

ЗМІСТ

ЗМІСТ	3
СПИСОК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАЧЕНЬ	5
ВСТУП	6
1 ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ГАЛУЗІ.....	8
1.1 Характеристика предметної галузі.....	8
1.2 Огляд існуючих аналогів.....	9
1.2.1 Первый тестовый.....	9
1.2.2 Quizful	10
1.2.3 InTester	12
2 ПОСТАНОВКА ЗАДАЧІ. ТЕХНІЧНЕ ЗАВДАННЯ	14
2.1 Вступ.....	14
2.2 Загальна характеристика системи.....	14
2.2.1 Сфера застосування.....	14
2.2.2 Функції системи	14
2.2.3 Джерело початкових даних.....	15
2.2.4 Результати, вихідні дані.....	15
2.3 Вимоги до обчислювального середовища.....	16
2.3.1 Склад та конфігурація апаратних засобів	16
2.3.2 Програмні засоби.....	16
2.3.3 Режим роботи	17
2.4 Зв'язок із зовнішнім середовищем.....	17
2.4.1 Формати і протоколи обміну для вхідних даних.....	17
2.4.2 Формати і протоколи для вихідних даних	18
2.4.3 Формати та правила зміни керуючих параметрів.....	19
2.5 Якість системи	20
2.5.1 Виконання стандартів та узгодження.....	20
2.5.2 Можливість перенесення	21
2.5.3 Надійність функціонування	21

					ЧДТУ 191539.003 ПЗ						
Зм.	Лист	№ документа	Підпис	Дата	«Web-додаток система тестування рівня кваліфікації працівника. Front-end Web-додатку» Пояснювальна записка				Літ.	Лист	Листів
Розроб.		Гаврилюк В.С.							Н	3	
Керівник		Півень О. Б.							ФІТІС, кафедра ПЗАС, ПЗ-154		
Н.контр.		Півень О. Б.									
Затв.		Первунінський С. М.									

2.6 Документація системи	22
2.6.1 Перелік та вимоги до опису посібника керівника	22
2.6.2 Вимоги до оформлення вихідних кодів	22
2.6.3 Вимоги до опису структури і формати даних.....	29
2.7 Вибір програмно-технічних засобів	29
3 ОПИС РОЗРОБЛЕНОЇ СИСТЕМИ	36
3.1 Опис шаблону сторінок	36
3.1.1 Сторінка Sign In.....	36
3.1.2 Сторінка Home.....	38
3.1.3 Сторінка Surveys List.....	41
3.1.4 Сторінка New Survey.....	43
3.1.5 Сторінка Survey Info.....	44
3.1.6 Сторінка Survey	46
3.2 Опис інтерфейсу та функціональних можливостей	46
3.2.1 Процес авторизації адміністратора	46
3.2.2 Процес перегляду детальної інформації по тестах	48
3.2.3 Процес створення нового тесту	50
3.2.4 Процес проходження створеного тесту	52
ВИСНОВКИ	54
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	55
Додаток А.....	57
Додаток Б.....	59
Додаток В.....	76
Додаток Г.....	88

СПИСОК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАЧЕНЬ

CSS	- Cascading Style Sheets
SCSS	- Sassy Cascading Style Sheets
SQL	- Structured query language
HTML	- Hypertext Markup Language
URL	- Uniform Resource Locator
FTP	- File Transfer Protocol
JSON	- JavaScript Object Notation
ОС	- Операційна Система
ПК	- Персональний Комп'ютер
БД	- База Даних
ТЗ	- Технічне Завдання
ІС	- Інформаційна Система
ЕОМ	- Електронно-обчислювальна машина

ВСТУП

Сьогодні в галузі ІТ існує дуже багато компаній із числом працівників понад тисячу і значно вище. Такі компанії дуже активно приймають на роботу нові кадри, але через високе завантаження відділу найму вони витрачають багато ресурсів на прості завдання, а саме попереднє тестування кандидата. Дана система має на меті спростити процес прийому на роботу нового працівника та зменшити витрати пов'язані з цим, за рахунок проведення попереднього стандартизованого тестування кандидата.

Також дана система стане в пригоді тоді, коли працівник зрозуміє, що вже готовий пройти тестування для підвищення своєї кваліфікації. Зазвичай таке тестування і всі пов'язані з ним витрати оплачує компанія. Але доволі часто буває і таке, що працівник провалює тест через свою самовпевненість, а компанія несе збитки. Тому перед відправкою запиту на саме тестування дана система зможе перевірити працівника на предмет готовності до власне самого тестування з підвищення рівня його кваліфікації.

Стандартизований тест або іспит являє собою тестування, яке проводиться та перевіряється заздалегідь установленим, або «стандартним», шляхом. Стандартизовані тести розроблені таким чином, що питання, умови проведення, оцінювання і тлумачення заздалегідь погоджені, процеси проведення і оцінювання визначені наперед стандартним чином. Стандартне тестування називають найбільш значним соціальним внеском сучасної психології.

Будь-який тест, в якому він подається ідентично для всіх тестованих, являє собою стандартизований тест. Стандартизовані тести — це не обов'язково тести з високими ставками, тести з обмеженим часом, або тести з декількома варіантами відповідей. Протилежністю стандартизованого тестування є нестандартизоване тестування, в якому або істотно різні випробування наведені до різних тестованих, або той же самий тест призначається при істотно різних умовах (наприклад, одній групі дозволено набагато менше часу, щоб завершити випробування, ніж наступній групі) або оцінюватися по-різному (наприклад, та ж відповідь вважається правильною для одного студента, але неправильною для

іншого студента). Підрахунок очок за розгорнуту відповідь проводить людина, яка має вищий кваліфікаційний рівень, ніж тестований працівник.

Стандартизовані тести сприймаються як більш справедливі, ніж не-стандартизовані тести, тому що кожен отримує той же самий тест і ту ж систему класифікації. Це є більш справедливим і більш об'єктивним, ніж система, в якій деякі студенти отримують більш простий тест і інші отримують більш важке випробування. Узгодженість також дозволяє більш надійне порівняння результатів у всіх тестованих, тому що кожен приймає один і той же тест.

Актуальність даної теми полягає у тому, що компаніям потрібно перевіряти рівень працівників, які в них працюють не витрачаючи багато часу на різні багато часові опитування. Також, даний web-додаток полегшить та спростить процедуру співбесіди та визначення рівня працівника.

Метою є розробка web-додатку, яким будуть користуватись компанії. Вони зможуть з легкістю додати в додаток свої бібліотеки питань та відповідей до них, та використовувати їх для опитування різного рівня працівників. Як результат кожен додаток буде мати свою базу даних, та свої унікальні питання, тому що як ми знаємо, рівень оцінювання працівників у кожного свій.

У всьому світі з ринковою економікою існував ринок праці програмістів, який відрізнявся високою конкуренцією. Отже, він завжди був здатний регулювати себе сам, не кажучи вже про те, що процес ліцензування — досить коштовна процедура. Хто має платити за неї? Що робити з тими, хто її не пройшов? Є й багато інших проблем, пов'язаних із проблемою ліцензування.

Об'єктом дослідження курсової роботи є інтерфейс системи попереднього тестування працівників перед основною співбесідою «Test system». Забезпечення користувача всім необхідним інструментарієм у форматі web-додатку.

Предметом дослідження даної курсової роботи є взаємодія користувача із інтерфейсом веб-додатку, можливість створення нових тестів, перегляд результатів вже пройдених та, власне, самого тестування.

1 ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Характеристика предметної галузі

Завжди цінність людини в суспільстві визначалася кількістю та якістю її знань. Особи із вищим рівнем знання як правило займали високе положення в суспільстві і мали багато привілеїв. Ще з часів первісних общин і до сьогодні це правило залишається істинним. Змінюються лише привілеї та суть користі людини для суспільства.

Розвиток сфери ІТ розпочався з 1960-х років ХХ століття, разом із появою і розвитком перших інформаційних систем та електронно-обчислювальних машин. З часом бізнес все більше зацікавлювався даною сферою і цим самим лише поширював технології ІС та ЕОМ, а також зацікавленість ними суспільством. Наприкінці 1990-х років ХХ століття інвестиції в інфраструктуру та сервіси Інтернету спричинили бурхливий розвиток даної сфери. На сьогодні кожного року відкриваються сотні нових компаній. Кількість працівників в даних компаніях зростає надзвичайно швидко. Але такий активний розвиток тягне за собою ускладнення процесів управління персоналом та підвищення витрат на нього.

При прийомі на роботу нових працівників з кожним із них потрібно провести співбесіду і визначити рівень технічної підкованості особи, щоб прийняти рішення про її найм та позицію в компанії, в разі успішного рішення. Зазвичай в середньому до 70% кандидатів відсіюються на етапі технічної співбесіди в компанію. Кожна така співбесіда – це фінансові та трудові витрати, які забирають на себе частину ресурсів. Дані ресурси можна було б спрямувати на інші проекти, які затребувані на даний момент.

Аналогічна ситуація склалася і в сфері підвищення рівня кваліфікації працівника. Молодших спеціалістів завжди ведуть старші спеціалісти, які їх навчають та допомагають розвиватися професійно. Заробітна плата працівника напряму залежить від його знань та навичок, а їх у свою чергу відображає його рівень кваліфікації. Тому підвищення останньої є невід'ємною частиною розвитку працівника. Дуже часто співбесіда на підвищення рівня кваліфікації

призначається на вимогу самого працівника, але, на жаль, не рідкістю є ситуації, коли він ще не готовий до неї. Тому ми тут також маємо даремні витрати дорогоцінних ресурсів компанії.

Щоб уникнути подібних ситуацій найкращим варіантом є проведення попереднього стандартизованого тестування, що виявить чи має кандидат необхідний рівень технічних знань для продовження співбесіди із інтерв'юером.

Проведення попереднього автоматичного стандартизованого тестування з використанням програмних комплексів та систем зменшує кількість витрачених людино-годин активних працівників, що в свою чергу веде до зменшення потреби в такому типіві робочих. Такий наслідок веде до загального зменшення витрат на підтримку штату а також до зменшення супутніх витрат в суміжних сферах. Також дане тестування гарантує, що до рекрутера потраплять лиш ті, хто володіє необхідними знаннями.

1.2 Огляд існуючих аналогів

1.2.1 Первый тестовый

Посилання: <http://testme.pro/>

Розробник: ООО "Облака-2010"

Архітектура: клієнт-сервер, сервер знаходиться у власності розробника

Мови реалізації:

- Front-end: HTML, CSS, JavaScript
- Back-end: PHP

Перелік функцій та характеристик:

- реєстрація;
- авторизація;
- створення тесту;
- база питань;
- списки співробітників;
- звітність.

На рисунку 1.1 зображено інтерфейс головної сторінки сайту даного програмного забезпечення «Первый тестовый».

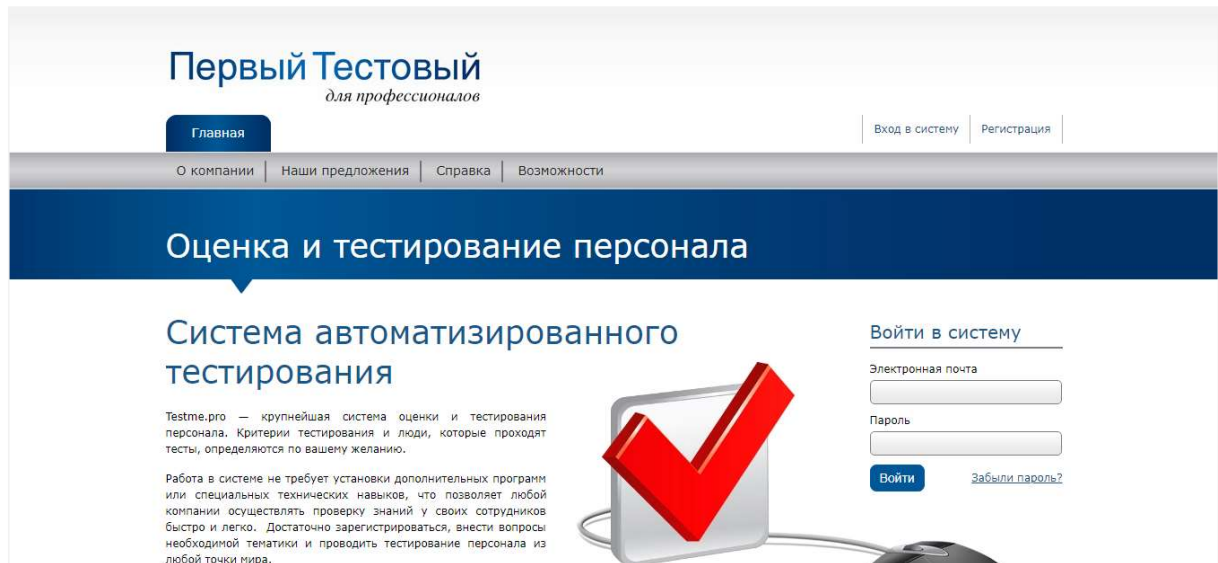


Рисунок 1.1 – Головна сторінка сайту продукту «Первый тестовый»

Опис: даний продукт має морально застарілий інтерфейс, який ґрунтується на застарілих бібліотеках та має велику кількість вразливостей в системі захисту. Вимагає постійного продовження підписки на використання продукту. Back-end даної системи знаходиться на серверах компанії виробника, що збільшує ризик витоку конфіденційних даних та передачу їх третім особам. Також, за Політикою конфіденційності даного сервісу, він збирає персональні дані працівників компаній. До його переваг можна віднести систему звітування по пройденим тестам. Так як сервер знаходиться під управлінням компанії розробника, то експлуатація даної системи є спрощеною та такою, що не вимагає затрати великої кількості додаткових ресурсів.

1.2.2 Quizful

Посилання: <http://www.quizful.net/>

Розробник: Forteko software development

Архітектура: клієнт-сервер

Мови реалізації:

- Front-end: HTML, CSS, JavaScript
- Back-end: PHP

Перелік функцій та характеристик:

- реєстрація;

- авторизація;
- створення тесту;
- створення власних питань;
- рейтингова таблиця;
- коментарі користувачів до питань.

На рисунку 1.2 зображено інтерфейс головної сторінки даного програмного забезпечення «Quizful».

