**Лабораторна робота №3**

з курсу «**Проектування інформаційних систем**»

на тему: «**Розробка життєвого циклу проекту та системи багтрекінгу. Розробка поетапного плану проекту.»**

Виконав:

студент групи ДА-61

Попов Григорій

**Мета роботи:** Вивчити типові життєві цикли розробки програмного забезпечення. Усвідомити методику роботи з багтрекінгом, навчитися створювати питання, задавати їх статус, а також навчитися оперувати статистикою багтрекінга. Скласти і описати поетапний план проекту.

**Завдання:**

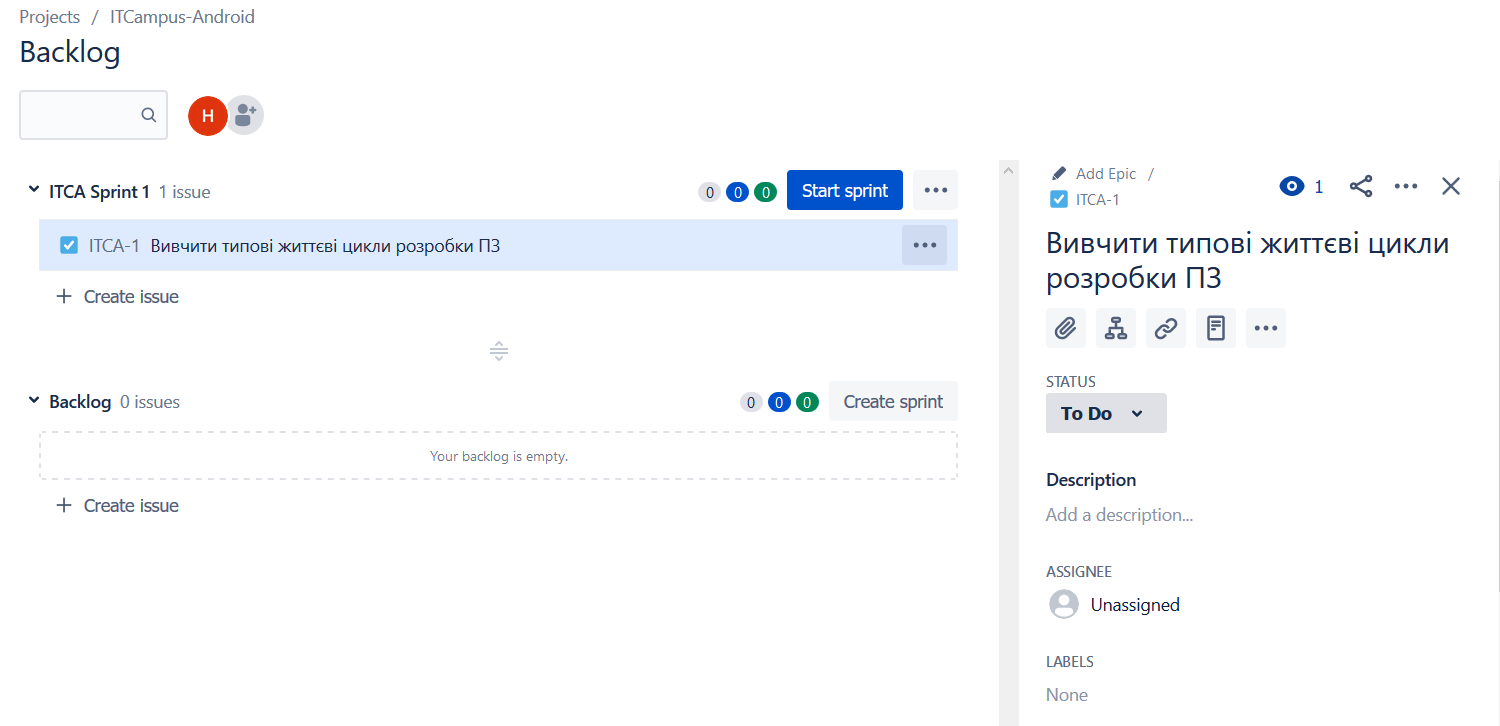
1. Використовуючи систему багтрекінга MantisBT або подібну необхідно виконати наступні дії:

* створити питання (ticket)
* видалити питання
* змінити статус
* додати коментар
* прикріпити файл
* ознайомитися з журналом змін
* ознайомитися з виведенням статистики.

