МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ”

Інститут Прикладного системного аналізу

Кафедра Системного проектування

**Лабораторна робота № 3.**

З дисципліни

«Проектування інформаційних систем»

На тему

***«****Розробка життєвого циклу проекту та системи багтрекінгу.*

*Розробка поетапного плану проекту.****»***

Виконав

студент групи ДА-61

Брижатенко Руслан

Київ - 2019

**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.

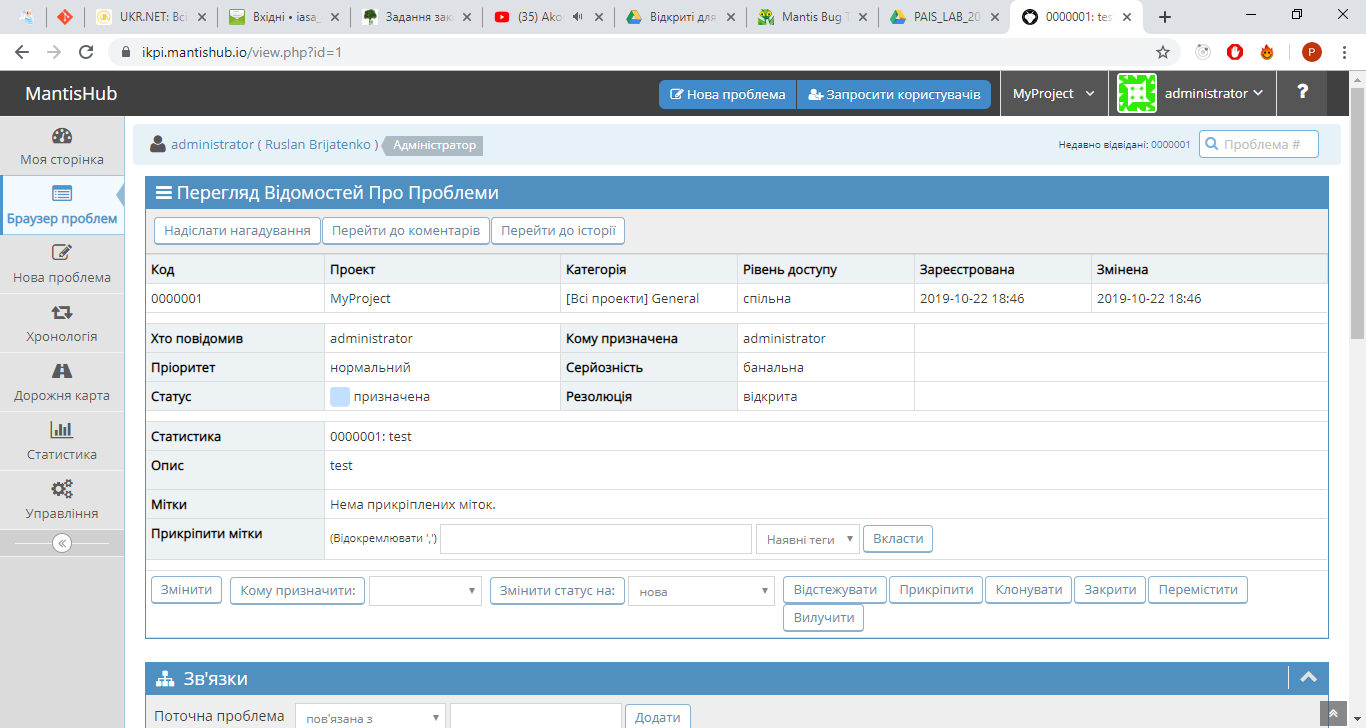
**Задача:** Вивчити зміст прикладу інформаційної системи та створити опис життєвого циклу розробки відповідного програмного забезпечення. Використовуючи систему багтрекінга MantisBT або подібну необхідно ознайомитися веденням журналу для повного циклу розробки та тестування, відтворити статуси для питання (ticket) та ролі у процесі розробки програмного забезпечення.

