МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ХАРКІВСЬКИЙ КОМП’ЮТЕРНО-ТЕХНОЛОГІЧНИЙ

КОЛЕДЖ НТУ «ХПІ»

**ЛАБОРАТОРНА РОБОТА №1**

Виконав: Перевірив:

Лещенко Андрій Фесенко Д. В.

Харків

2017

**ЛАБОРАТОРНА РОБОТА №1**

**«Загальні відомості про Git та GitHub. Основи роботи з репозиторіями»**

**1. Тема роботи:**  Створення свого проекту та встановлення його під версійний контроль.

**2 . Мета роботи:** Навчитись створювати та керувати власним репозиторієм.

**3. Теоретичний матеріал**

GitHub – это крупнейший веб-сервис для хостинга IT -проектов и их совместной разработки. Основан на системе контроля версий [Git](https://ru.wikipedia.org/wiki/Git).

Система контроля версий (СКВ/VCS) – это система, регистрирующая изменения в одном или нескольких файлах с тем, чтобы в дальнейшем была возможность вернуться к определённым старым версиям этих файлов.

Виды СКВ: локальные; централизованные; распределенные.

[Git](https://ru.wikipedia.org/wiki/Git) относится к распределённым СКВ. Главное отличие Git'а от любых других СКВ – это то, как Git смотрит на свои данные, он считает хранимые данные набором слепков небольшой файловой системы. Для совершения большинства операций в Git'е необходимы только локальные файлы и ресурсы, т.е. обычно информация с других компьютеров в сети не нужна.

В проектах, использующих Git, есть три части: каталог Git'а (Git directory), рабочий каталог (working directory) и область подготовленных файлов (staging area).



Рисунок 1.1 – Робочий каталог, область підготовленнях файлів, каталог Git'а.

Каталог Git'а — это место, где Git хранит метаданные и базу данных объектов вашего проекта. Это наиболее важная часть Git'а, и именно она копируется, когда вы клонируете репозиторий с другого компьютера.

Рабочий каталог — это извлечённая из базы копия определённой версии проекта. Эти файлы достаются из сжатой базы данных в каталоге Git'а и помещаются на диск для того, чтобы вы их просматривали и редактировали.

Область подготовленных файлов — это обычный файл, обычно хранящийся в каталоге Git'а, который содержит информацию о том, что должно войти в следующий коммит. Иногда его называют индексом (index), но в последнее время становится стандартом называть его областью подготовленных файлов (staging area).

Каждый файл в рабочем каталоге может находиться в одном из двух состояний: под версионным контролем (отслеживаемые/tracked) и нет (неотслеживаемые/untracked).

Отслеживаемые файлы — это те файлы, которые были в последнем слепке состояния проекта (snapshot). Они, в свою очередь, могут быть неизменёнными, изменёнными или подготовленными. К изменённым относятся файлы, которые поменялись, но ещё не были зафиксированы. Подготовленные файлы — это изменённые файлы, отмеченные для включения в следующий коммит.

Неотслеживаемые файлы — это всё остальное, любые файлы в вашем рабочем каталоге, которые не входили в ваш последний слепок состояния и не подготовлены к коммиту.

Когда вы впервые клонируете репозиторий, все файлы будут отслеживаемыми и неизменёнными, потому что вы только взяли их из хранилища (checked them out) и ничего пока не редактировали.



Рисунок 1.2 – Життєвий цикл станів файлів.

Основні команди Git:

git help – виклик довідки;

git config --global user.name “ім'я” – задає ім’я користувача;

git config --global user.email [ваша поштова скринька] – задає поштову скриньку користувача;

git config --list – показує список налаштувань;

git init – ініціалізація репозиторію у поточній директорії;

git clone [url] [directory name] – клонування репозиторію з серверу;

git status – виклик поточного статусу файлів;

git add – індексування змін;

git diff – конкретизований вивід змін;

git diff --cached/--staged – конкретизований вивід проіндексованих змін;

git commit – здійснення комміту;

git commit -m “коментар”– здійснення комміту з коротким коментарем;

git commit -a – здійснення комміту з автоматичною індексаціею;

git commit --amend – доповнення останнього зробленого комміту;

git rm [file] – видалення файлів;

git rm -f [file] – примусове видалення файлів;

git rm --cached [file] – видалення файлів тільки зі індексу;

git reset HEAD [file] – відміна індексування файлу;

git checkout -- [file] – відміна останніх змін у файлі;

git log – виклик історії коммітів;

git log –p – додає до виводу історії зміни кожного комміта;

git log --stat – виклик історії коммітів зі статистикою;

git log --shortstat – виклик історії коммітів зі скороченою статистикою;