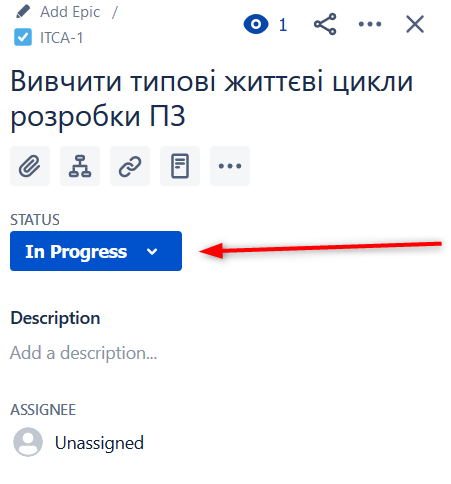
1. Використовувати методологію створення програмного забезпечення: upper/lower design methods.
2. Створити поетапний план для об'єкта проектування за допомогою OpenProj або аналогічного програмного забезпечення.
3. Оформити діаграму Ганта для етапів проекту.
4. Оформити для кожного етапу проекту список ризиків, їх вплив на проект та ймовірність виникнення.
5. Для кожного ризику розробити список анти-ризикових заходів.

**Оформлення результатів роботи**

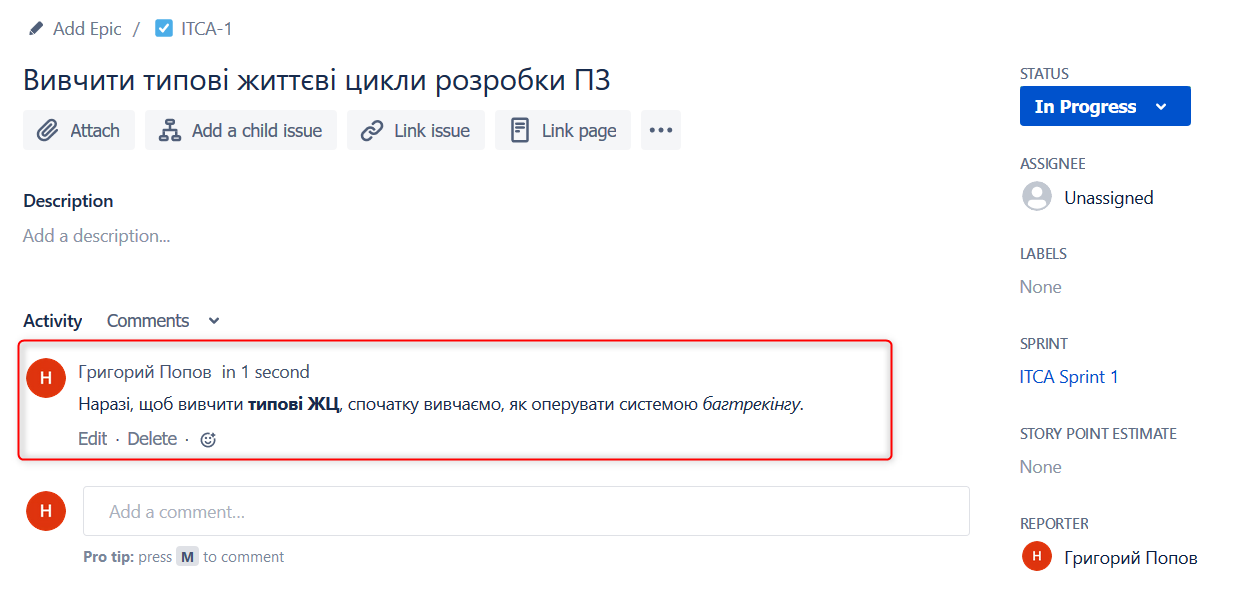
1. **Ознайомлення із системою багтрекінгу Jira**