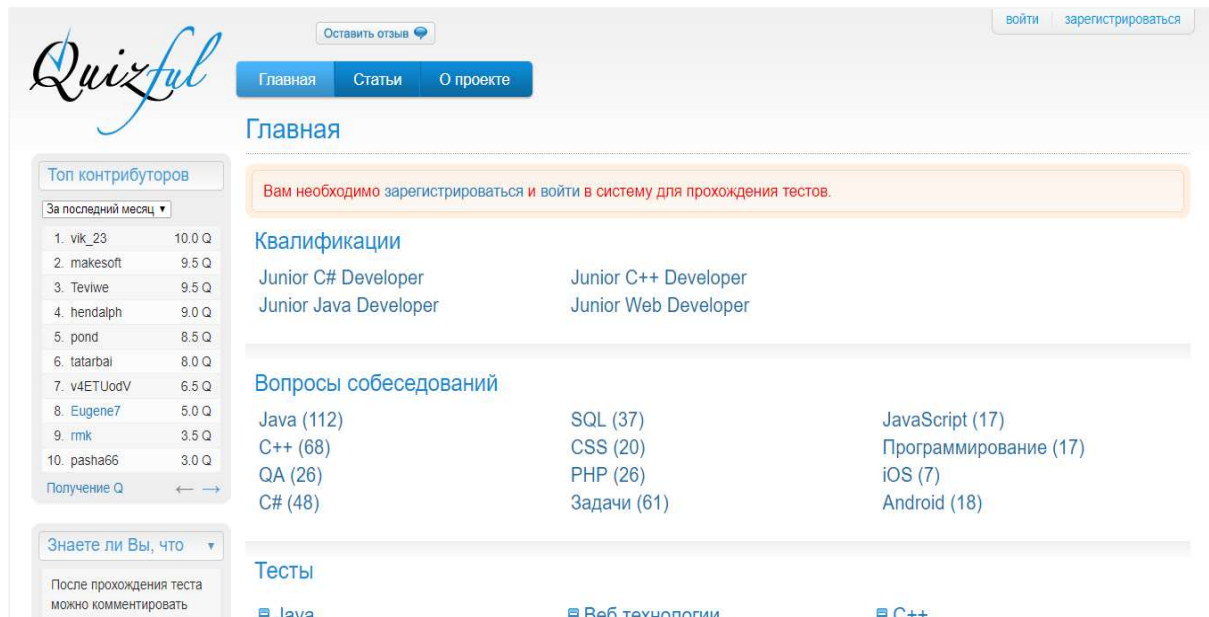


Рисунок 1.2 – Головна сторінка сайту «Quizful»

Опис: даний продукт не можливо використати для корпоративних потреб. Він призначений для персонального тестування з метою виявлення прогалин у своїх знаннях. Система коментарів до питань дозволяє користувачам спільно шукати відповіді на складні завдання. Але водночас дана особливість цього продукту є його і мінусом, тому що в корпоративному програмному забезпеченні така деталь повинна бути відсутньою – всі питання підібрані й перевірені старшим персоналом і не містять вимог вищих, ніж цього вимагає статут. Також тут градація відповідно до рівня кваліфікації працівника доступна лише для кількох мов програмування. Питання створюються тут самими користувачами для інших користувачів. Також на цьому сайті доступна рейтингова таблиця, де можна побачити прогрес найуспішніших осіб, які проходять або пройшли тестування.

1.2.3 InTester

Посилання: <https://intester.com/>

Розробник: intester.com

Архітектура: клієнт-сервер

Мови реалізації:

- Front-end: HTML, CSS, JavaScript
- Back-end: PHP

Перелік функцій та характеристик:

- реєстрація;
- авторизація;
- створення тесту;
- рейтингова таблиця;
- автоматична генерація;
- часовий ліміт на виконання одного завдання.

На рисунку 1.3 зображено інтерфейс головної сторінки даного програмного забезпечення «InTester».

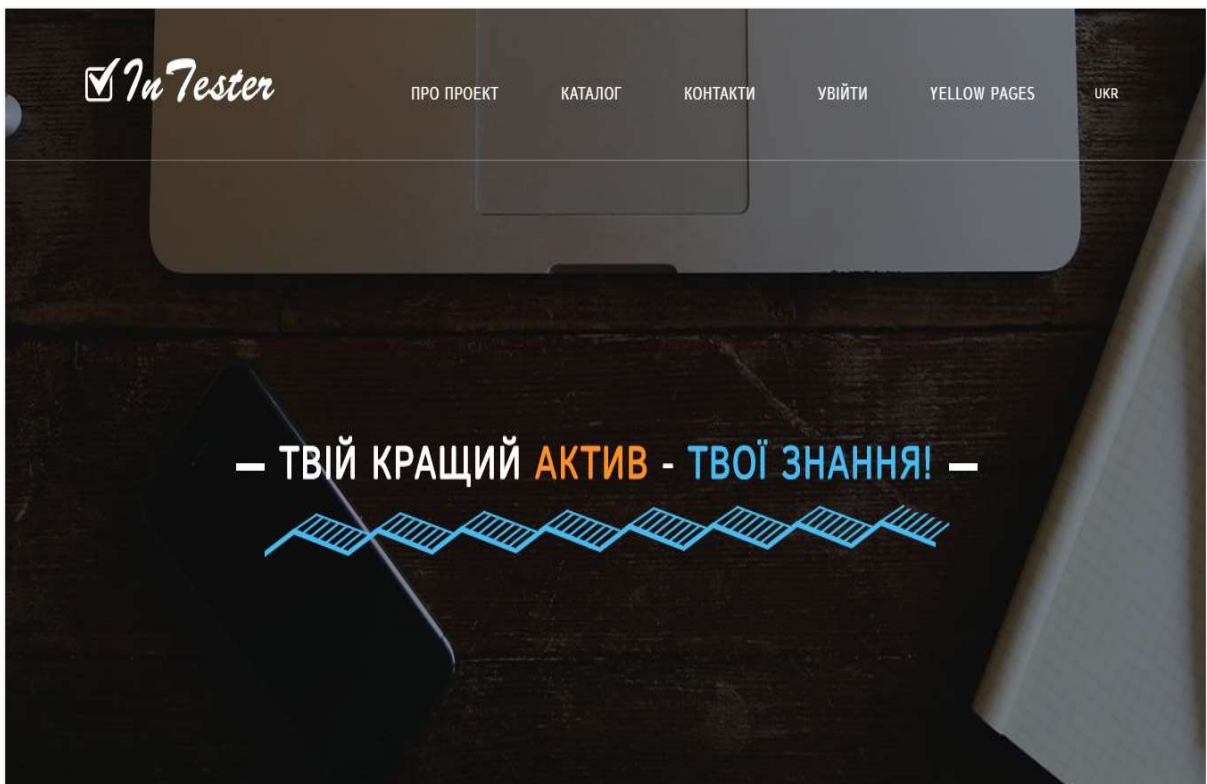


Рисунок 1.3 – Головна сторінка сайту «InTester»

На рисунку 1.4 зображений перелік мов програмування, по якому можна пройти тестування на даному сайті. Перелік невеликий, а частина взагалі не являється мовою програмування.

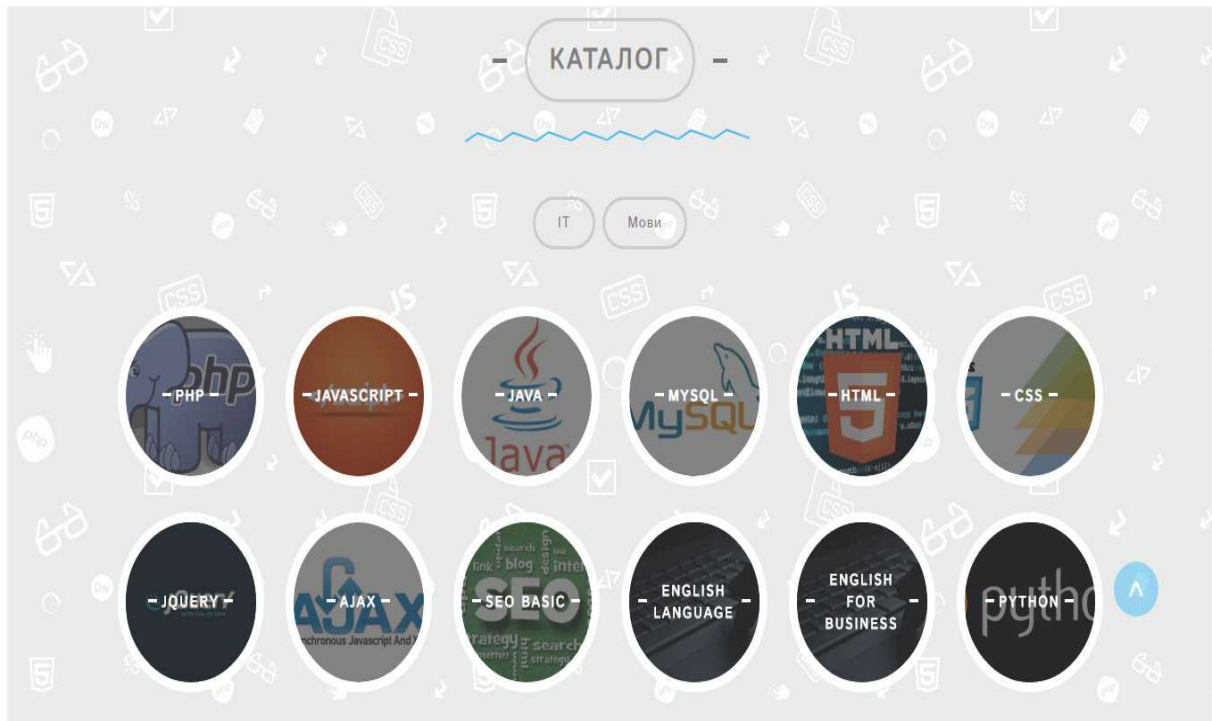


Рисунок 1.4 - Перелік мов програмування для тестування сайту «InTester»

Опис: даний продукт не можливо використати для корпоративних потреб. Він призначений для персонального тестування з метою виявлення прогалин у своїх знаннях прогалин. Перелік мов і технологій, по яким можна проходити тестування, невеликий. На сайті присутня рейтингова таблиця для тестованих, щоб можна було бачити свій прогрес. На сайті є встановлений часовий ліміт на виконання кожного завдання, що збільшує швидкість його проходження та змушує особу мислити лише в рамках тесту і не відволікатися на зайві речі. Приємним плюсом є наявність україномовного інтерфейсу. Але попри всі свої переваги даний програмний продукт не призначений для внутрішнього корпоративного використання.

2 ПОСТАНОВКА ЗАДАЧІ. ТЕХНІЧНЕ ЗАВДАННЯ

2.1 Вступ

Дане технічне завдання поширюється на розробку веб-системи «Система тестування рівня кваліфікації працівника», яка призначена для використання в межах корпоративної системи для тестування кандидатів на якусь посаду або кандидатів на підвищення кваліфікаційного рівня.

2.2 Загальна характеристика системи

2.2.1 Сфера застосування

Система буде використовуватись в корпоративному секторі галузі ІТ для визначення рівня кваліфікації та наявності мінімального набору необхідних знань працівників. Система призначена для людей, якій займаються підбором персоналу в компанію, для людей, які є менторами для молодших спеціалістів, та власне для кандидатів на позицію в компанії або на підвищення кваліфікаційного рівня. Це можуть бути:

- рекрутери;
- інтерв'юери;
- superior'и;
- Senior Developer або Middle Developer, які є менторами.

2.2.2 Функції системи

- вхід в систему;
- перегляд основної статистичної інформації системи;
- перегляд списку всіх пройдених тестів;
- перегляд детального звіту по конкретному пройденому тесті конкретного працівника;
- перегляд правильності / не правильності відповіді працівника на питання;
- пошук працівника в системі;
- вибір рівня тестування;
- створення нового тесту для працівника;
- проходження тесту в окремому вікні за окремим посиланням;

- відображення сповіщень про помилки або збої в системі, успішне / не успішне завершення якоїсь дії;
- валідація даних, які вводить працівник.

2.2.3 Джерело початкових даних

Дані на сторінці Sign In:

- поле Email є обов'язковим, а також підлягає перевірці на правильність введеної електронної скриньки;
- поле Password є обов'язковим, а також в ньому приховуються введені символи від сторонніх спостерігачів і небажаних третіх осіб.

Дані на сторінці New Survey:

- поле Employee є обов'язковим, присутня фільтрація від сторонніх символів;
- поле Degree Survey також є обов'язковим, наявна лише можливість вибору із вже зазначених параметрів, а свої дані вводити заборонено.

2.2.4 Результати, вихідні дані

Коли адміністратор заходить на головну сторінку, то йому відображається основна статистична інформація системи, де зазначено кількість працівників, які належать до тієї чи іншої внутрішніх груп.

На сторінці із списком всіх створених тестів виводяться їх основні дані, а також статус самого тесту. При переході за спеціальним посиланням можна подивитися весь список питань та відповідей працівника в цьому тесті, а також їх правильність. Якщо тест знаходиться на стадії виконання, то адміністратор буде сповіщений відповідним повідомленням у верхній частині екрану та перенаправлений на сторінку із списком створених тестів.

На сторінці створення нового опитування, після заповнення необхідних полів та відправки даних на сервер, користувачеві будуть відображені згенеровані посилання для проходження тесту, які можна скопіювати напямую або натиснувши спеціальну кнопку. На цій же сторінці при введенні імені працівника відображається випадаючий список із всіма працівниками, які підпадають під даний запит пошуку. Це зручно, якщо адміністратор не пам'ятає

повного імені працівника.

На сторінці з тестом виводиться весь список із 20 питань, які поділяються на два типи: питання з вибором правильної відповіді шляхом відмічення галочок біля правильних варіантів і питання з розгорнутою відповіддю, яку працівник повинен ввести особисто в спеціальне поле. Якщо тест вже був пройдений, то працівник отримає відповідне сповіщення про це та буде негайно припинене його проходження.

2.3 Вимоги до обчислювального середовища

2.3.1 Склад та конфігурація апаратних засобів

Мінімальна конфігурація:

- тип процесора: Intel Pentium II Xeon або вище, AMD Athlon Argon або вище;
- об'єм оперативної пам'яті: 512 Мб і більше;
- екран: 800х600 пікселів з базовим відео адаптером.

Рекомендована конфігурація:

- тип процесора: Intel Core i3 або вище, AMD Phenom Agena або вище;
- об'єм оперативної пам'яті: 2048 Мб і більше;
- екран: 1024х768 пікселів з базовим відео адаптером.

Об'єм пам'яті жорсткого диску на сервері залежить від об'єму інформації, яка може збільшуватись та зменшуватись.

2.3.2 Програмні засоби

Для роботи зі даною системою необхідний інтернет браузер. Операційна система не є важливим фактором, так як наразі браузери з високими показниками підтримки сучасних стандартів мов присутні на всіх платформах. Але попри це для комфортного користування додатком рекомендується використовувати наступні браузери:

- Google Chrome версії 63 і вище;
- Mozilla Firefox версії 67 і вище;
- Apple Safari версії 9 і вище;
- Opera версії 30 і вище.

Так як дана робота має на меті створення інтерфейсу даної системи з використанням фреймворку Angular, то після компіляції написаного проекту він перетворюється в набір .html, .css та .js файлів. Тому до серверної частини є єдина вимога – а саме наявність файлового сховища типу FTP або будь-якого схожого типу з можливістю налаштування прав доступу до окремих файлів. Такий спрощений перелік вимог стає можливим завдяки тому, що front-end системи та її back-end розділені і можуть знаходитися на незалежних серверах.

2.3.3 Режим роботи

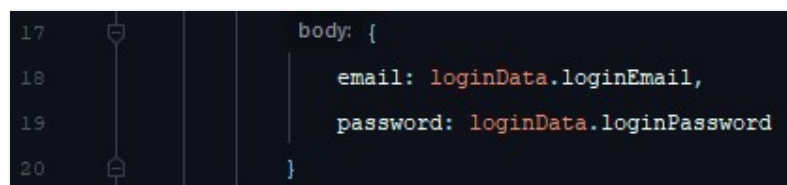
Розроблювана система технічно буде працювати в постійній основі в режимі 24/7. Програмно система матиме два режими роботи:

- адміністративний – доступ до управління системою, доступ до створення нових тестів, доступ до перегляду відповідей працівників та їх оцінки;
- користувацький – доступ до проходження тесту.