git log --pretty=oneline – виводить історію у форматі де кожний комміт друкується на одній строці;

git log -<n> – показує останню <n> кількість коммітів;

git remote - викликає список налаштованих віддалених(remote) репозиторіїв;

git remote add [short name] [url] – додавання віддаленого репозиторію;

git fetch [remote name] – отримання даних з віддаленого репозиторію;

git pull [remote name] [branch name] – злиття даних з віддаленого репозиторію до вказаної гілки;

git push [remote name] [branch name] – відправлення даних до вказаної гілки віддаленого репозиторію;

git show [remote name] – виклик інформації про віддалений репозиторій;

git remote rename [old name] [new name] – зміна імені віддаленого репозиторія;

git remote rm [name] – видалення віддаленого репозиторія;

git tag – виклик списку міток;

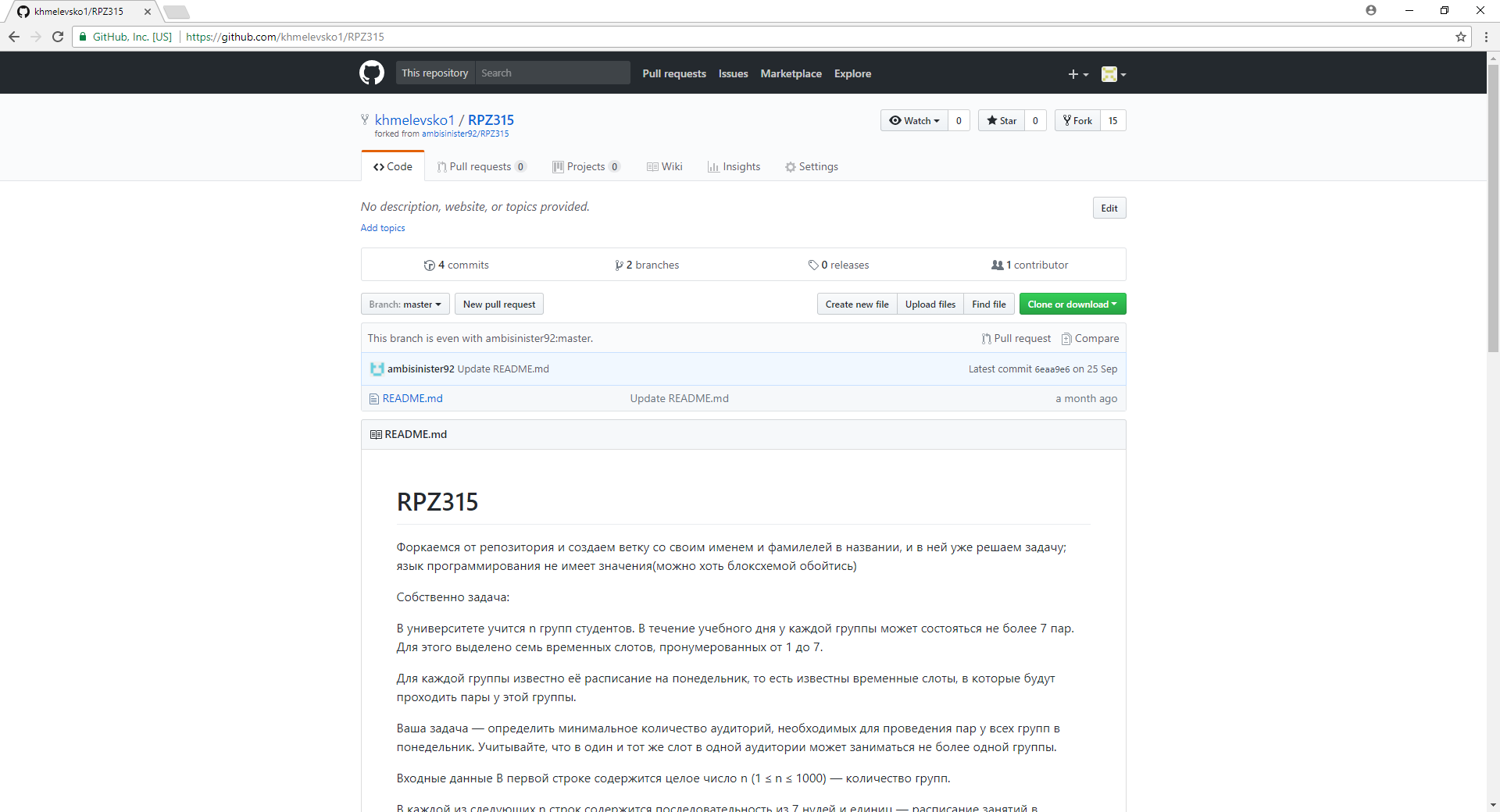
git tag [ім’я]– створення легковісної мітки;

