Приватний заклад вищої освіти
Одеський технологічний університет «ШАГ»
Кафедра інформаційних технологій та фундаментальної підготовки

Випускна кваліфікаційна робота бакалавра

«ІНФОРМАЦІЙНА СИСТЕМА УЗГОДЖЕННЯ ВЗАЄМОДІЇ КОРИСТУВАЧІВ»

на здобуття бакалаврського рівня вищої освіти
зі спеціальності «122 Комп'ютерні науки»

Виконавці проєкту

№	П.І.Б.	Група
1	Сарієв Г.	КН-П-191
2	Кравчук Д.	КН-П-191
3	Лоскутников И.	КН-П-191
4	Денисов Д.	КН-А-191
5	Самбаева К.	КН-Д-191

Автор звіту

_____ Сарієв Георгій Валентинович

Одеса – 2023

АНОТАЦІЯ

УДК 004.9
Д-30

Сарієв Г. В. Інформаційна система узгодження взаємодії користувачів: Автореферат випускної кваліфікаційної роботи на здобуття бакалаврського рівня вищої освіти зі спеціальності «122 Комп'ютерні науки». - Одеса: ОТУШ, 2023 - 12 с.

Робота присвячена дослідженням, розвиненню та впровадженню інформаційної системи, головним призначенням якої є узгоджена взаємодія між користувачами. Додаткові функції системи включають реєстрацію, авторизацію та автентифікацію, засоби обмеження доступу до контенту, ведення журналів активності користувачів системи.

Актуальність роботи визначається поширеністю задач, які ґрунтуються на оперативній фільтрації результатів, необхідністю врахування у складі комплексної оцінки волатильності результата місцезнаходження суб'єкта пошуку, часу у який потрібно виконати роботу, а також за категорією праці. Це дозволяє класифікувати задачі як такі, що формуються в умовах невизначеності або неповної визначеності вихідних даних.

Об'єктом дослідження у рамках даної роботи виступає алгоритм складання комплексної оцінки результатів за багатьма критеріями в умовах неповної визначеності вихідних умов. Предметом дослідження обрано вибір категорії за якою відбувається формування запиту. У якості задач дослідження було встановлене наступне:

- провести огляд інформаційних систем, що враховують відомості про локацію користувачів, встановити характерні риси та напрями удосконалення.
- визначити програмні засоби (інтерфейси API), які дозволяють отримувати дані про працю і працівників конкретної категорії, встановити їх особливості та ступінь вживаності для поставлених завдань.
- скласти програмне забезпечення, реалізувавши у ньому результати, досягнуті у попередніх задачах, провести випробування за різних умов та для різних пристроїв.
- впровадити у систему програмні засоби захисту даних для покращення показників інформаційної безпеки створеної системи.

Методи дослідження, що використані для досягнення поставлених задач, утворені суперпозицією методів системного аналізу, моделювання та когнітивної методології.

ЗМІСТ

ВСТУП

ТЕХНІЧНЕ ЗАВДАННЯ ДО ПРОЕКТУ

Мета проекту:

Створити інформаційну систему узгодження взаємодії користувачів під назвою "Busy Bee" для надання платформи, де користувачі зможуть розміщувати завдання та знаходити виконавців.

Основна функціональність:

Розміщення завдань, пошук виконавців, управління користувачами, платіжна система.

● Вимоги до бекенду:

- ○ Архітектура:
  - ■ Використання монолітної архітектури.
- ○ Технології:
  - ■ Мова програмування: C#.
  - ■ Фреймворк: ASP.NET WebApi.
  - ■ ORM: Entity Framework Core для роботи з базою даних PostgreSQL.
  - ■ Мапер: AutoMapper для перетворення даних між моделями.
  - ■ Валідація: FluentValidation для перевірки вхідних даних.
  - ■ Аутентифікація та авторизація: Використання Azure AD B2C для управління користувачами.
  - ■ Шаблон проектування: Clean architecture.
  - ■ Unit of Work / Repository Pattern: Для управління доступом до даних.
  - ■ Документація API: Використання Swagger для автоматичного генерування документації.
  - ■ Тестування: Використання Postman для тестування API.

