

**КПІ ім. Ігоря Сікорського
Інститут прикладного системного аналізу
Кафедра Системного проектування**

Лабораторна робота №1

«Системи контролю версій SVN, GIT»

з дисципліни «Проектування інформаційних систем»

Виконав:

Студент групи ДА-72 ННК «ПСА»

Мазуренко В.О.

Тема: “Калькулятор проміле”

Київ-2020

Мета роботи: за допомогою системи контролю версій завантажити коди

програми у репозіторій. Відтворити типовий цикл розробки програмного

забезпечення з використанням системи контролю версій.

Задача:

1. Вивчити основні команди роботи з репозиторіями.
2. Завантажити код програми у репозиторій.
3. Показати основний цикл роботи з програмним кодом за допомогою системи контролю версій.

Завдання:

1. Обрати безкоштовну систему репозиторія для системи контролю версіями, наприклад projectlocker, або інш.
2. Встановити клієнтське безкоштовне програмне забезпечення для роботи з системою контролю версій (GIT, SVN clients).
3. Протягом роботи над лабораторними роботами 2-6 використовувати систему контролю версіями.
4. Описати цикл розробки програмного забезпечення з використанням системи контролю версій.

Хід роботи:

1. Створимо репозиторій на хостингу проектів Bitbucket, що працює з системою контролю версій git.

Create a new repository [Import repository](#)

Workspace  Vlad Mazurenko

Project name *****

Repository name *****

Access level **Private repository**
Uncheck to make this repository public. Public repositories typically contain open-source code and can be viewed by anyone.

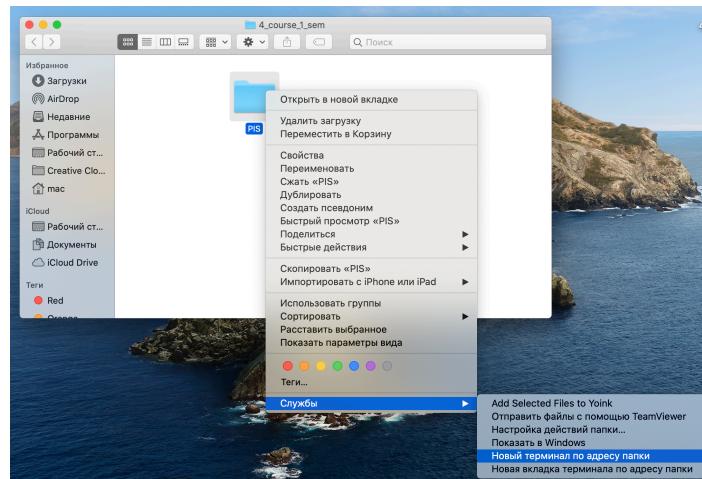
Include a README?

Include .gitignore?

[Advanced settings](#)

Репозиторій створено:

2. Склонуємо пустий репозиторій на робочий комп'ютер використовуючи термінал macOS.

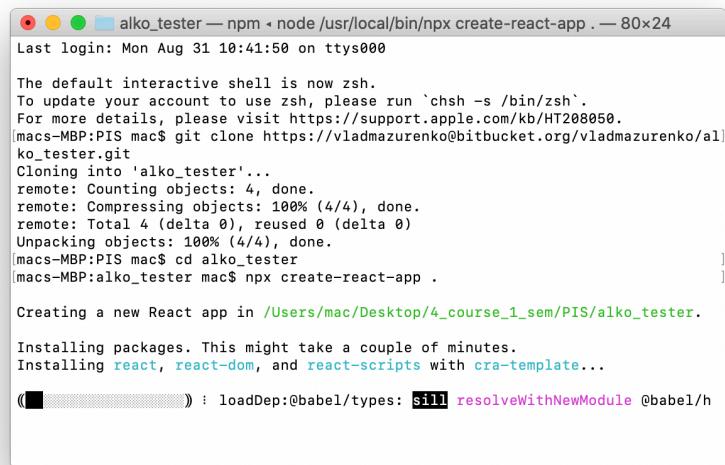


```

PIS — bash — 80x24
Last login: Mon Aug 31 10:41:50 on ttys000
The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT200050.
macs-MBP:PIS mac$ git clone https://vladmazurenko@bitbucket.org/vladmazurenko/alko_tester.git