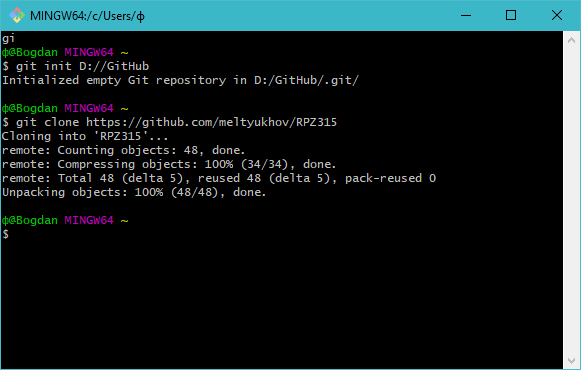
git tag –a [name] – створення аннотовної мітки;

git tag –a [name] –m ”comment” – створення аннотовної мітки з коротким коментарем;

git tag –a [name] [short hash] – помічення вже існуйочого комміту;

**4. Хід виконання практичної роботи**

Рисунок 1.3 – Відкриваю свій репозиторій

Рисунок 1.4 – Клоную свій репозиторій на комп'ютер

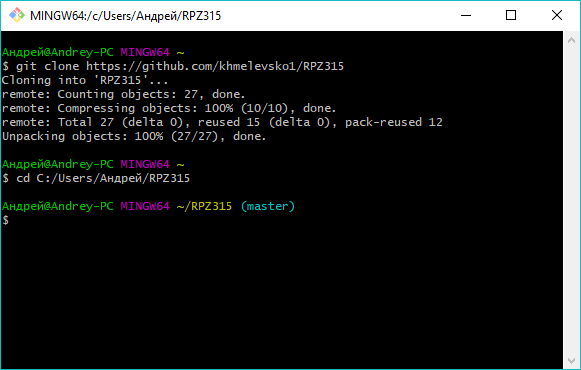


Рисунок 1.5 – Переходжу в папку з репозиторієм

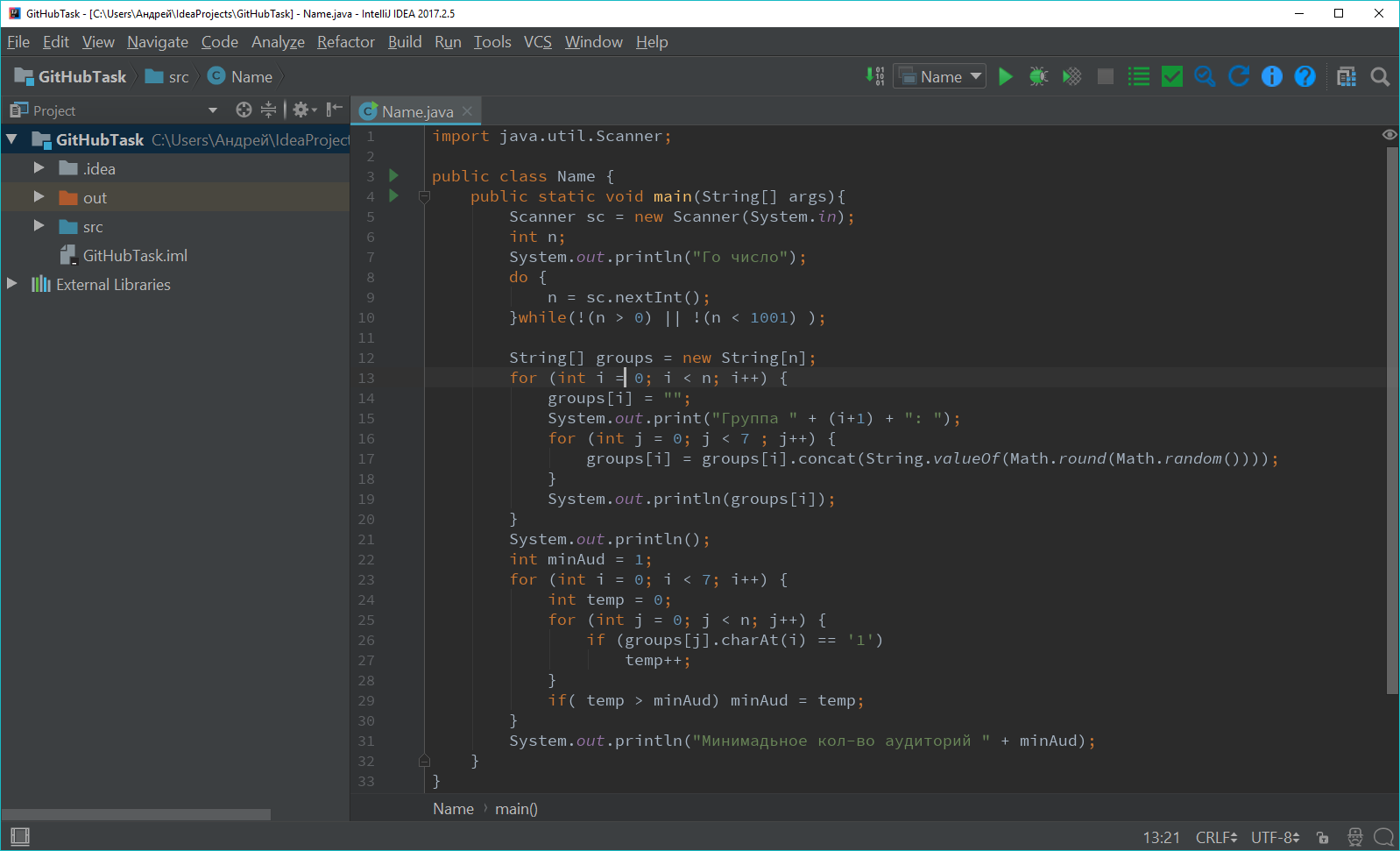


Рисунок 1.6 – Пишу програму

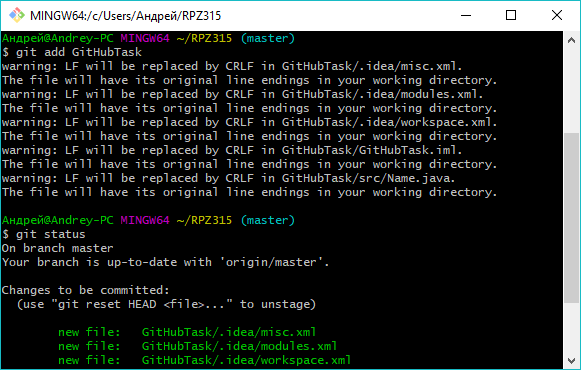


Рисунок 1.7 - Додаю папку з програмою в індекс

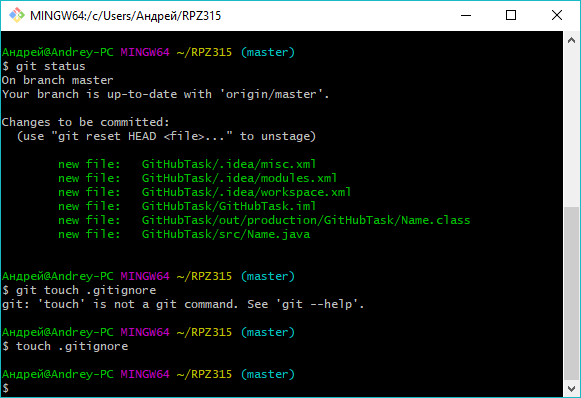


Рисунок 1.8 – Створюю файл .gitignore

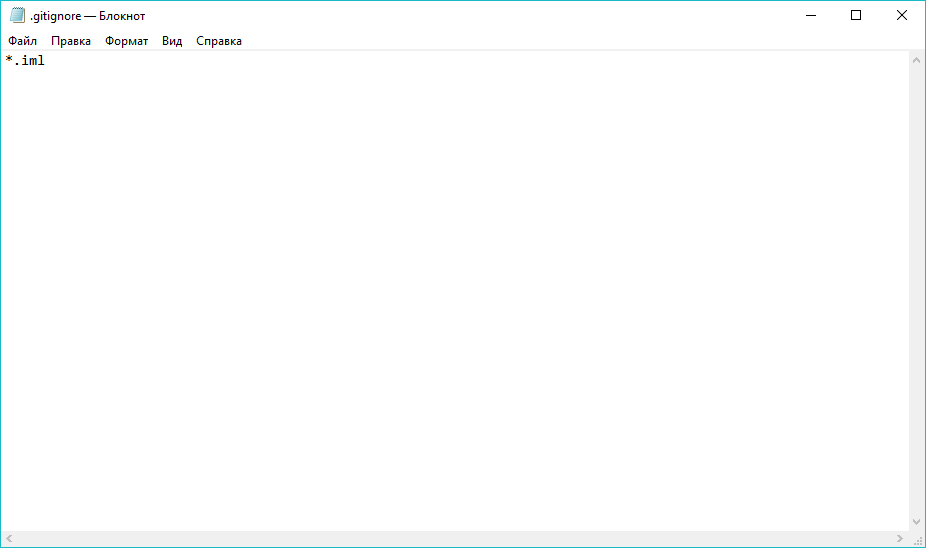


Рисунок 1.9 – Заповнюю його

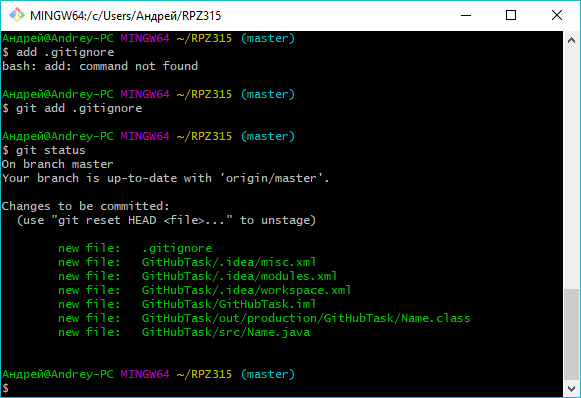