2.4 Зв'язок із зовнішнім середовищем

2.4.1 Формати і протоколи обміну для вхідних даних

Для відправки даних на сервер створюється JavaScript об'єкт із полями, в які прописуються необхідна інформація. На рисунку 2.1 зображено тіло запиту, яке містить в собі дані для входу в систему адміністратором.



```
17  body: {  
18    email: loginData.loginEmail,  
19    password: loginData.loginPassword  
20  }
```

Рисунок 2.1 – Тіло запиту на вхід до системи адміністратором

Таким чином дані на сервер відправляються у вигляді JSON об'єкта. На сервері відбувається автоматичне зворотне перетворення даних в об'єкт JavaScript для подальшої роботи з ними.

Відправка будь-яких даних на сервер здійснюється за допомогою методу POST. Це забезпечує деякий рівень захисту даних від несанкціонованого перехоплення. За допомогою методу GET на сервер відправляються лиш запити на перевірку активності сесії адміністратора в даний момент, тобто ніякі дані,

окрім токена власне самої сесії, не містяться в запиті.

2.4.2 Формати і протоколи для вихідних даних

Дані від серверу приходять у вигляді JSON об'єкта, який відразу перетворюється на JavaScript об'єкт із полями за допомогою фреймворку Angular. Будь-яка передача даних від сервера на Front-end додатку в обов'язковому порядку повинна бути описана з допомогою інтерфейсів, що виконують роль шаблонів. На рисунку 2.2 зображений код інтерфейсу SurveyDetails, що описує об'єкт загальних даних тесту. Ці дані відображаються в таблиці на сторінці Surveys list.

```
22  
23 export interface SurveyDetails {  
24     survey_id: number;  
25     firstname: string;  
26     lastname: string;  
27     status: string;  
28     description: string;  
29 }  
30
```

Рисунок 2.2 – Код інтерфейсу SurveyDetails

Це забезпечує системі захист від помилок, пов'язаних з типами змінних, відносну стабільність роботи. Всі дані, що не підпадають під опис інтерфейсу будуть просто ігноруватися системою. Також це правило розповсюджується і на внутрішнє спілкування між компонентами Front-end'у додатку за допомогою блоків даних. Кожен такий блок являє собою JavaScript об'єкт із полями.

Користувачеві дані відображаються зазвичай блочним або табличним методом. На сторінці Home основна статистична інформація системи відображається за допомогою блочного методу – це проілюстровано на рисунку 2.3.

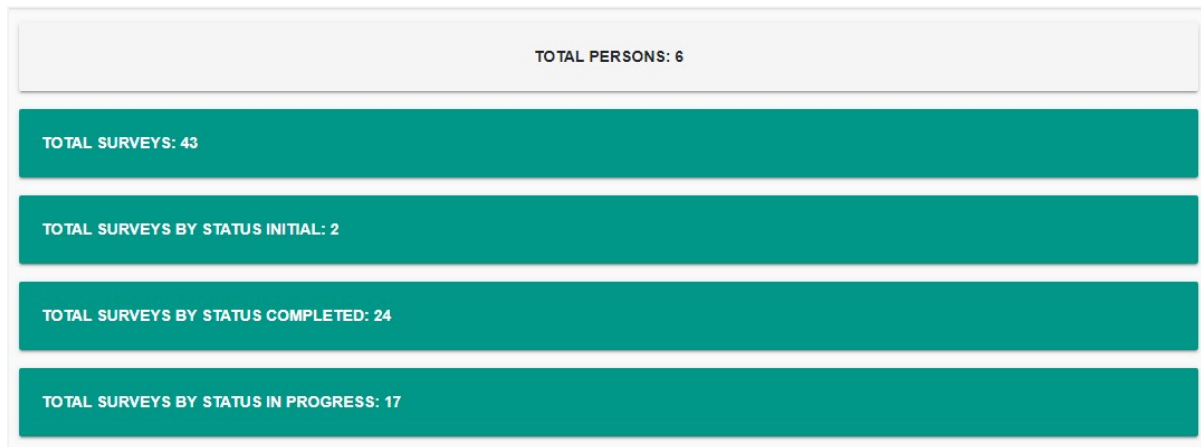


Рисунок 2.3 – Блочний метод відображення основної статистичної інформації

На сторінці Surveys list відображається таблиця з даними про всі створені тести за допомогою табличного методу інформації – це проілюстровано на рисунку 2.4.

ID	First Name	Last Name	Status	Description	Show info
1	vlad	admin	in progress	1 junior	SHOW
35	Alexis	Ortiz	completed	5 senior	SHOW
68	joker	.	completed	8 middle-1	SHOW
69	joker	.	in progress	8 junior	SHOW
2	vlad	admin	completed	1 middle-1	SHOW
3	vitalii	not admin	in progress	2 junior	SHOW
4	vitalii	not admin	in progress	2 middle-1	SHOW

Рисунок 2.4 – Табличний метод відображення інформації

2.4.3 Формати та правила зміни керуючих параметрів

Доступ до створення та перегляду і оцінки тестів мають тільки адміністратори системи. Прості працівники мають доступ лише до проходження тесту. Хоча така рівнева ієрархія користувачів не перешкоджає тому, щоб якийсь із адміністраторів пройшов тест в даній системі. Це забезпечується автоматичною генерацією та перевіркою тесту на серверній частині системи, куди доступ мають лише персонал найвищих класів.

2.5 Якість системи

2.5.1 Виконання стандартів та узгодження

Проектування та реалізація даного програмного продукту повністю відповідає стандарту ISO 9126:2001. Стандарт ISO/IEC 9126 регламентує зовнішні і внутрішні характеристики якості. Перші відображають вимоги до функціонування програмного продукту. Для кількісного встановлення критеріїв якості, за якими буде здійснюватися перевірка і підтвердження відповідності ПЗ заданим вимогам, визначаються відповідні зовнішні вимірювані властивості (зовнішні атрибути) ПЗ, метрики (наприклад, час виконання окремих компонентів), діапазони зміни значень і моделі їх оцінки. Метрики використовуються на стадії тестування або функціонування і називаються зовнішніми метриками. Вони являють собою моделі оцінки атрибутів.

Внутрішні характеристики якості і внутрішні атрибути ПЗ використовуються для складання плану досягнення необхідних зовнішніх характеристик якості продукту. Для квантифікації внутрішніх характеристик якості застосовують внутрішні метрики, як інструмент перевірки відповідності проміжних продуктів внутрішнім вимогам до якості, які формуються на процесах, що передують тестуванню.

Зовнішні і внутрішні характеристики якості відображають властивості самого ПЗ (працюючого або не працюючого), а також погляд замовника і розробника на таке ПЗ. Безпосереднього кінцевого користувача ПЗ цікавить експлуатаційна якість ПЗ – сукупний ефект від досягнення характеристик якості, що виміряється строком результату, а не властивістю самого ПЗ. Це поняття ширше, ніж будь-яка окрема характеристика (наприклад, зручність використання або надійність).

Множина характеристик і атрибутів за стандартом ISO 9126:2001, яким повинен відповідати програмний продукт та його розробка:

- функціональність (functionality) – здатність ПЗ в певних умовах вирішувати задачі, потрібні користувачам;

- надійність (reliability) – здатність ПЗ підтримувати визначену працездатність у заданих умовах;
- зручність використання (usability) або практичність – здатність ПЗ бути зручним у навчанні та використанні, а також привабливим для користувачів;
- продуктивність (efficiency) або ефективність – здатність ПЗ при заданих умовах забезпечувати необхідну працездатність стосовно виділюваного для цього ресурсам;
- зручність супроводу (maintainability) – зручність проведення всіх видів діяльності, пов'язаних із супроводом програм;
- переносимість (portability) – здатність ПЗ зберігати працездатність при перенесенні з одного оточення в інше, включаючи організаційні, апаратні й програмні аспекти оточення.

2.5.2 Можливість перенесення

Дана система призначена для використання у внутрішній корпоративній мережі компанії. Але за потреби вона може бути легко виведена в мережу Інтернет.

Для перенесення її на простенький вебхостинг знадобляться прості інструменти для роботи із FTP, наприклад, Filezilla. Більш сучасніші та захищеніші хостинги дозволяють і підтримують завантаження файлів системи на власні сервери за допомогою системи контролю версій GitHub. Такий спосіб має багато переваг як, наприклад, можливість контролю всього процесу розробки системи.

При перенесенні системи слід враховувати її апаратні та програмні вимоги для середовища виконання.

2.5.3 Надійність функціонування

Стабільність системи залежить від рівня та кількості неліквідованих дефектів, помилок та можливостей системи реагувати на їх появу так, щоб це не відображалось в показниках надійності.

Для забезпечення системи від аномальної поведінки на деякі аспекти її роботи накладаються обмеження. Це гарантує відсутність помилок, пов'язаних із некоректним введенням даних.

При виконанні всіх перелічених програмних та апаратних вимог до системи (п. 2.3.1-2.3.2), а також дотримання керівництва користувача – гарантується стабільна робота системи.

2.6 Документація системи

2.6.1 Перелік та вимоги до опису посібника керівника

Розроблювана система повинна мати головне меню, де будуть всі статичні посилання системи. Елементи системи (кнопки, посилання, поля введення даних, тощо) повинні мати підкази користувачу та мати вигляд, який дозволить чітко зрозуміти призначення того чи іншого елементу. Також всі поля для введення даних повинні мати валідацію відповідно до типу та вигляду даних, які вони приймають.

Вимоги до опису посібника керівника системи:

- опис основних вікон системи;
- опис полів введення даних;
- опис роботи функцій валідації полів введення даних;
- опис кнопок системи;
- опис посилань системи;
- інформація про розробника системи;
- призначення системи;
- опис функцій системи.

2.6.2 Вимоги до оформлення вихідних кодів

При написанні коду на мові JavaScript (а також мові TypeScript) потрібно дотримуватися ряду простих правил, які полегшать роботу з кодом та допоможуть уникнути зайвої витрати часу на його постійний аналіз, з метою розуміння його роботи.

1 Об'єкти:

- створення об'єкта – для створення об'єкта використовуйте фігурні дужки. Не створюйте об'єкти через конструктор `new Object`;
 - зарезервовані слова – не використовуйте зарезервовані слова в якості ключів об'єктів. Вони не будуть працювати в IE8;
 - ключові слова – не використовуйте ключові слова (в тому числі змінені). Замість них використовуйте синоніми;
 - створення об'єкта на 3 і більше елементів – при створенні об'єктів, так само як і масивів, що містять велику кількість властивостей (елементів), і тим самим утворюють рядки, довжиною понад 20 символів, необхідно виконувати ряд умов:
 - відкриваюча дужка розташовується на тому ж рядку;
 - кожна властивість оформляється на новому рядку;
 - пробіл після двокрапки;
 - закриваюча дужка розташовується на новому рядку.
- 2 Змінні – для іменування змінних використовуйте іменники англійською мовою (не трансліт!). Ім'я змінної має бути осмисленим. Ім'я може складатися з букв, цифр, символів \$ і _, воно не повинно починатися з цифри.
- 3 Функції – іменування функцій – ім'я функції має бути дієсловом англійською мовою або починатися з нього. Для імен, що складаються з декількох слів, використовуйте camelCase.
- 4 Відступи:
- горизонтальні відступи – відступ при вкладеності – 4 пробіли на кожен рівень вкладеності;
 - вертикальні відступи – між логічними блоками (циклами, функціями і т.д.) слід залишати порожній рядок. Це робить код більш читабельним. Уникайте блоків коду більше, ніж 9 рядків поспіль.
- 5 Пробіли:
- використовуйте пробіли між параметрами і не використовуйте між ім'ям функції і дужкою;

- при створенні анонімної функції необхідно використовувати пробіл перед дужкою;
 - використовуйте пробіли навколо операторів.
- 6 Дужки – відкриваюча фігурна дужка розташовується на тому ж рядку. Перед дужкою пробіл. Закриваюча дужка розташовується на новому рядку.
- 7 Лапки – завжди в коді скрипта використовуйте одинарні лапки, якщо не вимагається іншого. Подвійні лапки допустимі, якщо в цьому ж рядку необхідно використовувати апостроф (') або ж одинарні лапки для інших цілей.
- 8 Крапка з комою – в кінці виразу обов'язкова крапка з комою.
- 9 Коментарі:
- для пояснення коду використовуються коментарі. Коментарі не виконуються інтерпретатором JavaScript;
 - однорядкові коментарі починаються з подвійного слеша //. За ним обов'язково повинен йти пробіл;
 - багаторядкові коментарі розташовуються між /* і */. За символом початку коментаря обов'язково повинен йти пробіл. Символ кінця коментаря розташовується на новому рядку.

При написанні коду на мові розмітки HTML потрібно дотримуватися ряду простих правил, які полегшать роботу з кодом та допоможуть легко розуміти структуру вебсторінки.

1 Синтаксис:

- чотири пробіли в якості відступу – відступ допомагає візуально оцінити структуру документа і швидко перемикається між його фрагментами. Розмір відступу налаштовується в редакторі. Також у багатьох редакторах можна включити відображення символів пробілів і перетворення відступів;

- теги та атрибути записуються в нижньому регістрі – символи нижнього регістру не привертають до себе великої уваги, і вам легше буде знайти потрібний;
 - окремі логічні блоки відокремлюються символом нового рядка – це полегшує роботу з кодом і візуально створює структуру документа;
 - використовуйте тільки подвійні лапки – при написанні значень атрибутів використовуйте тільки подвійні лапки;
 - ніколи не ставимо пробілів перед і після символу (=) – оскільки права частина безпосередньо відноситься до лівої, то атрибут і його значення повинні бути написані без пробілів;
 - між атрибутами один пробіл – ніколи не використовуємо переносів рядків між атрибутами, тільки один пробіл. Перенесення рядків прийнятні в css-документах, але не в html-розмітці. Пишіть всі атрибути елементів в рядок;
 - поодинокі теги без закриваючого слеша – не використовуйте закриваючого слеша в одиночних тегів (,
 та інші). Цей елемент є пережитком минулого, коли HTML був більш суворим, а браузер не вміли розпізнавати і виправляти помилки.
- 2 Валідність – HTML-документ повинен проходити перевірку на валідність. Для перевірки використовуйте валідатор.
 - 3 !DOCTYPE – першим рядком в HTML-документі повинен йти актуальний doctype. Це необхідно щоб браузер вірно відображав сторінку. Це забезпечить однакове відображення у всіх сучасних браузерах.
 - 4 Кодування – кодування символів в html-документі завжди повинно бути вказано явно. Це забезпечить правильне відображення тексту. Кодування UTF-8 підходить завжди, за рідкісним винятком.
 - 5 Підключення стилів – файли зі стилями підключаються всередині тега <head> за допомогою <link>. Атрибут type для тега <link> вказувати не потрібно, так як значення text/css встановлюється за замовчуванням.

