Міністерство освіти і науки України

Львівський національний університет імені Івана Франка

Факультет електроніки та комп’ютерних технологій

Звіт про виконання лабораторної роботи №2

**“Ознайомлення з підходом до розробки Waterfall”**

Виконав студент групи ФеП-41:

Радь О. Ю.

Перевірив:

доцент Франів В. А.

**Львів 2024**

**Мета роботи:** Вивчити та зрозуміти основні принципи та етапи методології розробки Waterfall. Оволодіти ключовими концепціями цього підходу, включаючи послідовність етапів, взаємодію між ними та роль кожного етапу в життєвому циклі розробки програмного забезпечення.

**Введення:** У рамках цієї лабораторної роботи буде розглянуто процес розробки веб-сайту для бібліотеки, починаючи з аналізу вимог користувачів та закінчуючи випробуваннями та впровадженням готового продукту. Кожен етап розробки буде розглянутий та документований згідно з методологією Waterfall, щоб забезпечити якість та стабільність розробленого веб-сайту.

**Основна частина**

1. **Аналіз вимог**
   1. Функціональні вимоги:

* Система має забезпечувати можливість реєстрації та авторизації користувачів.
* Користувачі повинні мати доступ до каталогу книг з можливістю пошуку та фільтрації.
* Система повинна підтримувати процес видачі, повернення та доставки книг користувачами.
* Користувачі повинні мати можливість стежити за своєю бібліотекою (позначати книги як «прочитані», «лайкати» книги, відслідковувати замовлення та штрафи).
  1. Нефункціональні вимоги:
* Веб-сайт повинен мати зручний інтерфейс для користувачів.
* Система повинна бути стабільною та швидкою навіть при великому навантаженні.
* Захист особистих даних користувачів є важливим завданням.
  1. Специфікації вимог:
* Користувачі повинні мати можливість реєструватися за допомогою електронної пошти, номеру телефону та пароля, або за допомогою аккаунту Google.
* У каталозі книг користувачі можуть шукати за назвою, автором, жанром тощо.
* Для процесу замовлення книги користувачеві потрібно ввести свій ідентифікаційний номер, адресу та дату доставки.

1. **Проектування**
   1. **Архітектурне проектування:**
2. Структура системи:

* Система буде розділена на фронтенд і бекенд.
* Фронтенд буде розроблений за допомогою бібліотеки React, що забезпечить швидку та динамічну взаємодію з користувачем у веб-браузері.
* Бекенд буде реалізований на платформі Node.js, яка забезпечить обробку запитів користувача та взаємодію з базою даних.

1. Взаємозв'язки компонентів:

* Фронтенд буде взаємодіяти з бекендом через REST API.
* Бекенд в свою чергу буде звертатися до бази даних PostgreSQL для отримання необхідної інформації.
  1. **Дизайн інтерфейсу:**

1. Макети сторінок:

* Головна сторінка: міститиме пошукове поле та каталог книг.
* Сторінка реєстрації: форма для введення даних користувача.
* Сторінка каталогу книг: відображення списку книг та можливість фільтрації та пошуку.
* Сторінка книги: міститиме інформацію про книгу, та можливість замовити книгу

1. Взаємодія з користувачем:

* Форми реєстрації та входу будуть створені з використанням компонентів React, що забезпечить динамічну інтерактивність.
* Навігаційне меню забезпечить швидкий доступ до основних функцій системи.
  1. **Технічне проектування:**

1. Вибір технологій:

* Фронтенд: React.
* Бекенд: Node.js.
* База даних: PostgreSQL.

1. **Реалізація**
   1. **Фронтенд (React):**
2. Компоненти:

* Створення компонентів для головної сторінки, сторінки реєстрації, каталогу книг тощо з використанням бібліотеки React.
* Реалізація інтерактивних елементів і форм для забезпечення зручності взаємодії з користувачем.

1. Запити до сервера:

* Налаштування запитів до сервера для отримання та надсилання даних через REST API.
* Обробка отриманих даних та оновлення відповідних компонентів інтерфейсу.
  1. **Бекенд (Node.js):**

1. Створення сервера:

* Створення серверної частини за допомогою Node.js та фреймворка Express.
* Налаштування маршрутів для обробки запитів від клієнтської частини.

1. Зв'язок з базою даних:

* Підключення до бази даних PostgreSQL.
* Написання запитів SQL для зберігання та отримання даних про книги, користувачів тощо.
  1. **База даних (PostgreSQL):**
* Створення таблиць у базі даних для зберігання інформації про книги, користувачів, видачі книг тощо.
* Встановлення зв'язків між таблицями за допомогою ключів.

1. **Тестування**
   1. План тестування:
2. **Функціональне тестування:**

* Перевірка реєстрації користувача та входу в систему.
* Перевірка пошуку та фільтрації книг у каталозі.
* Тестування процесу доставки та повернення книг користувачами.

1. **Навантажувальне тестування:**

* Симуляція одночасної роботи багатьох користувачів для перевірки стабільності та продуктивності системи.

1. **Тестування безпеки:**

* Перевірка на вразливості веб-додатку до ін'єкцій SQL тощо.
* Аналіз заходів безпеки, таких як захищений пароль, шифрування даних тощо.
  1. Виконання тестів:

1. **Автоматизоване тестування:**

* Використання інструментів для автоматизованого тестування функціональності, таких як Jest або Mocha для тестування React компонентів.
* Використання інструментів для тестування API, таких як Supertest для тестування REST API.

1. **Ручне тестування:**

* Ручне тестування інтерфейсу користувача.
* Перевірка реакції системи на непередбачувані дії користувачів та введення некоректних даних.
  1. Документування результатів:

1. **Фіксація помилок:**

* Запис всіх виявлених під час тестування помилок у спеціальній системі управління помилками (наприклад Jira).

1. **Тестова звітність:**

* Створення звіту про результати тестування, що містить інформацію про кількість пройдених тестів, виявлені помилки та їх статус.

**Висновок:** Методологія Waterfall є традиційною моделлю управління проектами, що передбачає послідовний перехід через етапи розробки, починаючи від аналізу вимог і завершуючи впровадженням та підтримкою системи. Вона дозволяє докладно документувати кожен етап проекту та забезпечити високу якість та стабільність продукту. Однак вона може бути менш гнучкою в порівнянні з іншими методологіями, і не завжди підходить для проектів з високою ступенем невизначеності або вимогами, які швидко змінюються. Проте, Waterfall може бути ефективним методом управління проектом.