

Build a Strong MERN Foundation



Online Banking Web Application





Planning Project with Agile and Git

You Already Know

Before we begin, let's recall what we have covered till now:



git



Recap

Agile

It is an iterative approach used to manage the development of a software project.

Git

It is a distributed version control system used to handle software projects.

GitHub

It is a web-based platform for version control and collaboration, allowing users to store, manage, and track changes to their code projects.



A Day in the Life of a MERN Stack Developer

As a MERN stack web developers, our key role is to develop full-stack web applications using MongoDB, Express.js, React.js, and Node.js.

As developers, our given task is to develop an online banking application using React.js and Node.js with Express.js as the web framework.

In order to persist the data, the NoSQL database, MongoDB will be used.



A Day in the Life of a MERN Stack Developer

To begin with our project, we need to plan it properly. We will employ the Agile methodology.

First, we will create user stories for the epics, a web admin dashboard for bank authorities, and a web app for the end user.

We will thereafter see how to use Trello and plan the project with our stories. Finally, we will create the project structure and sync the code to GitHub.



Learning Objectives

By the end of this lesson, you will be able to:

- 🕒 Implement the Agile methodologies, such as Scrum or Kanban, in the planning and execution of web application projects
- 🕒 Design a comprehensive Agile strategy for a web application project
- 🕒 Develop a Git workflow strategy for a team-based web application project
- 🕒 Assess the use of Git in managing branches and merging code





Task 1: Plan the Project Using Agile Methodologies

Create a Kanban Board

Create admin user stories as the requirements of the project:

- Admin can do the signup for the dashboard with his/her credential.
- Admin can view all customer details.
- Admin can approve new customers to create the account.
- Admin can lock the account.
- Admin can search customer details based on email ID, account number, or phone number.
- Admin can provide special interest for special customer.

Create a Kanban Board

Create customer user stories as the requirements of the project:

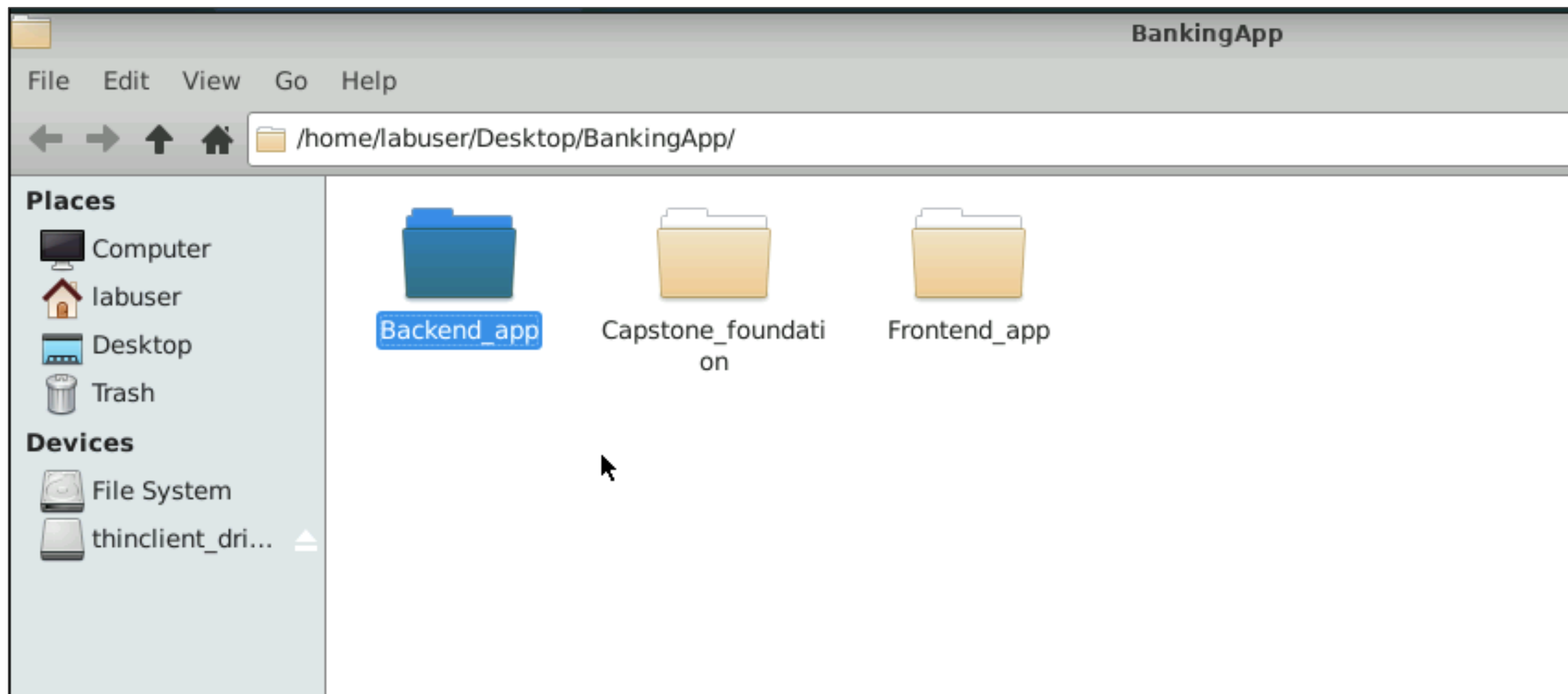
- Customer can sign up for the account
- Customer can sign into the account.
- Customer can request admin to activate or deactivate the account.
- Customer can do the signup for the net banking.
- Customer can view their account details.
- Customer can transfer the amount
- Customer can withdraw the amount
- Customer can deposit the amount



