3 – Початкова сторінка ресурсу LeetCode.

LeetCode пропонує різноманітні завдання з різних тем, що дозволяє користувачам практикуватися на різних рівнях складності та вирішувати проблеми різного характеру, є можливість швидко відсортувати задачі за складністю, задачі які використовують на співбесідах та задачі з різних мов програмування та на різні теми. Велика кількість завдань дозволяє збагатити навички у вирішенні різноманітних завдань, від простих до дуже складних, але зазвичай завжди знаходяться більш складні.

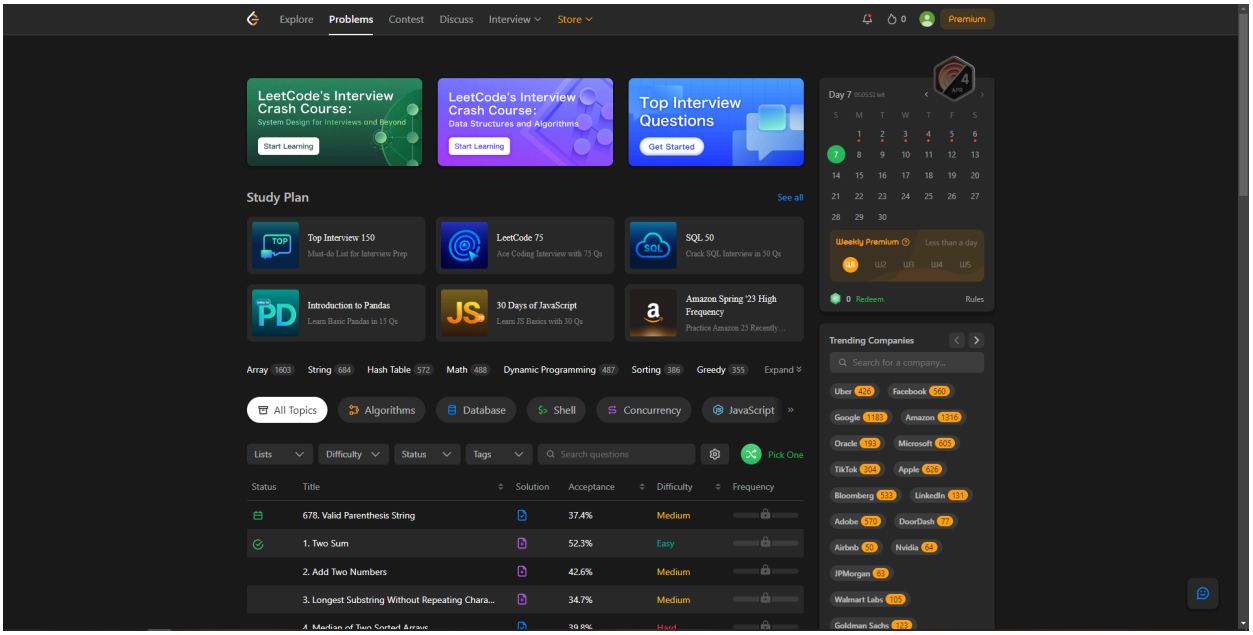

Рисунок 1.4 – Приклад наведеного матеріалу у LeetCode.

Що стосується якості матеріалу, LeetCode славиться своєю професійною платформою, платформа допомагає відслідковувати прогрес маючи календар. LeetCode пропонує як завдання, так і рішення високого рівня. Інтерфейс LeetCode дозволяє користувачам зручно та ефективно працювати над завданнями прямо у вбудованому онлайн-редакторі, перевіряти завдання на якість написання, витрати ресурсів системи, а також швидкодію, що значно полегшує процес вирішення задач.

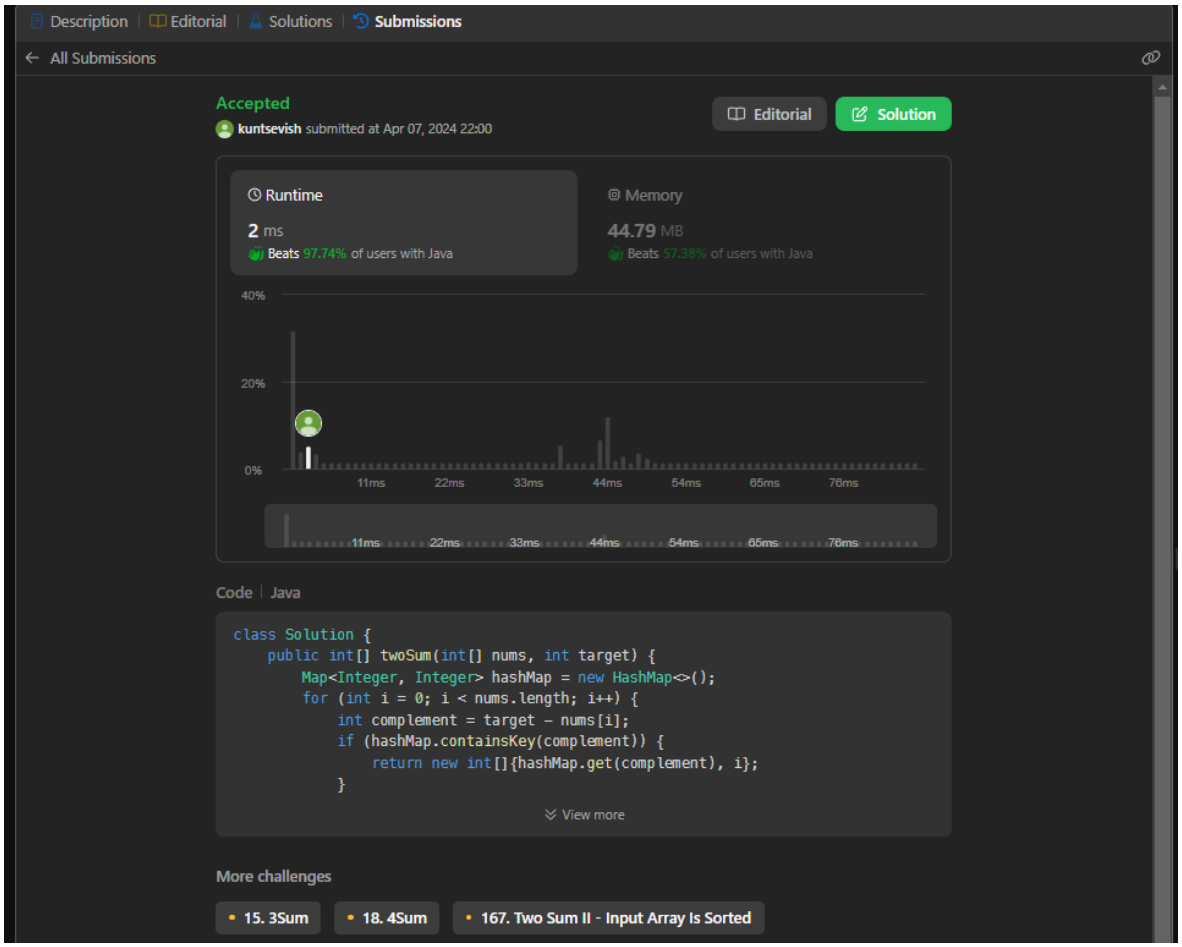

Рисунок 1.5 – Перевірки завдань у LeetCode.

Проте, варто відзначити, що LeetCode може бути викликом для початківців, оскільки деякі завдання можуть бути дуже складними. Також може відсутність додаткових інструкцій, що може ускладнити розуміння завдання. Іноді не зрозумілий інтерфейс та сортування завдань можуть стати перепоною.

Однак, завдяки розділу для обговорення рішень, користувачі мають можливість обмінюватися досвідом та отримувати зворотний зв'язок від інших учасників спільноти, що може значно полегшити процес вивчення.

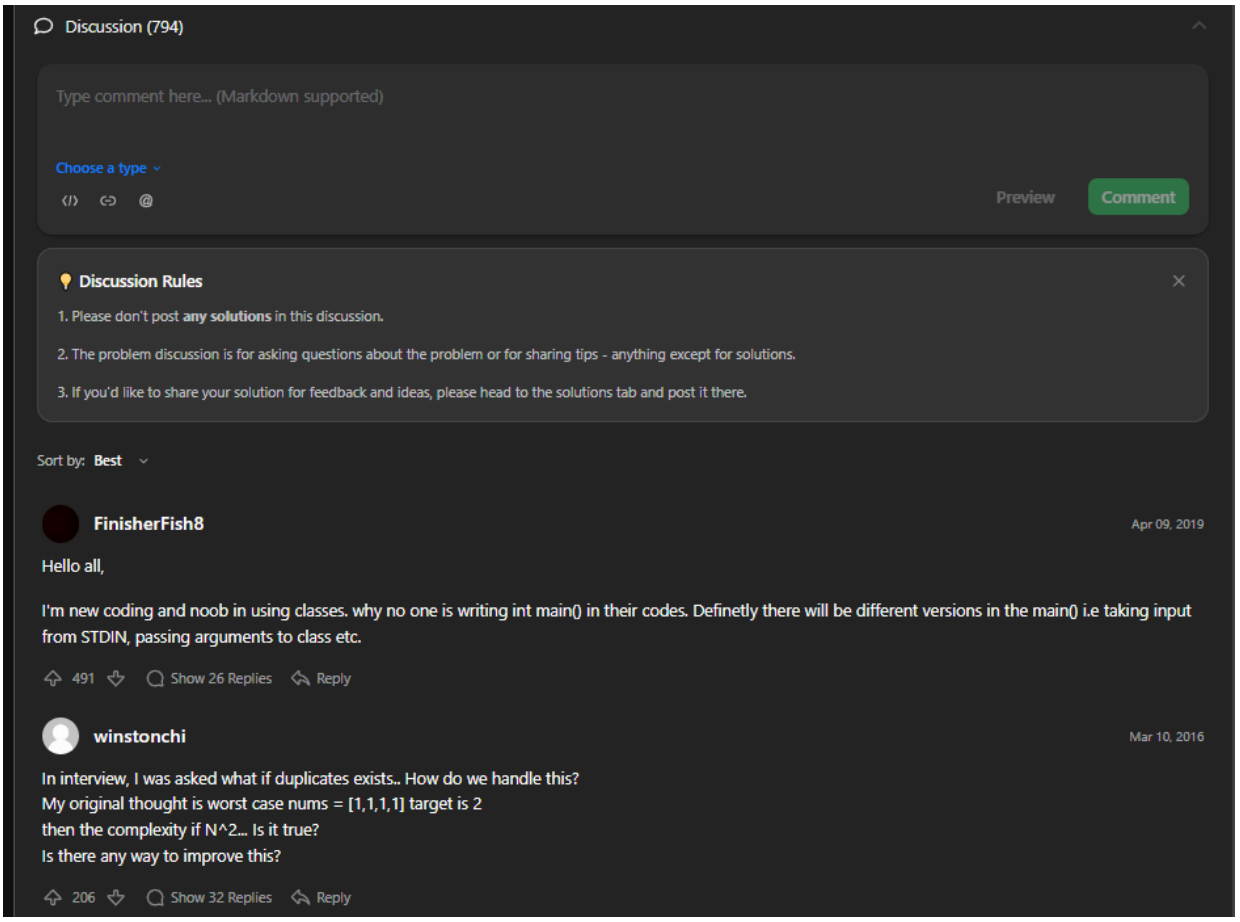

Рисунок 1.6 – Форум обговорень та зворотного зв’язку.

Підводячи підсумки, перейдемо до критеріїв. LeetCode

Функціональність:

- Платформа для вирішення задач з алгоритмів та структур даних.
- Відкритий доступ до більш ніж 2000 задач.
- Можливість участі у змаганнях та челенджах.

Переваги:

- Велика кількість задач різного рівня складності.
- Розділи за темами для цільової підготовки.
- Інтерактивні інструменти для написання та перевірки коду.

Недоліки:

- Більшість розширених функцій доступні лише за платною підпискою.
- Відсутність персоналізованих рекомендацій.
- Відсутність теорії для розширення знань з тем практичного виконання.

1.3 Онлайн платформа для навчання JavaRush

JavaRush – онлайн-платформа для навчання Java та веб-розробки. Варто відзначити її важливість для початківців та тих, хто бажає поглибити свої знання в цій конкретній області програмування. JavaRush пропонує унікальний підхід до навчання, який дозволяє користувачам вивчати мову програмування Java через структуровані курси, практичні завдання та ігри, що допомагають у закріпленні отриманих знань.

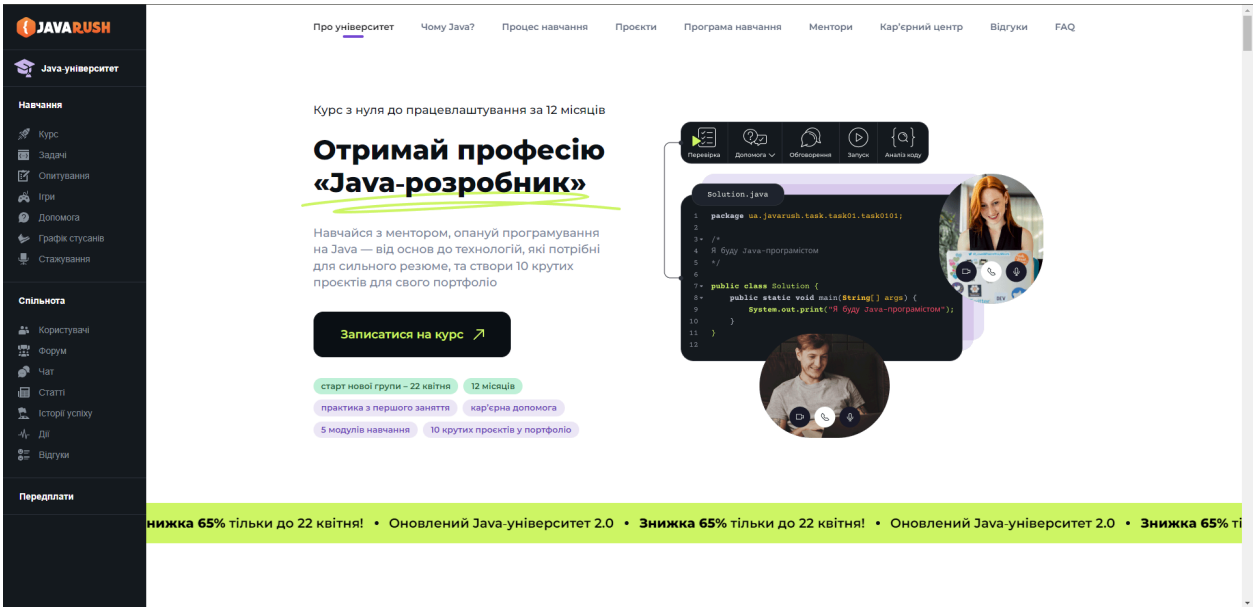

Рисунок 1.7 – Початкова сторінка ресурсу JavaRush .

Однією з ключових переваг JavaRush є його фокус на одній конкретній мові програмування – Java. Це дозволяє користувачам глибше заглибитися в цю мову та стати експертом цій області. Курси на платформі структуровані таким чином, що дозволяють користувачам поступово розвивати свої навички, починаючи з основ та закінчуючи більш складними темами. Користувач має можливість вивчати додатковий матеріал на який посилається веб-платформа.