● Вимоги до DevOps-інженера:

- ○ Контейнеризація: Використання Docker для контейнеризації додатка.
- ○ Хостинг: Розгортання на Azure Virtual Machine.
- ○ Система контролю версій: Використання Azure Repos для управління кодовою базою.
- ○ CI/CD інструменти: Використання Jenkins для автоматизації процесів CI/CD.

○ Моніторинг: Налаштування Prometheus і Grafana для моніторингу системи.

○ Проксі-сервер: Використання Nginx в якості проксі-сервера.

○ Планування та розгортання інфраструктури проекту.

○ Автоматизація процесів розгортання, моніторингу та масштабування системи.

○ Встановлення та налаштування інструментів для CI/CD.

○ Забезпечення безпеки та контролю якості проекту.

○ Управління конфігурацією та забезпечення надійності системи.

● Вимоги до дизайнера:

○ Проектування користувацького інтерфейсу та користувацького досвіду.

○ Використання інструментів для дизайну: Фігма, Ілюстратор, Фотошоп. (Фігма - посилання).

● Вимоги до фронтенду:

○ Фреймворк: Використання Angular для розробки фронтенд-частини сайту.

○ Інтегроване середовище розробки: Використання WebStorm для розробки на Angular.

○ Управління залежностями: Використання npm для управління пакетами.

Загальний опис проекту:

Опис функціональності:

Розміщення завдань:

● Створення нового завдання з вказанням категорії, опису та інших необхідних полів.

● Редагування та видалення завдання.

● Управління статусом завдання (відкрите, в роботі, завершене).

Пошук виконавців:

● Пошук виконавців за категоріями, навичками, локацією та іншими параметрами.

● Відображення списку доступних виконавців з основною інформацією та рейтингом.

● Фільтрація результатів пошуку та сортування за різними критеріями.

Оцінки та відгуки:

● Можливість оцінювати виконавців та залишати відгуки після завершення завдання.

● Відображення рейтингу та відгуків на профілях виконавців.

Цільова аудиторія:

● Визначення цільової аудиторії відповідно до виду завдань та послуг, пропонованих на сайті.

● Аналіз потреб і очікувань користувачів щодо функціональності та користувацького досвіду.

Розширюваність:

● Розробка модульної архітектури, що дозволяє легко додавати нові функції та можливості.

● Розгляд можливості інтеграції з іншими сервісами або платформами для розширення функціональності сайту.

Дизайн та користувацький інтерфейс:

Загальний стиль дизайну:

● Визначення бажаного стилю дизайну (мінімалістичний).

● Визначення колірної палітри, шрифтів та інших візуальних елементів, що відповідають обраному стилю.

Макети та прототипи:

● Створення макетів користувацького інтерфейсу для різних сторінок та функціональних елементів сайту.

● Розробка інтерактивних прототипів для демонстрації користувацької взаємодії та навігації по сайту.

База даних та зберігання даних:

Вибір та опис бази даних:

● Розробка структури бази даних, включаючи таблиці, зв'язки між ними та ключові поля.

● Врахування вимог до продуктивності та масштабованості бази даних.

Зберігання даних:

● Визначення даних, які повинні зберігатися в базі даних, включаючи інформацію про користувачів, завдання, коментарі тощо.

● Розробка моделей даних та визначення зв'язків між ними.

Безпека:

Аутентифікація та авторизація:

- Реалізація процесів аутентифікації користувачів з використанням Azure AD B2C.
- Встановлення прав доступу та ролей для різних типів користувачів.

Захист даних:

- Застосування заходів безпеки для захисту користувацьких даних, включаючи шифрування зберігання та передачі даних.

Інфраструктура:

Контейнеризація:

● Використання Docker для контейнеризації бекенд- та фронтенд-частини додатка.

● Створення Docker-контейнерів для різних компонентів додатка та їх оркестрація.