**Завдання**

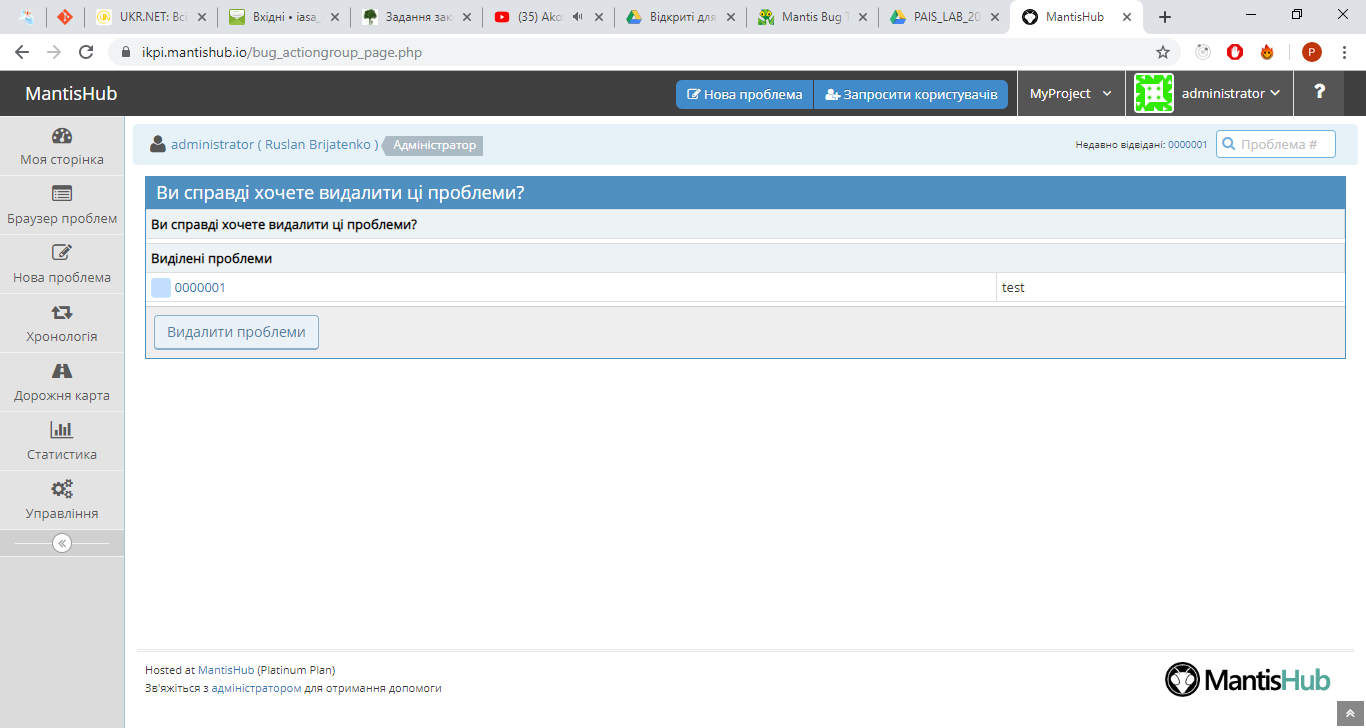
**Частина 1**. *Розробка життєвого циклу проекту та системи багтрекінгу.*

Використовуючи систему багтрекінга MantisBT або подібну необхідно виконати наступні дії:

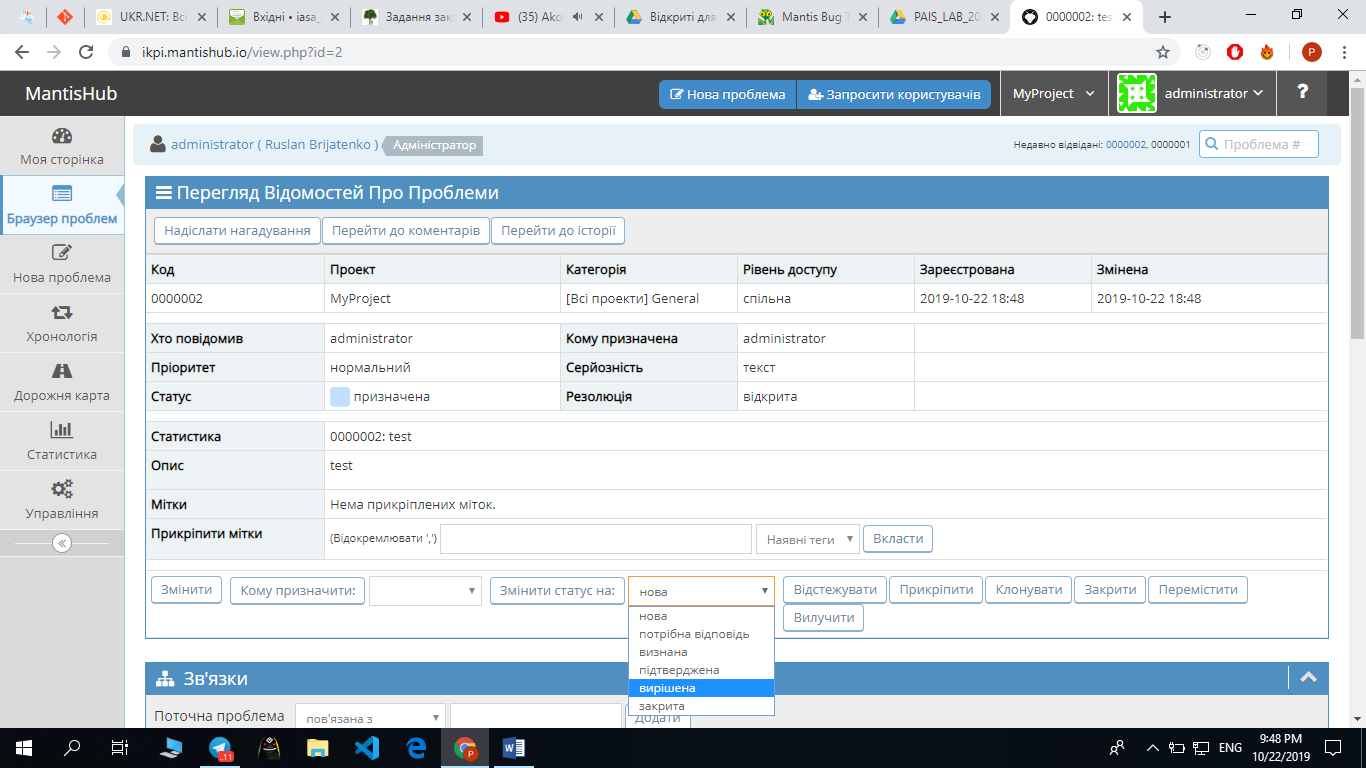
Створення питання (ticket)