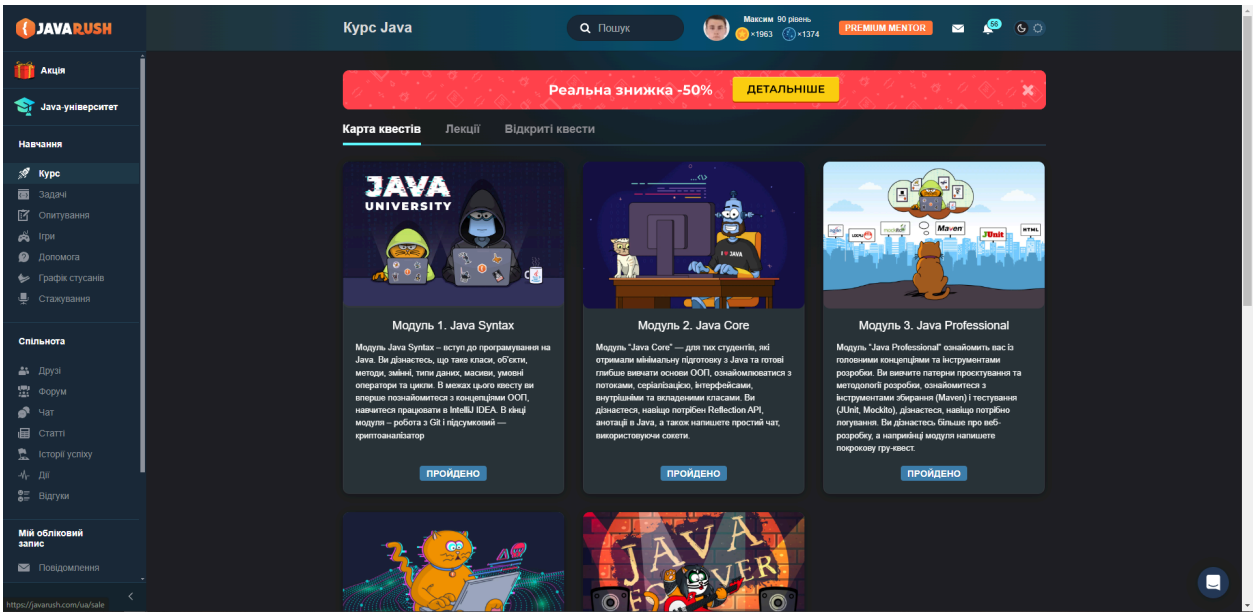

Рисунок 1.8 – Приклад наведеного матеріалу на JavaRush.

Також важливою перевагою є можливість виконувати практичні завдання безпосередньо на платформі. Інтерактивні завдання та проєкти дають змогу користувачам негайно випробувати свої знання та отримати зворотний зв'язок щодо їхньої роботи. Це сприяє кращому засвоєнню матеріалу та підготовці до реальних випробувань чи завдань, які можуть виникнути під час співбесіди.

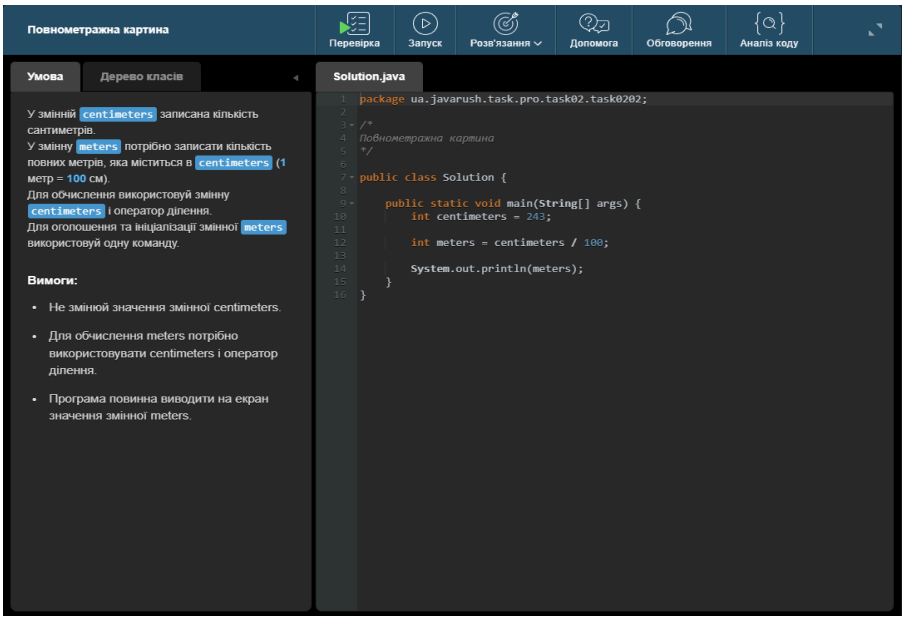

Рисунок 1.9 – Приклад завдань на платформі JavaRush.

Проте, варто відзначити, що JavaRush може бути менш відповідним для тих, хто шукає розширення своїх знань в інших областях веб-розробки, окрім Java. Обмежений фокус платформи на одній мові програмування може бути обмеженням для тих, хто бажає отримати більший обсяг знань у різних технологіях.

Що до спільноти, то можна сказати, що вона дуже активна, дуже багато статей, форумів, та можливість при складностях з виконанням завдання звернутись в чат який знаходиться під завданням. Також розробники розробили додаток або як ще звикли називати плагіном, який допомагає модифікувати ваше середовище розробки Intellij IDEA.

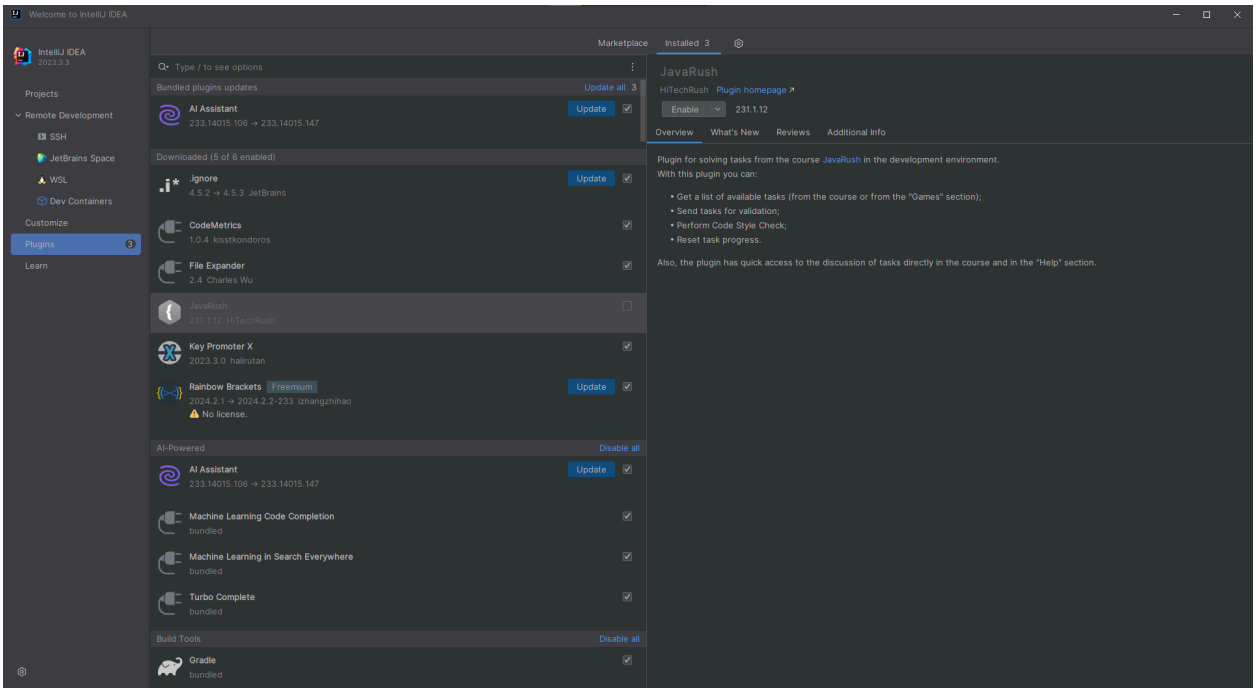

Рисунок 1.10 – Плагін платформи JavaRush.

Незважаючи на це, JavaRush може бути корисним інструментом для підготовки до технічних співбесід, особливо для тих, хто зацікавлений у вивченні Java та розвитку навичок веб-розробки на цій мові. Його структуровані курси, практичні завдання та можливість отримати зворотний

зв'язок дозволяють користувачам ефективно вивчати матеріал та підготуватися до викликів, що можуть зустрітися в реальному житті.

Підводячи підсумки, перейдемо до критеріїв. JavaRush

Функціональність:

- Платформа, що спеціалізується на навчанні Java.
- Інтерактивні курси, що охоплюють всі аспекти програмування на Java.
- Практичні завдання для закріплення теоретичних знань.
- Проекти для створення реальних додатків.

Переваги:

- Структурований курс, що підходить для новачків і просунутих користувачів.
- Велика кількість практичних завдань, що допомагають розвивати навички кодування.
- Зворотній зв'язок і підтримка від менторів та спільноти.
- Актуальний контент, що відповідає вимогам ринку праці.

Недоліки:

- Обмежений фокус на одній мові програмування (Java), що може не задовольняти потреби тих, хто хоче освоїти інші технології.
- Платна підписка для доступу до повного контенту і деяких розширених функцій.
- Відсутність спеціалізованих інструментів для підготовки до співбесід з інших технологій, окрім Java.

1.4 Сайт для розробників Codewars

Codewars - це сайт для розробників, на якому представлені складні практичні завдання і довідкові матеріали з різних існуючих мов програмування, які охоплюють більшість аспектів розробки та програмування.

У контексті підготовки до технічних співбесід, варто відзначити, що ця платформа пропонує унікальний підхід до навчання програмування. Її

основний принцип полягає в тому, щоб розв'язувати складні завдання, які представлені у вигляді "кат". Ката вимагає від користувача розробки програмного коду для вирішення конкретного алгоритмічного та програмного завдання. Такий підхід дозволяє не лише засвоювати нові знання, а й вдосконалювати свої навички, розвивати креативність та аналітичне мислення.

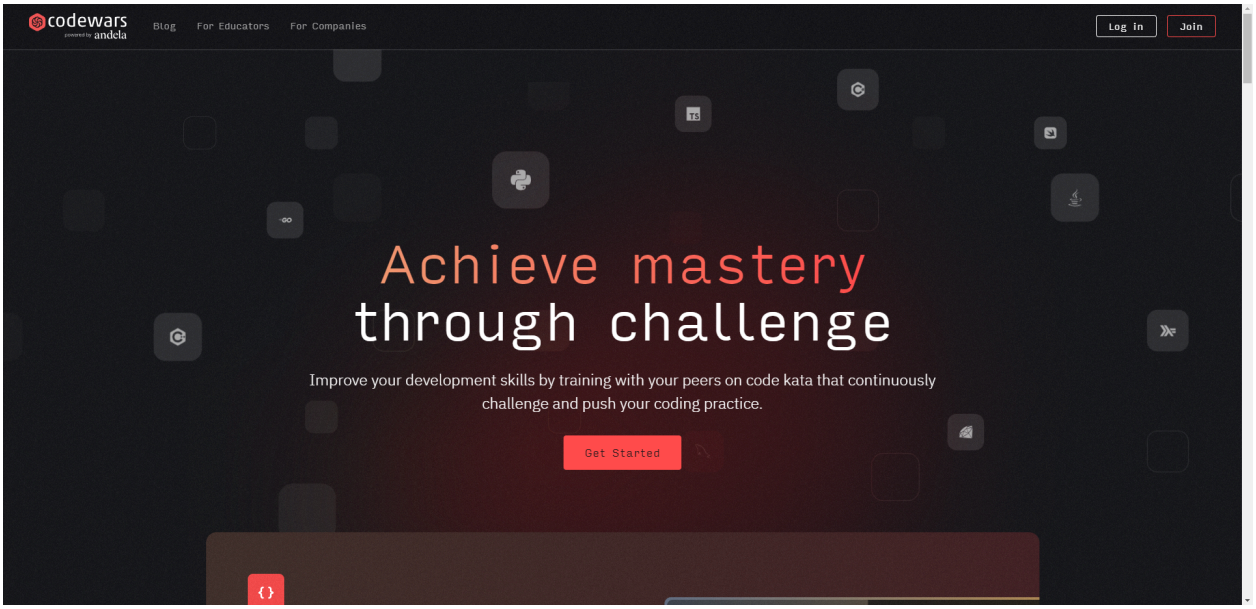

Рисунок 1.11 – Початкова сторінка, яка зустрічає користувача в Codewars.

Однією з переваг Codewars є його різноманітність. Платформа пропонує завдання для різних мов програмування, таких як JavaScript, Python, Java, C#, Ruby та інші. Це дозволяє користувачам вибирати та розвиватися у тій області, яка їх цікавить найбільше. Крім того, завдання представлені на різні рівні складності, від початкового до дуже складного, що дає можливість користувачам вибирати та підбирати завдання відповідно до їхніх потреб та рівня навичок.

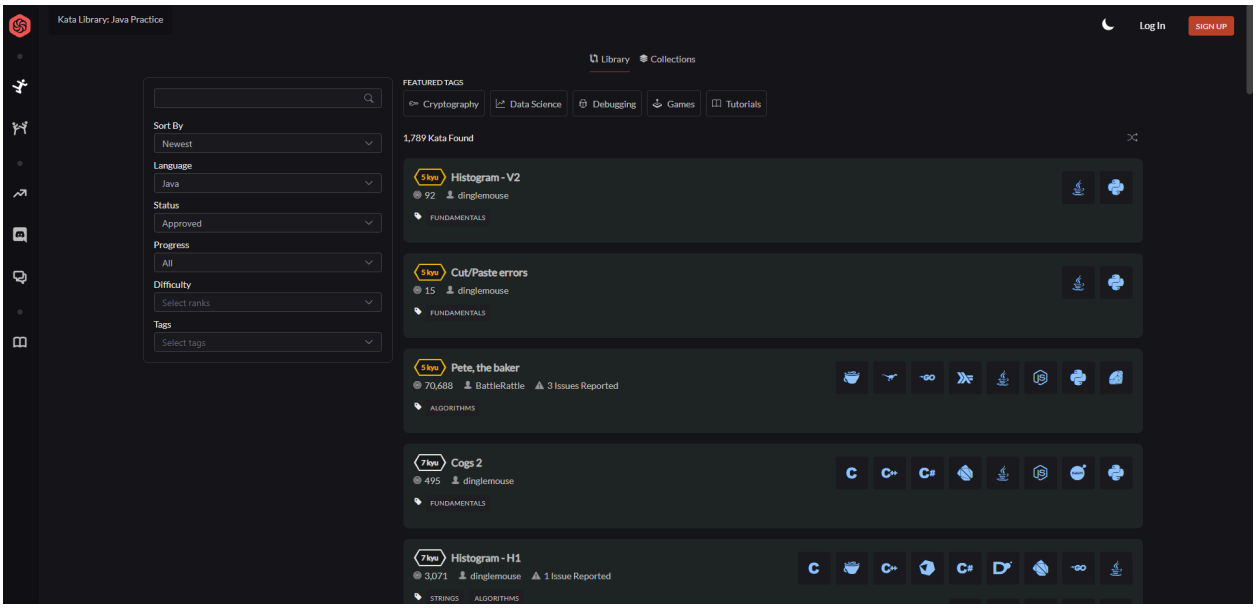

Рисунок 1.12 – Зовнішній вигляд “кат” та мов на яких можна вирішувати проблему.

Кожне завдання супроводжується великою кількістю тестів, що дозволяє користувачам перевірити правильність свого розв'язку та його ефективність. Крім того, на платформі є можливість переглядати та обговорювати різні способи розв'язання завдань, що дозволяє вчитися на прикладах та збагачувати свій досвід.