Task 2: Create Project and Sync Code to GitHub

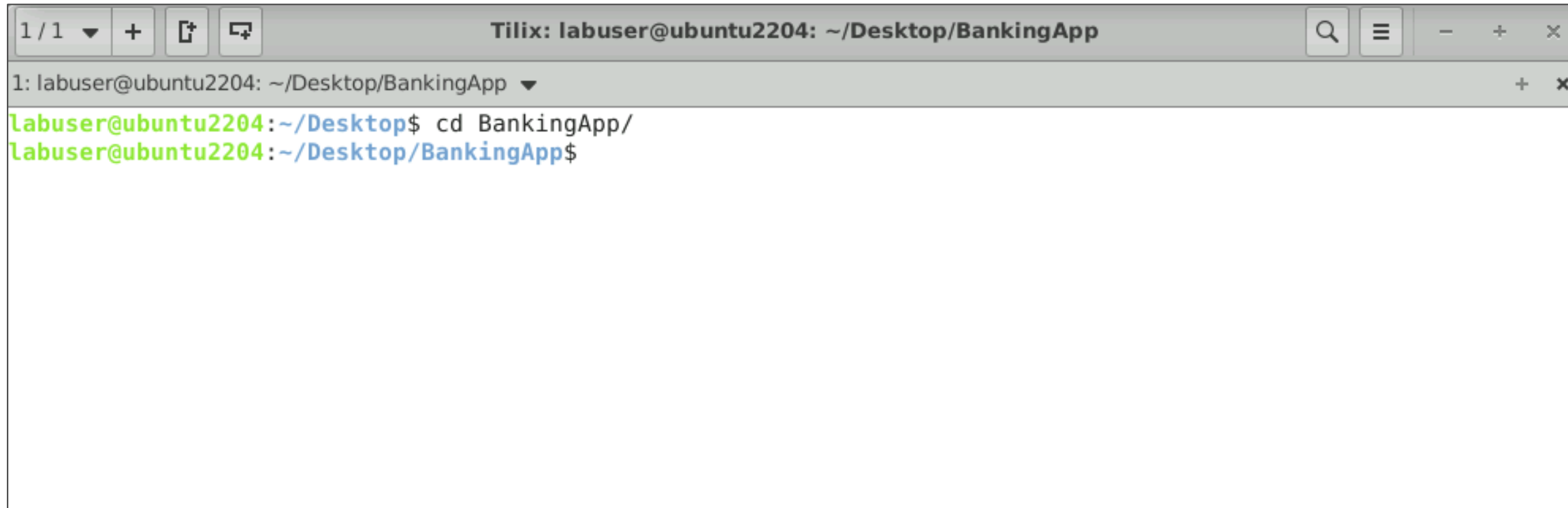
Create the Project Structure

Create a folder named **BankingApp** and three sub-folders named **Capstone_foundation**, **Frontend_app**, and **Backend_app**