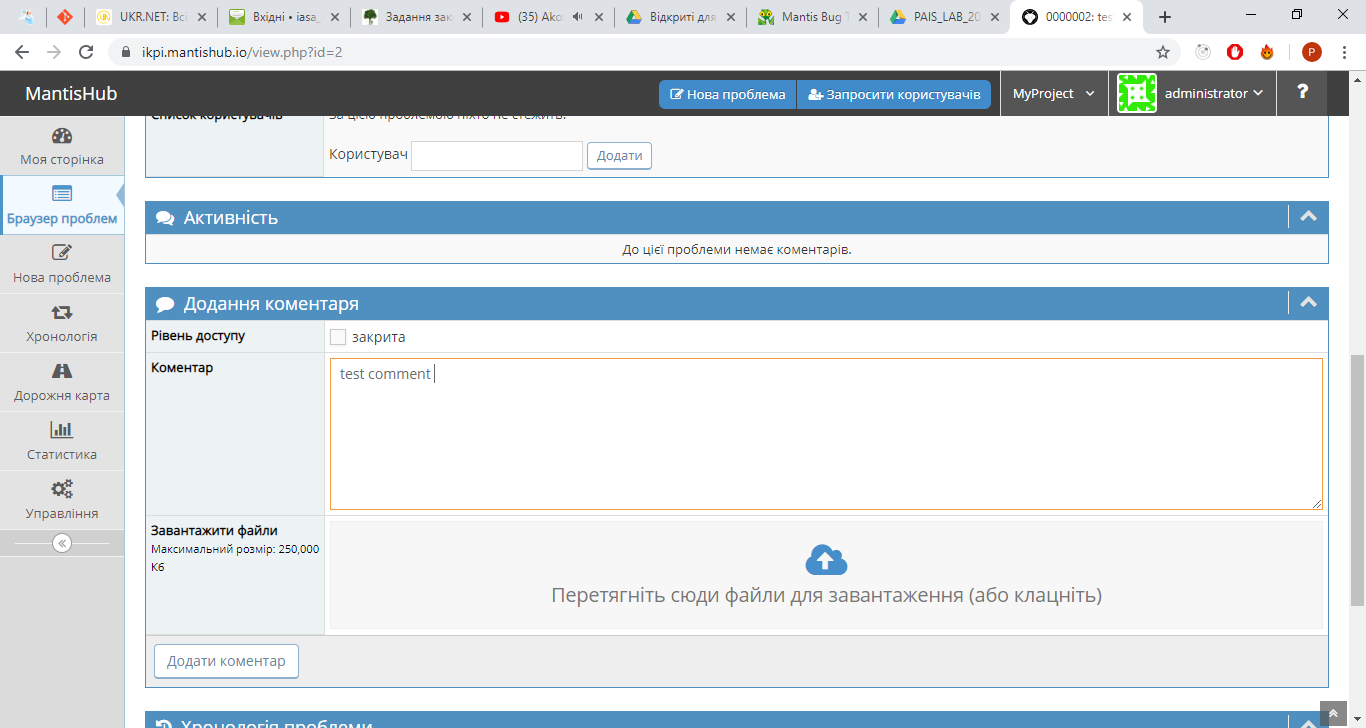
Видалити питання



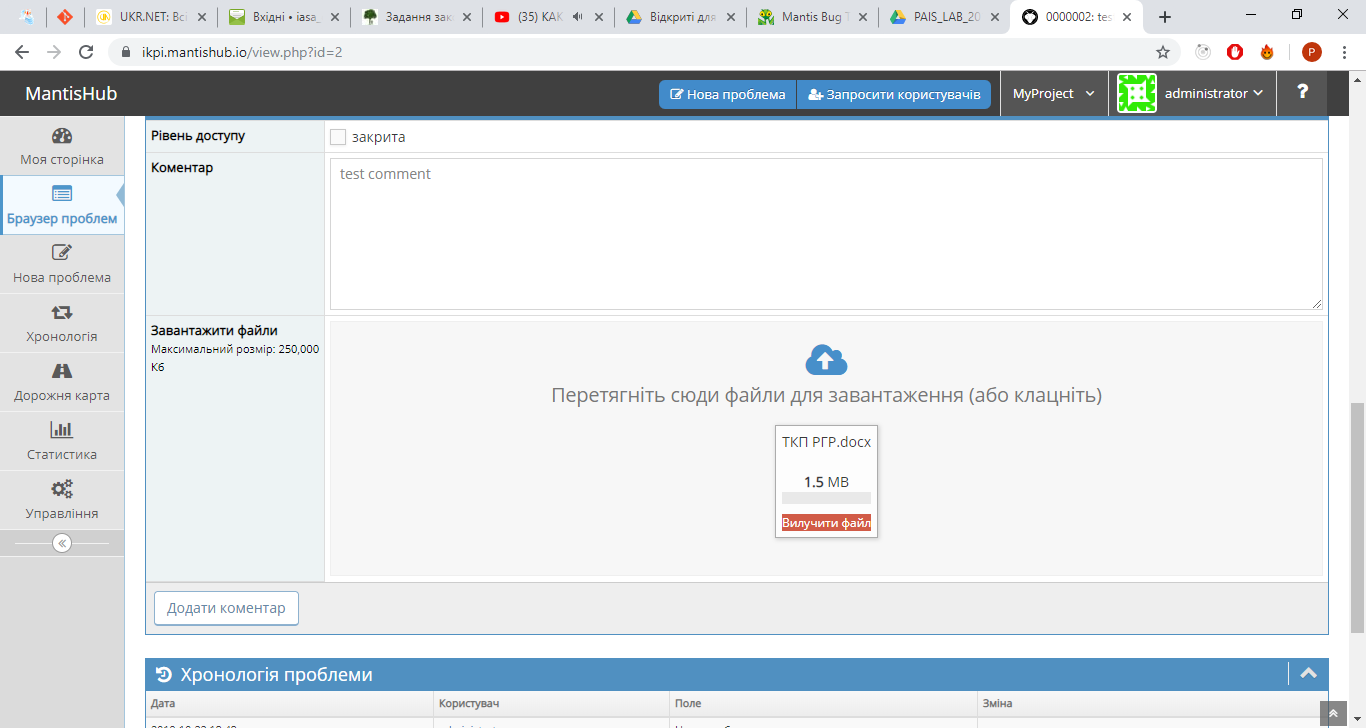
Змінити статус



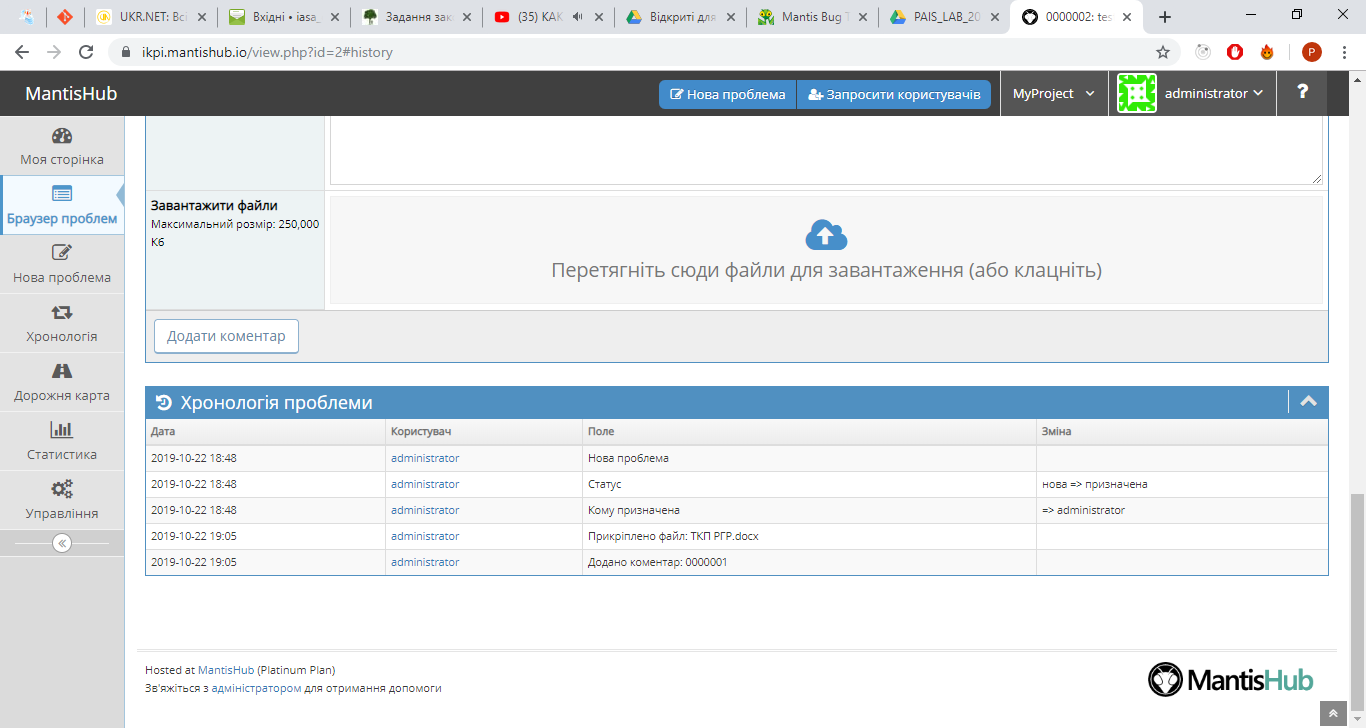
Додати коментар



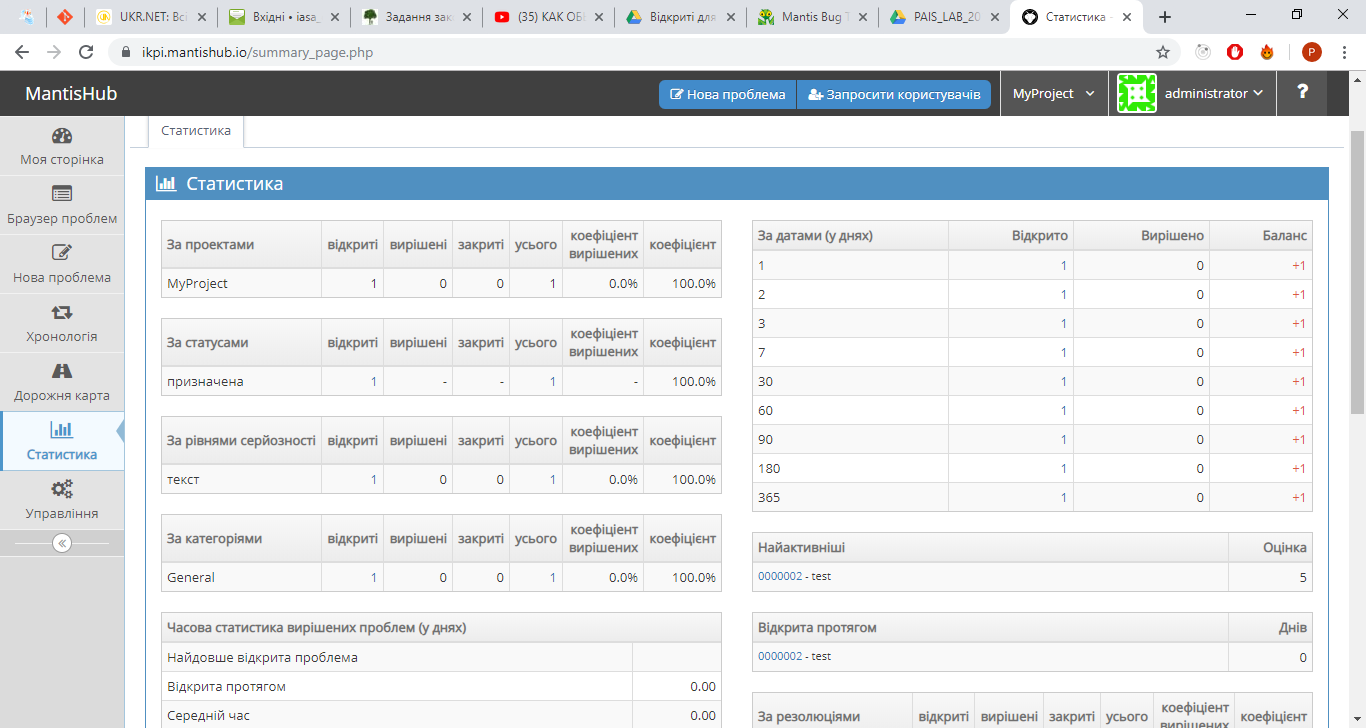
Прикріпити файл



Журнал змін



Статистика



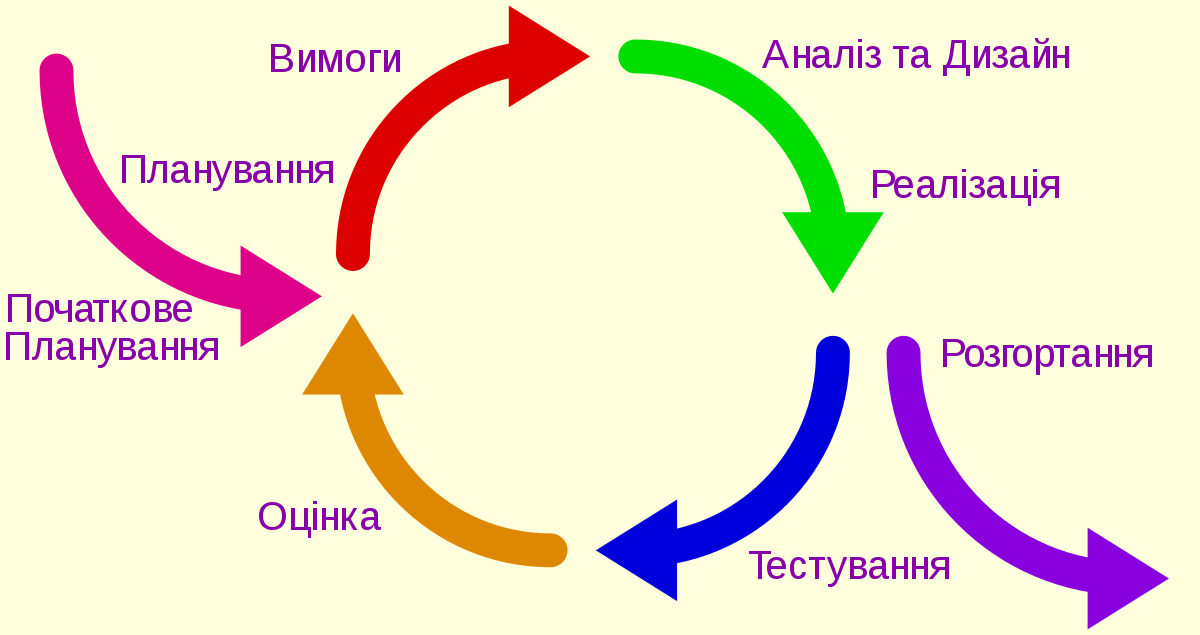
**Створити опис життєвого циклу розробки програмного забезпечення інформаційної системи.**

Каскадна модель життєвого циклу є ідеальною, оскільки лише дуже прості завдання проходять усі етапи без будь-яких ітерацій (повернень на попередні кроки процесу). Наприклад, при програмуванні може виявитися, що реалізація деякої функції неефективна і вступає у протиріччя з вимогами до продуктивності системи. У цьому випадку потрібні зміни проекту, а можливо, і переробка специфікацій. Для врахування повторюваності етапів процесу розроблення створювались альтернативи каскадної моделі.