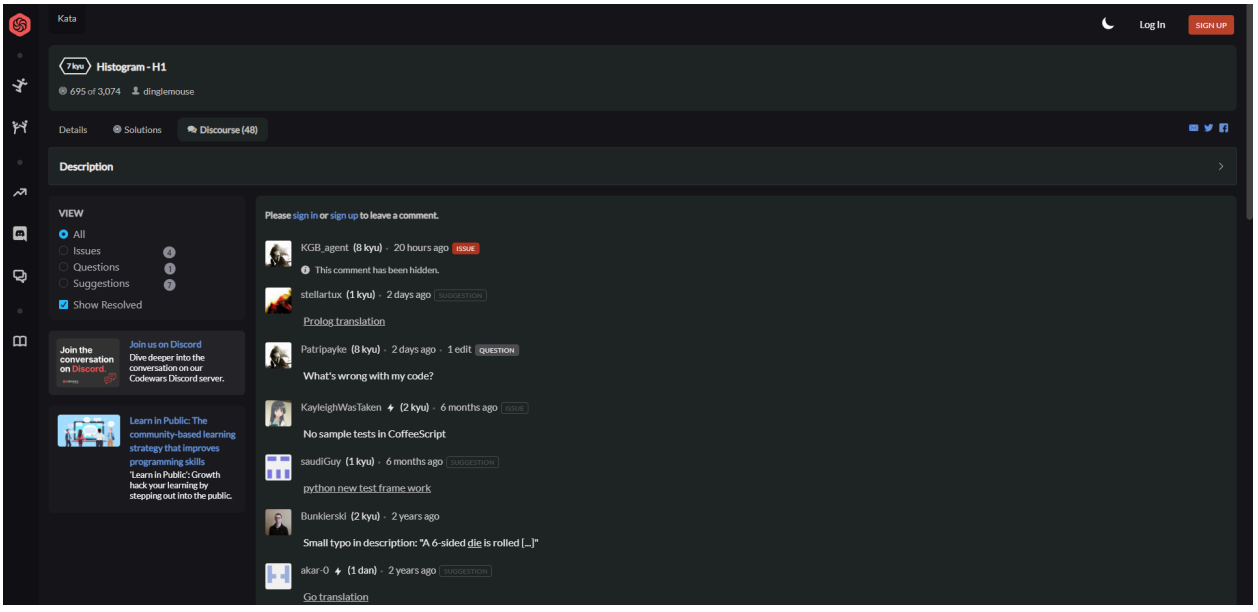

Рисунок 1.13 – Спільнота розробників для обговорення тем завдань.

Загалом, Codewars - це відмінний ресурс для тих, хто шукає викликів у навчанні програмування та бажає поглибити свої знання та навички шляхом практичного застосування. Завдяки різноманітним завданням, які платформа пропонує, користувачі мають можливість розвивати свої вміння у будь-якій області програмування та вдосконалювати свої навички вирішення алгоритмічних завдань.

Підводячи підсумки, перейдемо до критеріїв. Codewars

Функціональність:

- Платформа для вирішення програмних задач різного рівня складності.
- Підтримка багатьох мов програмування, включаючи JavaScript, Python, Ruby, Java, C++, і багато інших.
- Задачі (ката) створені та оцінені спільнотою користувачів.
- Можливість створення власних задач і участь у їх обговоренні.

Переваги:

- Велика різноманітність задач, що охоплюють різні аспекти програмування.
- Активна спільнота, яка допомагає у вирішенні задач та надає зворотній зв'язок.
- Гейміфікований підхід до навчання, що включає ранги та досягнення, які мотивують користувачів.
- Можливість порівняння своїх рішень з іншими, що сприяє покращенню навичок.

Недоліки:

- Відсутність структурованих курсів або планів навчання.
- Відсутність теоретичних матеріалів для новачків.
- Може бути складно для новачків, оскільки деякі задачі не мають достатньо пояснень.
- Фокус на задачах може не охоплювати всі аспекти підготовки до співбесід, такі як поведінкові питання або системний дизайн.

Висновок до розділу 1

У даному розділі було проведено аналіз існуючих аналогів веб-додатків, які можуть допомогти в підготовці до співбесід, розглядаючи їх функціональність, переваги та недоліки. Це допоможе визначити найбільш ефективні платформи для різних аспектів підготовки, забезпечуючи всебічний підхід до підготовки кандидатів.

W3Schools може бути менш відповідним для досвідчених розробників або тих, хто шукає більш глибокого розуміння складних концепцій. Навчальний контент на цьому ресурсі часто спрощений, що може не задовольняти потреби більш просунутих користувачів.

LeetCode може бути важливим інструментом у підготовці до технічних співбесід, особливо для тих, хто прагне покращити свої алгоритмічні навички та підготуватися до складних технічних завдань. Його висока якість та різноманітність завдань робить його привабливим варіантом для будь-якого програміста, який прагне розвивати свої навички у цій області.

JavaRush є гарним вибором для цілеспрямованої підготовки з Java, але для всебічної підготовки до технічних співбесід може знадобитися додаткове використання інших платформ.

Codewars є відмінною платформою для практики вирішення програмних задач і покращення навичок програмування в різних мовах. Активна спільнота та гейміфікований підхід роблять процес навчання цікавим і мотивуючим. Однак відсутність структурованих навчальних матеріалів та можливих труднощів для новачків може вимагати додаткового використання інших ресурсів для комплексної підготовки до співбесід.

Кожен з розглянутих веб-додатків має свої сильні та слабкі сторони. Вибір оптимального інструменту залежить від індивідуальних потреб, рівня підготовки та наявного бюджету. Для комплексної підготовки до співбесід доцільно комбінувати декілька платформ, що дозволить отримати як теоретичні знання, так і практичний досвід.

РОЗДІЛ 2
ІНСТРУМЕНТИ ТА ТЕХНОЛОГІЇ ДЛЯ СТВОРЕННЯ ВЕБ-ПЛАТФОРМИ ДЛЯ ПІДГОТОВКИ ДО СПІВБЕСІДИ

Створення ефективної платформи для підготовки до співбесід вимагає ретельного вибору програмних інструментів та технологій, що забезпечують функціональність, зручність використання та привабливий дизайн. У цьому розділі розглянуто основні технології, які можуть бути використані для реалізації функціоналу та дизайну платформи, оцінюючи їх переваги та недоліки.

2.1 Програмні інструменти та технології

2.1.1 Фронтенд технології

Angular – це комплексний фреймворк для розробки веб-додатків, створений і підтримуваний компінією Google. Він використовується для створення односторінкових додатків (SPA – Single Page Applications), що дозволяють користувачам взаємодіяти з веб-сторінкою без необхідності її повного перезавантаження.

Основні характеристики:

- Компонентний підхід: базується на компонентах, які є будівельними блоками інтерфейсу користувача. Кожен компонент складається з шаблону HTML, CSS та логіки на TypeScript.
- TypeScript: статично типізована надбудова до JavaScript.
- Декларативні шаблони: використання розширення HTML - шаблонів, для створення динамічних інтерфейсів користувача.
- Маршрутизація: Вбудована система маршрутизації дозволяє створювати та керувати навігацією односторінкових додатків.

HTML – стандартна мова розмітки для створення веб-сторінок та веб-додатків.

Основні характеристики:

- Структура сторінки: HTML визначає структуру сторінки використовуючи різні “теги”.
- Гіпертекстові посилання: “Теги” <a> дозволяють створювати посилання між різними сторінками або ресурсами в межах сторінки.
- Мультимедіа: підтримує вбудовування зображень, відео та аудіо.

CSS – мова стилів, яка використовується для опису зовнішнього вигляду документів, написаних мовою розмітки.

Основні характеристики:

- Стилізація: дозволяє змінювати кольори, шрифта та інше.
- Розміщення: управляє розташуванням елементів на сторінці.
- Каскадність: можливість стилів успадковуватись до різних елементів.
- Анімація та переходи: підтримує створення анімацій і плавних переходів між різними елементами.

2.1.2 Бекенд технології

Java Spring – це потужний фреймворк для розробки програмного забезпечення на мові Java.

Основні характеристики:

- Інверсія керування (IoC): це ключовий принцип Spring, що полягає в тому. що об’єкти не створюються напряму кодом, але замість цього контейнер Spring керує створенням об’єктів та їх залежностями.
- Введення залежностей (Dependency Injection - DI): надає механізм введення залежностей, що полегшує керування залежностями між об’єктами та сприяє створенню слабкосполучених компонентів.
- Модульність: Spring складається з численних модулів, які можна використовувати разом або окремо в залежності від потреб проєкту.

- Аспектно-орієнтоване програмування (Aspect-Oriented Programming - AOP): Spring підтримує AOP, що дозволяє виділяти крос-вирішуванні аспекти застосунку, такі як журналювання, безпека, транзакції, тощо.

MySQL – це потужна система управління базами даних (СУБД), яка використовується для зберігання та управління релігійними даними.

Основні характеристики:

- Реляційна структура даних: MySQL використовує модель реляційної бази даних, що дозволяє зберігати дані у вигляді таблиць, які мають відношення одна до одної або інше через ключі.
- Мова запитів SQL: підтримка стандартів мов запиту SQL (Structured Query Language), що дозволяє виконувати різноманітні операції з даними.
- Широкі можливості індексації: MySQL дозволяє створювати індекси на таблицях для покращення швидкої операції пошуку та фільтрації даних.

Hibernate – це потужний фреймворк для роботи з базами даних у Java програмах.

Основні характеристики:

- Об’єктно-реляційне відображення (ORM): одна з ключових функцій Hibernate - це можливість відображати об’єкти Java безпосередньо на таблиці бази даних і навпаки, що полегшує роботу з даними та зменшує необхідні
- Автоматична генерація SQL-запитів: Hibernate автоматично створює SQL-запити на основі об'єктів Java та їх відношень, що полегшує роботу з базою даних.
- Кешування даних: Hibernate надає можливості кешування, що дозволяє зберігати результати запитів у пам'яті для подальшого використання, що підвищує швидкодію додатка.

- Підтримка транзакцій: Hibernate інтегрується з різними механізмами управління транзакціями (наприклад, JTA або розширення Hibernate для управління транзакціями), що дозволяє забезпечувати консистентність даних у великих системах.

Lombok – це бібліотека для Java, яка дозволяє покращити продуктивність та зробити код більш зрозумілим та компактним.

Основні характеристики:

- Автоматичне генерування методів: Lombok дозволяє автоматично генерувати рутинний код, такий як гетери, сетери, методи toString(), equals() та hashCode(), що значно зменшує кількість написаного шаблонного коду.[3][17]
- Анотації для зменшення білого коду: Lombok надає анотації, які дозволяють замінити стандартний шаблоний код скороченими конструкціями. Наприклад, анотація @Data автоматично генерує гетери, сетери, метод toString(), equals() та hashCode() для класу. [3][17]
- Менша залежність від IDE: Використання Lombok дозволяє писати менше шаблонного коду, що допомагає покращити читабельність та зрозумілість коду, а також робить роботу з кодом менш залежною від конкретного інтегрованого середовища розробки (IDE).
- Підтримка для різних сценаріїв використання: Lombok може бути використаний для практично будь-якого типу проєкту, від простих програм до великих корпоративних додатків.

2.1.3 Середовища розробки

Intellij IDEA Ultimate – це інтегроване середовище розробки (IDE), створене компанією JetBrains, яке головним чином призначене для розробки на мові програмування Java, але також підтримує велику кількість інших мов та технологій.

Основні характеристики:

- Інтелектуальне редагування коду: доповнення коду або аналіз у реальному часі допомагають пришвидшити кодування.
- Підтримка різних мов та фреймворків: допомагає розширити області розробки програмного забезпечення або інших видів розробки.
- Інтеграція з системами контролю версій: допомагає зберігати прогресс розробки, та в будь-який момент часу повернутись до потрібного стану розробки.
- Рефакторинг коду: редагування коду, для покращення розуміння іншими програмістами.

Docker – це платформа для розробки, доставки та запуску додатків у контейнерах. Контейнери дозволяють ізолювати додатки та їх залежності, забезпечуючи повторюваність та переносимість програмного забезпечення між різними середовищами.

Основні переваги:

- Контейнеризація: контейнери - необхідні для запуску додатків налаштування (код, залежності, бібліотеки), образи Docker - файли, що містять інструкції для створення контейнерів.
- Ізоляція: контейнери ізолюють додатки один від одного та від хост-системи забезпечуючи більше безпеку та надійність.
- Портативність: завдяки контейнерам, додатки можуть бути легко перенесені між різними середовищами.
- Масштабність: дозволяє швидко створювати та керувати великою кількість контейнерів.

2.2 Принцип роботи клієнт-серверних моделей веб-додатків

“Хороша архітектура – це скоріше виняток, ніж правило.” - JavaRush [11]

В архітектурі серверних програм існують кілька популярних підходів, які кожен має свої переваги та недоліки. Однак, доки не зануритися в реальну практику, складно повністю розуміти їхню складність та вибірковість. Із назви “Клієнт-сервер” стає зрозумілим, що в концепції беруть участь дві

сторони: клієнт і сервер. Клієнт – це користувач, а сервер - постачальник послуг. [11]

Спілкування клієнту і сервера відбувається за стандартними протоколами, такими як HTTP, FTP або низькорівневий, такими як TCP або UPD. Протокол зазвичай обирається під тип послуги, яку надають сервери. Важливий аспект взаємодії клієнт-сервер лежить в принципі того, що таку взаємодію починає клієнт: сервер лише відповідає клієнту і повідомляє про те, чи може він надати послугу, і якщо може, то на яких умовах. [11]

Ця концепція зародилась як перший крок у бік спрощення складної система. Та має такі сильні сторони:

- Спрощення логіки: сервер нічого не знає про клієнта і як він використовуватиме його дані надалі.
- Можуть бути “слабкі клієнти”: все ресурсомісткі завдання можна перенести на сервер.
- Незалежний розвиток коду клієнтів та коду сервера.
- Багато різних клієнтів: Tomcat, різні браузери.

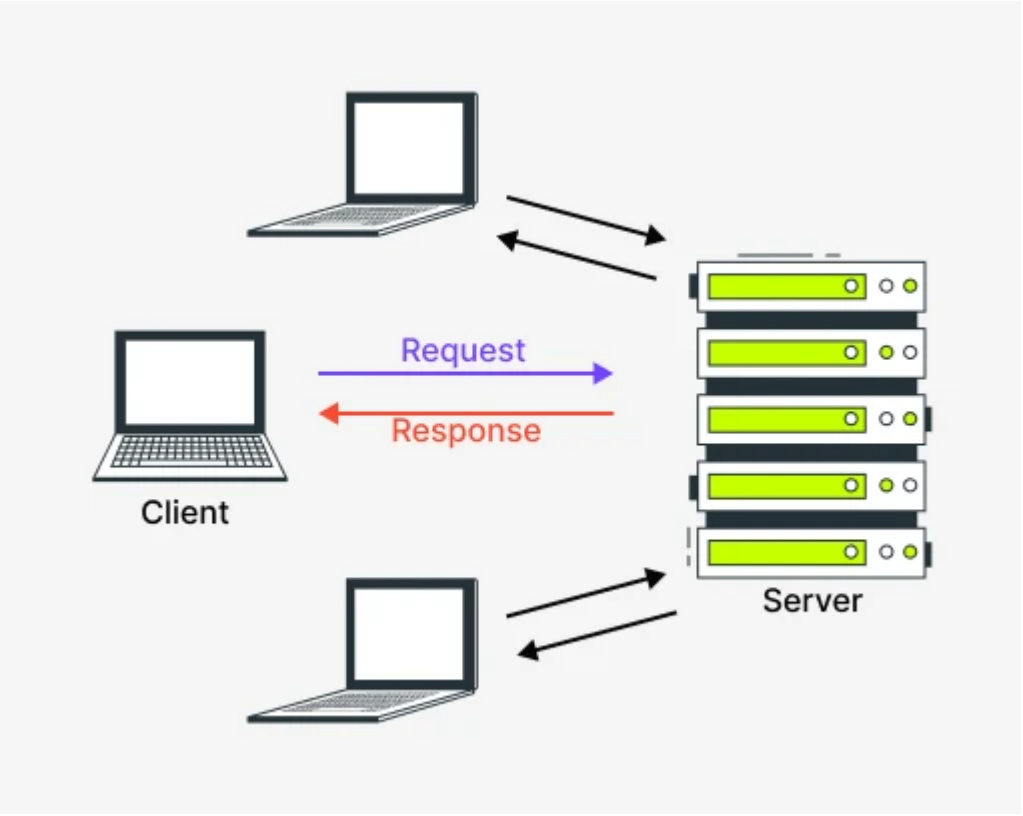

Рисунок 2.1 – Взаємодія клієнт-серверної моделі. [11]

Існує два види архітектури взаємодії клієнт-сервер:

1) Дворівнева архітектура.
2) Трирівнева архітектура.