Git Steps to Create Local Repository

Open the terminal and navigate inside the project directory

A screenshot of a terminal window titled "Tilix: labuser@ubuntu2204: ~/Desktop/BankingApp". The terminal shows the command "cd BankingApp/" being executed. The prompt changes from "labuser@ubuntu2204:~/Desktop\$" to "labuser@ubuntu2204:~/Desktop/BankingApp\$".

```
1 / 1 ▼ + [ ] [ ]  
Tilix: labuser@ubuntu2204: ~/Desktop/BankingApp  
1: labuser@ubuntu2204: ~/Desktop/BankingApp ▼ + x  
labuser@ubuntu2204:~/Desktop$ cd BankingApp/  
labuser@ubuntu2204:~/Desktop/BankingApp$
```

Git Steps to Create Local Repository

Execute the following command to initialize the Git repository in the local repository: **git init**, **git add .** and **git commit -m "Project structure ready"**

```
1 / 1 + [ ] [ ]
Tilix: labuser@ubuntu2204: ~/Desktop/BankingApp
1: labuser@ubuntu2204: ~/Desktop/BankingApp
labuser@ubuntu2204:~/Desktop/BankingApp$ git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:
hint:   git branch -m <name>
Initialized empty Git repository in /home/labuser/Desktop/BankingApp/.git/
labuser@ubuntu2204:~/Desktop/BankingApp$ git commit -m "Project structure ready"
[master (root-commit) 465bc81] Project structure ready
Committer: Ubuntu <labuser@ubuntu2204.2poyypvsuijhb51bmx4t2lvqb.rx.internal.cloudapp.net>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

3 files changed, 5 insertions(+)
create mode 100644 Backend_app/Readme.txt
create mode 100644 Capstone_foundation/Readme.txt
create mode 100644 Frontend_app/Readme.txt
labuser@ubuntu2204:~/Desktop/BankingApp$
```

Create a Remote Repository

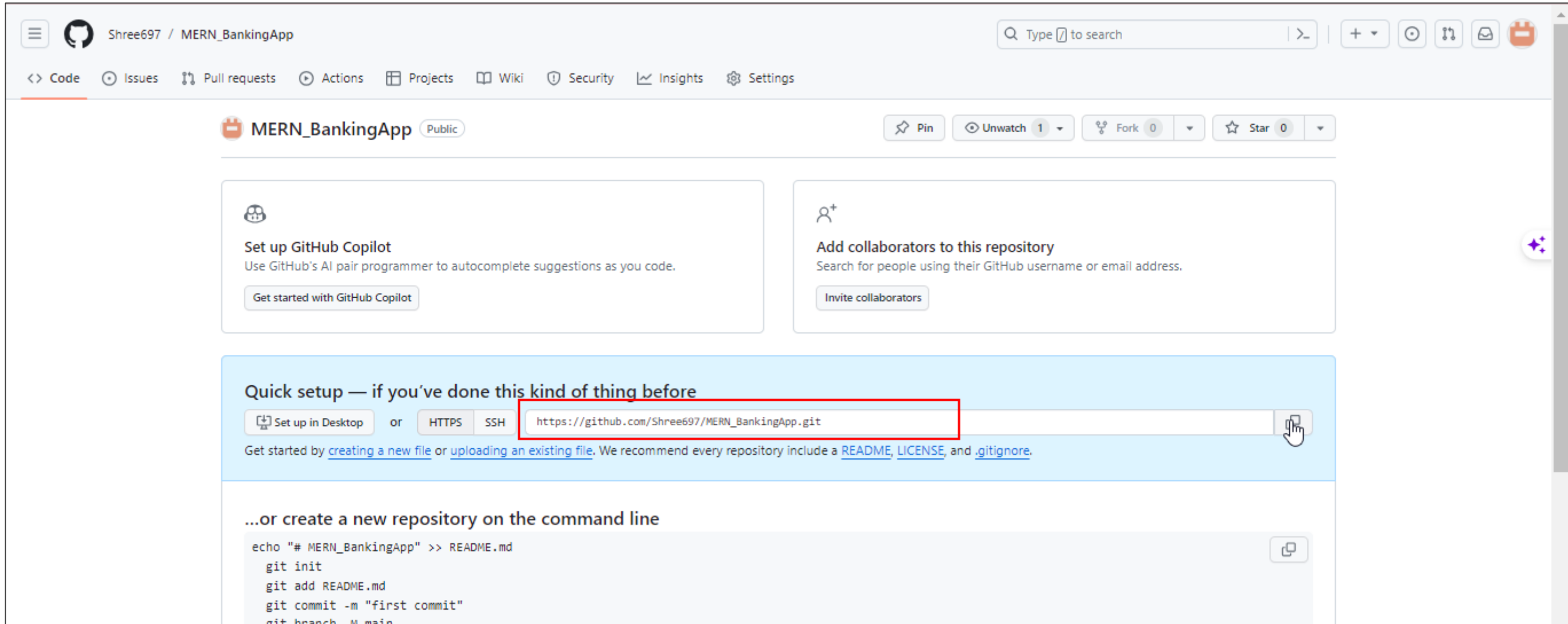
Create a remote GitHub repository named **MERN_BankingApp**