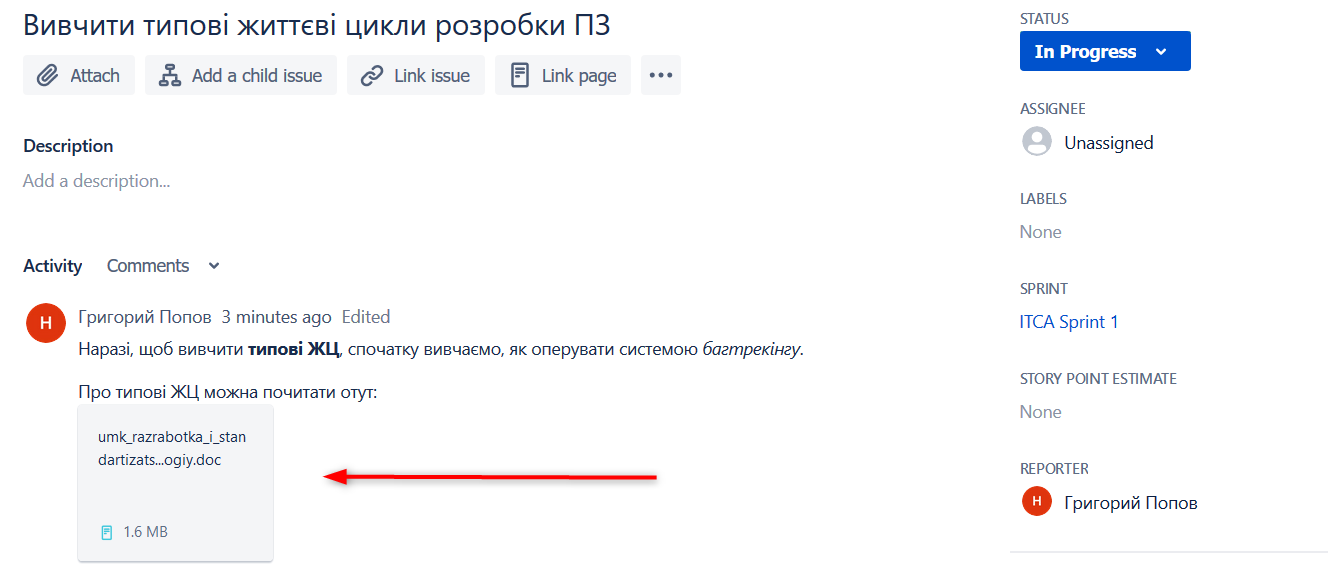
*Рис. 1а – Створення питання (issue)*

**

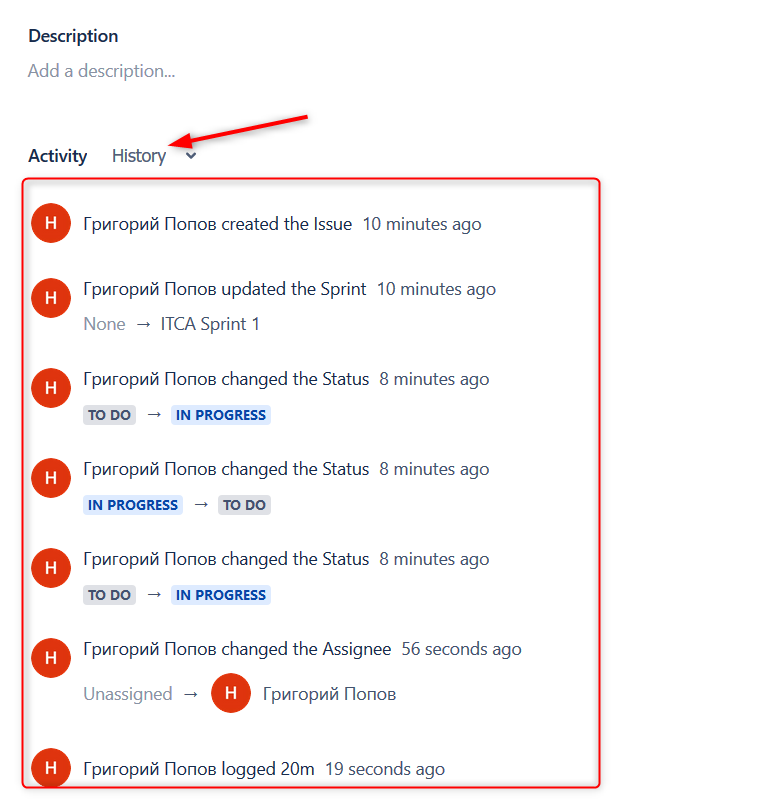
*Рис. 1б – Зміна статусу питання*

**

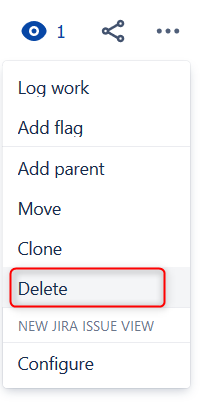
*Рис. 1в – Створення коментаря*

**

*Рис. 1г – Прикріплення файлу*

**

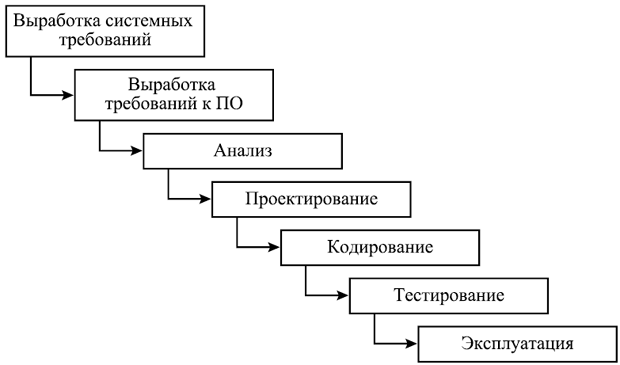
*Рис. 1д – Журнал змін*

**

*Рис. 1е – Видалення питання*

1. **Життєвий цикл розробки програмного забезпечення**

Існує декілька основних методологій розробки ПЗ. Із них найстаріший – каскадний підхід. Каскадний підхід моделює наступний ЖЦ розробки ПЗ:



*Рис. 2а – Схема ЖЦ проекту при каскадному підході*

На виході етапів 1-2 (відповідальний – бізнес-аналітик (BA) є набір вимог до ПЗ, етап 3 перевіряє їхню узгодженість та повноту, при необхідності проводяться додаткові дослідження. Зазвичай цей етап виконується також BA, але в деяких випадках йому може знадобитись поради різноманітних експертів. Етап 4 на виході видає сукупність проектної документації, тестові специфікації, архітектурну модель системи та іншу. За цей етап відповідальним є архітектор ПЗ (йому може допомагати в йьому бізнес-аналітик). Етап кодування (5) виконується, звісно, розробниками (DEV), які, додатково повинні також розробити набір модульних тестів для написаних власноруч модулів, що надалі використовуватимуться у якості регресійних тестів. Цей етап в якості вихідного продукту видає код, що описує модулі розробленої на попередніх етапах системи. Етап тестування (відповідальні тестувальники (QA), спираючись на описану раніше тестову специфікацію (acceptance criteria, unit&integration tests, non-functional tests(security, stability, performance), підтверджує певний рівень якості розробленого програмного продукту на попередньому етапі. Тоді проект переходить у стадію експлуатації та підтримки. В іншому випадку – відправляється на допрацювання на один із попередніх етапів. Каскадний підхід характеризується лінійністю та послідовністю у часі: кожний наступний етап починається лише після закінчення попереднього. В цьому полягає основний його недолік – він нестійкий до змін, які є дуже частим явищем на сучасних проектах.

Саме тому все більшу популярність отримують так звані «гнучкі» методології (agile methodologies), які, повторюють більшу частину каскадного циклу, але в значно менших масштабах, що дозволяє відносно легко повернутися до будь-якого попереднього етапу на будь-якому етапі розробки. Оскільки проект, що розробляється в результаті виконання усіх лабораторних робіт з даного предмету + РГР є дуже простим, для його виконання потребуватиметься лише одна ітерація у Scrum (різновид agile methodologies), а тому моделювання ЖЦ даного додатку простіше виконувати в межах каскадної моделі розробки.