```

3. Додамо React шаблон для майбутньої роботи за допогою команди:
 npx create-react-app



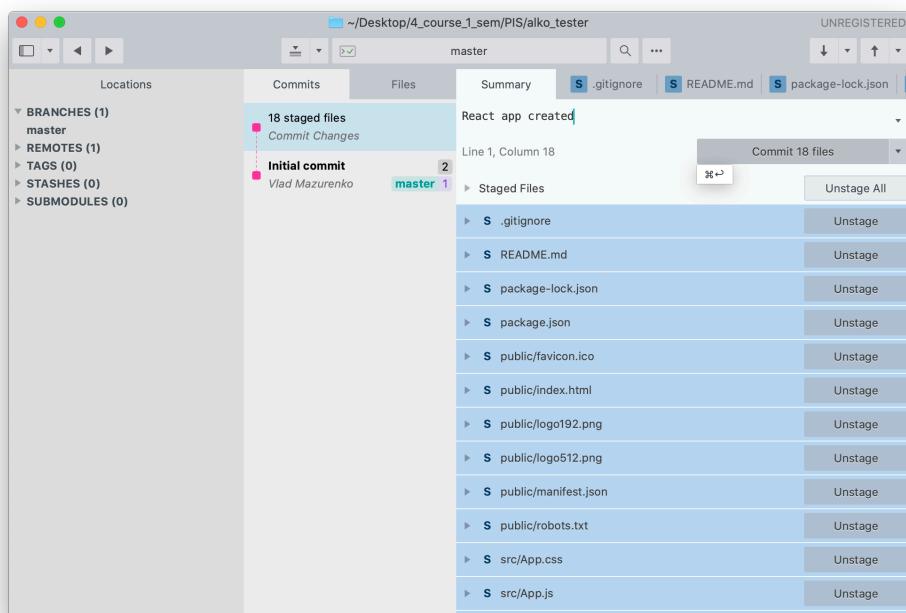
```

Terminal window showing the creation of a React app:
Last login: Mon Aug 31 10:41:50 on ttys000
The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
[macs-MBP:PIIS mac$ git clone https://vladmazurenko@bitbucket.org/vladmazurenko/alko_tester.git
Cloning into 'alko_tester'...
remote: Counting objects: 4, done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 4 (delta 0), reused 0 (delta 0)
Unpacking objects: 100% (4/4), done.
[macs-MBP:PIIS mac$ cd alko_tester
[macs-MBP:alko_tester mac$ npx create-react-app .
Creating a new React app in /Users/mac/Desktop/4_course_1_sem/PIIS/alko_tester.

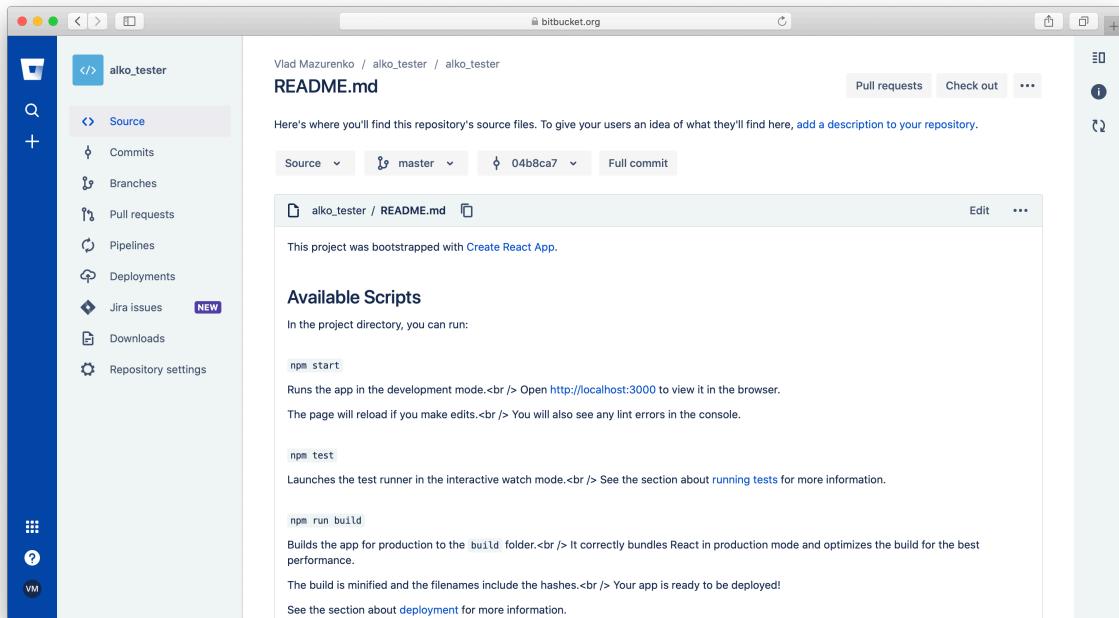
Installing packages. This might take a couple of minutes.
Installing react, react-dom, and react-scripts with cra-template...
( [ ] : loadDep:@babel/types: sill resolveWithNewModule @babel/h