The screenshot shows the GitHub interface for a newly created repository named "MERN_BankingApp". The repository is public and has 0 stars, 0 forks, and 1 watcher. The page includes navigation tabs for Code, Issues, Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings. Below the repository name, there are two main sections: "Set up GitHub Copilot" and "Add collaborators to this repository". A "Quick setup" section provides instructions for cloning the repository using HTTPS or SSH, with a pre-filled URL: `https://github.com/Shree697/MERN_BankingApp.git`. Below this, there is a section for creating a new repository on the command line, showing the following commands:

```
echo "# MERN_BankingApp" >> README.md
git init
git add README.md
git commit -m "first commit"
```


Create a Remote Repository

Copy the HTTPS remote URL and save it



The screenshot shows the GitHub interface for a repository named 'MERN_BankingApp' by user 'Shree697'. The repository is public. The 'Quick setup' section is highlighted in light blue and contains the following information:

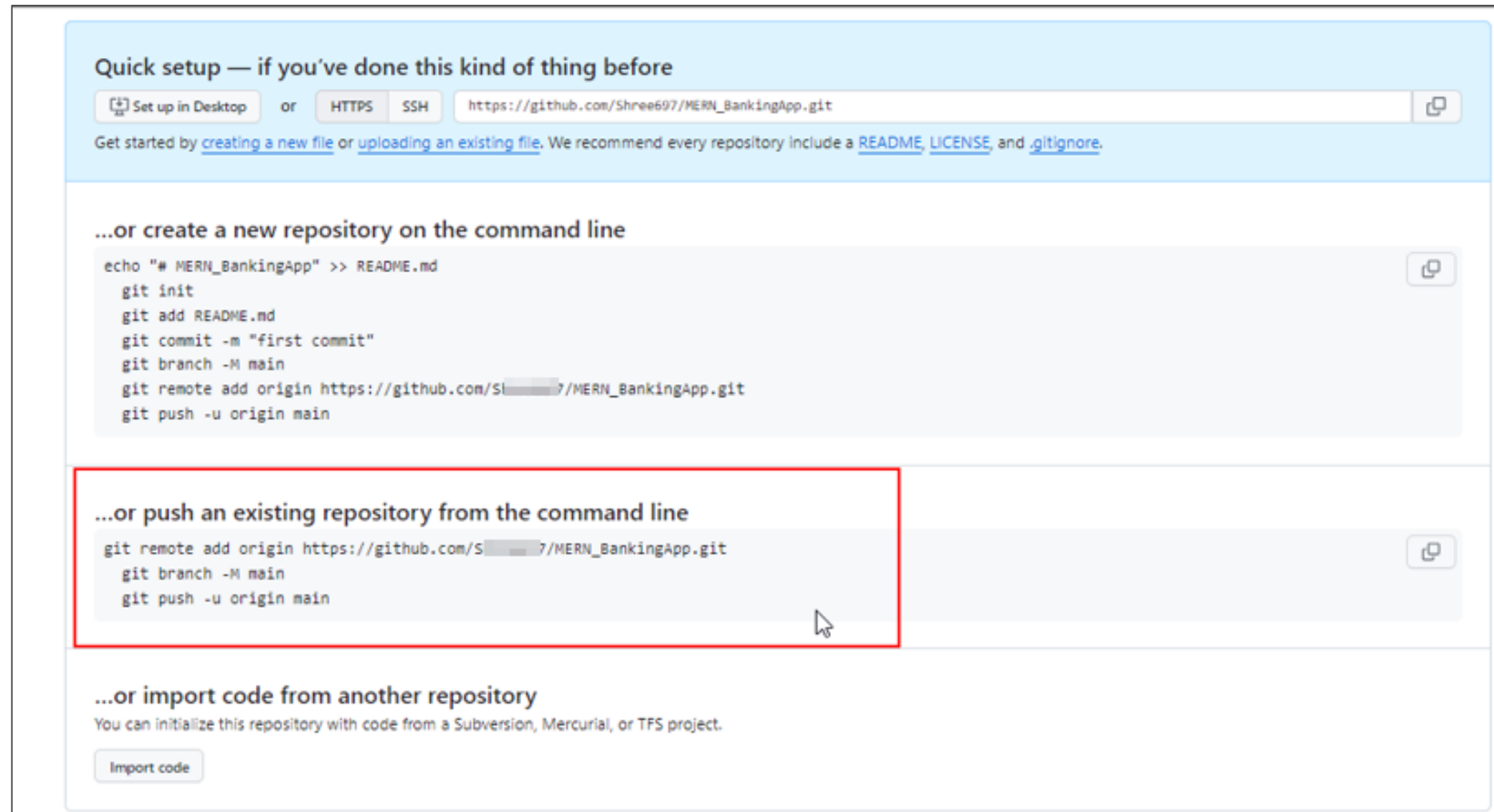
- Quick setup — if you've done this kind of thing before**
- Buttons for 'Set up in Desktop', 'HTTPS', and 'SSH'.
- A text input field containing the HTTPS URL: `https://github.com/Shree697/MERN_BankingApp.git`. This field is highlighted with a red rectangle.
- A copy icon (two overlapping squares) to the right of the input field.
- Text below the input field: 'Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).'

Below the 'Quick setup' section, there is a section titled '...or create a new repository on the command line' with a code block containing the following commands:

```
echo "# MERN_BankingApp" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
