МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

“ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ”

КАФЕДРА ПРОГРАМНОЇ ІНЖЕНЕРІЇ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГИЙ УПРАВЛІННЯ

Лабораторна робота № 5

з курсу «Проектування інформаційних систем»

«Документування вимог та проектування архітектури системи за допомогою мови UML. Робота з гілками у системі GIT»

Виконав:

студент групи КН 36-б

Чуркін Радислав

Перевірив:

Копп А. М.

ХАРКІВ 2019

Мета роботи:

Ознайомитись з документуванням вимог та проектуванням архітектури системи за допомогою мови UML, та навчитись працювати з гілками у системі GIT.

Завдання до лабораторної роботи:

Ознайомитись з документуванням вимог та проектуванням архітектури системи за допомогою мови UML, та навчитись працювати з гілками у системі GIT.

Хід виконання

1 Підготовка к виконанню роботи.

У створеному раніше, робочому каталозі, створено підкаталог у якому будуть виконуватись усі наступні кроки.

2 Виконання роботи

Історії користувачів.

Для документування користувацьких історій можна використовувати електронні таблиці Excel. Створивши у робочому каталозі (C:\...\Desktop\Program\lab5) файл Excel, а саме user\_story.xlsx, сформовано структуру таблиці, як показано на рисунку 1.

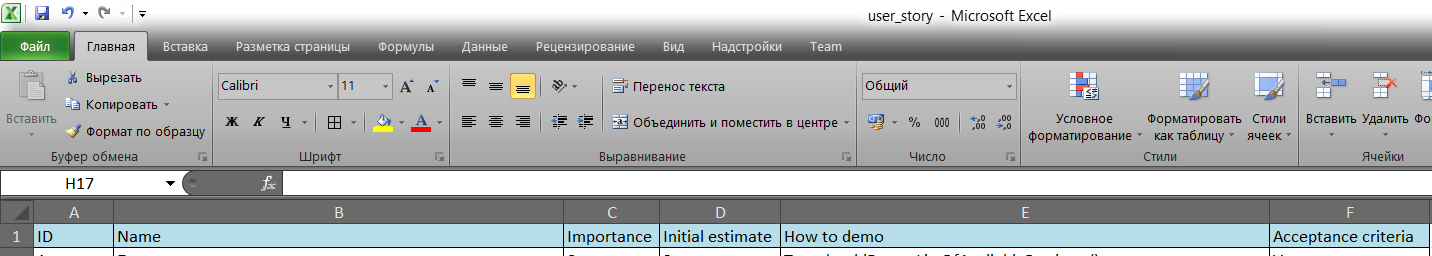


Рисунок 1 – Вміст таблиці для документування історій

Для розглянутої раніше предметної області, а саме «Деятельность предприятия по сборке и продаже компьютеров и ноутбуков» (Сборка и тестирование компьютеров) , користувацькі історії можуть виглядати, як показано на рисунку 2.

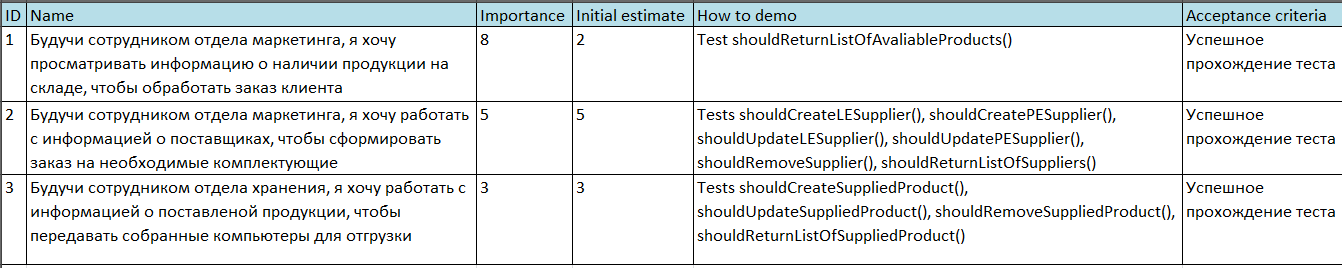


Рисунок 2 – Приклад користувацьких історій

На рисунку 3 показано, що виконанні вище дії було зафіксовано та прокоммічено.

