Завдання № 1
КОМАНДНА РОБОТА.
ІНТЕГРАЦІЯ ПРОЄКТУ ДО СИСТЕМИ КОНТРОЛЮ ВЕРСІЙ

Мета роботи: сформувати проєктну команду; навчитися адмініструвати систему контролю версій, розподіляти і відстежувати завдання за проєктом, а також інтегрувати проєкт до системи контролю версій; набути практичних навичок використання клієнта СКВ та веб-хостингу (GitHub, GitLab, Assembla).

Порядок виконання роботи:

1. Прочитати теоретичний матеріал до практичної роботи.
2. Розбитися на команди по 3 людини.
3. Виділити в команді ролі (тімліда, програміста, тестувальника).
4. Розробити простий додаток, який містить кілька класів, при цьому кожен клас повинен бути реалізований окремим програмістом, а інтерфейси для кожного класу описуються тімлідом.
5. *Розробку необхідно вести з використанням різних гілок:* відгалуження додавати або для кожної ролі учасника проєкту, або для розроблення/доопрацювання робочого/неробочого функціоналу.
6. У процесі виконання лабораторного практикуму необхідно зафіксувати («заскринити») всі помилки і проблеми, що виникли при використанні обраної системи керування версіями. Якщо подібних ситуацій не виникло, то зімітувати помилкове виконання операцій (хоча б одне). Детальний опис додати до звіту.

Завдання за ролями у команді:

Тімліду:

а) зареєструватися на ресурсах: Bitbucket (https://bitbucket.org/), GitHub (https://github.com/), GitLab (https://gitlab.com/gitlab-org/gitlab) та Assembla (https://get.assembla.com/);

б) створити репозитарій (який буде використано для виконання першої практичної роботи);

в) створити Групу користувачів;

г) додати до Групи користувачів членів команди і призначити їм повні права доступу;

д) створити Solution з проєктом, який містить опис інтерфейсів класів.

Програмістам:

а) встановити TortoiseHG (http://tortoisesvn.net/) або будь-яку іншу версію СКВ з графічним інтерфесом;

б) отримати копію сховища на локальний комп'ютер;

в) додати в Solution проєкти, які реалізують інтерфейс, і вкомітити його в репозитарій;

г) описати класи, що реалізують описані інтерфейси;

д) закомітити і зафіксувати всі зміни на сервері.

Тестувальникам:

а) підключитися до сховища проєкту;

б) «витягти» з репозитарію на комп'ютер робочу копію проєкту;

в) перевірити функціональність і взаємодію інтерфейсів. Усунути баги;

г) залити проєкт назад у SVN або будь-якої іншої СКВ на Ваш вибір.

7. Скласти матрицю відповідальних за виконання проєкту.

8. Оформити звіт, перевірити його у викладача.

9. Зберегти файл, який містить звіт з практичної роботи, та файл-архів проєкту.

Зміст звіту:

1. Постановка завдання.

2. Опис проєкту.

3. Обґрунтування вибору проєкту: в чому його актуальність, на яку аудиторію користувачів орієнтований.

4. Обґрунтування вибору системи контролю версій (СКВ). Короткі теоретичні відомості про те, для яких проєктів більше підходить та чи інша СКВ. Чому? (Дати коротке пояснення). Критерії вибору (перерахувати).

5. Аналіз коментарів експертів щодо використовуваних систем контролю версій. У ролі експертів, як правило, виступають розробники програмного забезпечення великих IT-підприємств, які мають великий досвід у сфері створення програмного продукту.

6. Обґрунтування вибору типу сховища (public, private).

7. Скриншоти створеного сховища та груп користувачів.

9. Текст програми (за необхідності – пояснення до реалізації).

10. Результати тестів (скриншоти).

11. Матриця відповідальних.

12. Історія комітів.

13. Висновки.

Теоретичні відомості

Робота через термінал.

Почати роботу з Git можна з кількох основних кроків.

1. Інсталяція Git.

Першим кроком є інсталяція Git на ваш комп'ютер. Git є вільною та відкритою системою керування версіями, і ви можете завантажити його з офіційного сайту Git: https://git-scm.com/

2. Конфігурація Git.