Із таких альтернатив утворилася **ітеративна модель**. Ця модель передбачає розбиття життєвого циклу проекту на послідовність ітерацій, кожна з яких нагадує "міні-проект" з усіма фазами життєвого циклу.

Класична ітераційна модель абсолютизує можливість повернень на попередні етапи. Ця обставина відбиває істотний аспект програмних розробок: прагнення заздалегідь передбачати всі ситуації використання системи та неможливість у переважній більшості випадків досягти цього. Усі традиційні технології програмування спрямовані лише на те, щоб мінімізувати повернення. Але суть від цього не змінюється: при поверненні завжди доводиться повторювати побудову того, що вже вважалося готовим.Відмінність цієї моделі від каскадної полягає в можливості багато разів повертатися до процесу формулювання вимог і до повторної розробки версії системи з будь-якого процесу моделі.

Мета кожної ітерації в розробленні ПЗ – отримання працюючої версії програмної системи, що включає функціональність, визначену інтегрованим змістом усіх попередніх і поточної ітерації. Результат фінальної ітерації містить усю необхідну функціональність продукту. Таким чином, із завершенням кожної ітерації продукт розвивається інкрементально (нарощує функціональність).



***Стисла характеристика:***

стадії повторюються неодноразаво.

***Недоліки:***

система часто погано структурована, проект **«не прозорий»**

після уточнення вимог відкидається частина раніше виконаної роботи;

потрібні засоби для швидкого розроблення*.*

***Використання****:*

підходить для малих та середніх *проектів.*

#### **Частина 2. Розробка поетапного плану проекту.**

#### **Мета роботи:** Скласти і описати поетапний план проекту.

#### Завдання

#### Опис об'єкту взяти з SRS в лабораторній роботі № 2.

#### 2) Використовувати методологію створення програмного забезпечення: upper/lower design methods.

В процесі розробки програмного забезпечення ключову роль відіграють підходи зверху-вниз і знизу-вгору.

Підходи зверху-вниз підкреслюють планування і повне розуміння системи. Неминуче, що ніяке кодування не може розпочатися до тих пір, поки не буде досягнутий достатній рівень деталізації при проектуванні, щонайменше, деякої частини системи. Підходи зверху-вниз реалізуються шляхом прикріплення заглушок замість модуля. Однак, це, затримує тестування кінцевих функціональних блоків системи до тих пір, поки не буде завершено значний проект. Внизу підкреслюється кодування і раннє тестування, яке може початися, як тільки буде вказано перший модуль. Однак такий підхід пов'язаний з ризиком того, що модулі можуть бути закодовані без чіткого уявлення про те, як вони зв'язуються з іншими частинами системи, і що таке зв'язування може бути не таким простим, як здається на перший погляд. Повторне використання коду є одним з основних переваг висхідного підходу.