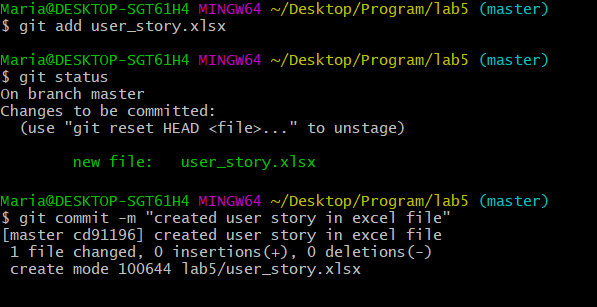
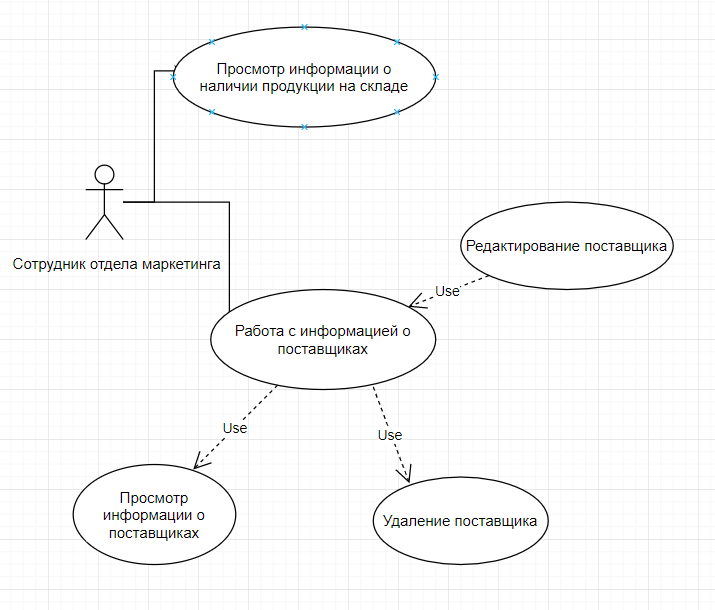


Рисунок 3 – Додавання документа у відстеження та його комміт

Сценарії використання.

За допомогою інструментарію Draw.io, виконаємо документування сформованих раніше користувацьких історій у вигляді сценарію використання.

Взаємодія системи із співробітником відділу маркетингу та відділу зберігання (рисунок 4).



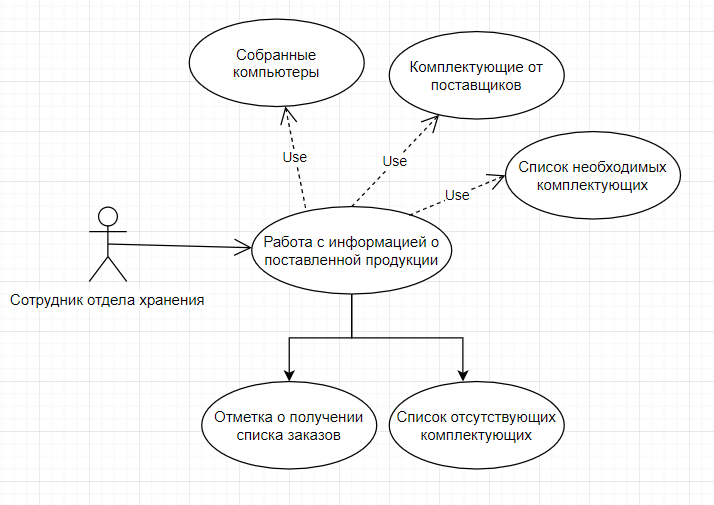


Рисунок 4 – Сценарій використання системи з відділом маркетингу та відділом зберігання

Створений файл додано до системи відстеження та закоммічено (рисунок 5).

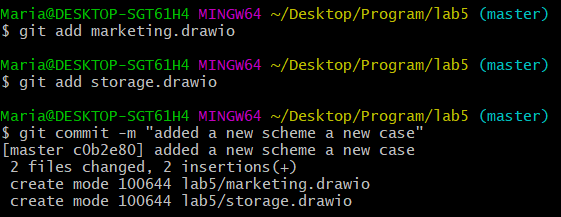


Рисунок 5 – Додавання створених схем у відстеження та їх комміти

Детальні вимоги.