- 6 Підключення скриптів – скрипти при завантаженні блокують відображення вмісту сторінки. З цієї причини слід підключати їх в самому кінці html-документа. При підключенні скриптів в тегові `<script>` атрибут `type` вказувати не потрібно, так як значення `text/javascript` встановлюється за замовчуванням.
- 7 Атрибути і їх порядок – у HTML-елементів атрибут `class` пишеться першим. Однакове написання допомагає легше читати код і швидше розбиратися в призначенні блоків по класах. Решта атрибути можуть бути розставлені в будь-якому порядку, але теж однаково для схожих елементів.
- 8 Логічні атрибути – для логічних атрибутів (наприклад, `checked`, `disabled`, `required`) значення не вказується, а самі атрибути вказуються останніми і в однаковій послідовності в усьому документі.
- 9 Підписи `input` – для поліпшення досвіду користувача, при натисканні на підпис поля, саме поле повинно активуватися. Для цього поле форми зв'язується з описом за допомогою ідентифікатора і атрибута `for` тега `<label>`.
- 10 Атрибут мови – для елемента `<html>` в атрибуті `lang` повинен вказуватися використовувана мова документа. Це допомагає системам перекладу визначити, які мовні правила використовувати.

При написанні коду на мові каскадних стилів CSS потрібно дотримуватися ряду простих правил, які полегшать роботу з кодом та допоможуть уникнути помилок при створенні стильового оформлення елементів.

1 Синтаксис:

- у кінці рядка повинна стояти крапка з комою – після пари (властивість: значення) обов'язково ставиться крапка з комою. Без цього знаку не працюватиме правило в цьому рядку і наступне за ним;
- для відступів всередині правил використовуйте чотири пробіли – правила, які перераховуються всередині фігурних дужок, повинні

мати відступ від початку рядка. Для цього використовуйте 4 пробіли. Це дозволяє відразу бачити блоки властивостей, що відносяться до одного селектора;

- значення кольорів не скорочується – якщо колір заданий в шістнадцятковій системі, то значення не скорочується, а пишеться повний код з усіх шести символів. Для запису використовуйте малі літери, наприклад, #e3e3e3;
- все пишеться літерами в нижньому регістрі – всі назви тегів і властивості пишуться малими літерами;
- нулі не пропускаються – якщо число дробове і починається з нуля, то він не пропускається (наприклад, .5 замість 0.5);
- використовуйте подвійні лапки – у стилях завжди без винятку використовуються подвійні лапки. Якщо лапки не обов'язкові, то вони пишуться все одно;
- пробіл після двокрапки – у правилах після двокрапки ставиться один пробіл (top: 10px;). При цьому перед двокрапкою пробіл не потрібен;
- пробіли після коми в кольорах – після ком всередині значень rgb (), rgba (), hsl (), hsla () або rect () пишуться пробіли. Це покращує читабельність;
- пропуск до і після комбінатора – між селекторами до і після комбінатора (наприклад, p > a) ставиться один пробіл;
- кожна властивість з нового рядка – одна властивість – один рядок. Кожне оголошення в правилі пишеться на новому рядку;
- пропуск перед фігурною дужкою – після селектора і перед відкриваючою фігурною дужкою ставиться один пропуск. Після відкриваючої дужки запис йде з нового рядка;
- закриваюча фігурна дужка на новому рядку – закриваюча фігурна дужка після набору властивостей пишеться на новому рядку і без відступу. Вона повинна бути на одному рівні з селектором. Наступне після цього правило відділяється символом нового рядка;

- опускайте одиниці виміру – одиниці виміру не потрібно писати там, де без них можна обійтися. Наприклад, border: 0.
- 2 Порядок властивостей – порядок логічно пов'язаних властивостей повинен бути згрупований наступним чином:
- позиціонування;
 - блокова модель;
 - типографіка;
 - оформлення;
 - анімація;
 - інше.

Позиціонування слід писати першим, оскільки воно впливає на становище блоків в потоці документа. Блокова модель визначає розміри і розташування блоків і йде наступною. Всі інші правила, які змінюють вигляд внутрішніх частин блоків і не впливають на інші блоки, йдуть в останню чергу. Згруповані властивості в правилі відокремлюються один від одного порожнім рядком. Порядок оголошення докладних правил, таких як font-size, font-family, line-height, повинен відповідати порядку в скороченій версії правила. У разі використання детальних і скорочених правил, першою повинна йти скорочена версія.

- 3 Імена класів – імена класів пишуться малими літерами, між декількома словами використовується дефіс (але не знак нижнього підкреслення або camelCase). Дефіси служать роздільниками у взаємозалежних класах (наприклад, .button і .button-cancel). Імена повинні бути такими, щоб по них можна було швидко визначити, якого елемента на сторінці заданий клас: уникайте скорочень (єдиний виняток - .btn для кнопок), але не робіть їх занадто довгими (не більше трьох слів). Для іменування класів використовуються англійські слова і терміни. Не використовуйте трансліт для назви класів і атрибутів.
- 4 Правило @import – це правило працює повільніше, ніж тег <link>. У стилях @import використовувати не бажано.

5 Варіанти шрифту – альтернативні варіанти шрифту і тип сімейства вказуються в кінці перерахування значень font-family. У разі використання нестандартних шрифтів обов'язково вказувати альтернативний веб-безпечний шрифт і тип сімейства, щоб в разі відсутності нестандартного шрифту в системі зміни зовнішнього вигляду сторінки були мінімальні. Нестандартний і альтернативний шрифти повинні бути одного типу. Порядок шрифтів наступний:

- нестандартний шрифт;
- веб-безпечний шрифт;
- тип сімейства шрифту.

2.6.3 Вимоги до опису структури і формати даних

На етапі розробки система повинна мати формальні моделі свого існування:

- DFD – діаграма потоків даних. Вона показує структурний аналіз системи, описує зовнішні по відношенню до системи джерела даних, логічні функції, потоки та сховища даних до яких є доступ;
- діаграма варіантів використання;
- IDEF0 – методологія функціонального моделювання і графічна нотація, призначена для формалізації і опису бізнес-процесів. Відмінною особливістю IDEF0 є її акцент на підпорядкованість об'єктів;
- діаграма взаємодії – діаграма, на якій для деякого набору об'єктів на єдиній тимчасовій осі показаний життєвий цикл будь-якого певного об'єкта і взаємодія акторів (дійових осіб) системи в рамках якого-небудь певного прецеденту (відправка запитів і отримання відповідей);
- діаграма декомпозиції – система розбивається на підсистеми і кожна підсистема описується окремо.

2.7 Вибір програмно-технічних засобів

В якості мови програмування серверної частини була вибрана мова JavaScript [14]. Серверна частина була реалізована на програмній платформі NodeJs. В якості бази даних було вибрано об'єктно орієнтовану БД PostgreSQL.

Її об'єктна модель як ніяка інша краще підходить для реалізації поставлених завдань.

Для написання графічного інтерфейсу даної системи було обрано фреймворк Angular [1]. Для створення набору стилізованих елементів було застосовано бібліотеку Bootstrap 4 [9] з модифікованими CSS стилями згідно з принципами дизайну нового покоління Material Design [10, 18, 19]. Для написання коду було обрано кілька мов програмування. До стандартних для вебпрограмування мов HTML [4, 5], CSS [2, 3] та JavaScript було додано мову, на якій написано сам фреймворк Angular, – TypeScript, та препроцесор для написання стилів SCSS [6], який дозволяє писати каскадні стилі більш структуровано та компактніше.

Angular (зазвичай так називають фреймворк Angular 2 або Angular 2+, тобто вищі версії) — написаний на TypeScript front-end фреймворк з відкритим кодом, який розробляється під керівництвом Angular Team у компанії Google, а також спільнотою приватних розробників та корпорацій. Angular — це AngularJS, який переосмислили та який був повністю переписаний тією ж командою розробників.

Спочатку переписаний AngularJS отримав назву Angular 2 від команди розробників, яка над ним працювала, але це призвело до плутанини серед інших розробників. Аби пояснити різницю між ними та наголосити, що це окремі проекти, команда вирішила для фреймворків версій 1.X застосовувати назву AngularJS, а для версій, починаючи з 2.0, – Angular без JS.

До основних відмінностей між Angular та AngularJS можна віднести архітектуру додатка на Angular. Основними елементами в розробці є модулі, компоненти, шаблони, метадані, біндінг даних, директиви, сервіси та ін'єкції залежностей.

Як згадано вище, Angular — це ретельно переписаний AngularJS:

- додано Angular CLI, що дає змогу розпочати створення нового додатка, просто написавши команду `ng new [app name]`;

- Angular не використовує концепцію "області видимості" або контролерів, натомість як головну архітектурну концепцію він застосовує ієрархію компонентів;
- Angular має інакший синтаксис написання виразів, застосовуючи "[]" для біндингу даних властивостей, і "()" для біндингу даних івентів;
- модульність – значна частина основного функціоналу перенесена у модулі.

Вибір цього фреймворку обґрунтований тим, що розробнику не потрібно розпорошувати свою увагу на незначні базові речі – за нього це робить Angular, а розробник займається більшими і складнішими питаннями. Зокрема до таких особливостей належать:

- 1 Angular може приймати рішення. Протягом всього процесу розробки доводиться приймати масу рішень - починаючи від характеристик продукту і до технологічних аспектів, які в майбутньому вплинуть на оперативність бази коду. Angular може приймати рішення і пропонує розробникам параметри за замовчуванням для таких речей як мережеве з'єднання, адміністративне управління станами, вибір мови, набір інструментів конфігурації.
- 2 Angular створений для Scale. Angular був створений в Google, щоб вирішити проблеми Google-scale. Для Google це означає мільйони рядків коду, тисячі інженерів, різноманітні графіки виконання проекту, вимоги і потік дій. Платформа була розроблена так, щоб дати можливість створювати і управляти загальним кодом, а також розділяти роботу між відповідними ролями.
- 3 Зручність супроводу - це наступна турбота багатьох розробників. Angular вирішує цю задачу двома шляхами. По-перше, розробники Angular використовують TypeScript, що дозволяє їм швидко знаходити баги і помилки. Крім цього, TypeScript дає можливість розробникам, які знайомі з базою коду, швидше включитися в роботу, так як вони зможуть швидше ознайомитися з типами даних в додатку. Команда Angular

створила Angular з TypeScript саме тому. І нарешті, Angular робить ставку на зручність тестування. Введення залежностей - це ключовий елемент Angular, який полегшує завдання написання тестів. Angular включає і підтримку наскрізного тестування за допомогою Protractor.

- 4 Angular – надійний. У зв'язку з тим, що Angular - це продукт Google, він використовує всі переваги його інфраструктури тестування. Кожна зміна в Angular перевіряється на відповідність з кожним проектом Angular в межах Google. Це означає, що ще до будь-якого публічного релізу фреймворк вже використовують сотні проектів, а такий підхід збільшує шанси на відсутність непередбачених критичних змін або регресій.

Angular має потужну підтримку. В інтернеті існує безліч багаторазово використовуваних інструментів, бібліотек і прикладів коду для Angular і AngularJS, і велика кількість цих інструментів або були оптимізовані під роботу з Angular, або зараз знаходяться в процесі оптимізації. Такі розробники як VMWare, Teradata, ag-Grid, NativeScript і інші вже повністю підтримують Angular.

Angular рекомендує та застосовує розроблену Microsoft мову — TypeScript, що містить такі можливості, як:

- класи, а отже Об'єктно-орієнтоване програмування;
- система типізації;
- узагальнене програмування.

TypeScript [7] — надмножина ECMAScript 6 (ES6), і є зворотно сумісним зі стандартом ECMAScript 5 (тобто JavaScript). Angular також має такі ES6-можливості, як:

- анонімні функції;
- ітератори;
- цикли типу For/Of;
- Python-подібні генератори;
- рефлексія;
- динамічне завантаження;

- асинхронна компіляція шаблонів;
- заміна контролерів та `$scope`(області видимості) компонентами та директивами – компонент є директивою з шаблоном;
- ітеративні колбеки завдяки використанню RxJS. RxJS дещо обмежує видимість станів та можливості дебагінгу, але, застосовуючи такі плагіни, як `ngReact` та `ngRx`, це легко вирішується.

TypeScript — мова програмування, представлена Microsoft восени 2012; позиціонується як засіб розробки веб-застосунків, що розширює можливості JavaScript.

Розробником мови TypeScript є Андерс Гейлсберг (англ. Anders Hejlsberg), який створив раніше C#, Turbo Pascal і Delphi.

Код експериментального компілятора, котрий транслює код TypeScript в представлення JavaScript, поширюється під ліцензією Apache, розробка ведеться в публічному репозиторії через сервіс CodePlex. Специфікації мови відкриті і опубліковані в рамках угоди Open Web Foundation Specification Agreement (OWFa 1.0).

TypeScript є зворотно сумісним з JavaScript. Фактично, після компіляції програму на TypeScript можна виконувати в будь-якому сучасному браузері або використовувати спільно з серверною платформою Node.js.

Переваги над JavaScript:

- можливість явного визначення типів (статична типізація);
- підтримка використання повноцінних класів (як в традиційних об'єктно-орієнтованих мовах);
- підтримка підключення модулів.

За задумом ці нововведення мають підвищити швидкість розробки, читабельність, рефакторинг і повторне використання коду, здійснення пошуку помилок на етапі розробки та компіляції, а також швидкодію програм.

Планується, що в силу повної зворотної сумісності адаптація існуючих застосунків на нову мову програмування може відбуватися поетапно, шляхом поступового визначення типів. Підтримка динамічної типізації зберігається —

компілятор TypeScript успішно обробить і не модифікований код на JavaScript. Основний принцип мови — весь існуючий код на JavaScript сумісний з TypeScript, тобто в програмах на TypeScript можна використовувати стандартні JavaScript-бібліотеки і раніше створені напрацювання. Більш того, можна залишити існуючі JavaScript-проекти в незмінному вигляді, а дані про типізації розмістити у вигляді анотацій, які можна помістити в окремі файли, які не заважатимуть розробці і прямому використанню проекту (наприклад, подібний підхід зручний при розробці JavaScript-бібліотек).

На момент релізу представлені файли для сприйняття розширеного синтаксису TypeScript для Vim і Emacs, а також плагін для Microsoft Visual Studio.

Одночасно з виходом специфікації розробники підготували файли з деклараціями статичних типів для деяких популярних JavaScript-бібліотек, серед яких jQuery.

Так як Angular під час етапу розробки проєкту працює на основі програмної платформи Node.js, то більшість спеціальних бібліотек, що не входять в його ядро, беруться із репозиторію npm.

До таких бібліотек зокрема можна віднести і Ng Http Loader [16]. Вона дозволяє відображати заздалегідь налаштований лоадер, блокуючи тим самим доступ до інтерфейсу, під час виконання http-запитів до сервера. Це забезпечує захист системи від помилкових та надмірних дій користувача.

Бібліотека NgX Clipboard [15, 17] дозволяє працювати із буфером обміну ЕОМ без необхідності використовувати застарілу та небезпечну технологію Flash. Ця бібліотека повністю написана на мові програмування JavaScript. Вона підтримується всіма браузерами та системами, що з'явилися за останні 12 років – це дозволяє охопити більше 80% користувачів.

В якості середовища розробки було вирішено обрати WebStorm [20] від компанії JetBrains. Це інтегроване середовище розробки для JavaScript, HTML та CSS від компанії JetBrains, розроблена на основі платформи IntelliJ IDEA. WebStorm є спеціалізованою версією PhpStorm, пропонуючи підмножину з його

можливостей. WebStorm постачається з перед-установленим плагінами JavaScript (такими як для Node.js), котрі доступні для PhpStorm безкоштовно. Він підтримує мови JavaScript, CoffeeScript, TypeScript та Dart.