Принцип роботи дворівневої архітектури взаємодії клієнт-сервер полягає в тому, що обробка запиту відбувається на одному сервері без звернення до інших серверів у процесі цієї обробки. [11]

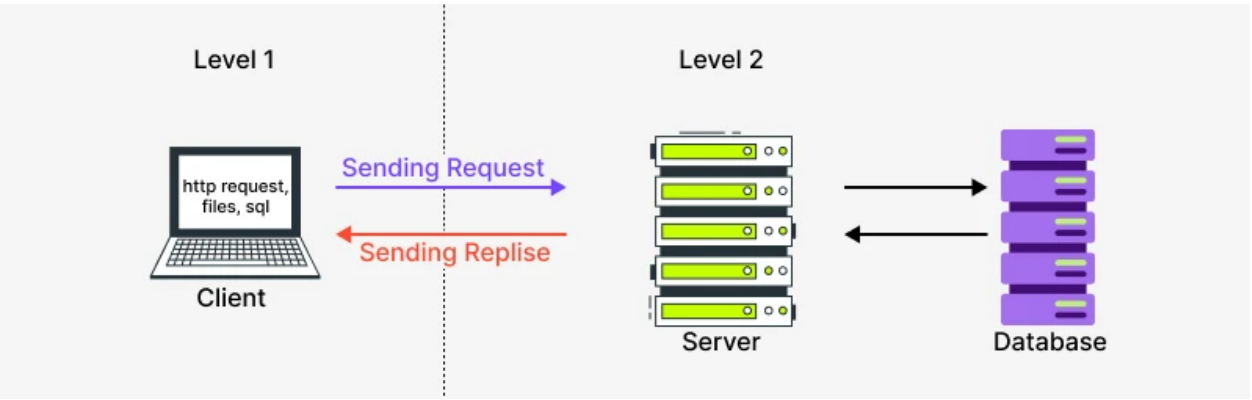

Рисунок 2.2 – Вигляд дворівневої архітектури “Клієнт-сервер” [11]

Трирівнева архітектура - це найпоширеніша архітектура взаємодії в інтернеті. Вона з’явилася коли серверну части дворівневої архітектури розділили на дві частини:

- Шар логіки.
- Шар даних.

Появлення шару даних спричинено з багатьох причин, але найголовніша з них - це приріст навантаження на сервер. Бази даних почали вимагати багато пам’яті та процесорного часу на обробку даних, а також бази даних стали більш розумними – у них виникла власна бізнес-логіка. Вони стали підтримувати процедури, що зберігаються, тригери, власні мови по типу PLSQL.

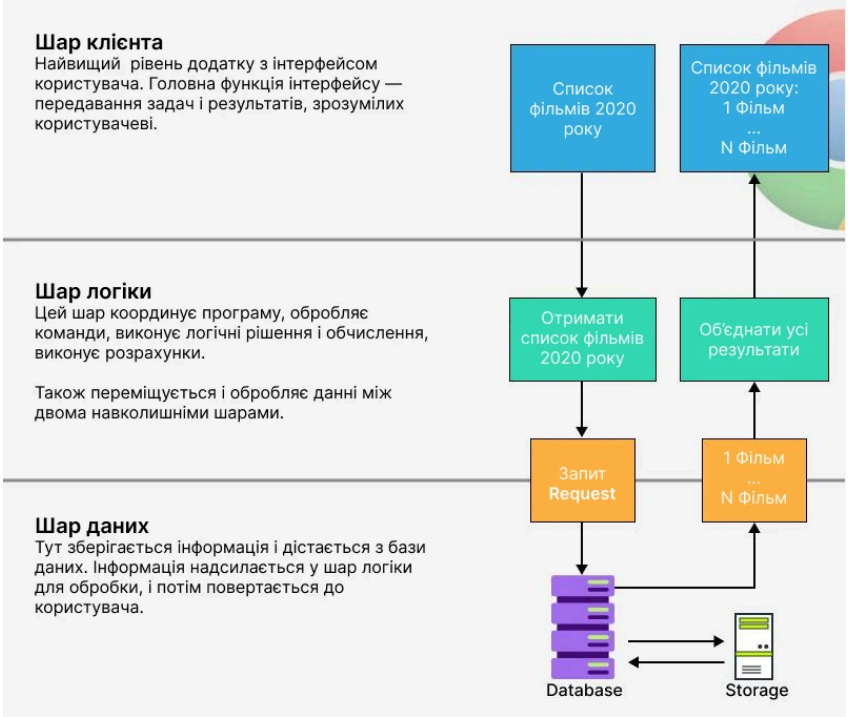

Рисунок 2.3 – Вигляд трирівневої архітектури “Клієнт-сервер” [11]

Також є й недоліки – це неоднозначність розташування логіки програми. Не завжди можна правильно визначити, в яке саме місце системи потрібно додавати нову частину бізнес-логіки (новий код).

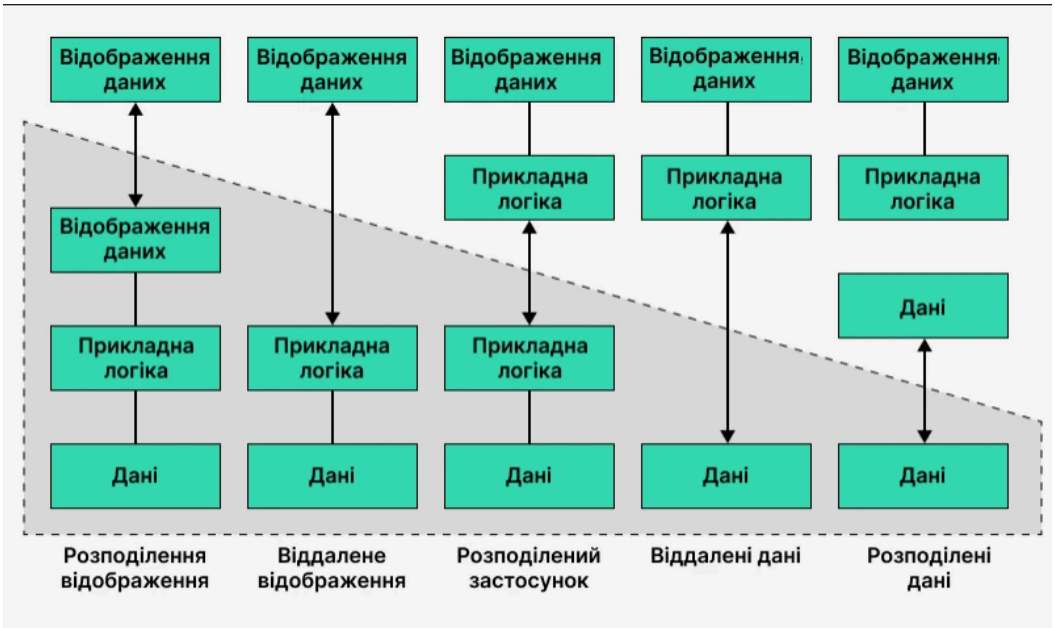

Рисунок 2.4 – Проблематика розташування бізнес-логіки трирівневої архітектури [11].

Сірим фоном залита серверна частина, білим – клієнтська. Хороший приклад останнього підходу (крайній правий варіант) – це сучасні мобільні програми. На стороні клієнта (на телефоні) міститься представлення (відображення), логіка та дані. І лише інколи ці дані синхронізуються із сервером.

Приклад крайнього лівого варіанту – це типовий PHP-сервер, у якого вся логіка знаходиться на сервері, і він віддає клієнту вже статичний HTML.

Завдяки комбінованим технологіям стало можливим створити ефективну, програмну модель веб-додатку для підготовки до співбесіди на посаду Junior Java Developer в IT компанії (рис. 2.5)

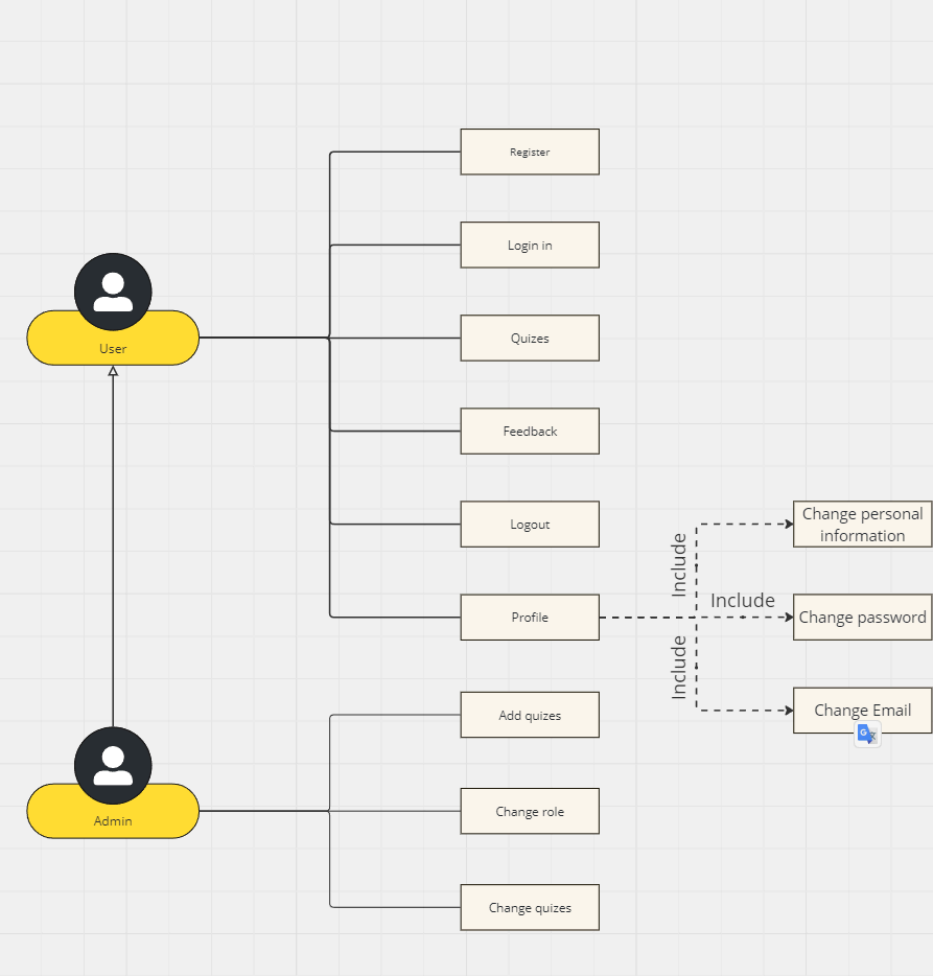

Рисунок 2.5 – Use case діаграма взаємодії користувачів з веб-додатком.

Висновок до розділу 2

Вибір інструментів та технологій для реалізації платформи залежить від багатьох факторів, включаючи вимоги проекту, технічні знання команди, бюджет та терміни. Для фронтенду доцільно використовувати сучасні фреймворки такі як Angular, які забезпечують високу продуктивність та зручність розробки. Для бекенду можна обрати Java Spring як шар логіки для обробки вхідних даних з фронтенду. Вибір бази даних залежить від потреб у зберіганні та обробці даних, де MySQL є надійним та продуктивним варіантом та чудовим “шаром даних” в трирівневій архітектурі. Також доцільно використовувати контейнеризація для гнучкого запуску на різних машинах для тестування роботи баз даних та самого додатку.

Комбіновані технології дозволять створити ефективну, продуктивну та привабливу платформу для підготовки до співбесід.

РОЗДІЛ 3.
ПРОГРАМНА МОДЕЛЬ ВЕБ-ДОДАТКУ ДЛЯ ПІДГОТОВКИ ДО СПІВБЕСІДИ НА ПОСАДУ JUNIOR JAVA DEVELOPER В IT КОМПАНІЇ.

Створення програмної моделі є багатоступеневим процесом, який розпочинається з налаштування середовища розробки. Це є ключовим етапом, оскільки правильне налаштування середовища може суттєво вплинути на продуктивність роботи та якість кінцевого продукту.

Насамперед, необхідно обрати відповідне середовище розробки (IDE) або текстовий редактор, що підтримує всі необхідні функції для розробки програмного забезпечення. У даному випадку вибір впав на Intellij IDEA

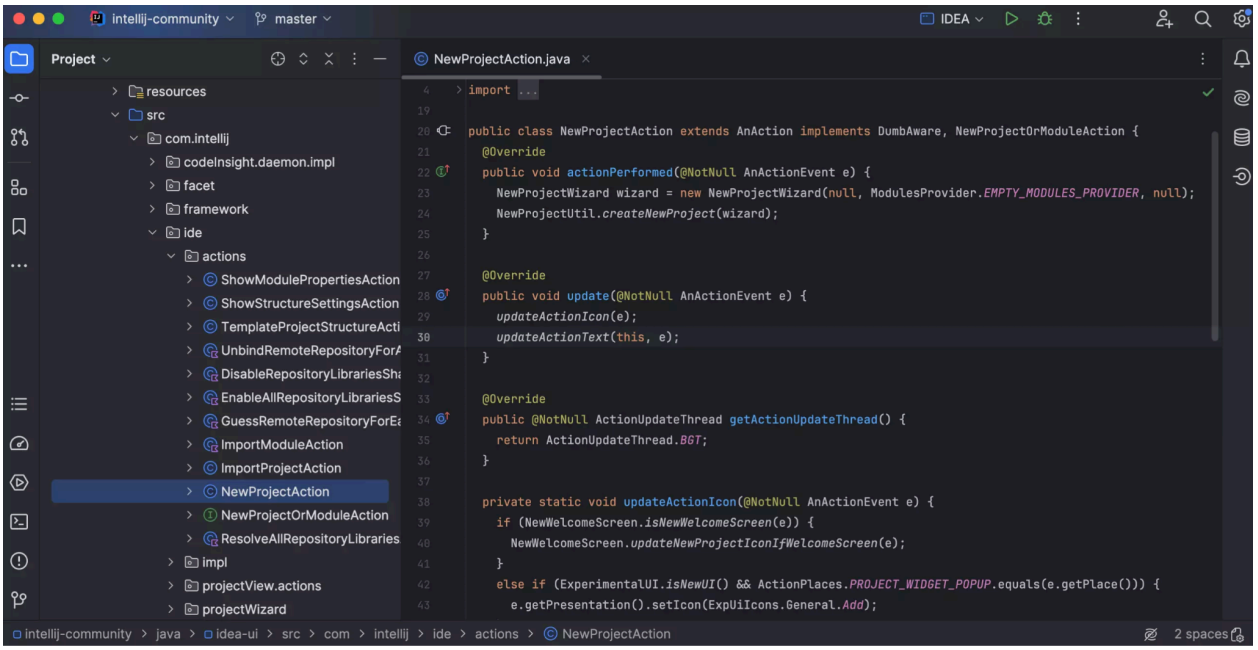

Рисунок 3.1 – Зовнішній вигляд Intellij IDEA.

Наступним кроком є підключення необхідних бібліотек, фреймворків та залежностей. Це може включати як стандартні бібліотеки мови програмування, так і зовнішні пакети, що додають додаткові функціональні можливості. Для спрощення підключення зазвичай використовують Maven.

Maven є "фреймворком" для керування складанням програми, який стандартизує опис проєкту, сценарії складання проєктів та залежності між бібліотеками (рис. 3.2).