Хостинг:

● Розгортання додатка на Azure Virtual Machine або іншому облачному сервісі.

CI/CD:

● Використання Jenkins для налаштування процесів CI/CD.

● Автоматизація процесів розгортання, тестування та випуску нових версій додатка.

Моніторинг:

● Налаштування Prometheus та Grafana для моніторингу системи.

● Відстеження метрик продуктивності, завантаження та інших параметрів для забезпечення стабільності та швидкодії сайту.

Система контролю версій:

Використання системи контролю версій:

● Інтеграція проекту з Git для управління версіями та історією змін.

● Створення репозиторію Git та налагодження процесу спільної роботи розробників.

Гілкування та злиття:

- Визначення стратегії гілкування та злиття коду, наприклад, використання моделі Git Flow.
- Створення та керування різними гілками для розробки нових функцій, виправлення помилок та випуску стабільних версій.

Робота з командою розробників:

- Опис угод щодо найменування гілок, коментарів до комітів та інших правил для спільної роботи над проектом.
- Визначення процесу рецензування коду та схвалення змін.

Інтеграція з CI/CD:

- Використання інструментів CI/CD, таких як Jenkins, для автоматизації процесів збірки, тестування та розгортання додатка.
- Налаштування пайплайнів CI/CD, що включають етапи збірки, тестування, аналізу коду та розгортання на сервері.

Управління версіями та тегування:

- Опис правил для присвоєння версій релізам, виправленням помилок та іншим змінам.
- Використання тегів Git для маркування важливих точок в історії проекту, таких як релізи або майлстоуни.

Резервне копіювання:

- Розробка стратегії резервного копіювання репозиторію Git для забезпечення збереженості коду та історії змін.

Управління проектом:

Створення плану проекту:

● Визначення етапів розробки, термінів та завдань.

● Встановлення пріоритетів та розподіл ресурсів. Комунікація та співпраця:

● Встановлення засобів комунікації та співпраці між членами команди.

● Використання інструментів для спільного редагування та обміну документами.

Звітність:

- Вихідний код повинен оформлятися у відповідності до загальноприйнятих стандартів, на зразок стандартів GNU http://www.gnu.org/prep/standards/, у тому числі з додержанням вимог до коментування та документування коду.
- Зберігання вихідних кодів ПЗ повинно здійснюватись за допомогою системи контроля версій (вибір конкретної системи узгоджується групою проекту). Структура файлів та каталогів повинна відбивати архітектуру ПЗ, на зразок:
  - root
  - Backend
  - Web
- Підготовка звітів про стан проекту та результати роботи.
- Презентація результатів перед командою та замовником проекту.

РОЗДІЛ 1

ЗАГАЛЬНА ХАРАКТЕРИСТИКА СИСТЕМИ

Мета проекту полягає у створенні інформаційної системи "Busy Bee", яка надасть платформу для узгодження взаємодії користувачів. Ця система дозволить користувачам розміщувати завдання та знаходити виконавців.

Основна функціональність включає такі можливості: розміщення завдань, пошук виконавців, управління користувачами та платіжну систему.

- Для реалізації бекенду використана монолітна архітектура. Мовою програмування для бекенду використано C#, а фреймворком - ASP.NET WebApi. Для роботи з базою даних PostgreSQL використано Entity Framework Core. Для перетворення даних між моделями використано AutoMapper, а для перевірки вхідних даних - FluentValidation. Для аутентифікації та авторизації користувачів буде використовується Azure AD B2C. Для організації проекту застосовано шаблон Clean Architecture. Документація API згенерована автоматично за допомогою Swagger, а для тестування API використовується Postman.

- DevOps-інженер зробив контейнеризацію додатка з використанням Docker та розгортання його на Azure Virtual Machine. Для управління кодовою базою використано Azure Repos, а процеси CI/CD автоматизовані з використанням Jenkins. Для моніторингу системи налаштовані Prometheus і Grafana, а Nginx використовують як проксі-сервер. Також проведені планування та розгортання інфраструктури проекту, а також забезпечення безпеки та контролю якості.