WebStorm забезпечує автодоповнення, аналіз коду на льоту, навігацію по коду, рефакторинг, зневадження та інтеграцію з системами управління версіями. Важливою перевагою інтегрованого середовища розробки WebStorm є робота з проектами[3] (у тому числі, рефакторинг коду JavaScript, що міститься в різних файлах і теках проекту, а також вкладеного в HTML). Підтримується множинна вкладеність (коли в документ на HTML вкладений скрипт на Javascript, в який вкладено інший код HTML, всередині якого вкладений Javascript) — в таких конструкціях підтримується коректний рефакторинг.

До його основних можливостей належить:

- інтеграція з системами управління версіями Subversion, Git, GitHub, Perforce, Mercurial, CVS підтримуються з коробки з можливістю побудови списку змін і відкладених змін;
 - інтеграція з системами відстеження помилок;
 - модифікація файлів .css, html, .js з одночасним переглядом результатів (Live Edit, в деяких джерелах ця функціональність називається «редагування файлів на льоту» або «в реальному часі» або «без перезавантаження сторінки»);
 - віддалене розгортання за протоколами FTP, SFTP, на монтованих мережних дисках тощо з можливістю автоматичної синхронізації;
- можливості Zen Coding і Emmet.

3 ОПИС РОЗРОБЛЕНОЇ СИСТЕМИ

3.1 Опис шаблону сторінок

3.1.1 Сторінка Sign In

Так як фреймворк Angular вимагає дотримання синтаксису і структури проекту у форматі компонентів, всі сторінки складаються із окремих незалежних компонентів. Через це шаблони сторінок не є суцільними в процесі розробки. Але в процесі компіляції проекту у фінальну версію готову до публікації на хостинг весь код цих компонентів, включно із їх стилями, переноситься в js-файли. Тому файл шаблону має монолітну структуру лише коли відображається у браузері.

Дана сторінка Sign In, що зображена на рисунку 3.1, є першою сторінкою, яка відображається користувачеві при спробі увійти в систему. На ній знаходиться лише форма для введення даних для авторизації в системі.

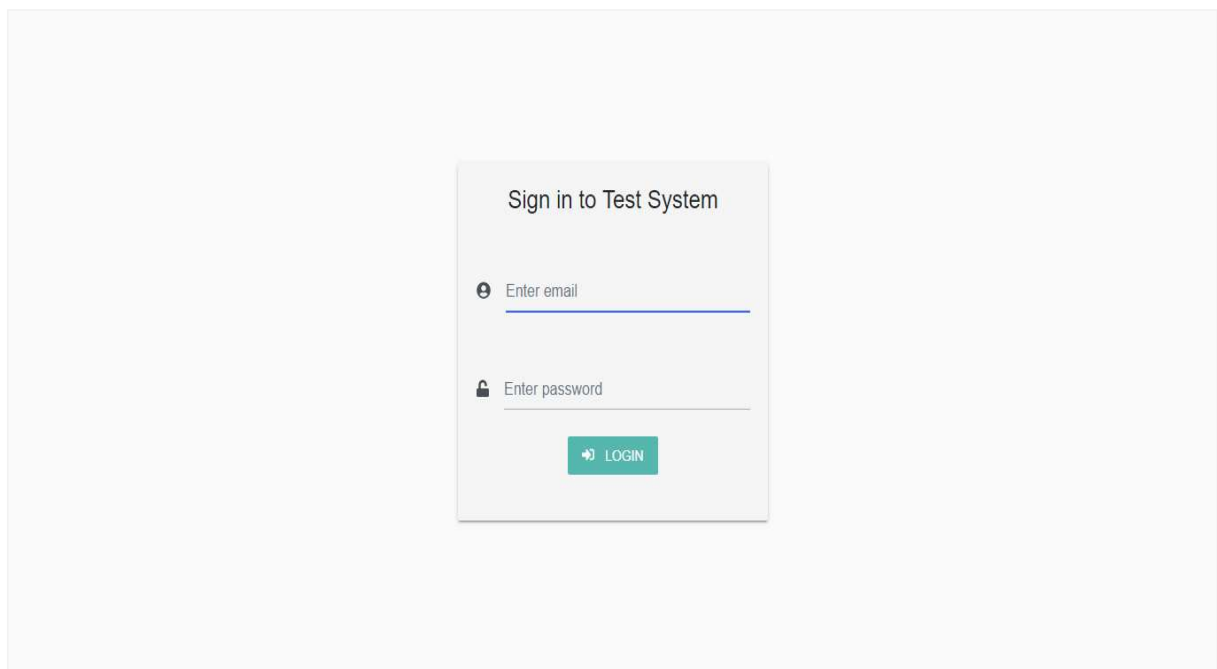


Рисунок 3.1 – Сторінка Sign In

Сторінка складається з таких компонентів:

- AppComponent – головна компонента, яка є батьківською для всіх компонент додатку;
- AlertCloseableComponent – компонента для виведення та відображення користувачеві помилок, що виникають в процесі експлуатації додатку;

- NgHttpLoaderComponent – компонента із стороннього пакету, що відображає лоадер, який перекриває весь інтерфейс додатку, коли виконується http-запит до серверу і очікуються дані;
- LoginFormComponent – компонента, що містить форму для введення даних для авторизації в системі.

Компонента AlertCloseableComponent може виводити адаптовані для розуміння користувачем повідомлення про деякі найчастіші помилки, що виникають в процесі експлуатації системи. Наприклад на рисунку 3.2 зображено повідомлення про невірні дані для авторизації, що їх увів користувач.

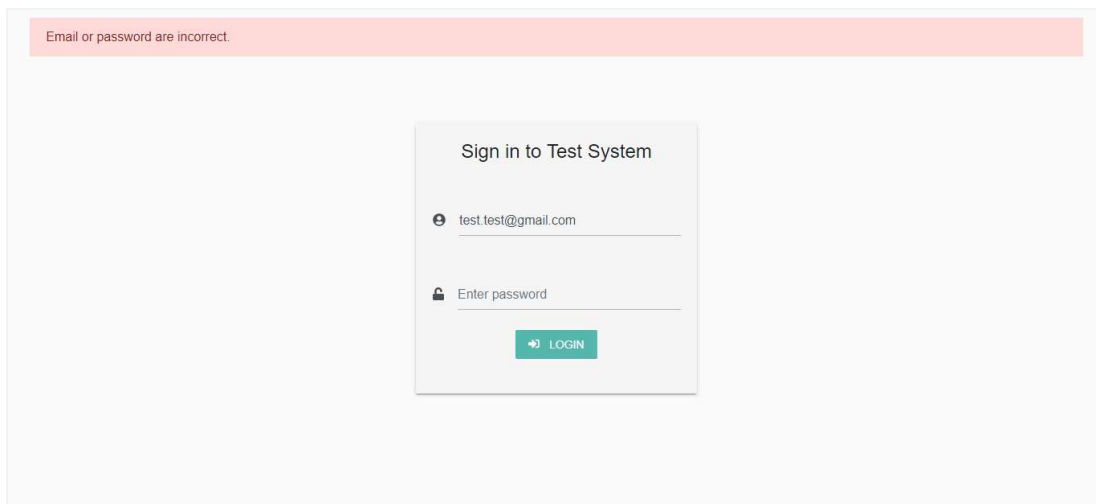


Рисунок 3.2 – Повідомлення про помилку, коли логін чи пароль невірні

А на рисунку 3.3 зображено повідомлення про помилку, яка виникає в разі відсутності доступу до інтернету або специфічної помилки на сервері.

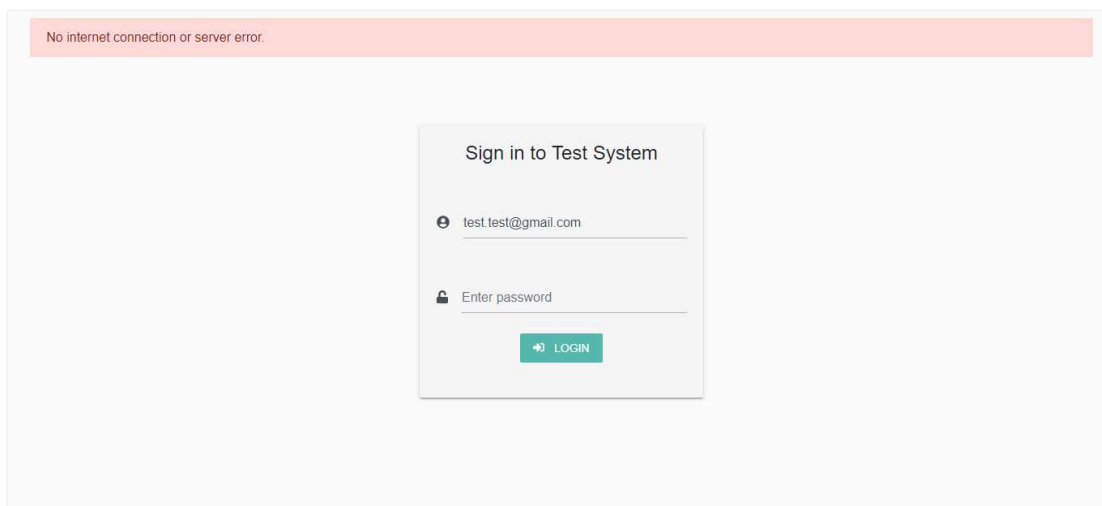


Рисунок 3.3 – Повідомлення про помилку, коли відсутнє інтернет з'єднання

Компонента `NgHttpLoaderComponent` блокує весь інтерфейс додатку та відображає блок із лоадером, коли відбувається http-запит на сервер та очікується відповідь з даними. Це робиться для того, щоб користувач не призвів своїми діями до виникнення помилки в роботі додатку. Даний елемент зображено на рисунку 3.4.

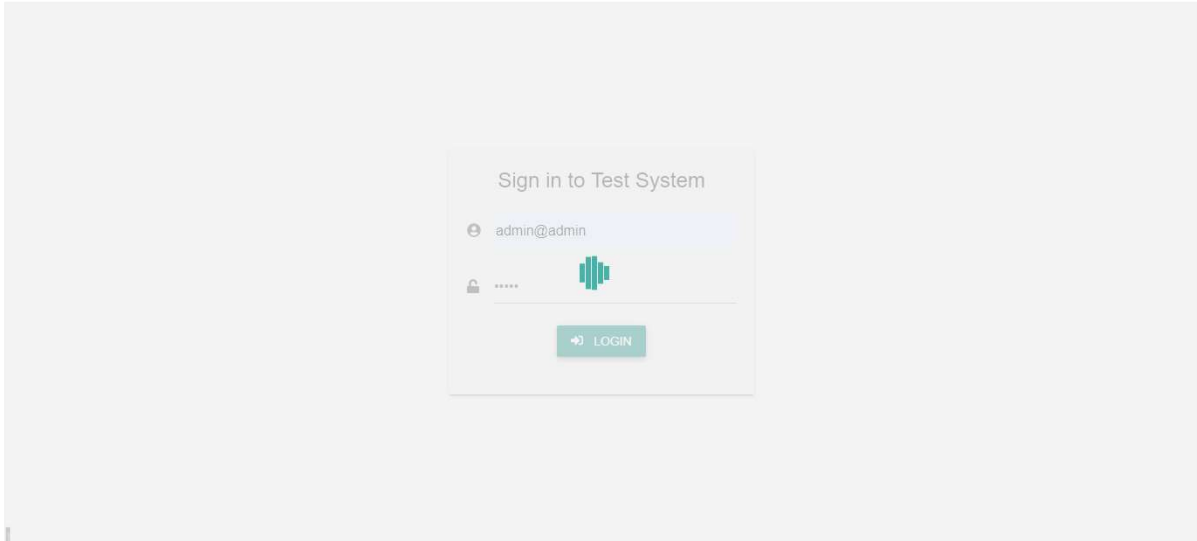


Рисунок 3.3 – Повідомлення про помилку, коли відсутнє інтернет з’єднання

3.1.2 Сторінка Home

Зважаючи на інформацію щодо роботи і створення графічного інтерфейсу додатку за допомогою фреймворку Angular, яка зазначена в пункті 3.1.1 даної пояснювальної записки, сторінка Home має таку ж структуру – складається із кількох компонентів. Вона зображена на рисунку 3.4.

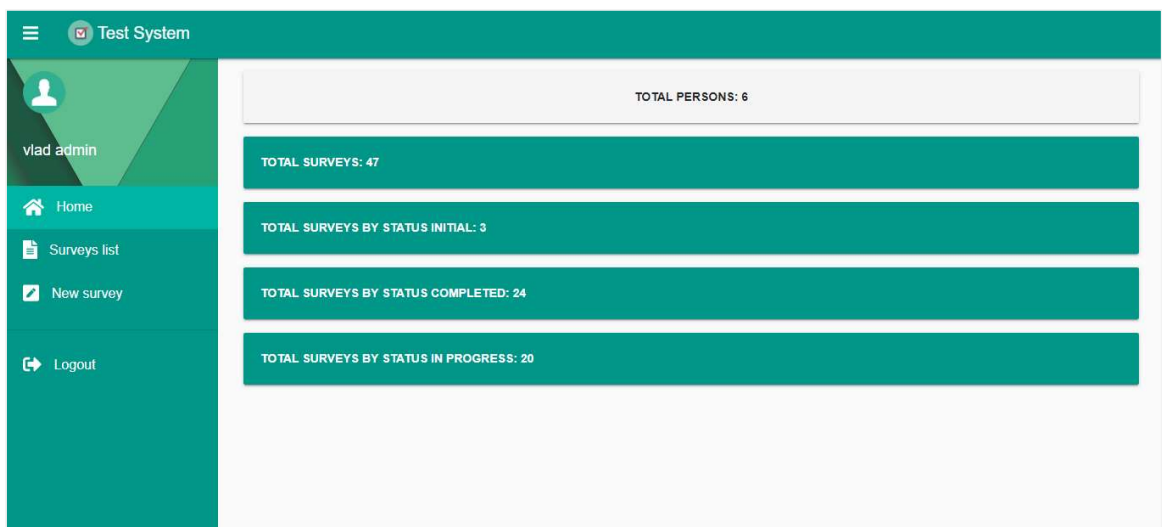


Рисунок 3.4 – Сторінка Home

Дана сторінка складається із таких компонент:

- AppComponent – головна компонента, яка є батьківською для всіх компонент додатку;
- AlertCloseableComponent – компонента для виведення та відображення користувачеві помилок, що виникають в процесі експлуатації додатку;
- NgHttpLoaderComponent – компонента із стороннього пакету, що відображає лоадер, який перекриває весь інтерфейс додатку, коли виконується http-запит до серверу і очікуються дані;
- AppWrapperComponent – компонента, що виступає обгорткою для всіх сторінок додатку, при умові, що вони доступні лише після авторизації;
- SidebarMenuItemComponent – дана компонента є блоком для кожного з пунктів головного меню додатку, що розташоване зліва;
- HomeComponentComponent – компонента, що містить контент кожної сторінки додатку;
- HomeStatisticItemComponent – компоненти, за допомогою яких виводиться основна статистична інформація системи.

Детальніше про компоненти AppComponent написано у пункті 3.1.1 даної пояснювальної записки.