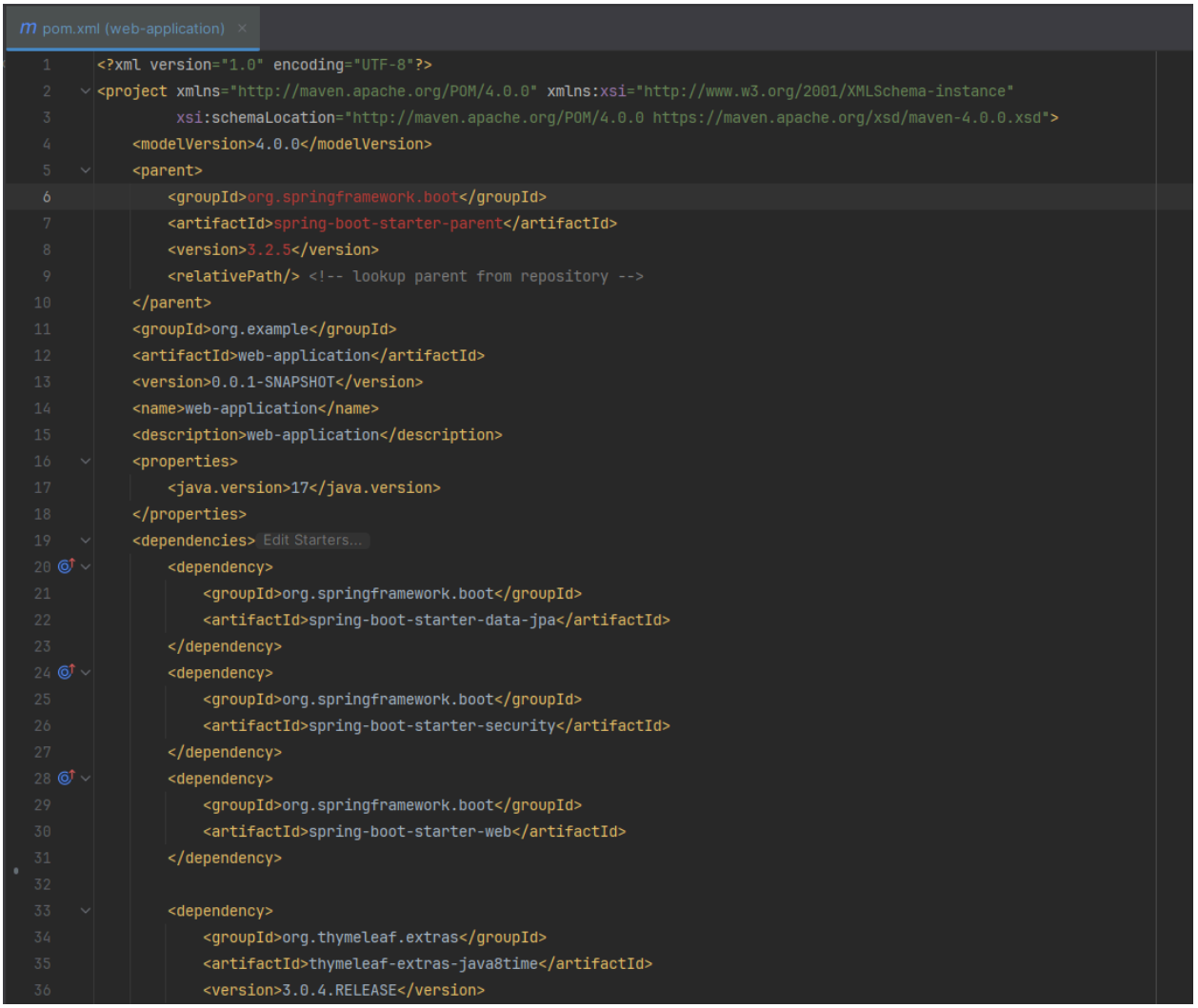

Рисунок 3.2 – POM.xml з підключеними фреймворками та бібліотеками.

Особливу увагу слід приділити системі керування версіями, такій як Git, яка дозволяє відстежувати зміни в коді та працювати над проектом спільно з іншими розробниками за потреби (рис. 3.3).

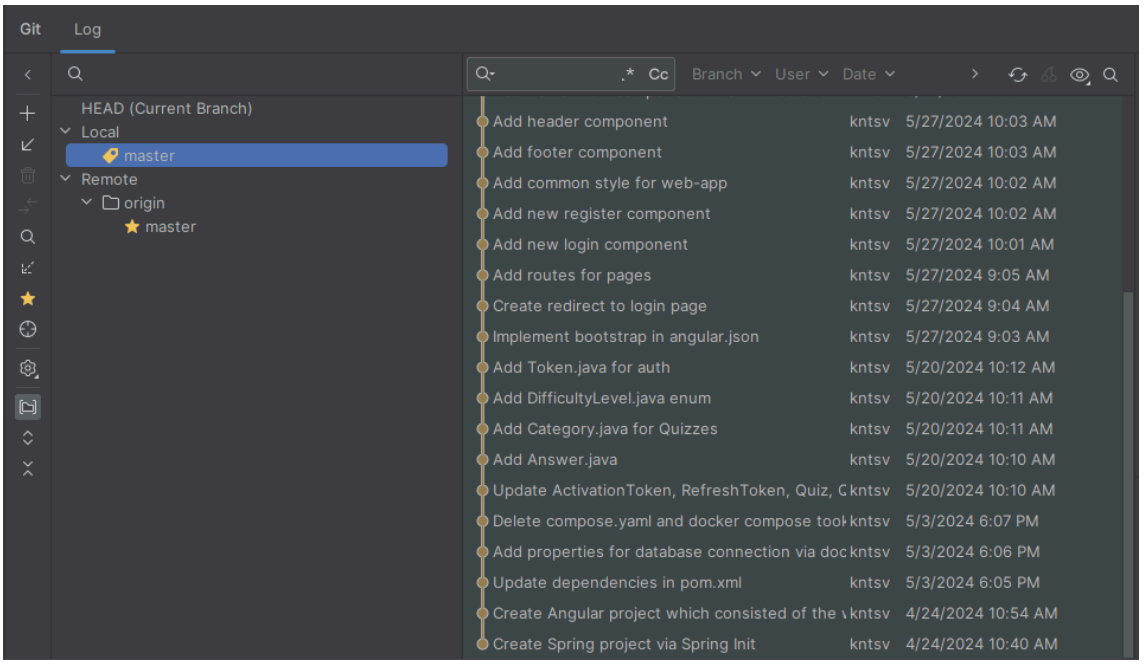

Рисунок 3.3 – Git та коміти проекту.

Програма була розділена на дві складові: Backend та Frontend. До backend віднесено роботу бізнес-логіки проєкту, а саме обробку HTTP-запитів для роботи з даними від frontend. Проект складається з багатьох пакетів, які вміщують різні функціонали (рис. 3.4).

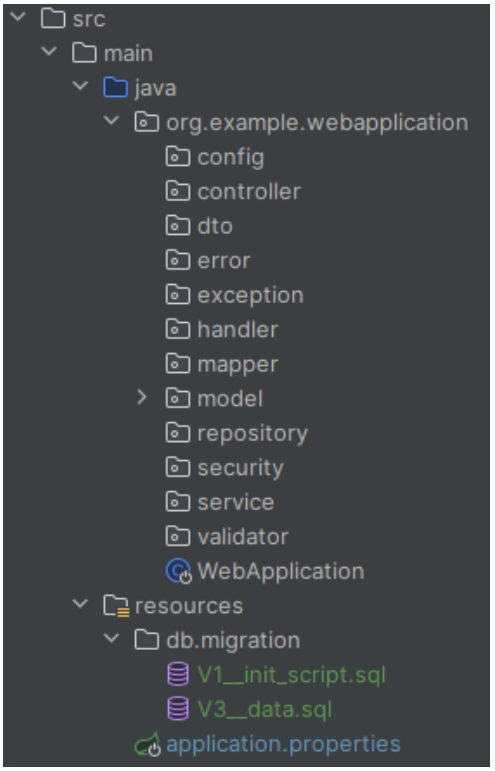

Рисунок 3.4 – Файлова структура backend частини проекту.

Короткий опис кожного пакету:

- Config – містить основні налаштування програми.
- Controller – контролери для обробки HTTP-запитів.
- DTO (Data Transfer Object) – об’єкти для передачі даних між підсистемами застосунку.
- Error – класи, що вміщують помилки роботи програми.
- Exception – класи виключень для представлення деяких некритичних помилок та інструкцій щодо їх обробки.
- Handler – використовується для обробки певних подій.
- Model – класи-моделі, що використовуються як сутності в застосунку та зазвичай відповідають таблицям бази даних (рис. 3.5–3.6)
- Repository – взаємодія з базою даних, зокрема методи створення, читання, оновлення або видалення сутностей.
- Security – класи, пов’язані з безпекою застосунку.
- Service – бізнес-логіка застосунку та взаємодія з репозиторіями для отримання та збереження даних, що викликаються контролерами.
- Validator – перевірка даних на відповідність певним критеріям.

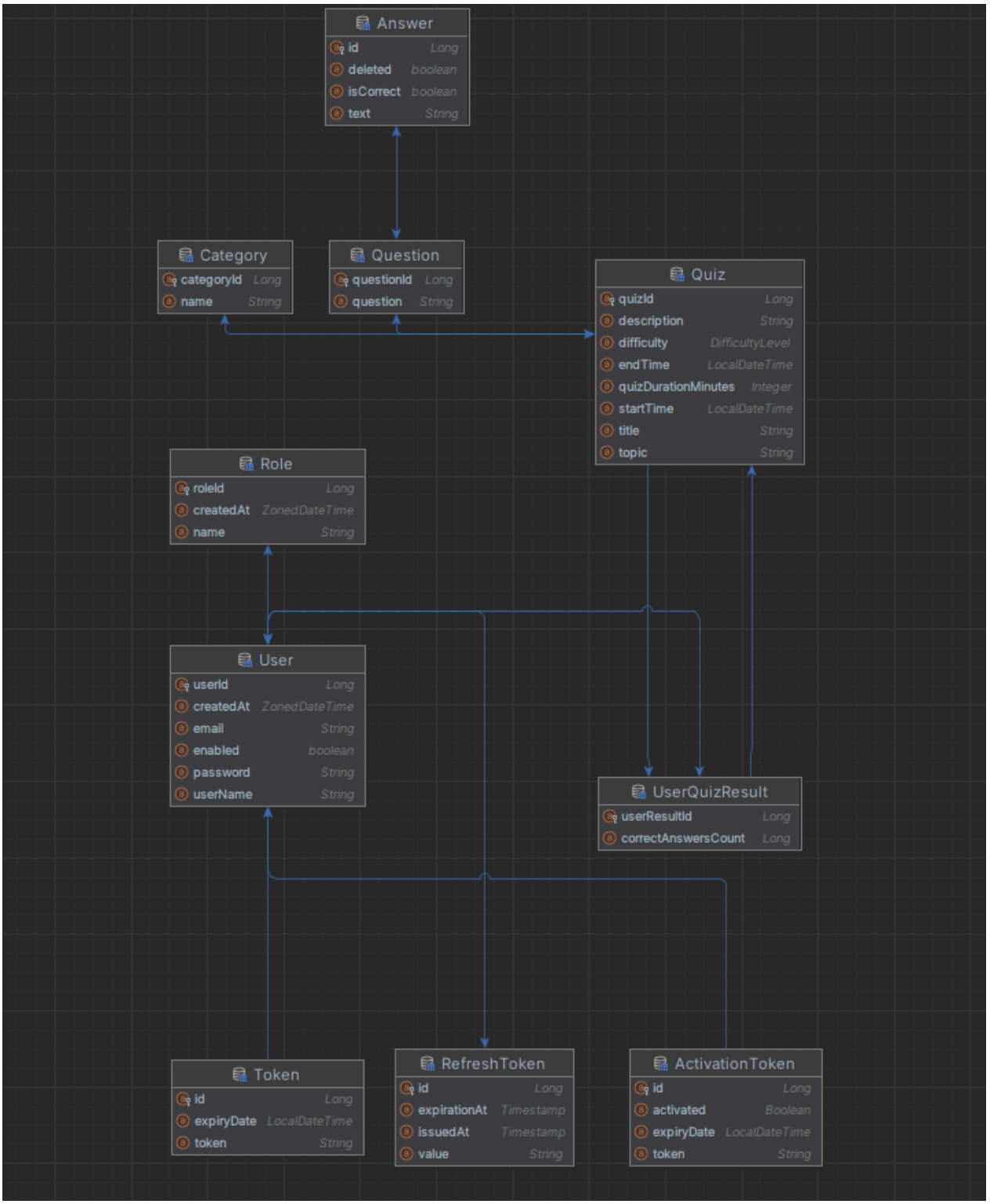

Рисунок 3.5 – Модель бази даних програми.

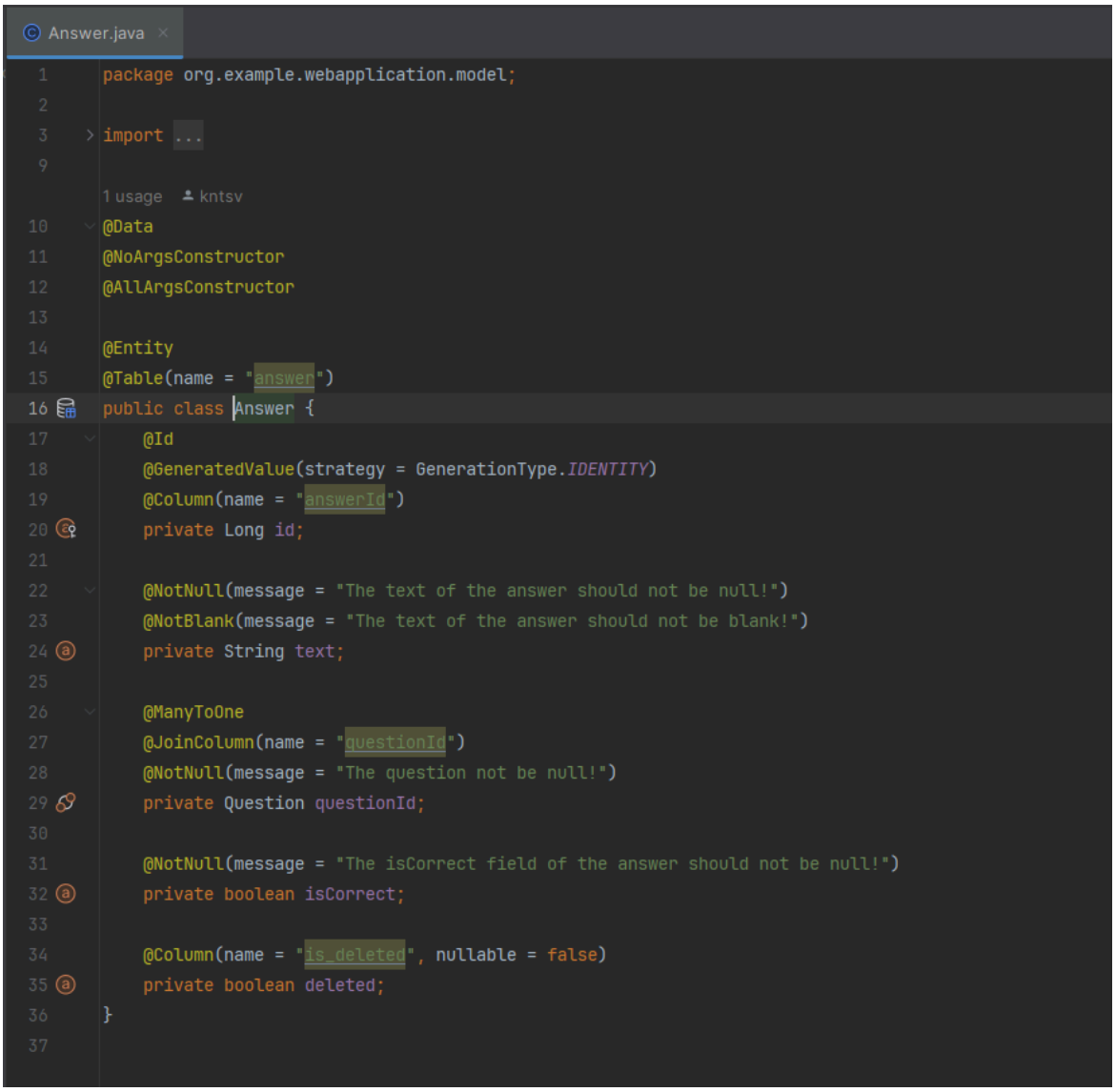

Рисунок 3.6 – Приклад сутності “Answer” з пакету “model”.

