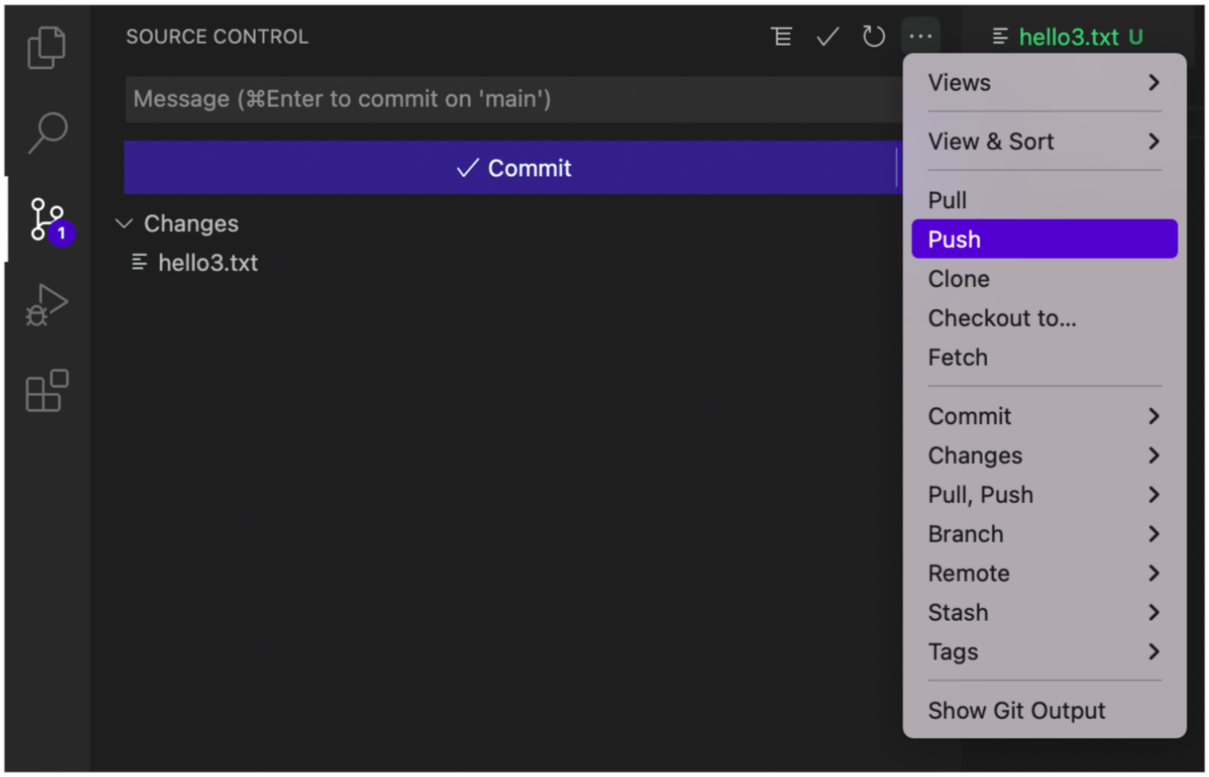
Week 1: Project Management and Git

Exercise Instructions:

1. Read the exercise to the end.
2. Identify the tasks that need to be performed.
3. Create a dedicated Jira project and create the aforementioned tasks, organizing them into relevant Epics.
4. Create a sprint and add all the tasks to it.
5. Start the sprint and start working on the tasks, maintaining their status. Namely, when you start a task, move it to "in progress" and when you finish a task, move it to "done".
6. Continue until the sprint is completed.
7. During this exercise, document your progress using screenshots (of the **entire** screen) to demonstrate that you have indeed completed all the required tasks (both the Jira part and Git part).
8. Save all the screenshots in the GitHub project, in a dedicated folder called "proof," and present them chronologically in the README file.
9. **There's no need to make hundreds of screenshots, but please take enough to convince the person checking your submission that you completed the exercise as instructed.**
10. Your README file must also include a link to the repository.
11. When you are ready to submit, download the repository from GitHub (the entire repository) and submit it. Your submission **MUST NOT** be compressed in any way, nor contain links to other locations where the files are (like Google Drive or a link to the GitHub repository instead of the actual files).
12. The Moodle has a hard limit on the upload size, and will not allow you to upload files beyond 5MB. **There is nothing I can do about it.** Therefore, remove from the submission screenshots until your submission is in the right size. **The screenshots still need to exist in Github.**

Steps:

1. Create a new **private** GitHub repository (**not** a public repository), with a README file and .gitignore file.
2. Open the “Show Git Output”: (bottom-right corner of the next picture)



This will allow you to see that whenever you do a Git-related operation in visual studio code, behind the scenes, visual studio code actually perform git commands.
You will see many commands per each operation. That’s ok, you do not need to understand all of them. If you use an IDE which is not visual studio code, look for the equivalent feature in that IDE.

3. **Clone** the repository locally (to your computer). Can you see the clone operation in the “Git Output” window?

4. Add, locally, a file with some text. **Commit** the changes. Can you see the commit operation in the “Git Output” window?
5. Use the .gitignore file to exclude some (other) file and some folder with files in it.
6. Commit the changes in **two** separate commits. Namely, do “git add” to some of the files and commit only them, then, in a separate commit, commit the rest of files (except the ignored ones of course).
7. Go to Github and see that the commits are still not there (they currently only exist locally on your computer).
8. **Push** the commits to Github. Can you see the push operation in the “Git Output” window? Can you now see the commits in Github?

End of week 1 mini-exercise