Раніше було розглянуто С-вимоги (вимоги замовника).

Створено також D-вимоги (детальні вимоги).

1. Діаграма діяльності для сценарію «Створення файлу нового заказу» (рисунок 6).

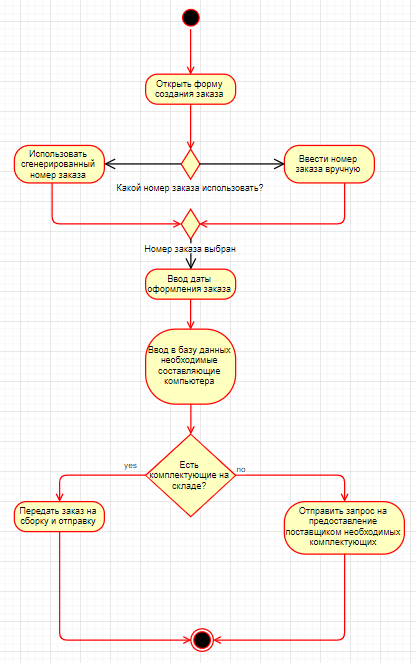


Рисунок 6 – Діаграма діяльності

1. Діаграма послідовності для сценарію «Створення файлу нового заказу» (рисунок 7).



Рисунок 7 – Діаграма послідовності

1. Діаграма класів для сценарію «Створення файлу нового заказу» (рисунок 8).

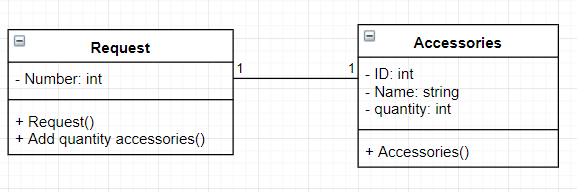


Рисунок 8 – Діаграма класів

Зафіксуємо зміни відповідним коммітом (рисунок 9).

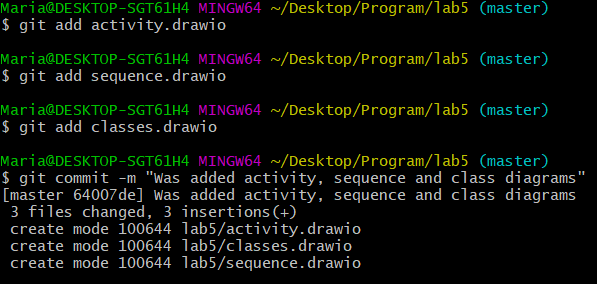


Рисунок 9 – Комміт дій, що були проведені раніше

Проектування архітектури системи.

Діаграма компонентів, системи що проектується, показано на рисунку 10.

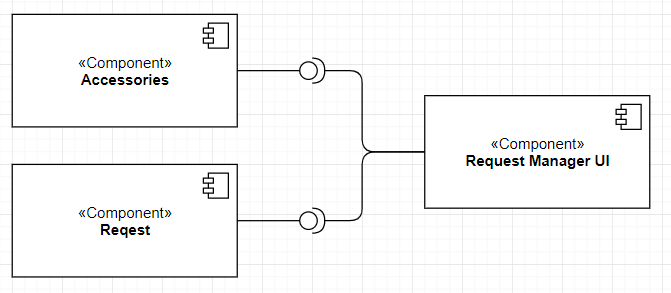


Рисунок 10 – Діаграма компонентів

Діаграму розгортання можна побачити на рисунку 11.

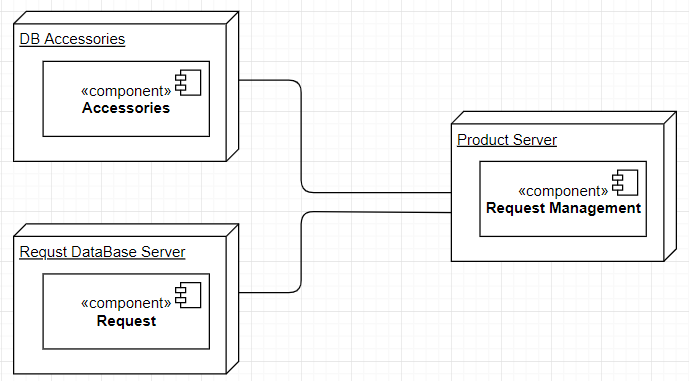


Рисунок 11 – Діаграма розгортання

Робота з голками у системі GIT.