До frontend частини віднесено зовнішній вигляд застосунку, який має бути гарно оформлений та простим у використанні. Для розробки frontend використано Angular (рис. 3.7).

Angular використовує модульність при роботі з frontend, що означає наявність модулів, таких як "Auth", який відповідає за аутентифікацію та реєстрацію користувачів. Кожен модуль містить компоненти, що в свою чергу можуть містити сервіси для взаємодії з HTTP-запитами до backend (рис. 3.7).

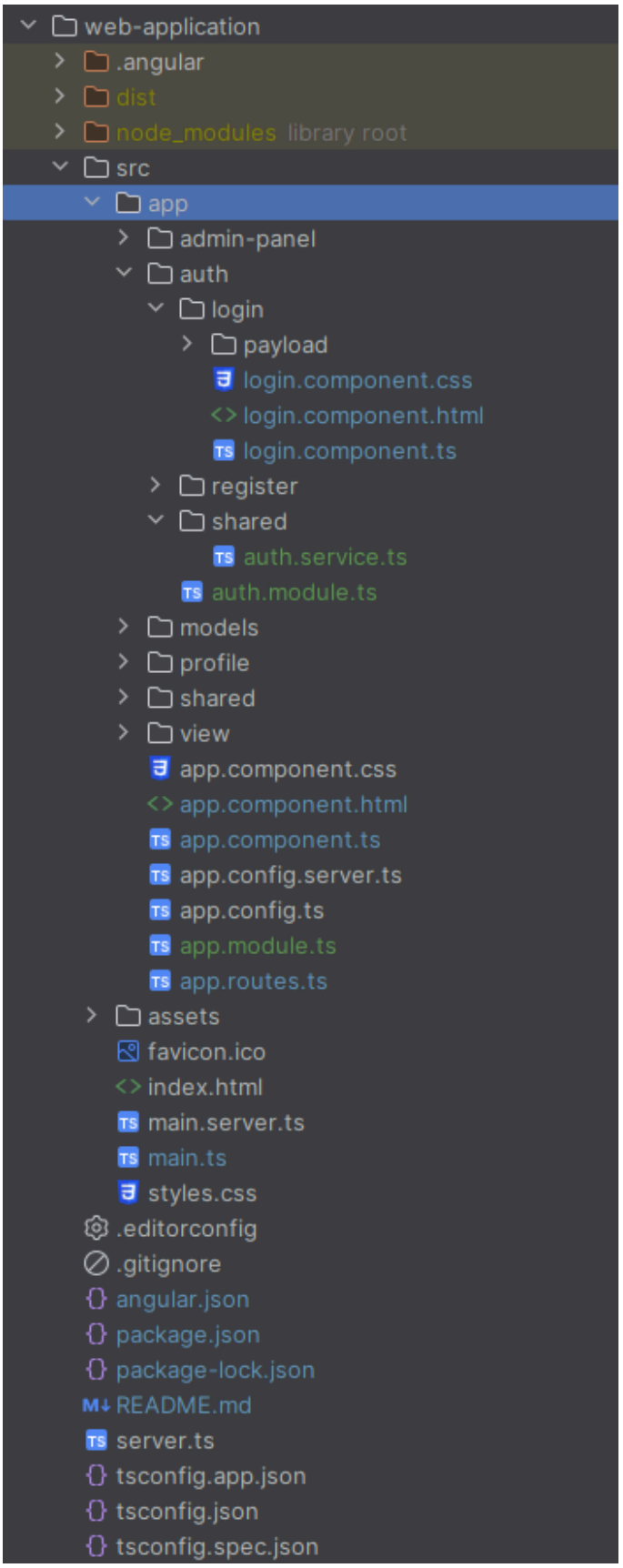

Рисунок 3.7 – Структура frontend частини застосунку.

Зовнішній вигляд веб-застосунку включає сторінки:

- Home – головна сторінка з прогресом користувача по тестах. (рис. 3.8)
- Quizzes – сторінка проходження тестів (рис. 3.9).
- Profile - сторінка профілю користувача з можливістю змін (рис. 3.10).
- Log out - функція виходу користувача з акаунту (рис. 3.10).

Компоненти описують свою роботу у файлах name.component.ts, де за допомогою декоратора @Component задаються основні властивості компоненту: selector, templateURL та styleURL (рис. 3.11) [13] .

Програма має модульну структуру, що включає модулі з декоратором @NgModule, який повідомляє Angular, як компілювати та запускати код модуля. Він визначає власні компоненти, директиви та канали модуля, роблячи деякі з них загальнодоступними для зовнішніх компонентів. @NgModule може додавати постачальників послуг до інжекторів залежностей програм (рис. 3.12) [13].

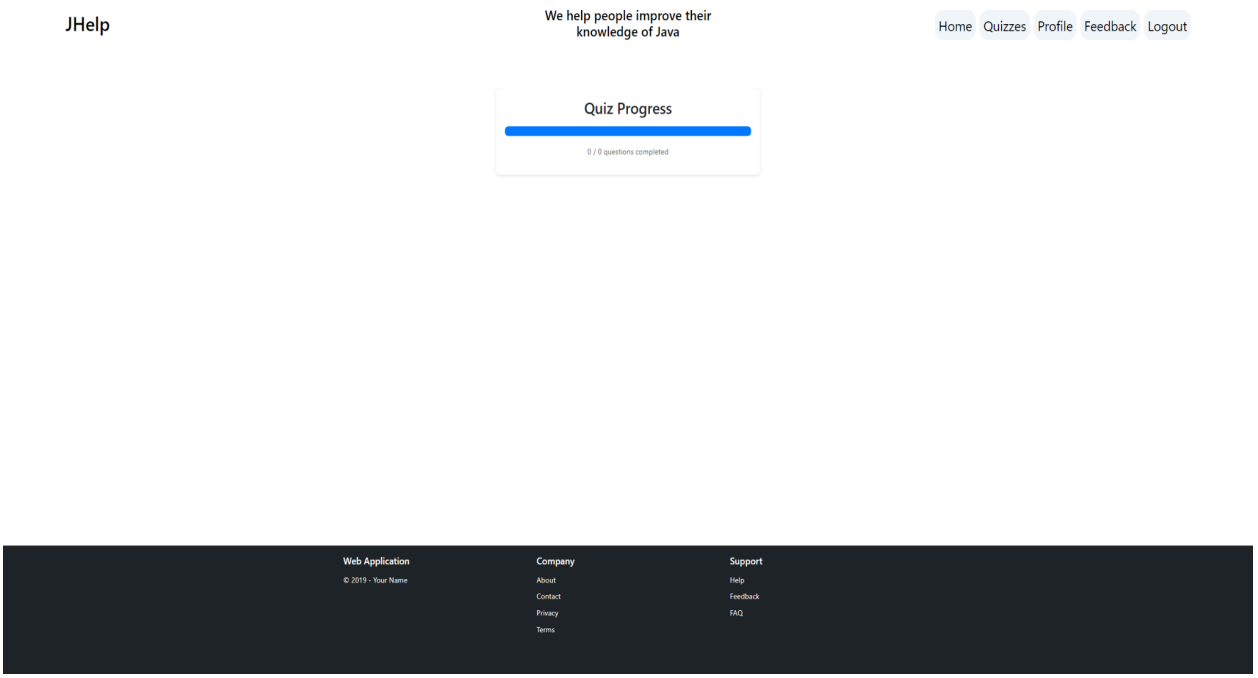

Рисунок 3.8 – Зовнішній вигляд сторінки Home.

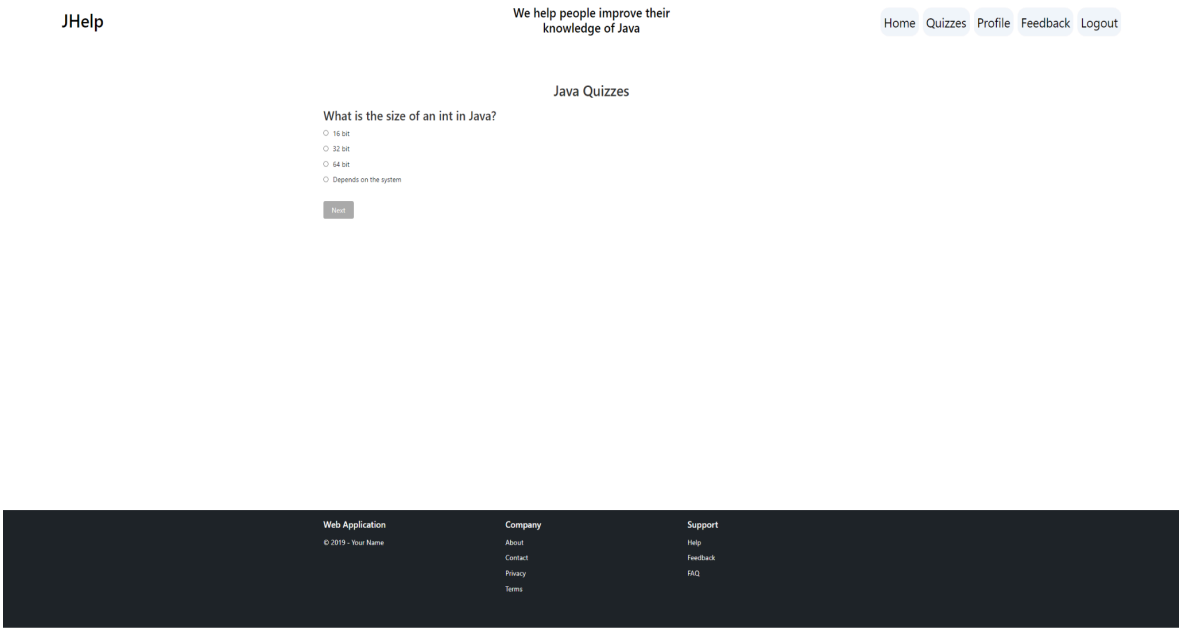

Рисунок 3.9 – Зовнішній вигляд сторінки Quizzes.

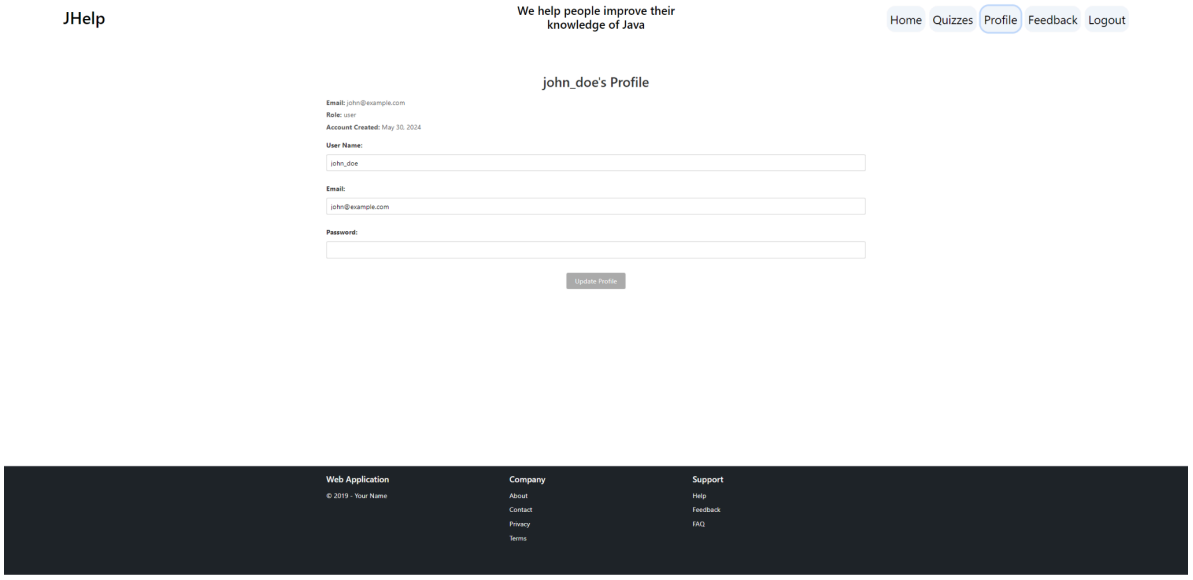

Рисунок 3.10 – Зовнішній вигляд сторінки Profile.

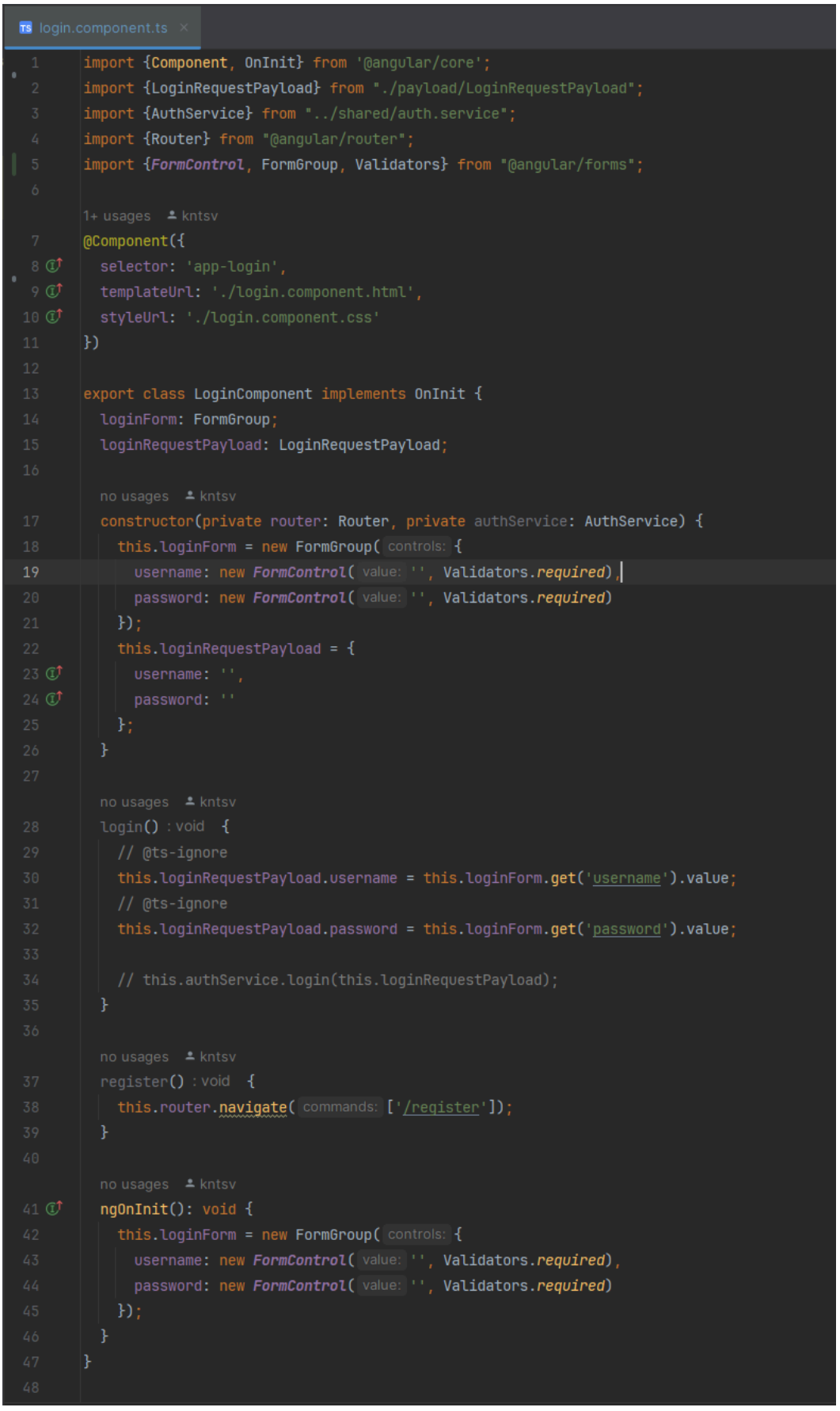

Рисунок 3.11 – Вигляд файлу компоненту “login.component.ts”.