Компонента SidebarMenuItemComponent створена для зменшення кодової бази. Так як одним із основних принципів фреймворку Angular є те, що додаток повинен складатися із компонент, які можна легко використати знову, то ліва навігаційна панель являє собою звичайну обгортку та статичні елементи і набір таких компонент. Це корисно, так як зменшує об'єм кодової бази та файлів, що передаються користувачеві, та зручно, так як не потрібно постійно писати одні й ті ж шматки коду. Ліва навігаційна панель з компонентами SidebarMenuItemComponent зображена на рисунку 3.5.

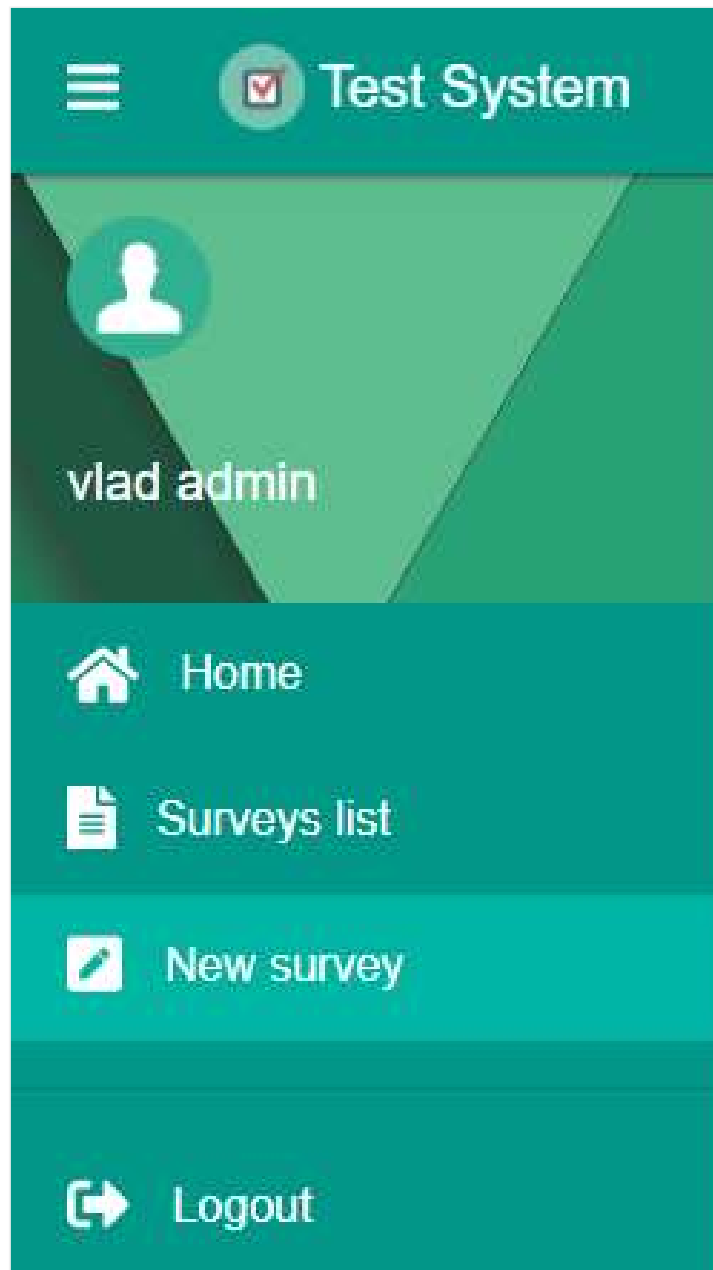


Рисунок 3.5 – Ліва навігаційна панель

Ця панель прихована на мобільних пристроях, тому що додаток має оптимізований під пристрої з різними розмірами екрану інтерфейс. За відкриття та показ даної панелі відповідає спеціальна кнопка.

Компонента `HomeContentComponent` складається із компонент-блоків `HomeStatisticItemComponent`, що містять основну статистичну інформацію системи, як то кількість адміністраторів, кількість взагалі створених тестів чи кількість тестів по категоріям, тощо. На рисунку 3.6 зображено дану компоненту з інформацією станом на період тестування.



Рисунок 3.6 – Компонента HomeComponentComponent з набором компонент HomeStatisticItemComponent

Дана сторінка відображається користувачеві при вході в систему і є стартовою для роботи з нею. Звідси можна перейти на всі інші сторінки системи.

3.1.3 Сторінка Surveys List

Сторінка Surveys List містить список всіх створених тестів у вигляді таблиці (рисунок 3.8). При натисненні на кнопку «SHOW» (рисунок 3.7) можна переглянути деталі за кожним тестом, якщо він має статус «completed».



Рисунок 3.7 – Кнопка «SHOW»

ID	First Name	Last Name	Status	Description	Show info
1	vlad	admin	in progress	1 junior	SHOW
35	Alexis	Ortiz	completed	5 senior	SHOW
68	joker	..	completed	8 middle-1	SHOW
69	joker	..	in progress	8 junior	SHOW
2	vlad	admin	completed	1 middle-1	SHOW
3	vitalii	not admin	in progress	2 junior	SHOW
4	vitalii	not admin	in progress	2 middle-1	SHOW
5	vitalii	not admin	in progress	2 junior	SHOW

Рисунок 3.8 – Сторінка Surveys List

Дана сторінка також має компонентну структуру, а тому складається з таких компонент:

- AppComponent – головна компонента, яка є батьківською для всіх компонент додатку;
- AlertCloseableComponent – компонента для виведення та відображення користувачеві помилок, що виникають в процесі експлуатації додатку;
- NgHttpLoaderComponent – компонента із стороннього пакету, що відображає ладер, який перекриває весь інтерфейс додатку, коли виконується http-запит до серверу і очікуються дані;
- AppWrapperComponent – компонента, що виступає обгорткою для всіх сторінок додатку, при умові, що вони доступні лише після авторизації;
- SidebarMenuItemComponent – дана компонента є блоком для кожного з пунктів головного меню додатку, що розташоване зліва;
- HomeComponentComponent – компонента, що містить контент кожної сторінки додатку;
- SurveysListComponent – компонента, що представляє собою таблицю з основною інформацією по всім створеним тестам.

Компонента SurveysListComponent (рисунок 3.9) має всього лише один елемент – таблицю. Вона являє собою список всіх створених тестів. Тут же відображається ім'я працівника, для якого створений тест, рівень тесту і його статус.

ID	First Name	Last Name	Status	Description	Show info
1	vlad	admin	in progress	1 junior	SHOW
35	Alexis	Ortiz	completed	5 senior	SHOW
68	joker	.	completed	8 middle-1	SHOW
69	joker	.	in progress	8 junior	SHOW
2	vlad	admin	completed	1 middle-1	SHOW
3	vitalii	not admin	in progress	2 junior	SHOW
4	vitalii	not admin	in progress	2 middle-1	SHOW

Рисунок 3.9 – Компонента SurveysListComponent

3.1.4 Сторінка New Survey

Сторінка New Survey (рисунок 3.10) призначена для створення нового тесту. Вона складається з таких компонент:

- AppComponent – головна компонента, яка є батьківською для всіх компонент додатку;
- AlertCloseableComponent – компонента для виведення та відображення користувачеві помилок, що виникають в процесі експлуатації додатку;
- NgHttpLoaderComponent – компонента із стороннього пакету, що відображає лоадер, який перекриває весь інтерфейс додатку, коли виконується http-запит до серверу і очікуються дані;
- AppWrapperComponent – компонента, що виступає обгорткою для всіх сторінок додатку, при умові, що вони доступні лише після авторизації;
- SidebarMenuItemComponent – дана компонента є блоком для кожного з пунктів головного меню додатку, що розташоване зліва;
- HomeComponentComponent – компонента, що містить контент кожної сторінки додатку;
- CreateNewSurveyComponent – компонента, що форму для введення даних, які потрібні для створення нового тесту.

The screenshot displays the 'Test System' web application interface. On the left is a teal sidebar with a hamburger menu icon and a user profile icon labeled 'vlad admin'. The sidebar contains navigation links: 'Home', 'Surveys list', 'New survey' (which is highlighted with a white checkmark), and 'Logout'. The main content area has a teal header with the text 'Test System'. Below the header is a white form titled 'Create new survey'. The form contains two input fields: 'Select employee' and 'Select degree survey', with the value 'junior' entered in the second field. A teal 'CREATE SURVEY' button is positioned below the second field. At the bottom of the form is a section titled 'Generated links for survey' with an empty table below it.

Рисунок 3.10 – Сторінка New Survey

На компоненті `CreateNewSurveyComponent` окрім форми для створення нового тесту є блок із полями, куди виводяться посилання на згенеровані тести (рисунок 3.11).

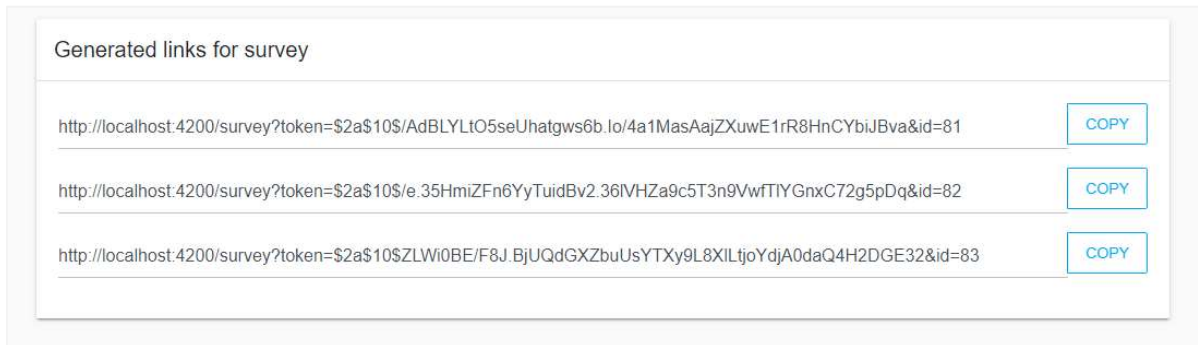


Рисунок 3.11 – Посилання на згенеровані тести

3.1.5 Сторінка Survey Info

На сторінці `Survey Info` (рисунок 3.12) міститься детальна інформація по кожному з тестів, що знаходяться в статусі «completed». Щоб переглянути цю сторінку, потрібно натиснути кнопку «Show» біля потрібного тесту (пункт 3.1.3).

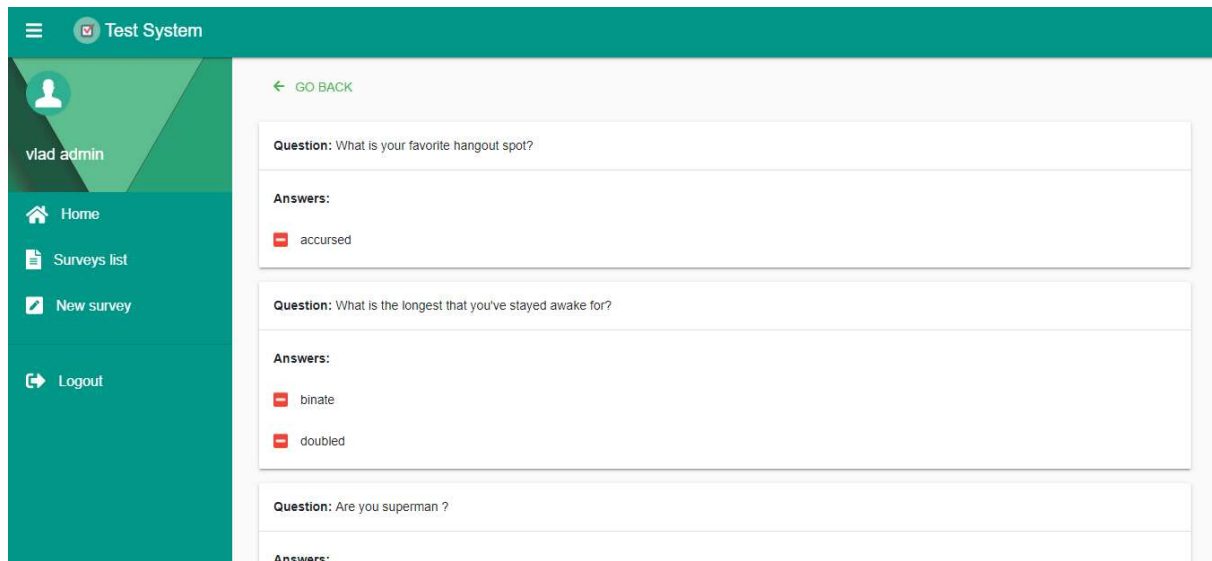


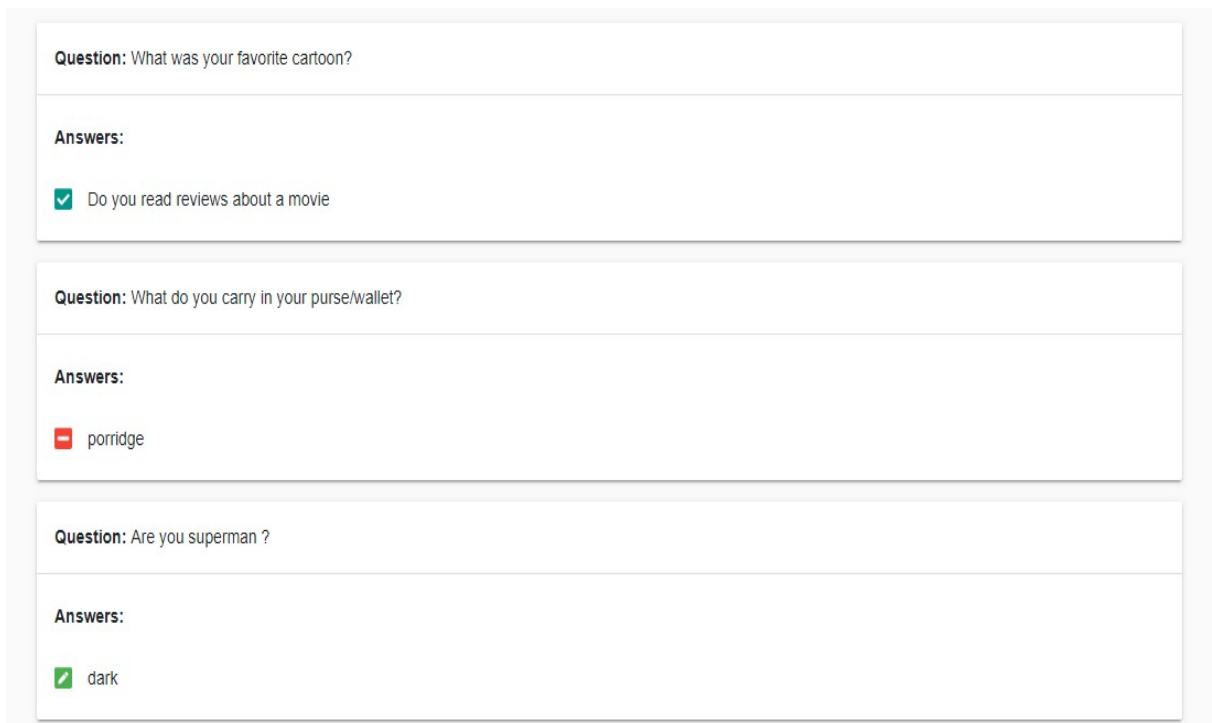
Рисунок 3.12 – Сторінка Survey Info

Дана сторінка складається з таких компонент:

- `AppComponent` – головна компонента, яка є батьківською для всіх компонент додатку;
- `AlertCloseableComponent` – компонента для виведення та відображення користувачеві помилок, що виникають в процесі експлуатації додатку;

- `NgHttpLoaderComponent` – компонента із стороннього пакету, що відображає лоадер, який перекриває весь інтерфейс додатку, коли виконується http-запит до серверу і очікуються дані;
- `AppWrapperComponent` – компонента, що виступає обгорткою для всіх сторінок додатку, при умові, що вони доступні лише після авторизації;
- `SidebarMenuItemComponent` – дана компонента є блоком для кожного з пунктів головного меню додатку, що розташоване зліва;
- `HomeContentComponent` – компонента, що містить контент кожної сторінки додатку;
- `CompletedSurveyInfoComponent` – компонента із інформацією по даному тесту;
- `SurveyAnswerItemComponent` – компонента, в якій відображається питання та відповідь на нього, яку дав працівник.