У результаті попередньої роботи, у гілці master, вже є декілька коммітів (рисунок 12).

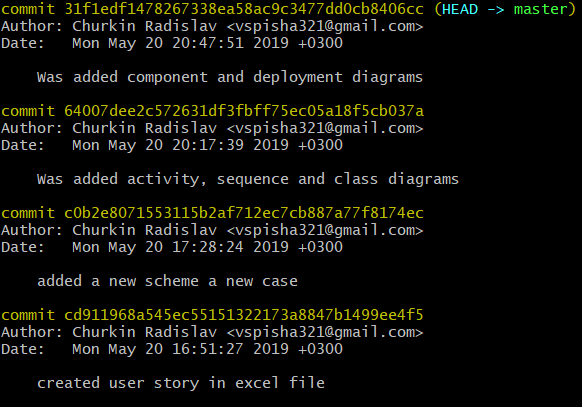


Рисунок 12 – Комміти гілки master

Створено нову гулку та переключено на неї ( рисунок 13).

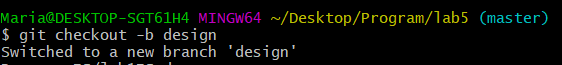


Рисунок 13 – Створення нової гілки

Зробили необхідні зміни у файлі index.html (рисунок 14).

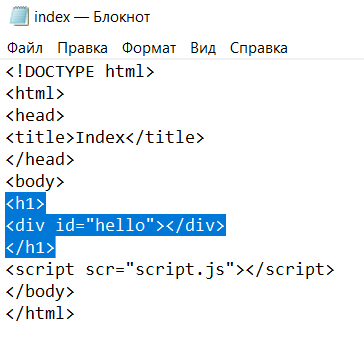


Рисунок 14 – Файл index.html

Після зміни структури файлу на рисунку 14, зроблено комміт, та перевірено його наявність (рисунок 15).

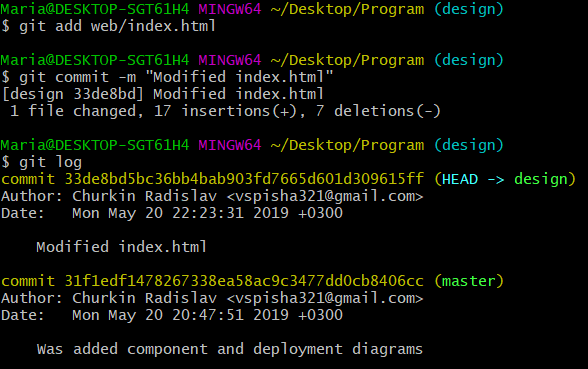


Рисунок 15 – Перевірка коммітів

Переключено на гілку master , та зроблено нову гілку (рисунок 16).

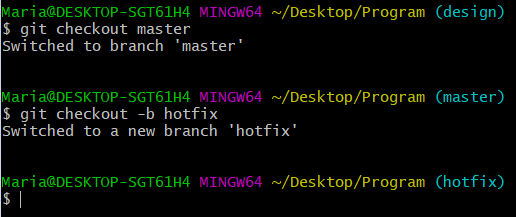


Рисунок 16 – Зафіксовані зміни

Внесено зміни, що було необхідно внести, та зроблено комміт (рисунок 17).

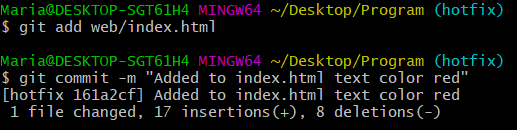


Рисунок 17 – Робота з гілкою hotfix

Проведено злиття гілок master і hotfix (рисунок 18).