Рисунок 3.12 – Вигляд файлу модулю “auth.module.ts”.

Висновок до розділу 3

У розділі описано процес створення програмної моделі веб-додатку для підготовки до співбесіди на посаду в IT-компанії. Весь процес розробки складався з кількох важливих етапів, кожен з яких мав ключове значення для успішної реалізації проекту.

На першому етапі було здійснено налаштування середовища розробки, де було обрано Intellij IDEA як основне середовище завдяки його потужним можливостям та зручності використання. Це забезпечило ефективне управління проектом та оптимізувати процес розробки.

Другим кроком стало підключення необхідних бібліотек та фреймворків за допомогою Maven, щоб стандартизувати процес управління залежностями та спростити підключення зовнішніх пакетів. Особлива увага була приділена системі керування версіями Git, що дозволяє відстежувати зміни в коді та забезпечило можливість спільної роботи над проектом, що є критичним для сучасної розробки програмного забезпечення.

Проект був поділений на дві основні складові: Backend та Frontend. Backend частина відповідає за обробку бізнес-логіки та HTTP-запитів, використовуючи різні пакети для налаштувань, контролерів, DTO, обробки помилок, моделей, репозиторіїв, безпеки, сервісів та валідації даних. Це забезпечило високу модульність та масштабованість додатку.

Frontend частина була реалізована за допомогою Angular, що дозволило створити модульну структуру інтерфейсу користувача. Кожен модуль відповідав за окремий функціональний блок, такий як аутентифікація, проходження тестів та управління профілем користувача. Завдяки цьому вдалося створити інтуїтивно зрозумілий та зручний інтерфейс.

У підсумку, розробка веб-додатку пройшла всі необхідні етапи від налаштування середовища розробки до інтеграції та тестування всіх компонентів. Важливість кожного етапу та використання сучасних інструментів та технологій забезпечили створення якісного продукту, що відповідає вимогам користувачів та готовий до подальшого вдосконалення.

ВИСНОВКИ

У ході виконання кваліфікаційної роботи було представлено модель веб-додатку для для підготовки до співбесіди на посаду в ІТ-компанії.

Було проведено аналіз існуючих аналогів веб-додатків, які можуть допомогти в підготовці до співбесід, розглядаючи їх функціональність, переваги та недоліки. Це допоможе визначити найбільш ефективні платформи для різних аспектів підготовки, забезпечуючи всебічний підхід до підготовки кандидатів.

Проведено обґрунтування вибору інструментів та технологій для реалізації платформи, зважаючи на вимоги проекту, технічні знання команди, бюджет та терміни. Для фронтенду використані сучасні фреймворки такі як Angular, що забезпечують високу продуктивність та зручність розробки. Для бекенду було обрано Java Spring як шар логіки для обробки вхідних даних з фронтенду. Вибір бази даних залежав від потреб у зберіганні та обробці даних, де MySQL став надійним та продуктивним варіантом та чудовим “шаром даних” в трирівневій архітектурі. Також доцільно використана контейнеризація для гнучкого запуску на різних машинах для тестування роботи баз даних та самого додатку.

У роботі висвітлено процес створення програмної моделі веб-додатку для підготовки до технічної співбесіди на ІТ-посаду, що включає кілька важливих етапів, кожен з яких має вирішальне значення для успіху проекту.

На першому етапі здійснено налаштування середовища розробки, зокрема, було обрано Intellij IDEA як основне середовище завдяки його функціональним можливостям та зручності.

Наступним кроком стало підключення необхідних бібліотек і фреймворків за допомогою Maven. Проект поділено на дві основні частини: Backend та Frontend. Backend відповідав за обробку бізнес-логіки та HTTP-запитів, використовуючи різні пакети для налаштувань. Frontend реалізовано з використанням Angular, що дозволило створити модульну

структуру інтерфейсу користувача. Кожен модуль відповідав за окремий функціональний блок, такий як аутентифікація, проходження тестів та інше. В комбінації ці технології дозволили створити ефективну, продуктивну та привабливу платформу для підготовки до співбесід.

Розробка програмної моделі веб-додатку для підготовки до технічної співбесіди на посаду Junior Java Developer в IT-компанії має значну практичну цінність для студентів факультету комп'ютерних наук. Веб-додаток надає студентам можливість систематично готуватися до співбесід на посаду Junior Java Developer, охоплюючи як теоретичні, так і практичні аспекти. Це включає як практичні завдання на програмування, теоретичні питання, що часто зустрічаються на співбесідах та моделювання реальних ситуацій, які можуть виникнути під час інтерв'ю.

Розвиток практичних навичок: під час використання веб-додатку студенти можуть вдосконалювати свої знання з Java та суміжних технологій, отримувати практичний досвід розв'язання алгоритмічних завдань та працювати над реальними проектами та завданнями, що можуть бути запропоновані під час співбесіди. Опанування сучасних технологій: студенти матимуть можливість ознайомитися з різними технологіями та інструментами, що використовуються у сучасній розробці веб-додатків. Підвищення конкурентоспроможності на ринку праці: веб-додаток сприяє підвищенню конкурентоспроможності студентів на ринку праці. Завдяки систематичній підготовці та відточенню навичок вони будуть більш підготовленими до реальних співбесід, що збільшить їх шанси на успішне працевлаштування. Веб-додаток сприяє інтегрованому підходу до навчання: студенти отримують теоретичні знання, виконують практичні завдання та кейси та проходять тести та отримують зворотний зв'язок щодо своїх знань та навичок. Співпраця та обмін досвідом: веб-додаток може стати платформою для обміну досвідом та знаннями серед студентів. Вони можуть обговорювати завдання, ділитися своїми рішеннями та підходами, що сприяє колективному навчанню та підвищенню загального рівня компетентності.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Cay S. Horstmann «Core Java: Fundamentals», Pearson P T R; 10th edition 2016р. 1007с.
2. Олександр Швець «Занурення в Патерни проектування» Refactoring.Guru, пер. Віталій Гальцев, Олександр Швець. 2022р. 396с. [Електронний ресурс]. – режим доступу: URL: https://refactoring.guru
3. Java Platform, Standard Edition & Java Development Kit Version 21 API Specification. [Електронний ресурс]. – режим доступу: URL:https://docs.oracle.com/en/java/javase/21/docs/api/index.html (Дата звернення 30.12.2023р)
4. Craig Walls «Spring in Action», Manning, 5th edition 2018, 520с.
5. ГОСТ 2.105-95 Єдина система конструкторської документації. Загальні вимоги до текстових документів. Міждержавний стандарт. К.: Держстандарт України. Наказ від 27.06.1996 № 259.
6. Текстові документи. Загальні вимоги СОУ 207.01:2017 / Ю. М. Бойко, Г. В. Красильнікова, Л. І. Першина, Т. Ф. Косянчук. 2-е вид., виправлене. Хмельницький: ХНУ, 2018р. 45 с.
7. ДСТУ 3008:2015 Інформація та документація. Звіти у сфері науки і техніки. Структура та правила оформлювання. Український інститут науково-технічної і економічної інформації; Технічний комітет стандартизації «Інформація і документація». Наказ від 22 червня 2015р. № 61 з 2017-07-01.
8. ДСТУ 3582:2013 Інформація та документація. Бібліографічний опис. Скорочення слів і словосполучень українською мовою. Загальні вимоги та правила (ISO 4:1984, NEQ; ISO 832:1994, NEQ)
9. W3Schools школа для веб-розробників. [Електронний ресурс]. – режим доступу: URL: https://www.w3schools.com/ (Дата звернення 05.01.2024р)

10. Leetcode. Platform to help you enhance your skills, expand your knowledge and prepare for technical interviews. [Електронний ресурс]. – режим доступу: URL: https://leetcode.com/ (Дата звернення 30.12.2023р)
11. JavaRush. Інтерактивний онлайн курс по вивченню Java-програмування з нуля. [Електронний ресурс]. – режим доступу: URL: https://javarush.com/ (Дата звернення 30.02.2024р)
12. Codewars. Site which helps to improve your development skills by training with your peers on code kata that continuously challenges and pushes your coding practice. [Електронний ресурс]. – режим доступу: URL: https://www.codewars.com/ (Дата звернення 30.01.2024р)
13. Angular. Web framework that empowers developers to build fast, reliable applications. [Електронний ресурс]. – режим доступу: URL: https://angular.dev/ (Дата звернення 12.03.2024р)
14. MySQL. The world's most popular open-source database. [Електронний ресурс]. – режим доступу: URL: https://dev.mysql.com/ (Дата звернення 17.01.2024р)
15. Docker. Accelerate how you build, share, and run applications. [Електронний ресурс]. – режим доступу: URL: https://www.docker.com/ (Дата звернення 21.04.2024р)
16. Hibernate. MORE THAN AN ORM, DISCOVER THE HIBERNATE GALAXY. [Електронний ресурс]. – режим доступу: URL: https://hibernate.org/ (Дата звернення 12.02.2024р)
17. Lombok. Lombok is a Java library that automatically plugs into your editor and builds tools, spicing up your Java. [Електронний ресурс]. – режим доступу: URL:https://projectlombok.org/ (Дата звернення 09.03.20243р)

ДОДАТКИ

Додаток А

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Харківський національний університет імені В. Н. Каразіна

Факультет комп'ютерних наук
Кафедра теоретичної та прикладної системотехніки
Рівень вищої освіти (освітньо-кваліфікаційний рівень) **бакалавр**
Галузь знань: **12 – Інформаційні технології**
Спеціальність: **123 – Комп'ютерна інженерія.**

ЗАТВЕРДЖУЮ
Завідувач кафедри теоретичної
та прикладної системотехніки
_____ д.т.н., проф. Шматков С. І.
«21» грудня 2023 року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ

КУНЦЕВИЧА МАКСИМА ПАВЛОВИЧА

1. Тема роботи **«Веб-платформа для підготовки до співбесіди на посаду в IT-компанії»**

керівник роботи: Мороз Ольга Юріївна, PhD, старший викладач ЗВО
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «03» травня 2024року № 4101-5/909

2.Строк подання студентом роботи 31 травня 2024року

3. Перелік питань, які потрібно розробити
1. Дослідження та аналіз існуючих аналогів веб-додатків для підготовки до співбесіди на посаду Junior Java Developer в IT компанії.
2. Аналіз кращих принципів побудови та розробки веб-додатків для підготовки до співбесід.
3. Аналіз принципів роботи клієнт-серверних моделей веб-додатків.
4. Вибір платформи для розробки додатку проходження співбесіди на посаду в IT.
5. Розробка програмної моделі веб-додатку для підготовки до співбесіди на посаду Junior Java Developer в IT компанії.
6. Розробка рекомендацій по впровадженню технологій, які використовуються в роботі.

4. План роботи

№ з/п	Назви етапів роботи	Термін виконання етапів роботи
1	Аналіз та пошук методичних матеріалів для розробки веб-додатків для підготовки до співбесіди на посаду Junior Java Developer в IT компанії.	21.12.2023 - 31.12.2023
2	Аналіз та розробка бази даних для вивчення ключових моментів роботи Junior Java Developer'а.	1.01.2024 - 29.02.2024
3	Огляд і аналіз написання серверної сторони	1.02.2024 - 1.03.2024
4	Розробка технічного завдання на розроблену модель	1.03.2024 - 15.03.2024
5	Реалізація клієнта та клієнт-серверного діалогу	16.03.2024 - 30.04.2024
6	Розробка програми та методики тестування розробленої моделі	16.03.2024- 1.04.2024
7	Розробка інструкції користувача для користування веб-додатком	1.04.2024- 16.04.2024
8	Підготовка доповіді на тему кваліфікаційної роботи	25.04.2024- 25.05.2024
7	Оформлення пояснювальної записки	1.04.2024 - 15.04.2024
8	Оформлення звіту за результатами переддипломної практики	22.04.2024 - 31.05.2024
9	Представлення кваліфікаційної роботи керівнику та рецензенту	20.05.2024 - 27.05.2024

5. Дата видачі завдання 30.10.2023

Студент М. П. Кунцевич _____
ініціали, прізвище підпис

Керівник роботи О. Ю. Мороз _____
ініціали, прізвище підпис

Додаток Б

Затверджую
_____
«_____»_____2024 р.

Технічне завдання
на розробку програмного виробу «Веб-платформа для підготовки до співбесіди на посаду в ІТ-компанії»

1.	Введення	1.1. Назва: Веб-платформа для підготовки до співбесіди на посаду в ІТ-компанії 1.2. Галузь застосування: **Інформаційні технології**.
2.	Підстава для розробки	2.1. Навчальний план за спеціальністю 123 – Комп'ютерна інженерія 2.2. Завдання на кваліфікаційну роботу бакалавра № 4101-5/909 від 03.05.2024 року (представити як Додаток А до пояснювальної записки до кваліфікаційної роботи).
3.	Призначення розробки	3.1. Мета розробки: розробка і реалізація веб-платформи, яка спрощує і автоматизує процес проведення співбесіди на посаду в ІТ компанії. 3.2. Призначення розробки: покращити систематизований аналіз знань набутих під час навчання та підготовки технічної частини до проходження етапу відбору на відповідні позиції в ІТ-компаніях 3.3. Вихідні дані розробки: програмно реалізований продукт призначений для проходження тестування для перевірки засвоєного матеріалу протягом навчання. Визначення можливості проходження реального технічного інтерв'ю на поточні позиції в компаніях.
4.	Технічні вимоги до програмного виробу	4.1. Вимоги до функціональних характеристик: можливість провести авторизацію користувача для входження в обліковий запис, зберігання результату

<table>
<tr><td></td><td></td><td colspan="2">можливість проходити технічне інтерв'ю в вибраних темах, розділення на складності.
4.2. Вимоги до надійності: Захищення даних користувача (паролі, пошта та інша конфіденційна інформація).
4.3.Вимоги до умов експлуатації: немає
4.4. Вимоги до складу і параметрів технічних засобів: звичайне обчислювальне обладнання, ПК, тощо
4.5. Вимоги до інформаційної та програмної сумісності: сумісність зі смартфонами, а також браузерами різних типів.
4.6. Вимоги до маркування та упаковки: немає
4.7. Вимоги до транспортування і зберігання: немає
4.8. Спеціальні вимоги: немає.</td></tr>
<tr><td>5.</td><td>Вимоги до програмної документації</td><td colspan="2">Програмною документацією до виробу «Веб-платформа для підготовки до співбесіди на посаду в ІТ-компанії» вважати:
1) Справжнє Технічне завдання на розробку виробу (представити у вигляді Додатку Б до пояснювальної записки до кваліфікаційної роботи).
2) Опис виробу (представити в розділі 3 пояснювальної записки до кваліфікаційної роботи)</td></tr>
<tr><td>6.</td><td>Вимоги до техніко-економічних показників</td><td colspan="2">Програмною документацією до виробу «Веб-платформа для підготовки до співбесіди на посаду в ІТ-компанії» вважати:
1) Справжнє Технічне завдання на розробку виробу (представити у вигляді Додатку Б до пояснювальної записки до кваліфікаційної роботи).
2) Опис програмного виробу (представити в Розділі 3 пояснювальної записки до кваліфікаційної роботи).