- Дизайнер спроектував користувацький інтерфейс та користувацький досвід. Для цього використовуються інструменти дизайну, такі як Фігма, Ілюстратор і Фотошоп. (Фігма - посилання)

- Фронтенд розроблений з використанням фреймворка Angular, а розробка відбувається в інтегрованому середовищі розробки WebStorm. Управління залежностями здійснюватись за допомогою npm.

Проект передбачає розміщення завдань з можливістю створення, редагування та видалення завдань, а також управлінням їх статусом. Крім того, користувачі зможуть здійснювати пошук виконавців за різними параметрами, переглядати профілі виконавців з основною інформацією та рейтингом, а також залишати оцінки та відгуки.

Розробка системи зроблена модульною, що дозволить легко додавати нові функції та розглядати можливості інтеграції з іншими сервісами або платформами для розширення функціональності сайту.

РОЗДІЛ 2

ПРОЦЕС РОЗРОБКИ

Розробка інформаційної системи узгодження взаємодії почалась з дослідження функціоналу та вимог до проекту. Кожен учасник команди вирішував які технології використовувати та як оптимізувати процес розробки задля більшої гібкості внесення змін у майбутньому.

На початку було вирішено, що фронтенд буде розроблятися за допомогою бібліотеки React, але пізніше було виявлено, що фреймворк під назвою Angular підходить більше …

Далі був вибір назви проекту та на основі чого, був обраний подальший стиль усього проекту. Так як уся команда вирішила зупинитися на варіанті “Busy Bee”, то і весь стиль проекту був обраний відповідно до кольорів бджоли.

На початку було завдання зробити першу сторінку проекту. Бекенд був відповідальний за розробку системи зберігання і створення ієрархії “Категорії”. Фронтенд був відповідальний за створення удобного інтерфейсу відображення цих категорій з можливістю поширення кількості без втручання в код проекту.

Дев-опс інженер обрав, де підняти сервер і налаштував конфігурацію за для автоматичного деплою проекту при оновленні коду як фронтенда, так і бекенда.

Після створення головної сторінки, було вирішено добавити головний функціонал, а саме створення та прийняття заказів на сайті. Коли був створений шаблон, за допомогою якого можна було налаштовувати кожну категорію(сторінку створення заказу), було вирішено зробити авторизацію\реєстрацію. Так як нам потрібні працівники і замовники, що могли взаємодіяти з новим функціоналом.

Коли були створені сторінки користувачів, а авторизація\реєстрація були налаштовані та підключені, залишалось добавити сторінки виводу працівників які бажали приймати закази конкретного вида праці, та сторінку самих заказів від замовників.

РОЗДІЛ 3

АНАЛІЗ ПРОЕКТУ

Проект “BusyBee” націлений на працездатну аудиторію, яка шукає швидку роботу, яка відповідає їх вмінням. Або замовники, які шукають спеціаліста у конкретній сфері, які можуть швидко виконати поставлену задачу.

При обиранні типу проекту, було вирішено, що кількість сайтів які надають можливість розміщення вакансій на різни види праці було забагато. Але проектів що надають можливість швидко шукати людей, або зацікавлених осіб під конкретні завдання, було недостатньо. У важкі часи коли багато людей не можуть знайти стабільне місце праці, наш проект допоможе знайти швидку роботу та допомогу тим, хто шукає можливості.

З цією ідеєю і був створений проект “BusyBee”. Бажання допомогти усім, хто шукає допомоги. Усім, хто хоче допомогти іншим. І тим, хто не байдужий до інших. Тому ми хочемо щоб кожен мав шанс на покращення життя, яким би воно не було важке.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Angular - The web development framework for building the future
   https://angular.io
2. Angular Material - Material Design components for Angular
   https://material.angular.io
3. Bootstrap widgets - The angular way (Angular widgets built from the ground up using Bootstrap 5 CSS with APIs designed for the Angular ecosystem.)
   https://ng-bootstrap.github.io/#/home
4. Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency.
   https://git-scm.com