****

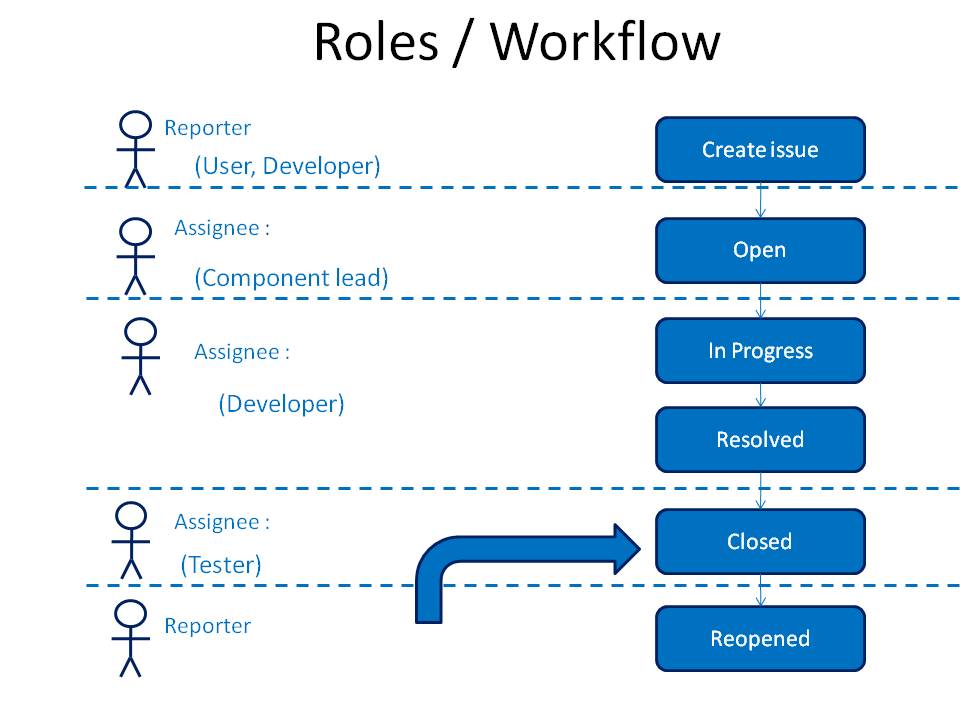
*Рис. 2в – Діаграма станів питання (ticket) даного додатку*

Може існувати кілька типів питання, як-то: задача (task), баг (bug), інцидент (incident), юсторія (story) та інші. Відповідно, кожен тип питання може мати свою діаграму станів, свої особливості та обмеження, що урізноманітнює ЖЦ програмного забезпечення.

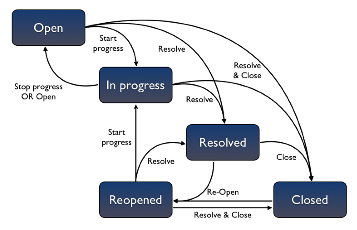
Наприклад, може існувати наступна домовленість:

* Задача створюється BA чи SA (або dev team lead в залежності від ситуації) для розробників, проходить статуси «Відкрита», «У процессі», «Імплементована», «Протестована», «Закрита».
* Баг створюється тестувальниками, має схожу діаграму станів з задачею;
* Інцидент чи проблема може бути створена для розробників підтримки, що мають обробити її «In Progress», винести рекомендацію по покращенню (“Triaged”)/закрити як дублікат чи з причини неактуальності питання (“Closed”). Із статусу “Triaged” питання має пройти шлях через відтворення тестувальником, потім оцінку бізнес-аналітика, а вже потім потрапити до черги розробника, що імплементує виправлення для даної проблеми. Тоді питання даного типу проходитиме ті ж самі статуси, що й питання типу «задача».