```

4. Зробимо перший комміт за допомогою програми Sublime Merge



README.md



bitbucket.org

Vlad Mazurenko / alko_tester / alko_tester

README.md

Here's where you'll find this repository's source files. To give your users an idea of what they'll find here, add a description to your repository.

Source master 04b8ca7 Full commit

Edit ...

This project was bootstrapped with [Create React App](#).

Available Scripts

In the project directory, you can run:

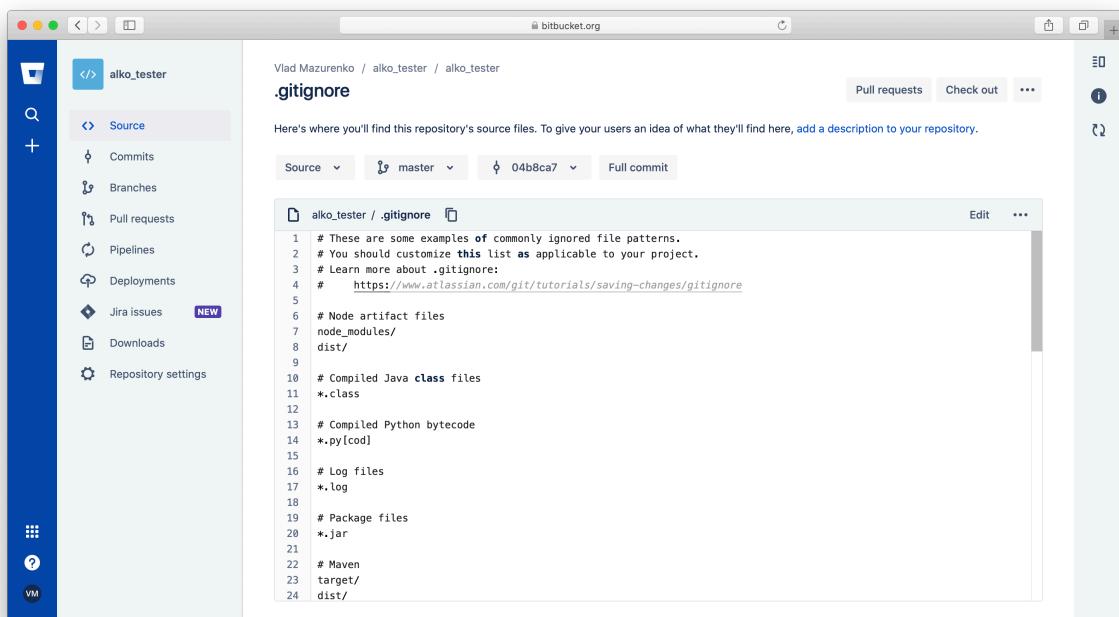
```
npm start
Runs the app in the development mode. Open http://localhost:3000 to view it in the browser.
The page will reload if you make edits. You will also see any lint errors in the console.

npm test
Launches the test runner in the interactive watch mode. See the section about running tests for more information.

npm run build
Builds the app for production to the build folder. It correctly bundles React in production mode and optimizes the build for the best performance.

The build is minified and the filenames include the hashes. Your app is ready to be deployed!
See the section about deployment for more information.
```