Після інсталяції Git Вам потрібно налаштувати основні налаштування, такі як ім'я та електронна адреса. Це дозволить Git ідентифікувати Вас при внесенні змін. Ви можете це зробити за допомогою команд:

git config --global user.name "Your Name"

git config --global user.email "youremail@example.com"

3. Створення нового репозиторію.

Після налаштування Git Ви можете створити новий репозиторій для Вашого проекту. Це можна зробити як локально, так і на віддаленому сервері, такому як GitHub або GitLab. Для створення нового локального репозиторію Вам потрібно перейти в каталог Вашого проекту та виконати команду:

git init

4. Додавання файлів.

Після створення репозиторію Ви можете додати файли до нього, які будуть відстежуватися Git. Для цього Ви можете використовувати команду:

git add <file>

5. Створення коміту:

Після додавання файлів Ви можете створити коміт, що представляє собою фіксацію змін. Для цього Ви можете використовувати команду:

git commit -m "Your commit message"

6. Завантаження на віддалений сервер.

Якщо Ви хочете зберегти свій код на віддаленому сервері, наприклад, на GitHub, Ви можете додати віддалений репозиторій та відправити свої коміти. Для цього використовуються команди:

git remote add origin <remote_repository_URL>

git push -u origin master

Щоб отримати доступ до довідкової інформації за допомогою команди Git, Ви можете використати команду `git help` з потрібним ключем. Наприклад:

git help <команда>

Замість `<команда>` ви повинні вказати саму команду Git, про яку ви хочете отримати довідку. Наприклад:

git help commit

Це відкриє довідкову сторінку для команди `commit`, де буде надано докладну інформацію про синтаксис, опції та приклади використання цієї команди.

Також, якщо Ви не впевнені, як саме називається потрібна вам команда, ви можете використати команду `*git help -g*`, яка відкриє глобальну довідку, в якій ви зможете переглянути всі доступні команди Git, а також докладну інформацію про них.

Робота з графічним клієнтом.

Ось кроки, які можна виконати, використовуючи графічний клієнт для роботи з Git:

1. Встановлення графічного клієнта.

Спочатку потрібно встановити графічний клієнт Git на ваш комп'ютер. Популярними варіантами є GitHub Desktop, GitKraken, SourceTree тощо.

2. Авторизація в клієнті.

Відкрийте графічний клієнт та авторизуйтеся в ньому за допомогою вашого облікового запису GitHub або іншого сервісу хостингу репозиторіїв.

3. Клонування репозиторію.

За допомогою графічного клієнта склонуйте репозиторій на свій комп'ютер. Для цього виберіть опцію "Clone" та вкажіть URL репозиторія.

4. Внесення змін.

Зробіть необхідні зміни у файлі або створіть нові файли у Вашому репозиторії, використовуючи Ваш редактор коду.

5. Збереження змін.

Після внесення змін у Вашому редакторі коду, поверніться до графічного клієнта. Ви побачите зміни, які були внесені вами. Виберіть файли, які потрібно додати до збереження (commit).

6. Збереження змін (commit):

Вкажіть коментар, який пояснює, що змінилось. Після цього збережіть зміни, вибравши опцію "Commit" у графічному клієнті.

7. Відправлення змін (push).

Після збереження змін локально, натисніть кнопку "Push", щоб відправити ваші зміни на віддалений сервер, наприклад, GitHub.

8. Оновлення локальної копії (pull).

Якщо інші користувачі зробили зміни у віддаленому репозиторії, Ви можете оновити свою локальну копію за допомогою кнопки "Pull" у графічному клієнті.

9. Вирішення конфліктів (optional):

Якщо під час злиття (merge) виникли конфлікти, графічний клієнт надасть інтерфейс для їх вирішення.

Порада. Якщо починаєте користуватися Git-ом, почніть зі звичайних текстових файлів, тобто відпрацюйте основні команди на них і лише потім переходьте до створення проєкту. Також краще починати взаємодію з СКВ через графічний клієнт.