Загальний вигляд ЖЦ можна описати наступними рисунками:



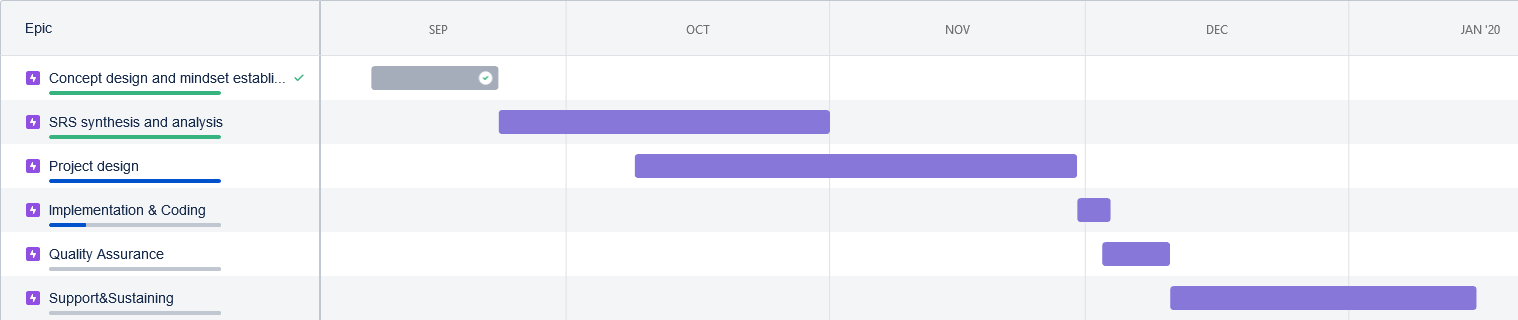
*Рис. 2г – Основні стани task та актори, що змінюють ці стани*



*Рис. 2д – Основні стани issue/bug*

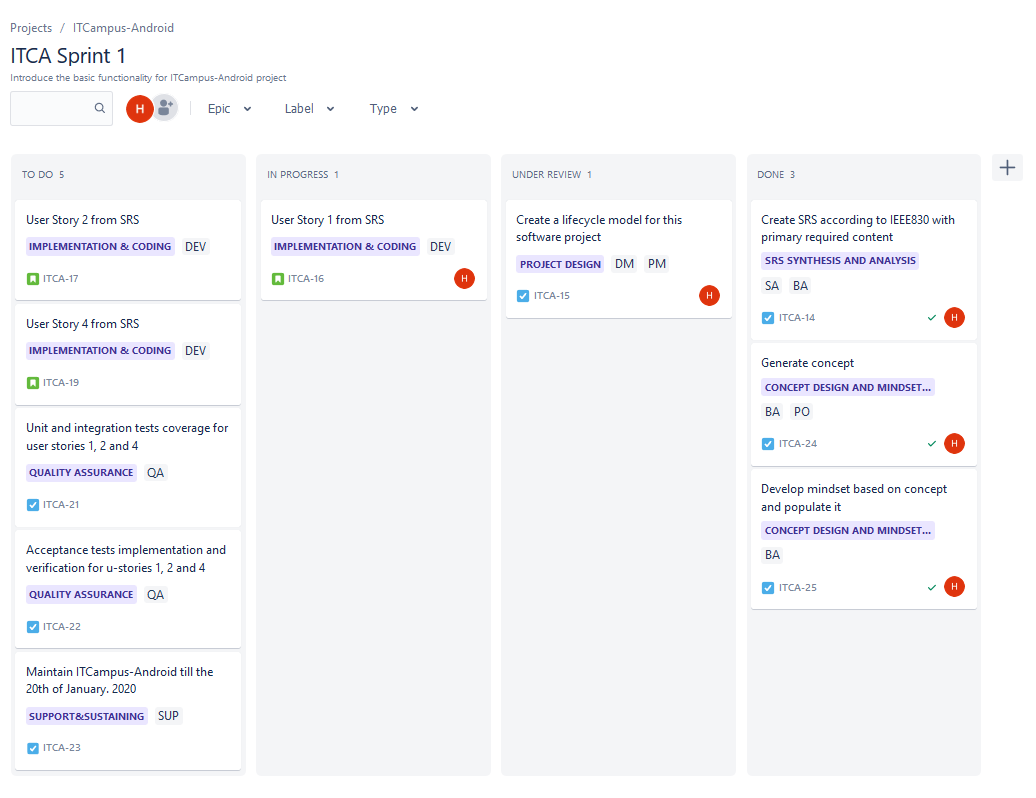
1. **Розробка поетапного плану проекту.**

Нижче наведена діаграма Ганта, що описує основні пункти плану проекта з прив’язкою до часових проміжків, коли ці пункти мають бути виконаними.