.gitignore



bitbucket.org

Vlad Mazurenko / alko_tester / alko_tester

.gitignore

Here's where you'll find this repository's source files. To give your users an idea of what they'll find here, add a description to your repository.

Source master 04b8ca7 Full commit

Edit ...

```
1 # These are some examples of commonly ignored file patterns.
2 # You should customize this list as applicable to your project.
3 # Learn more about .gitignore:
4 # https://www.atlassian.com/git/tutorials/saving-changes/gitignore
5
6 # Node artifact files
7 node_modules/
8 dist/
9
10 # Compiled Java class files
11 *.class
12
13 # Compiled Python bytecode
14 *.py[cod]
15
16 # Log files
17 *.log
18
19 # Package files
20 *.jar
21
22 # Maven
23 target/
24 dist/
```

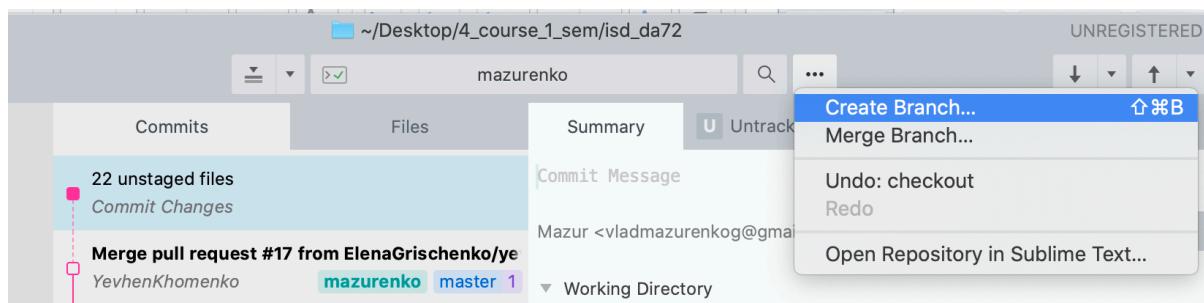
5. Додавання файлів проекту до учебового репозиторію на GitHub