Підхід зверху-вниз був запропонований дослідниками IBM Харланом Миллсом і Ніклаус Віртом. Міллс розробив концепції структурованого програмування для практичного використання і перевірив їх в проекті 1969 року по автоматизації індексу моргу New York Times. Інженерний і управлінський успіх цього проекту привів до поширення спадного підходу через IBM і решту комп'ютерної індустрії. Серед інших досягнень Ніклаус Вірт, розробник мови програмування Pascal, написав впливову статтю «Розробка програм шляхом покрокового уточнення». Оскільки Ніклаус Вірт продовжував розробляти такі мови, як Modula і Oberon (де можна визначити модуль, не знаючи всієї специфікації програми), можна зробити висновок, що спадний програмування не було строго тим, що він просував. До кінця 1980-х років в розробці програмного забезпечення використовувалися спадні методи, а об'єктно-орієнтоване програмування допомогло продемонструвати ідею, що можна використовувати як аспекти спадного, так і висхідного програмування.

Сучасні підходи до розробки програмного забезпечення зазвичай поєднують підходи як зверху вниз, так і знизу вгору. Хоча розуміння всієї системи зазвичай вважається необхідним для хорошого проектування, що теоретично веде до низхідного підходу, більшість програмних проектів намагаються до певної міри використовувати існуючий код. Попередньо існуючі модулі дають дизайн від низу до верху. Деякі підходи до проектування також використовують підхід, при якому частково функціональна система розробляється і кодується до завершення, а потім ця система розширюється для виконання всіх вимог до проекту.

#### 3) Створити поетапний план для об'єкта проектування за допомогою OpenProj або аналогічного програмного забезпечення.

#### 4) Оформити діаграму Ганта для етапів проекту.

#### 

#### 

#### 5) Оформити для кожного етапу проекту список ризиків, їх вплив на проект та ймовірність виникнення.

#### ***Ризик:*** Ризики поганої взаємодії між замовником і виконавцем.

#### ***Причини:*** *Р*изики, пов'язані з відсутністю комунікації між виконавцем і замовником або їх представниками. Недостатнє обговорення завдань або архітектури може негативно позначитися на ПЗ що розробляється.

#### ***Етапи: 2***

#### ***Вплив на проект:*** Значний

#### ***Ймовірність виникнення:*** Середня

#### ***Ризик:*** Ризики управління проектом

#### ***Причини:*** Ризики, пов'язані з відсутністю навичок проектного менеджменту у менеджера проекту, а також з відсутністю інтересу або мотивації у нього.

#### ***Етапи: 2-5***

#### ***Вплив на проект:*** Значний

#### ***Ймовірність виникнення:*** Середня

#### ***Ризик:*** Ризики, пов'язані з недостатньою обізнаністю керуючого проектом про точний стан проекту

#### ***Причини:*** Вид ризиків, пов'язаних з відсутністю зворотного зв'язку. Він виникає, коли проектний менеджер не вибудував робочий процес таким чином, щоб контролювати хід виконання проекту на всіх його етапах.

#### ***Етапи: 3-5***

#### ***Вплив на проект:*** Незначний

#### ***Ймовірність виникнення:*** Низька

#### ***Ризик:*** Ризики планування

#### ***Причини:*** Ризики, які можуть бути пов'язані з відсутністю навичок планування за проектом як менеджером, так і виконавцями, якщо вони готують інформацію про терміни виконання робіт.

#### ***Етапи: 2-5***

#### ***Вплив на проект:*** Середній

#### ***Ймовірність виникнення:*** Середня

#### ***Ризик:*** Ризики відсутності системи контролю

#### ***Причини:*** Ризики обумовлені великою кількістю аспектів у області проектного менеджменту при розробці ПЗ, коли складно врахувати всі можливі ситуації.

#### ***Етапи: 3-5***

#### ***Вплив на проект:*** Середній

#### ***Ймовірність виникнення:*** Висока

#### ***Ризик:*** Ризик появи нових вимог

#### ***Причини:*** Ризик виникає в процесі розробки ПЗ, коли з'являються все нові і нові вимоги, які відсувають терміни і оцінку конкретних завдань.

#### ***Етапи: 2***

#### ***Вплив на проект:*** Середній

#### ***Ймовірність виникнення:*** Висока

#### ***Ризик:*** Ризик суперечливості у вимогах (декомпозиція специфікації)

#### ***Причини:*** Ризики, пов'язані з виявленням суперечливості в вимогах замовника на етапі програмування або інтеграції проекту.

#### ***Етапи: 1-2***

#### ***Вплив на проект:*** Середній

#### ***Ймовірність виникнення:*** Низька

#### ***Ризик:*** Ризики неправильно сформульованих певних системних вимог

#### ***Причини:*** Ризики, коли на самому початку проекту були некоректно сформульовані характеристики цільової системи, для якої розробляється програмне забезпечення: програмне оточення (операційна система, встановлені компоненти, сервіси і т.п.) або вимоги до апаратної частини (частота процесора, об'єм жорсткого диска, обсяг оперативної пам'яті і т.п.).