*Рис. 3а – Діаграма Ганта*

Нижче наведені задачі, оформлені як карточки (tickets) на Kanban-дошці. Оскільки має місце застосування низхідного проектування (upper/down), то й планування відбувається від загального («upper») до часткового («down»).



*Рис. 3б – Kanban-дошка проекту*

1. **Ймовірні ризики та відповідні антиризикові заходи**

|  |  |  |  |
| --- | --- | --- | --- |
| *Ризик* | Причини | ***Вплив на проект*** | ***Ймовірність виникнення*** |
| *Неправильна реалізація функціональності проекту* | Відсутність або погана комунікація між замовником та виконавцями різних рівнів (бізнес-аналітиком, проектним менеджером, розробниками). Недостатнє обговорення ТЗ або архітектури ПЗ, що розробляється. | ***Значний*** | ***Середня*** |
| *Неорганізованість робочого процесу* | Відсутність навичок проектного менеджменту або власне мотивації у керівника (менеджера) проекту. | ***Значний*** | ***Висока*** |
| *Неякісне планування етапів проекту, невиконання намічених дедлайнів* | Відсутність зворотного зв'язку між менеджером та командою. Виникає, коли проектний менеджер не вибудував робочий процес таким чином, щоб контролювати хід виконання проекту на всіх його етапах. | ***Середній*** | ***Низька*** |
| Відсутність навичок оцінки власних можливостей, оцінки часових витрат на розв’язання тієї чи іншої задачі в усіх членах проекту, хоча напередусім у розробників та тестувальників. | ***Високий*** | ***Помірна*** |
| *Втрата частини інформаційних деталей, пов’язаних з проектом* | Проект містить велику кількість аспектів, немає чіткого регламентованого засобу зберігання та структурування інформації стосовно як вимог так і самого процесу розробки. | ***Середній*** | ***Висока*** |
| *Занадто велика кількість непланованих змін у розкладі/вимогах до проекту* | Виникає в процесі розробки ПЗ, коли з'являються все нові і нові вимоги, які відсувають терміни виконання конкретних завдань, збільшують їхню оцінку. | ***Середній*** | ***Висока*** |
| *Суперечливість у вимогах* | Виникають у результаті некоректної декомпозиції вимог та/або паралельного уведення кількох вимог, що суперечать одна одній. На жаль, вона частіше за все виявляється вже на етапі інтеграції проектних модулів вже після їхньої реалізації | ***Середній*** | ***Низька*** |
| *Помилки у реалізації, що не пов’язані напряму з неправильною реалізацією функціональності* | Пов'язані з використанням нових технологій, які ще не пройшли апробацію у виробництві або інших проектах. | ***Значний*** | ***Низька*** |
| *Низька продуктивність команди* | Обумовлений відсутністю мотивації членів команди та/або їхньою низькою кваліфікацією. | ***Середній*** | ***Середня*** |
| Склад команди динамічно змінюється, більш досвідчені співробітники постійно замінюються новими, які частину часу витрачають, щоб «увійти» в проект, що знижує загальну продуктивність команди. | ***Середній*** | ***Низька*** |
| *Нелегальне поширення або використання вихідного коду проекту* | Невстановлення, недотримання чи недостатня якість запобігання порушень у сфері захисту влсаної інтелектуальної власності. | ***Значний*** | ***Низька*** |
| *Поширення або використання без дозволу чужого вихідного коду або інших матеріалів інтелектуальної власності* | Аналогічно попередньому за вийнятком того, що стосується не власної інтелектуальної власності, а власності іншої компанії чи окремого розробника. | ***Середній*** | ***Середня*** |
| *Недостача фінансових коштів для потреб проекту* | Може виникнути як з вини менеджера, який планував бюджет проекту, так і з інших причин, як-то, наприклад, нестабільність валютного курсу, що не було закладено при планування. | ***Значний*** | ***Середня*** |
| *Неактуальніть проекту на момент його завершення* | Обумовлена зміною економічної ситуації, яка складалася на ринку при плануванні.Також може бути наслідком несвоєчасного завершення проекту. | ***Помірний*** | ***Висока*** |