Компонента `SurveyAnswerItemComponent` окрім питання та відповіді на нього містить мітку, що позначає правильна чи неправильна відповідь дана або чи не являє собою ця відповідь розгорнуту (рисунок 3.13).



The image displays three instances of the `SurveyAnswerItemComponent` stacked vertically. Each instance consists of a question box and an answer box. The first item has the question 'What was your favorite cartoon?' and the answer 'Do you read reviews about a movie' with a green checkmark icon. The second item has the question 'What do you carry in your purse/wallet?' and the answer 'porridge' with a red minus sign icon. The third item has the question 'Are you superman ?' and the answer 'dark' with a green checkmark icon.

Question	Answer	Status
Question: What was your favorite cartoon?	Do you read reviews about a movie	Correct (Green Checkmark)
Question: What do you carry in your purse/wallet?	porridge	Incorrect (Red Minus)
Question: Are you superman ?	dark	Correct (Green Checkmark)

Рисунок 3.13 – Компонента `SurveyAnswerItemComponent` із мітками до відповідей

3.1.6 Сторінка Survey

Сторінка Survey (рисунок 3.14) призначена для власне проходження тесту. А тому являє собою тимчасову сторінку. Вона складається із полів та чекбоксів, в залежності від типу питання.

Рисунок 3.13 – Сторінка Survey

Дана сторінка складається з наступних компонентів:

- AppComponent – головна компонента, яка є батьківською для всіх компонент додатку;
- AlertCloseableComponent – компонента для виведення та відображення користувачеві помилок, що виникають в процесі експлуатації додатку;
- NgHttpLoaderComponent – компонента із стороннього пакету, що відображає лоадер, який перекриває весь інтерфейс додатку, коли виконується http-запит до серверу і очікуються дані;
- UserSurveyPassComponent – компонента, що містить форми для проходження тесту.

3.2 Опис інтерфейсу та функціональних можливостей

3.2.1 Процес авторизації адміністратора

В даній системі авторизація допускається лише для адміністраторів. Для того, щоб потрапити на сторінку авторизації, потрібно зайти на будь-яку сторінку додатку, окрім Survey. Якщо користувач буде не авторизований, то його

автоматично перенаправить на неї. Після цього потрібно ввести свої дані – логін та пароль – в поля форми (рисунок 3.14). В поле «Enter email» потрібно ввести адресу електронної скриньки, зважаючи на підказки валідатора (рисунок 3.15, 3.16).

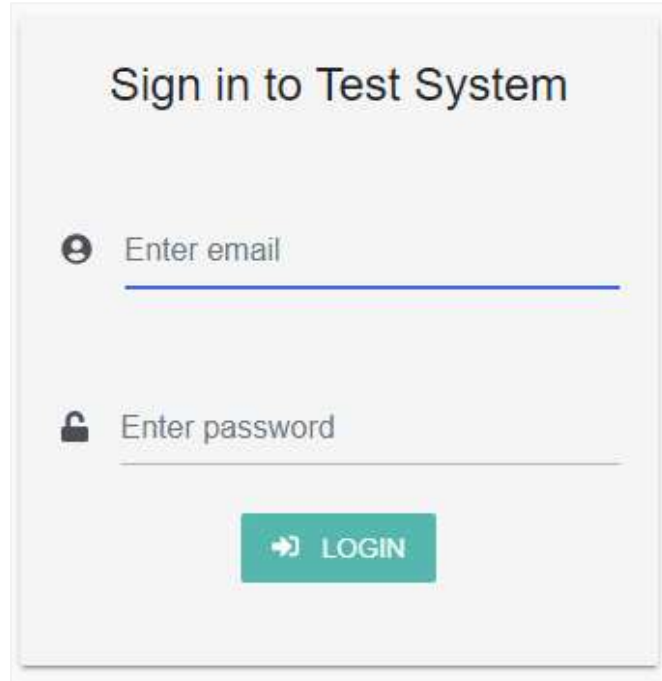


Рисунок 3.15 – Форма авторизації в системі

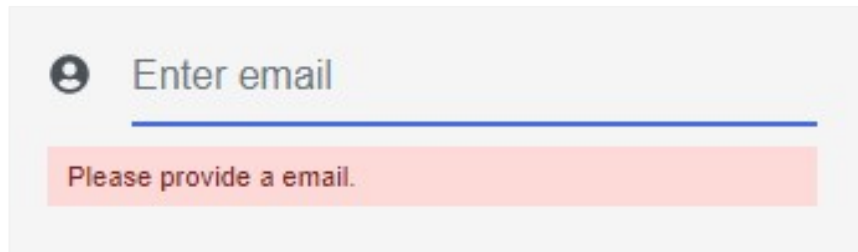


Рисунок 3.15 – Підказка, що дане поле не може бути порожнім

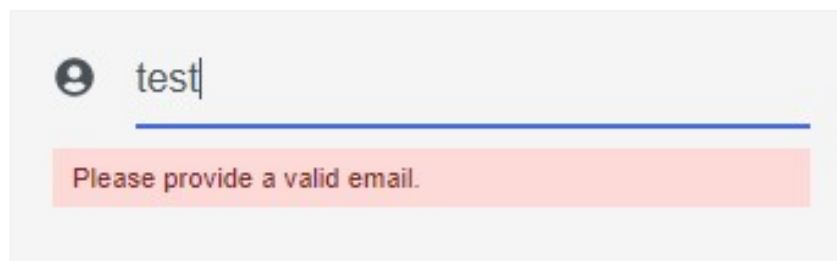


Рисунок 3.16 – Підказка, що введено невірний email

Потім в поле «Enter password» потрібно ввести пароль, дотримуючись підказок валідатора (рисунок 3.17).

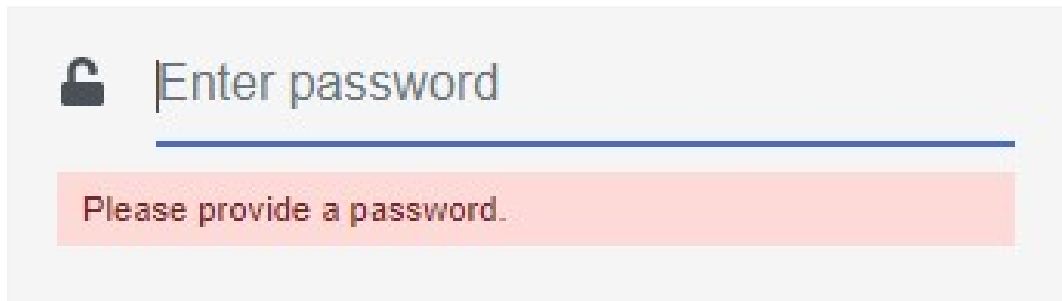


Рисунок 3.17 – Підказка, що дане поле не може бути порожнім

Після введення даних, що проходять валідацію, вони відправляються на сервер для перевірки наявності користувача з такими даними. У разі помилкових або неправильних даних буде показано помилку у верхній частині екрану (рисунок 3.18).



Рисунок 3.18 – Повідомлення про невірні логін або пароль

В разі відсутності інтернет з'єднання або помилки на сервері буде показано наступне повідомлення, що зображене на рисунку 3.19.

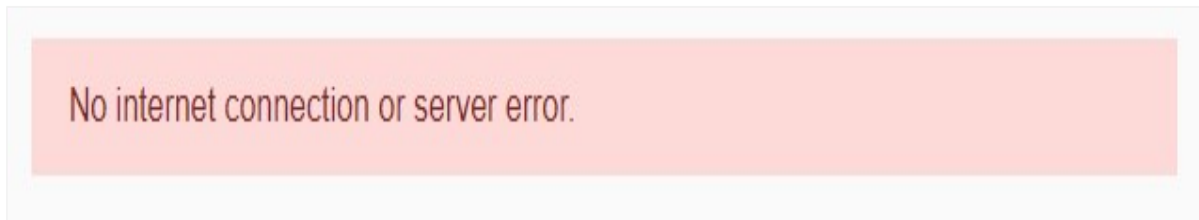
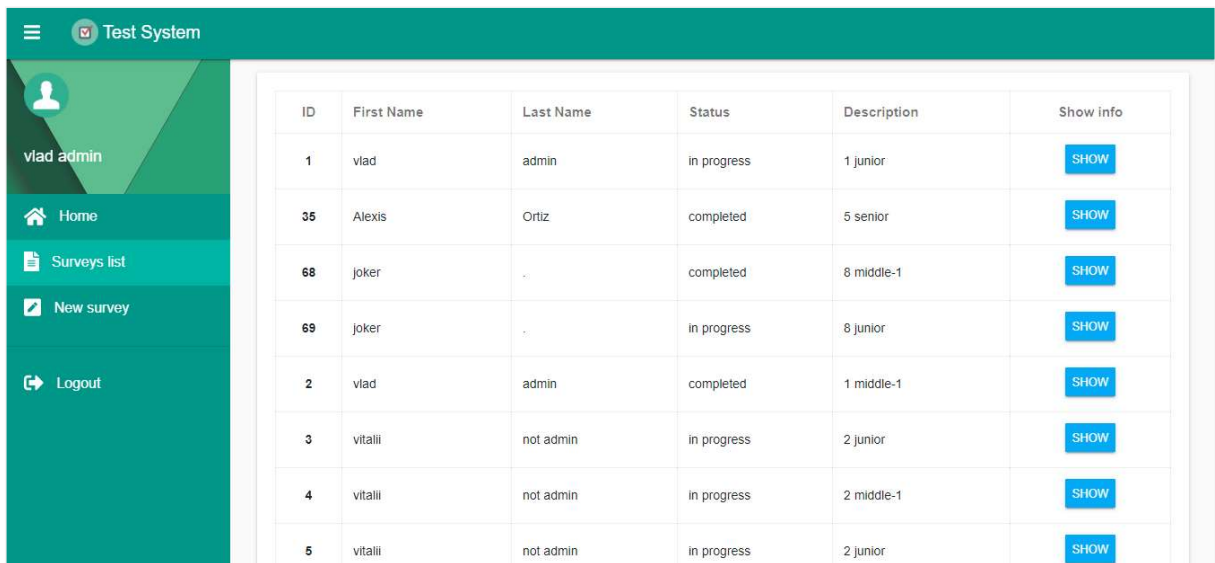


Рисунок 3.19 – Повідомлення про відсутність інтернету або помилки на сервері

Якщо ж в процесі авторизації помилок не було і користувач по введеним даним існує, то адміністратора буде перенаправлено на сторінку Home.

3.2.2 Процес перегляду детальної інформації по тестах

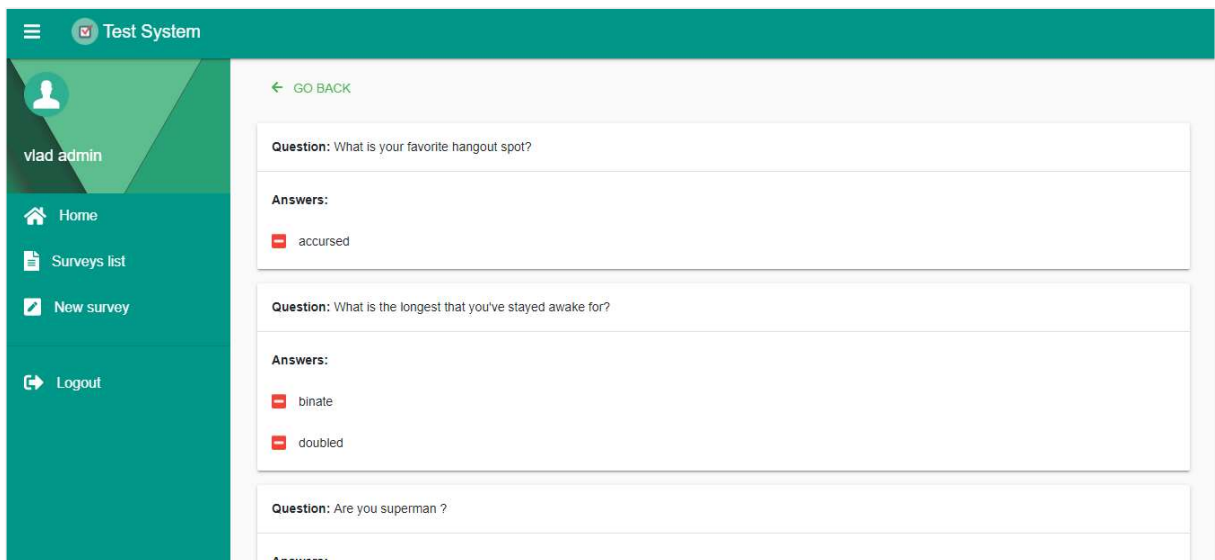
На сторінці Surveys List знаходиться список всіх тестів із загальною інформацією по них. Але цього не достатньо, щоб оцінити кваліфікаційний рівень співробітника, тому можна переглянути детальну інформацію по тестах на сторінці Survey Info. Щоб зробити це потрібно перейти на сторінку Surveys List (рисунок 3.20).



ID	First Name	Last Name	Status	Description	Show info
1	vlad	admin	in progress	1 junior	SHOW
35	Alexis	Ortiz	completed	5 senior	SHOW
68	joker	-	completed	8 middle-1	SHOW
69	joker	-	in progress	8 junior	SHOW
2	vlad	admin	completed	1 middle-1	SHOW
3	vitalii	not admin	in progress	2 junior	SHOW
4	vitalii	not admin	in progress	2 middle-1	SHOW
5	vitalii	not admin	in progress	2 junior	SHOW

Рисунок 3.20 – Сторінка Surveys List

При натисненні на кнопку «SHOW» (рисунок 3.22), користувача буде перенаправлено на сторінку Survey Info (рисунок 3. 21), де буде відображено детальний звіт по вибраному тестові. Але ця дія буде успішною в разі якщо тест має статус «completed», а інакше користувачеві буде показана помилка (рисунок 3.23).



← GO BACK

Question: What is your favorite hangout spot?

Answers:

- accursed

Question: What is the longest that you've stayed awake for?

Answers:

- binate
- doubled

Question: Are you superman ?

Answers:

Рисунок 3.21 – Сторінка Survey Info

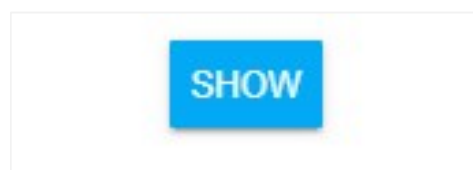


Рисунок 3.22 – Кнопка «SHOW»

№1 Persons Questions list is Empty

Рисунок 3.23 – Повідомлення про помилку при спробі перегляну результати тесту із статусом відмінним від «completed»

3.2.3 Процес створення нового тесту

Для того, щоб створити новий тест, потрібно перейти на сторінку New Survey (рисунок 3.24) та заповнити поля форми (рисунок 3.25).