#### ***Етапи: 1-2***

#### ***Вплив на проект:*** Значний

#### ***Ймовірність виникнення:*** Висока

#### ***Ризик:*** Ризики використання нестабільних технологій

#### ***Причини:*** Ризики, пов'язані з використанням нових технологій, які ще не пройшли апробацію у виробництві або інших проектах.

#### ***Етапи: 3-5***

#### ***Вплив на проект:*** Значний

#### ***Ймовірність виникнення:*** Низька

#### ***Ризик:*** Ризики, пов'язані з нездатністю впоратися зі складністю проекту

#### ***Причини:*** Іноді проект може бути настільки складним, що команда просто може з ним не упоратися.

#### ***Етапи: 1-4***

#### ***Вплив на проект:*** Значний

#### ***Ймовірність виникнення:*** Низька

#### ***Ризик:*** Ризик низької продуктивності

#### ***Причини:*** Обумовлений тривалістю реалізації проекту. Це на самому початку проекту створює велику втрату часу, яку складно буде надолужити. При цьому доводиться або переносити терміни, або працювати в більш динамічному режимі на більш пізніх етапах проекту.

#### ***Етапи: 3-5***

#### ***Вплив на проект:*** Середній

#### ***Ймовірність виникнення:*** Середня

#### ***Ризик:*** Ризик зміни співробітників

#### ***Причини:*** Коли проект покидають ключові співробітники, які максимально володіють інформацією.

#### ***Етапи: 4-5***

#### ***Вплив на проект:*** Середній

#### ***Ймовірність виникнення:*** Середня

#### ***Ризик:*** Ризики розкрадання вихідного коду

#### ***Причини:*** Виникають, коли розробники, йдучи з компанії, забирають з собою проект, що розробляється ними і, трохи модифікувавши вихідний код, можуть продати його або використовувати в інших проектах, наприклад, у конкурентів.

#### ***Етапи: 4-5***

#### ***Вплив на проект:*** Значний

#### ***Ймовірність виникнення:*** Низька

#### ***Ризик:*** Ризики порушення закону про авторське право

#### ***Причини:*** Можуть виникнути при використанні розробниками без відома проектного менеджера чужого вихідного коду, алгоритму або бібліотеки, які захищені законом про авторське право, але не придбано або їх використання не узгоджене з автором.

#### ***Етапи: 4-5***

#### ***Вплив на проект:*** Значний

#### ***Ймовірність виникнення:*** Низька

#### ***Ризик:*** Ризики фінансових обмежень

#### ***Причини:*** Можуть виникнути як з вини менеджера, який планував бюджет проекту, так і з інших причин.

#### ***Етапи: 1-6***

#### ***Вплив на проект:*** Значний

#### ***Ймовірність виникнення:*** Середня

#### ***Ризик:*** Ризики зміни кон'юнктури ринку

#### ***Причини:*** Обумовлені зміною економічної ситуації, яка складалася на ринку при плануванні. При цьому могли закладатися фактори, актуальні на момент планування, а їх зміна не була врахована.

#### ***Етапи: 1-5***

#### ***Вплив на проект:*** Значний

#### ***Ймовірність виникнення:*** Висока

#### ***Ризик:*** Валютні ризики

#### ***Причини:*** Ризики, пов'язані з можливим виникненням збитків або додаткових доходів внаслідок несприятливої або сприятливої зміни курсів іноземних валют.

#### ***Етапи: 1-5***

#### ***Вплив на проект:*** Значний

#### ***Ймовірність виникнення:*** Низька

#### 6) Для кожного ризику розробити список анти-ризикових заходів.

#### ***Ризики:***

*Ризики поганої взаємодії між замовником і виконавцем*

***Протиризиковий захід:***

*Ретельний підбір виконавця і узгодження ключових питань заздалегіть*

#### ***Ризики:***

*Ризики управління проектом*

*Ризики, пов'язані з недостатньою обізнаністю керуючого проектом про точний стан проекту*

***Протиризиковий захід:***

*Співпраця з відповідальним і досвідченим ПМ*

#### ***Ризики:***

*Ризики планування*

*Ризик суперечливості у вимогах (декомпозиція специфікації)*

*Ризики неправильно сформульованих певних системних вимог*

*Ризик появи нових вимог*

***Протиризиковий захід:***

*Правильне та грамотне формулювання технічної документації вимог*

#### ***Ризики:***

*Ризики використання нестабільних технологій*

***Протиризиковий захід:***

*Обговорення і використання стабільних технологій, узгодження всіх технологій з архітектором чи іншим досвідченим співробітником*

#### ***Ризики:***

*Ризики відсутності системи контролю*

*Ризики, пов'язані з нездатністю впоратися зі складністю проекту*

*Ризик низької продуктивності*

*Ризик зміни співробітників*

***Протиризиковий захід:***

*Ефективний менеджмент проекту і конкурентна спроможність на фоні ринку праці*

#### **Висновки**: в ході виконання лабораторної роботи було розроблено поетапний план проекту, створено діаграму Ганта, розбито проект на етапи. Для кожного етапу розраховано ризики та розроблено список анти-ризикових заходів.