Рисунок 1.10 – Додаю в індекс .gitignore

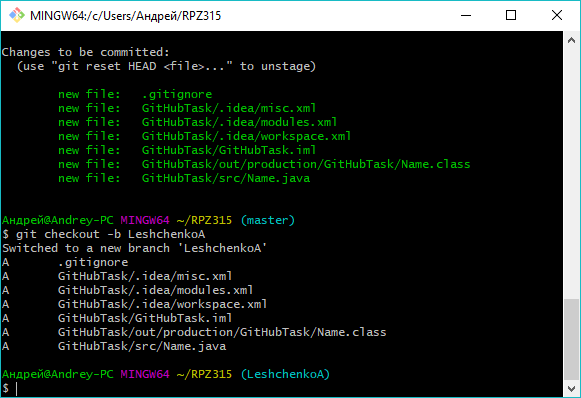


Рисунок 1.11 – Створюю гілку та переходжу на неї

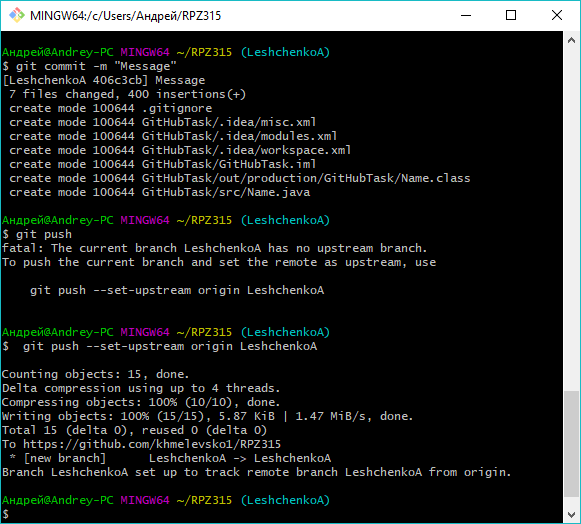


Рисунок 1.12 – Виконую комміт та відправляю на GitHub

**Питання до захисту :**

**Що таке СКВ, Види СКВ, до якої СКВ належить Гіт, відмінності від інших СКВ?**

Відповідь: Система контролю версій – це система, що реєструє зміни в одному або декількох файлах, щоб в подальшому мати можливість повернутися до старих версій цих файлів.

СКВ бувають:

1. Локальна
2. Централізована
3. Розподілена

Git належить до розподілених СКВ, це означає, що дані зберігаються не тільки на сервері, а на кожному з комп’ютерів.

**Що таке Репозиторій, які вони бувають, та з яких частин складаються?**

Відповідь: Репозиторій – це місце, в якому відбувається зберігання та підтримка будь-яких даних. Репозиторій git складається з робочого каталогу, індексу та .git.

**У яких станах можуть знаходитись файли у Git, життєвий цикл файлів?**

Відповідь: Кожен файл в робочому каталозі може знаходитися в одному з двох станів: що відстежуються або що не відстежується. Файли, що відстежуються, можуть бути зміненими, незміненими або підготовленими.

  
Рисунок 1.11 – Життєвий цикл станів файлів

**Чим є насправді гілки в Git?**

Відповідь: гілка насправді є легким вказівником на комміт, що може пересуватися.

**Розкажіть про способи злиття у Git?**

Відповідь: існує два способи злиття: перемотка(fast-forward) і трьохточкове (rebase).

**Що відбувається у робочій директорії при перемиканні гілок?**

Відповідь: переміщуються файли з останнього комміту в нову гілку, та змінюється HEAD.