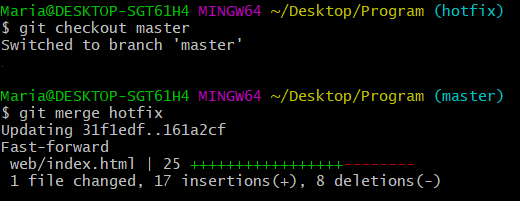


Рисунок 18 – Злиття гілок master і hotfix

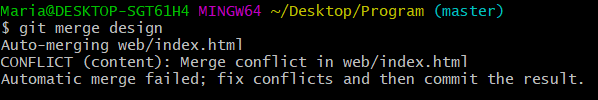
Видалимо непотрібну гілку hotfix, як на рисунку 19.



Рисунок 19 – Видалення гілки hotfix

Конфлікти при злитті.

На рисунку 20 показано, що буде конфлікт злиття гілок master і design, тому що обидві гілки мають однаковий файл.

Рисунок 20 – Відображення конфлікту при злитті гілок

Перегляд файлів, які не дозволили зробити злиття показано на рисунку 21.

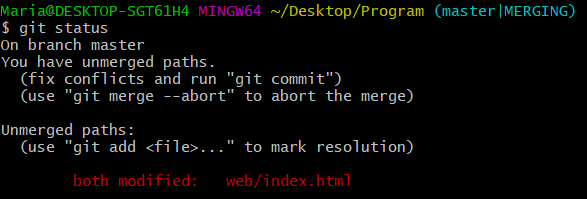


Рисунок 21 – Перегляд помилки злиття

Система Git автоматично додає стандартні маркери до файлів, які мають конфлікт, тож відкрито файл (рисунок 22).

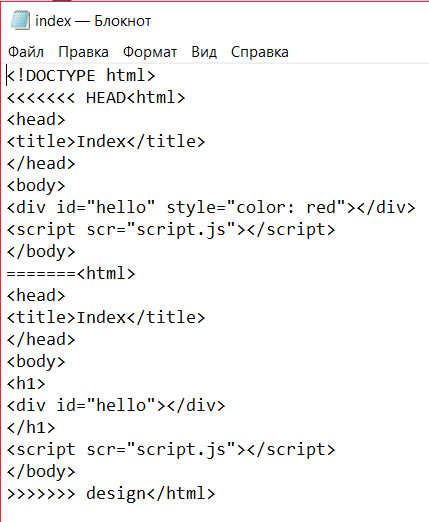


Рисунок 22 – Вигляд файлу index.html після злиття

Вирішено конфлікт наступним чином (рисунок 23).

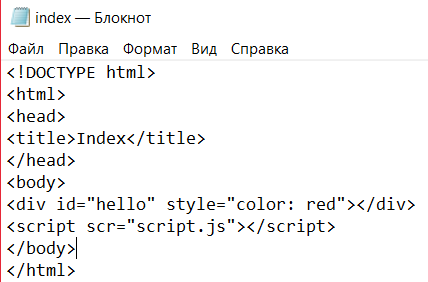


Рисунок 23 – Файл index.html

Виконано команди як показано на рисунку 24-25.

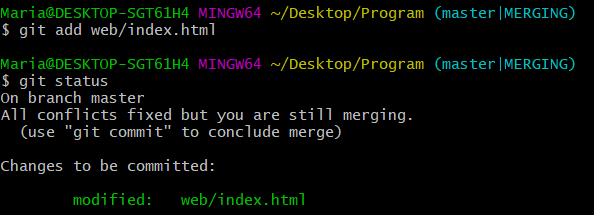


Рисунок 24 – Додавання файлу до відслідження

Конфлікт ліквідовано (рисунок 25).

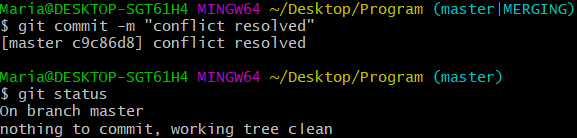


Рисунок 25 – Комміт та перевірка статусу

Висновки:

Виконуючи лабораторну роботу було отримано навичка у створенні UML діаграм, а також отримано наступні навички у використані Git:

1. створення гілок;
2. злиття гілок;
3. вирішення конфліктів злиття.