Рисунок 3.24 – Сторінка New Survey

Рисунок 3.25 – Форма створення нового тесту

В поле «Select employee» потрібно почати вводити ім'я або прізвище співробітника. Система відразу розпочне пошук по БД всіх, хто відповідає такому пошуковому запиту, і покаже їх у списку у вигляді випадаючого меню під полем, з якого можна вибрати співробітника (рисунок 3.26). Це поле також

має валідатор, що не дозволить відправити дані на сервер до тих пір, поки в поле не буде введено співробітника з бази даних (рисунок 3.27).

Рисунок 3.26 – Поле «Select employee» із списком знайдених співробітників

Рисунок 3.27 – Підказка, що дане поле не може бути порожнім

Після зроблених кроків в блоці Generated links for survey буде показаний список із згенерованих посилань для проходження створених тестів (рисунок 3.28). При натисненні на кнопку «COPY» посилання буде скопійовано в буфер обміну пристрою.

Рисунок 3.28 – Посилання на згенеровані тести

3.2.4 Процес проходження створеного тесту

Для проходження тесту потрібно перейти за посиланням, яке видається працівникові адміністратором. При відкритті даного посилання у особи буде запитана готовність до проходження тесту (рисунок 3.29). Якщо погодитися, то розпочнеться безпосередньо проходження самого тесту. Якщо ж відмовитися, то працівника перенаправить на попередню сторінку в цій вкладці браузера.

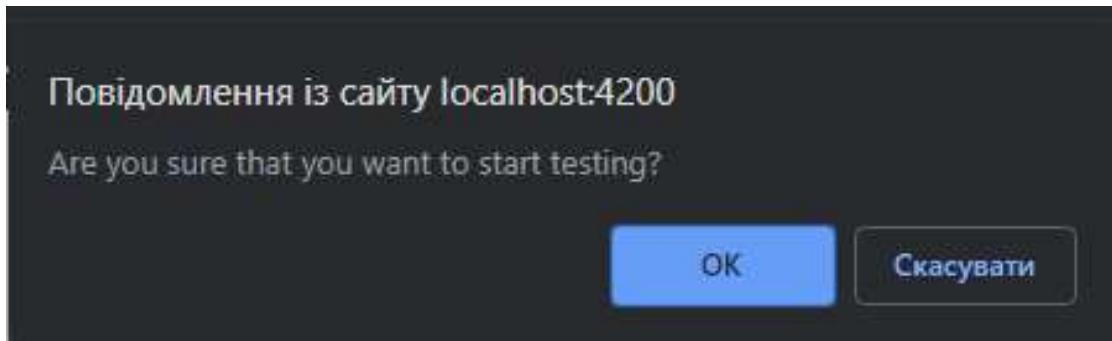


Рисунок 3.29 – Запит підтвердження на готовність до проходження тесту

Тест відображається на сторінці Survey (рисунок 3.30), яка виведена із основної частини додатку. Доступ на цю сторінку можливий лише за посиланням, яке містить спеціально згенерований токен. На сторінці знаходиться інформація про тест, а також два види питань: питання з вибором кількох правильних відповідей (рисунок 3.31) та питання з розгорнутою відповіддю (рисунок 3.32).

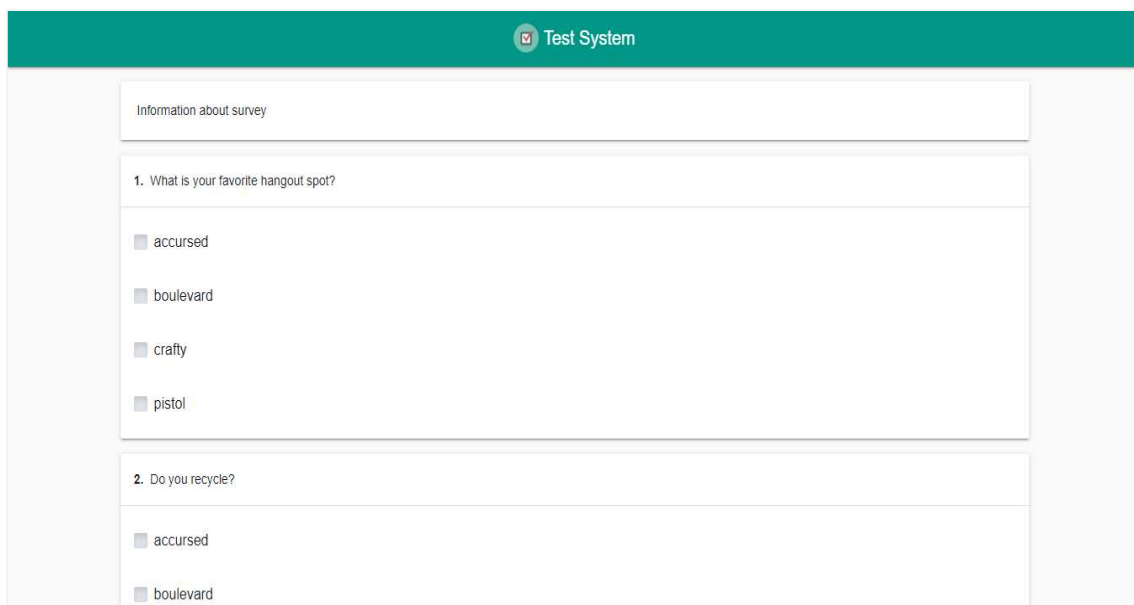
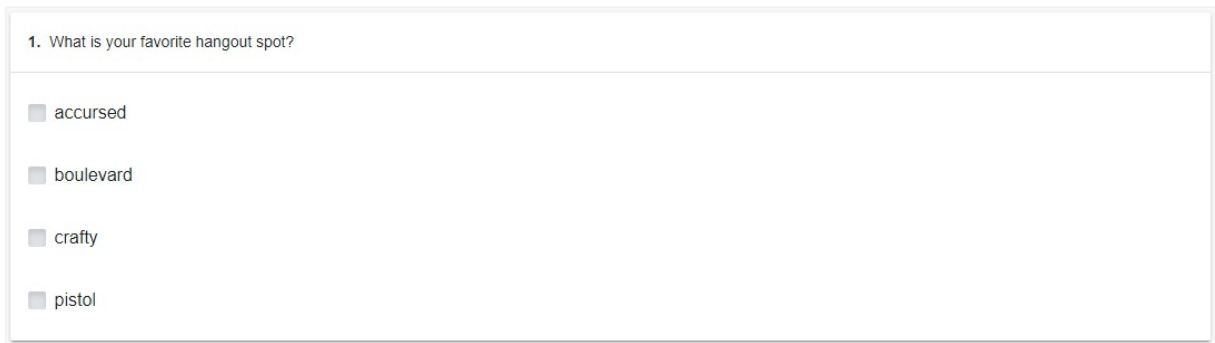


Рисунок 3.30 – Сторінка Survey



1. What is your favorite hangout spot?

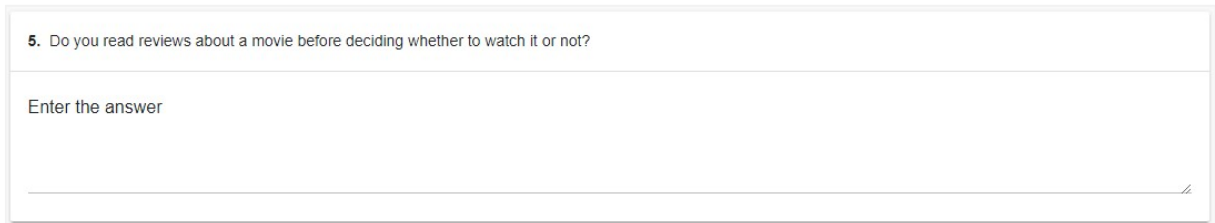
☐ accursed

☐ boulevard

☐ crafty

☐ pistol

Рисунок 3.31 – Питання з вибором кількох правильних відповідей



5. Do you read reviews about a movie before deciding whether to watch it or not?

Enter the answer

Рисунок 3.32 – Питання з розгорнутою відповіддю

Після виконання всіх завдань користувач натискає кнопку відправки даних на сервер. Коли дані відправились сервер для перевірки, то користувачеві відображається повідомлення про успішне завершення тесту з можливістю повернутися на попередньо відкритий веб-сайт в цій вкладці браузера (рисунок 3.33).

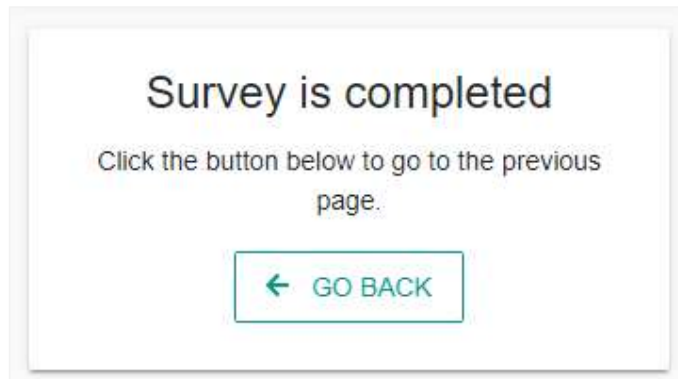


Рисунок 3.33 – Повідомлення про успішне завершення тесту

При спробі пройти раніше складений тест буде показано повідомлення із помилкою наступного характеру – рисунок 3.34.

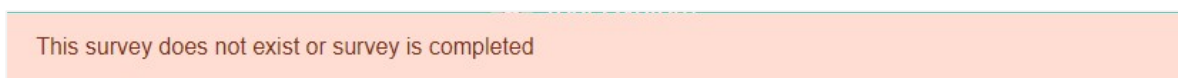


Рисунок 3.34 – Повідомлення про недоступність даного тесту, так як він вже був пройдений

ВИСНОВКИ

Web-додатки є актуальними в наш час. Це технологія, яка стрімко розвивається і покликана зекономити час та ресурси на створенні універсальних додатків (програм), які в свою чергу дають можливість працювати в будь-якому місці, у будь-який час та з будь-яких девайсів не залежно від операційної системи чи технічних характеристик.

В даній курсовій роботі був розроблений інтерфейс web-додатку «Система тестування рівня кваліфікації працівника». Для досягнення поставленої цілі було виконано технічне завдання.

До недоліків роботи можна віднести наступне:

- не реалізована можливість реєстрації різних користувачів;
- не реалізована можливість виставлення балів.

За допомогою цього додатку можна проводити попереднє тестування кандидатів на посади в компанії, зменшуючи таким чином фінансові витрати та витрати часу на даний процес.

При оформленні курсової роботи були одержані навички написання програмного забезпечення, а також великий практичний досвід роботи з:

- Angular;
- HTML5 ;
- CSS/SCSS ;
- JavaScript;
- TypeScript.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

- 1 angular.io [Електронний ресурс] : [Інтернет-портал]. – Електронні дані.– Режим доступу: <https://angular.io/> (дата звернення 10.03.2019). – Angular.
- 2 developer.mozilla.org [Електронний ресурс] : [Інтернет-портал]. – Електронні дані.– Режим доступу: <https://developer.mozilla.org/ru/docs/Learn/CSS3> (дата звернення 11.03.2019). – CSS.
- 3 webref.ru [Електронний ресурс] : [Інтернет-портал]. – Електронні дані.– Режим доступу: <https://webref.ru/css> (дата звернення 11.03.2019). – CSS.
- 4 developer.mozilla.org [Електронний ресурс] : [Інтернет-портал]. – Електронні дані.– Режим доступу: <https://developer.mozilla.org/ru/docs/Learn/HTML> (дата звернення 11.03.2019). – HTML.
- 5 webref.ru [Електронний ресурс] : [Інтернет-портал]. – Електронні дані.– Режим доступу: <https://webref.ru/html> (дата звернення 11.03.2019). – HTML.
- 6 sass-lang.com [Електронний ресурс] : [Інтернет-портал]. – Електронні дані.– Режим доступу: <https://sass-lang.com/> (дата звернення 27.03.2019). – SCSS.
- 7 www.typescriptlang.org [Електронний ресурс] : [Інтернет-портал]. – Електронні дані.– Режим доступу: <http://www.typescriptlang.org/> (дата звернення 01.04.2019). – TypeScript.
- 8 stackoverflow.com [Електронний ресурс] : [Інтернет-портал]. – Електронні дані.– Режим доступу: <https://stackoverflow.com/> (дата звернення 01.04.2019). – Angular.
- 9 getbootstrap.com [Електронний ресурс] : [Інтернет-портал]. – Електронні дані.– Режим доступу: <https://getbootstrap.com/> (дата звернення 06.04.2019). – Bootstrap.
- 10 fezvrasta.github.io [Електронний ресурс] : [Інтернет-портал]. – Електронні дані.– Режим доступу: <https://fezvrasta.github.io/bootstrap-material-design/> (дата звернення 06.04.2019). – Bootstrap.
- 11 metanit.com [Електронний ресурс] : [Інтернет-портал]. – Електронні дані.– Режим доступу: <https://metanit.com/web/angular2/> (дата звернення 06.04.2019). – Angular.

- 12 kushagragour.in [Електронний ресурс] : [Інтернет-портал]. – Електронні дані.– Режим доступу: <https://kushagragour.in/lab/hint/> (дата звернення 10.04.2019). – CSS.
- 13/popper.js.org [Електронний ресурс] : [Інтернет-портал]. – Електронні дані.– Режим доступу: <https://popper.js.org/> (дата звернення 11.04.2019). – Bootstrap.
- 14/jquery.com [Електронний ресурс] : [Інтернет-портал]. – Електронні дані.– Режим доступу: <https://jquery.com/download/> (дата звернення 12.04.2019). – JavaScript.
- 15/clipboardjs.com [Електронний ресурс] : [Інтернет-портал]. – Електронні дані.– Режим доступу: <https://clipboardjs.com/> (дата звернення 12.04.2019). – JavaScript.
- 16/npmjs.com [Електронний ресурс] : [Інтернет-портал]. – Електронні дані.– Режим доступу: <https://www.npmjs.com/package/ng-http-loader> (дата звернення 14.04.2019). – Angular.
- 17/npmjs.com [Електронний ресурс] : [Інтернет-портал]. – Електронні дані.– Режим доступу: <https://www.npmjs.com/package/ngx-clipboard> (дата звернення 14.04.2019). – Angular.
- 18/npmjs.com [Електронний ресурс] : [Інтернет-портал]. – Електронні дані.– Режим доступу: <https://www.npmjs.com/package/@ng-bootstrap/ng-bootstrap> (дата звернення 25.04.2019). – Bootstrap.
- 19/ng-bootstrap.github.io [Електронний ресурс] : [Інтернет-портал]. – Електронні дані.– Режим доступу: <https://ng-bootstrap.github.io/#/getting-started> (дата звернення 25.04.2019). – Bootstrap.
- 20/jetbrains.com [Електронний ресурс] : [Інтернет-портал]. – Електронні дані.– Режим доступу: <https://www.jetbrains.com/webstorm/> (дата звернення 25.04.2019). – WebStorm.