МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Кафедра «Програмна інженерія та

інформаційні технології управління»

мЕТОДИЧНІ ВКАЗІВКИ

до виконання лабораторної роботи № 3

з дисципліни

«Архітектура та проектування програмного забезпечення (частина 2)»

Розробив

Доцент кафедри ПІІТУ

к.т.н. доцент

Дмитро ДВУХГЛАВОВ

Харків 2022

ВИКОРИСТАННЯ РЕПОЗИТОРІЮ GITHUB

Мета

1. Закріпити знання з організації зберігання командного коду проекту та організації контролю версій під час розробки проекту.
2. Набути навичок створення сховища на GitHub та його підключення до локального сховища на локальному комп'ютері.
3. Набути навичок отримання коду із зовнішнього сховища.

Завдання

1. Створення облікового запису в GitHub.
2. Створення зовнішнього сховища в GitHub.
3. Пов’язання зовнішнього та локального сховища.
4. Збереження змін у проекті у віддаленому репозиторії з згодою інших розробників та без неї.

Методичні вказівки

**Завдання Git-5. Створити акаунт GitHub та завантажити проект до віддаленого репозиторію**

1. Створіть обліковий запис на GitHub

<https://git-scm.com/book/en/v2/GitHub-Account-Setup-and-Configuration>

1. Створіть віддалене сховище у своєму обліковому записі з назвою “MyRR\_FIO”. Приклад – MyRR\_DDE.
2. Створіть ключ ssh та інтегруйте його з GitHub. До початку виконання вивчіть всі частини матеріалу за посиланням - <https://docs.github.com/en/authentication/connecting-to-github-with-ssh/about-ssh>.
3. Пов’яжіть локальне та віддалене сховища.

<https://git-scm.com/book/en/v2/Git-Basics-Working-with-Remotes>

1. Перенесіть усі зміни з усіма вашими гілками до віддаленого сховища (*$ git push <remote repository name> --all*).
2. Перевірте завантаження робочого сховища у віддалене сховище за допомогою браузера.

**Завдання Git-6. Модифікація проекту та збереження змін у віддаленому репозиторії**

Це завдання дозволяє навчити навичкам збереження змін у віддаленому репозиторії без згоди інших розробників.

1. Створіть папку з іменем “LR\_FIO\_GIT2” (приклад - LR\_DDE\_GIT2).
2. Відкрийте Git Bash. Клонуйте проект у цю папку. Після клонування у вашій папці буде створена папка з назвою віддаленого репозиторію. Для початку роботи з поточною версією вашого проекту необхідно переміститися в цю папку.
3. Виконайте оновлення свого проекту – впровадьте нову функцію. Перед цим створити гілку *develop2* та перейти на неї.

Імплементація нової функції може полягати, наприклад, у створенні нової веб-сторінки, чи у створенні нового стилю, чи у оновленні існуючих веб-сторінок у проекті. Якщо реалізується проект у певній IDE, то під розробкою нової функції можна розуміти додавання нових класів або розробку нових методів.

1. Зробіть 2-3 коміти під час розробки нової функції.
2. Після повного оновлення виконайте merge *develop2* до *master*.
3. Внесіть зміни з вашого локального репозиторію у віддалене сховище на GitHub. Для цього необхідно в команді *$ git push* указати в якості джерела гілку *master* або використовувати параметри *- - all*.
4. Перевірте наявність нової версії проекту у віддаленому репозиторії.

**Завдання Git-7. Модифікація проекту та збереження змін, використовуючи pull request**

Це завдання дозволяє навчати навичкам збереження змін у віддаленому репозиторії за згодою інших розробників.

При виконанні цього завдання слід продовжите працювати з клоном проекту з віддаленого репозиторію (він розміщений в папці LR\_<FIO>\_GIT2/MyRR\_<FIO>). Якщо є бажанно, то можна клонувати проект в іншу папку.

1. Використовуючи **Git Bash** створити гілку *develop3* для реалізації нової функції та перейти на неї.
2. Зробити 2-3 комміти під час розробки нової функції. Після завершення розробки цієї функції проект не виконується ні MERGE, ні REBASE до *master* та виконується перехід до наступного кроку.
3. Завантажте зміни, отримані на гілці *develop3* з вашого локального репозиторію до віддаленого репозиторію на GitHub. У порівнянні з тим же кроком у завданні Git-6, необхідно встановити гілку *develop3* як гілку для завантаження.
4. Відкрийте віддалений репозиторій у своєму обліковому записі **GitHub**. На закладці «Pull request» створіть запит на пул для злиття *develop3* до *master*. Переконайтеся, що ви намагаєтеся завантажити правильну версію нової реалізації функції. Переконайтеся, що нове програмне забезпечення не конфліктує з поточною версією програмного забезпечення. Додайте повідомлення про це як коментар до запиту на пул.

Про pull requests - <https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/proposing-changes-to-your-work-with-pull-requests/about-pull-requests>

Створення pull request - <https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/proposing-changes-to-your-work-with-pull-requests/creating-a-pull-request>

Злиття pull request - <https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/incorporating-changes-from-a-pull-request/merging-a-pull-request>

1. Завантажте нові гілки у репозиторій, використовуючи варіант "Create merge commit", який передбачає, що весь коміт у гілці develop3 буде додано до master шляхом виконання команди merge.
2. Перевірте результат об'єднання в списку гілок.
3. Поверніться до **Git Bash**. Перейдіть на гілку *master* (!!!). Підготуйтеся до наступного оновлення вашого проекту. Рекомендується використовувати команду **$ git pull** з необхідними для цього опціями. Також можна використовувати команду **$ git fetch** і **merge** в *master* на локальному репозиторії.
4. Створіть нову гілку, щоб розробити ще одну функцію або виправити баг у коді. Назва для гілки – вибір студента. Зробіть на цій гілці 2-3 комміти. Завантажте гілку зі змінами у віддалений репозиторій також без об'єднання з *master*.
5. Відкрийте віддалений репозиторій у своєму обліковому записі **GitHub**. Створіть pull request для інтеграції нового коду. Натисніть нові гілки в репозиторій, який використовується опція “*Rebase and merge*”, яка включає в себе те, що всі коміти у гілці будуть перебазовані і додані до *master*.
6. Перевірте результат об'єднання в списку гілок. Представте його у звіті.
7. Поверніться до **Git Bash**. Перейдіть до *master*, щоб отримати зміни з віддаленого сховища.

12 Показати історію коммітів у вигляді дерева. Наведіть знімок екрана дерева історії коммітів. Розглянте, як **GitHub** зробив злиття ваших гілок. Результати цього аналізу внести у звіт.

Вимоги до звіту

У звіті про лабораторні роботи слід представити:

* тему, мету, завдання на лабораторну роботу;
* скріншоти з результатами виконання, відповідно до вказівок;
* висновки.