3) Джерела базової інформації.</td></tr>
<tr><td rowspan="2">7.</td><td rowspan="2">Стадії і етапи розробки</td><td>Дата</td><td>Назва етапу</td></tr>
<tr><td>від 21 грудня 2023
до 31 грудня 2023</td><td>Аналіз та пошук методичних матеріалів для розробки веб-додатків для підготовки до співбесіди на посаду</td></tr>
</table>

			веб-додатків для підготовки до співбесіди на посаду Junior Java Developer в IT компанії.
		від 1 січня 2024 до 29 лютого 2024	Аналіз та розробка бази даних для вивчення ключових моментів роботи Junior Java Developer'a.
		від 1 лютого 2024 до 1 березня 2024	Огляд і аналіз написання серверної сторони
		від 1 березня 2024 до 15 березня 2024	Розробка технічного завдання на розроблену модель
		від 16 березня 2024 до 30 квітня 2024	Реалізація клієнта та клієнт-серверного діалогу
		від 16 березня 2024 до 1 квітня 2024	Розробка програми та методики тестування розробленої моделі
		від 1 квітня 2024 до 16 квітня 2024	Розробка інструкції користувача для користування веб-додатком
		від 25 квітня 2024 до 25 травня 2024	Підготовка доповіді на тему кваліфікаційної роботи
		від 1 квітня 2024 до 15 квітня2024	Оформлення пояснювальної записки
		від 22 квітня 2024 до 31 травня 2024	Оформлення звіту за результатами переддипломної

		від 20 травня 2024 до 27 травня 2024	Представлення кваліфікаційної роботи керівнику та рецензенту
8.	Порядок контролю і приймання програмного продукту (моделі)	1. Перевірку ходу розробки програми виконувати раз в 1 тиждень. 2. Захист програмної моделі провести на засіданні Атестаційної комісії. 3. Пояснювальну записку подати в електронному вигляді в 1 примірнику.	

Виконавець
студент групи КІ- 41
КУНЦЕВІЧ М. П.
_____

Замовник
PhD, ст. викладач ЗВО
МОРОЗ О. Ю.
_____

Додаток В

Програма і методика випробувань програмного виробу
«Веб-платформа для підготовки до співбесіди на посаду в IT-компанії»

1 Об'єкт випробувань

1.1 Назва програмного виробу: «Веб-платформа для підготовки до співбесіди на посаду в IT-компанії»

1.2 Галузь застосування: Інформаційні технології

1.3 Перераховані відомості запозичуються з відповідних розділів Технічного завдання.

2. Мета випробувань

Перевірка відповідності функціональності програмної реалізації системи заявленим функціональним можливостям в технічному завданні (Додаток Б до пояснювальної записки до кваліфікаційної роботи).

3. Загальні положення

3.1 Підстави для проведення випробувань

Підставою для проведення випробувань є наказ про призначення атестаційної комісії.

3.2 Місце і тривалість випробувань

Приймальні (приймально-здавальні) випробування проводяться на базі комп'ютерного класу кафедри в період роботи атестаційної комісії.

3.3 Обсяг випробувань

Приймальні випробування програмного виробу проводяться в обсязі відповідному цієї програми і методики випробувань.

3.4 Організації, які беруть участь у випробуваннях

Приймальні випробування проводяться атестаційною комісією напередодні засідання (або в процесі засідання) за участю Замовника, Виконавця та інших осіб, присутніх на засіданні.

4. Вимоги до програми або програмного виробу

Модель повинна задовольняти наступним вимогам:

4.1. Вимоги до функціональних характеристик: можливість провести авторизацію користувача для входження в обліковий запис, зберігання результату проходження тестування протягом 15 днів, можливість проходити технічне інтерв'ю в вибраних темах, розділення на складності.

4.2. Вимоги до надійності: Захищення даних користувача (паролі, пошта та інша конфіденційна інформація).

4.3.Вимоги до умов експлуатації: немає

4.4. Вимоги до складу і параметрів технічних засобів: звичайне обчислювальне обладнання, ПК, тощо

4.5. Вимоги до інформаційної та програмної сумісності: сумісність зі смартфонами, а також браузерами різних типів.

4.6. Вимоги до маркування та упаковки: немає

4.7. Вимоги до транспортування і зберігання: немає

4.8. Спеціальні вимоги: немає. Спеціальні вимоги (не пред'являються).

5. Вимоги до програмної документації

Документацією до виробу «Веб-платформа для підготовки до співбесіди на посаду в IT-компанії» вважати:

1) Документація по мові програмування та додаткові мануали.

2) Програму і методику випробувань розробленої програми (представити як Додаток В до пояснювальної записки до кваліфікаційної роботи).

3) Опис програмного виробу (представити в Розділі 3 пояснювальної записки до кваліфікаційної роботи).

4) Джерела базової інформації.

6. Засоби і порядок випробувань

6.1 Засоби випробувань

Засоби випробувань представлено на ПК на яких встановлено наступні програмні засоби: інтерпретатор мови програмування.

6.2 Порядок проведення випробувань

Як правило, випробування проводяться в два етапи:

- ознайомчий (1-й етап);
- власне випробування програмного виробу (2-й етап).

Перелік перевірок, що проводяться на 1 етапі випробувань, включає в себе:

1) Перевірку комплектності складу програмної документації здійснюється за критерієм наявності зазначеної в ТЗ документації.

2) Перевірку якості програмної документації. Перевірку здійснювати за критерієм відповідності вимогам ГОСТ 19.301-79 ЕСПД. «Програма і методика випробувань».

Перелік перевірок, що проводяться на 2 етапі випробувань, включає в себе:

1) Перевірку відповідності технічних характеристик програми вимогам технічного завдання.

2) Перевірку ступеня виконання функціональних вимог до програми.

3) Методику проведення перевірок:

a) Запустити програмне забезпечення.

b) Порядок проведення випробувань:

- Зробити налаштування.
- Перевірити чи виконуються запити.

- Перевірити чи формується звіт.

4) Якщо перевірки на першому та другому етапах виконано успішно, то виріб вважається таким, що пройшов випробування.

Для проведення випробувань пропонується тест 1, тест 2 та тест 3.

Тест 1

1. Перевірка виконання програми;
2. Підключення локального серверу;
3. Отримання відповіді про успішне підключення.

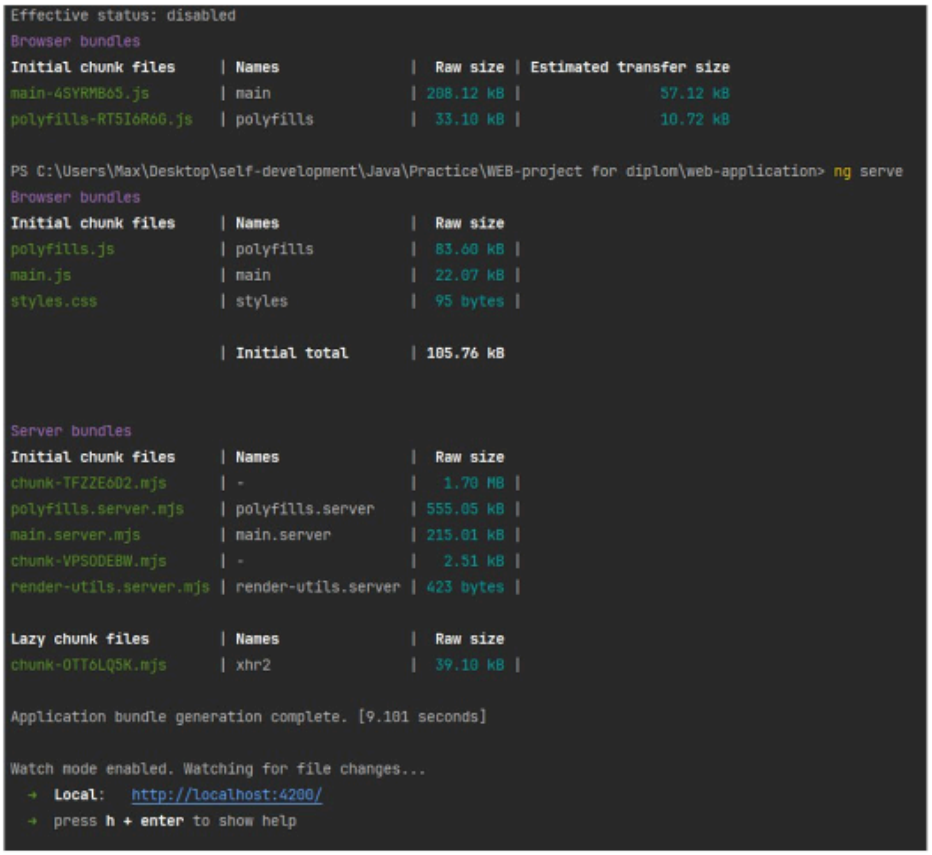

Рис. В.1 - Тест 1. проведення випробувань

Тест 2

1. Перевірка виконання програми
2. Реєстрація користувача;
3. Отримання результатів.

Рис. В.2 - Тест 2

Тест 3

4. Перевірка виконання програми
5. Успішне створення акаунту;
6. Отримання результату.

Рис. В.3 - Тест 3

Тест вважається пройденим, якщо відбуваються вказані операції і їх відображення у програмному продукті.

Висновки: тест 1 успішно пройшов випробування, тест 2 успішно пройшов випробування і тест 3 успішно пройшов випробування. Випробування пройшло успішно.

Виконавець: студент групи КІ-41, Кунцевич М. П.

Додаток Г

Фрагмент коду frontend частини.

```
login.component.ts
import {Component, OnInit} from '@angular/core';
import {LoginRequestPayload} from "./payload/LoginRequestPayload";
import {AuthService} from "../shared/auth.service";
import {Router} from "@angular/router";
import {FormControl, FormGroup, Validators} from "@angular/forms";

1 usages  kntsv
@Component({
  selector: 'app-login',
  templateUrl: './login.component.html',
  styleUrl: './login.component.css'
})

export class LoginComponent implements OnInit {
  loginForm: FormGroup;
  loginRequestPayload: LoginRequestPayload;

  no usages  kntsv
  constructor(private router: Router, private authService: AuthService) {
    this.loginForm = new FormGroup( controls: {
      username: new FormControl( value: '', Validators.required),
      password: new FormControl( value: '', Validators.required)
    });
    this.loginRequestPayload = {
      username: '',
      password: ''
    };
  }

  no usages  kntsv
  login() {
    // @ts-ignore
    this.loginRequestPayload.username = this.loginForm.get('username').value;
    // @ts-ignore
    this.loginRequestPayload.password = this.loginForm.get('password').value;

    // this.authService.login(this.loginRequestPayload);
  }

  no usages  kntsv
  register() {
    this.router.navigate( commands: ['/register']);
  }

  no usages  kntsv
  ngOnInit(): void {
    this.loginForm = new FormGroup( controls: {
      username: new FormControl( value: '', Validators.required),
      password: new FormControl( value: '', Validators.required)
    });
  }
}
```

register.component.ts

```
import {Component, OnInit} from '@angular/core';
import {AuthService} from "../shared/auth.service";
import {Router} from "@angular/router";
import {RegisterRequestPayload} from "./payload/RegisterRequestPayload";
import {FormControl, FormGroup, Validators} from '@angular/forms';


1+ usages  kntsv *
@Component({
  selector: 'app-register',
  templateUrl: './register.component.html',
  styleUrl: './register.component.css'
})
export class RegisterComponent implements OnInit {
  registerForm: FormGroup;
  registerRequestPayload: RegisterRequestPayload;

  no usages  kntsv *
  constructor(private router: Router, private authService: AuthService) {
    this.registerForm = new FormGroup( controls: {
      email: new FormControl( value: '', Validators.required),
      name: new FormControl( value: '', Validators.required),
      password: new FormControl( value: '', Validators.required),
      surname: new FormControl( value: '', Validators.required)
    });
    this.registerRequestPayload = {
      name: '',
      surname: '',
      email: '',
      password: ''
    };
  }

  no usages  kntsv
  register() : void  {
    // @ts-ignore
    this.registerRequestPayload.name = this.registerForm.get('name').value;
    // @ts-ignore
    this.registerRequestPayload.surname = this.registerForm.get('surname').value;
    // @ts-ignore
    this.registerRequestPayload.email = this.registerForm.get('email').value;
    // @ts-ignore
    this.registerRequestPayload.password = this.registerForm.get('password').value;

    // this.authService.register(this.registerRequestPayload);
  }

  no usages  kntsv
  login() : void  {
    this.router.navigate( commands: ['/login']);
  }
```

home-view.component.html

```
<div class="wrapper">
  <app-header></app-header>
  <div class="main-content">
    <router-outlet></router-outlet>
  </div>
  <app-footer></app-footer>
</div>
```

header.component.html

```
<header id="header">
  <nav class="navbar navbar-expand-lg">
    <div class="nav-logo">
      <a class="navbar-brand" href="#"><h1>JHelp</h1></a>
    </div>
    <div class="header-center-text">
      <h3>We help people improve their <br> knowledge of Java </h3>
    </div>
    <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNav"
            aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
      <span class="navbar-toggler-icon"></span>
    </button>
    <div class="collapse navbar-collapse" id="navbarNav">
      <ul class="navbar-nav">
        <li class="nav-item active">
          <a class="nav-link" routerLink="/home">Home</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" routerLink="/quizzes">Quizzes</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" routerLink="/profile">Profile</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" routerLink="/feedback">Feedback</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="#">Logout</a>
        </li>
      </ul>
    </div>
  </nav>
</header>
```

Token.java

```java
package org.example.webapplication.model;

import ...

no usages  kntsv
@Entity
@Table(name = "token")
@Data
@AllArgsConstructor
@NoArgsConstructor
@Builder(builderMethodName = "with", toBuilder = true)
public class Token {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "id")
    private Long id;

    @Column(name = "expiry_date")
    private LocalDateTime expiryDate;

    @Column(name = "token")
    private String token;

    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "user_id", referencedColumnName = "user_id")
    private User user;
}
```

User.java

```java
package org.example.webapplication.model;

import ...

5 usages  kntsv
@Entity
@Table(name = "\"user\"",
        uniqueConstraints = @UniqueConstraint(name = "uq_user_email", columnNames = "email"))
@Data
@AllArgsConstructor
@NoArgsConstructor
@Builder(builderMethodName = "with")
public class User {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "user_id")
    private Long userId;

    @Email
    @NotBlank(message = "Email is mandatory")
    @NaturalId
    private String email;

    @NotBlank(message = "Username is mandatory")
    @Size(min = 2, max = 30, message = "Username must be between 2 and 30 characters")
    private String userName;

    @NotBlank(message = "Password is mandatory")
    @Size(min = 8, message = "Password must be at least 8 characters")
    private String password;

    @ManyToOne
    @JoinColumn(name = "roleId")
    private Role role;

    private boolean enabled;

    @CreationTimestamp
    @Column(name = "created_at", nullable = false, updatable = false)
    private ZonedDateTime createdAt;

    @ToString.Exclude
    @OneToMany(mappedBy = "user", cascade = CascadeType.ALL, orphanRemoval = true)
    private List<RefreshToken> refreshTokens;

    @OneToMany(mappedBy = "user", cascade = CascadeType.ALL, orphanRemoval = true)
    private List<UserQuizResult> userQuizResults = new ArrayList<>();

}
```

shared.module.ts

```ts
import { NgModule } from '@angular/core';
import { CommonModule } from '@angular/common';
import {FooterComponent} from "./footer/footer.component";
import {HeaderComponent} from "./header/header.component";
import {RouterModule} from "@angular/router";



1+ usages  new *
@NgModule({
  declarations: [
    HeaderComponent,
    FooterComponent
  ],
  exports: [
    HeaderComponent,
    FooterComponent
  ],
  imports: [
    CommonModule,
    RouterModule,
  ]
})
export class SharedModule { }

```