1) Клонування репозиторію isd_da72

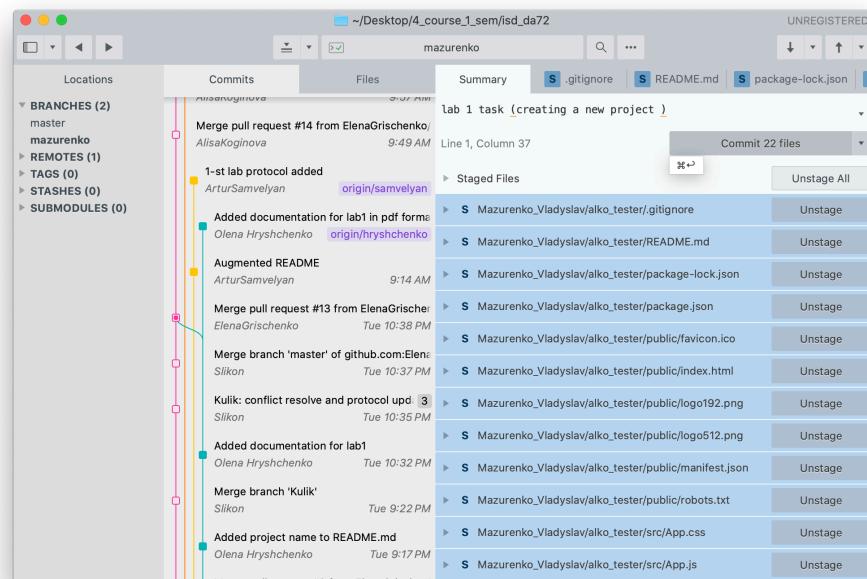
```
4_course_1_sem — bash — 80x24
Last login: Tue Sep 29 10:55:37 on ttys002

The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
[macs-MBP:~ mac$ cd desktop/4_course_1_sem
[macs-MBP:4_course_1_sem mac$ git clone https://github.com/ElenaGrischenko/isd_da72.git
Cloning into 'isd_da72'...
remote: Enumerating objects: 251, done.
remote: Counting objects: 100% (251/251), done.
remote: Compressing objects: 100% (198/198), done.
remote: Total 251 (delta 48), reused 212 (delta 28), pack-reused 0
Receiving objects: 100% (251/251), 15.32 MiB | 9.20 MiB/s, done.
Resolving deltas: 100% (48/48), done.
macs-MBP:4_course_1_sem mac$
```

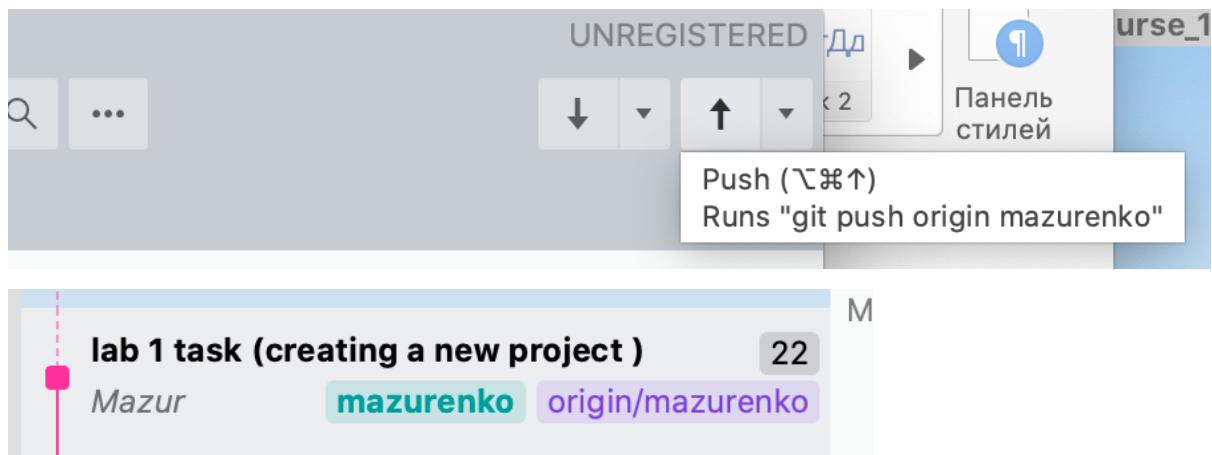
2) Створення нової гілки



3) Commit



4) Push



Висновок: Під час даної лабораторної роботи було створено репозиторій на платформі Bitbucket та виконано перший комміт за допомогою Sublime Merge.

При подальшій розробці план використання системи контролю версій git полягає у створенні нових коммітів при кожному закінченні робочої сесії або ж при виконанні окремо взятих задач.