|  |  |
| --- | --- |
| *Ризик* | *Протиризиковий захід* |
| *Неправильна реалізація функціональності проекту* | *Ретельний підбір виконавця і узгодження ключових питань заздалегіть. Налагодження та регламентування спеціалізованних засобів комунікаціїї між учасниками проекту.* |
| *Неорганізованість робочого процесу* | *Співпраця з відповідальним і досвідченим проджект менеджером. У разі необхідності, організація курсів підвищення кваліфікації* |
| *Неякісне планування етапів проекту, невиконання намічених дедлайнів* |
| *Ризики планування* | *Узгодити методику планування заздалегідь відповідно до особливостей проекту* |
| *Втрата частини інформаційних деталей, пов’язаних з проектом* | *Використання системи багтрекінгу/управління задачами, наприклад, Jira, а також спеціального сервісу управління документацією, як-то Confluence* |
| *Занадто велика кількість непланованих змін у розкладі/вимогах до проекту* | *Проаналізовати проект заздалегідь, обрати кращу гнучку методологію розробки, у випадку очікуванної великої кількості змін* |
| *Суперечливість у вимогах* | *Узгоджувати усі паралельні вимоги між бізнес-аналітиками, якщо такі виникнуть. Обирати досвідчених спеціалістів для проведення грамотної декомпозиції вимог.* |
| *Помилки у реалізації, що не пов’язані напряму з неправильною реалізацією функціональності* | *Обговорення і використання стабільних технологій, узгодження всіх технологій з архітектором чи іншим досвідченим співробітником* |
| *Низька продуктивність команди* | *Кращий підбір команди, члени команди мають бути мотивовані у якісній роботі над проектом. Організація якісного робочого місця в офісі/гнучкого графіку роботи/можливості працювати віддалено. Робочий процес має бути узгодженим з Кодексом захисту праці і не суперечити дійсним законам стосовно робочого навантаження* |
| *Ризики, пов'язані з нездатністю впоратися зі складністю проекту* | *Розробка проекту під наглядом більш досвідченого розробника. Постійний код рев`ю перед релізом. Командні обговорення питань по проекту.* |
| *Нелегальне поширення або використання вихідного коду проекту* | *Офіційне працевлаштування, підписання договору про нерозголошення даних і неможливість використання вихідного коду поза проектами компанії* |
| *Поширення або використання без дозволу чужого вихідного коду або інших матеріалів інтелектуальної власності* | *Надати команді обширні роз’яснення щодо можливості використання опенсорс ресурсів. Купівля ліценції на матеріали, які потрібно застосувати в проекту. Попередження щодо кримінальної відповідальності за порушення авторських прав* |
| *Недостача фінансових коштів для потреб проекту* | *Створення фінансового плану до старту проекту, контроль запланованих витрат та реальних витрат на кожному етапі проекту. У випадку виявлення нестачі фінансів на конкретному етапі – спробувати залучити додаткові джерела фінансування або переглянути об`єм робіт.* |
| *Неактуальніть проекту на момент його завершення* | *Залучення на етапі планування роботи аналітика для чіткого бізнес-прогнозу. Моніторинг ситуації на ринку в процесі розробки проекту, внесення корективів або можливо переорієнтація проекту, якщо розробка перестає бути актуальною.* |

**Висновок**

В даній лабораторній роботі був сформований життєвий цикл для даного проекту, досліджено методології розробки, життєві цикли задач у вигляді діаграм станів. Проаналізовано можливі ризики, з якими зіткнеться проект, а також обрано кращі стратегії їхнього запобігання.