```

Push the Project into the Remote Repository

Push the local project directory into the remote repository using the following commands:



The screenshot shows the GitHub 'Quick setup' interface. At the top, there's a section titled 'Quick setup — if you've done this kind of thing before' with buttons for 'Set up in Desktop', 'HTTPS', and 'SSH'. A text input field contains the repository URL 'https://github.com/Shree697/MERN_BankingApp.git'. Below this, a note says 'Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).' The main content area has three sections: 1. '...or create a new repository on the command line' with a code block containing:

```
echo "# MERN_BankingApp" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/S[redacted]/MERN_BankingApp.git
git push -u origin main
```

 2. '...or push an existing repository from the command line' with a code block containing:

```
git remote add origin https://github.com/S[redacted]/MERN_BankingApp.git
git branch -M main
git push -u origin main
```

 This section is highlighted with a red rectangle. 3. '...or import code from another repository' with a note 'You can initialize this repository with code from a Subversion, Mercurial, or TFS project.' and an 'Import code' button.

Quick setup — if you've done this kind of thing before

[Set up in Desktop](#) or [HTTPS](#) [SSH](#)

Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

...or create a new repository on the command line

```
echo "# MERN_BankingApp" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/S[redacted]/MERN_BankingApp.git
git push -u origin main
```

...or push an existing repository from the command line

```
git remote add origin https://github.com/S[redacted]/MERN_BankingApp.git
git branch -M main
git push -u origin main
```

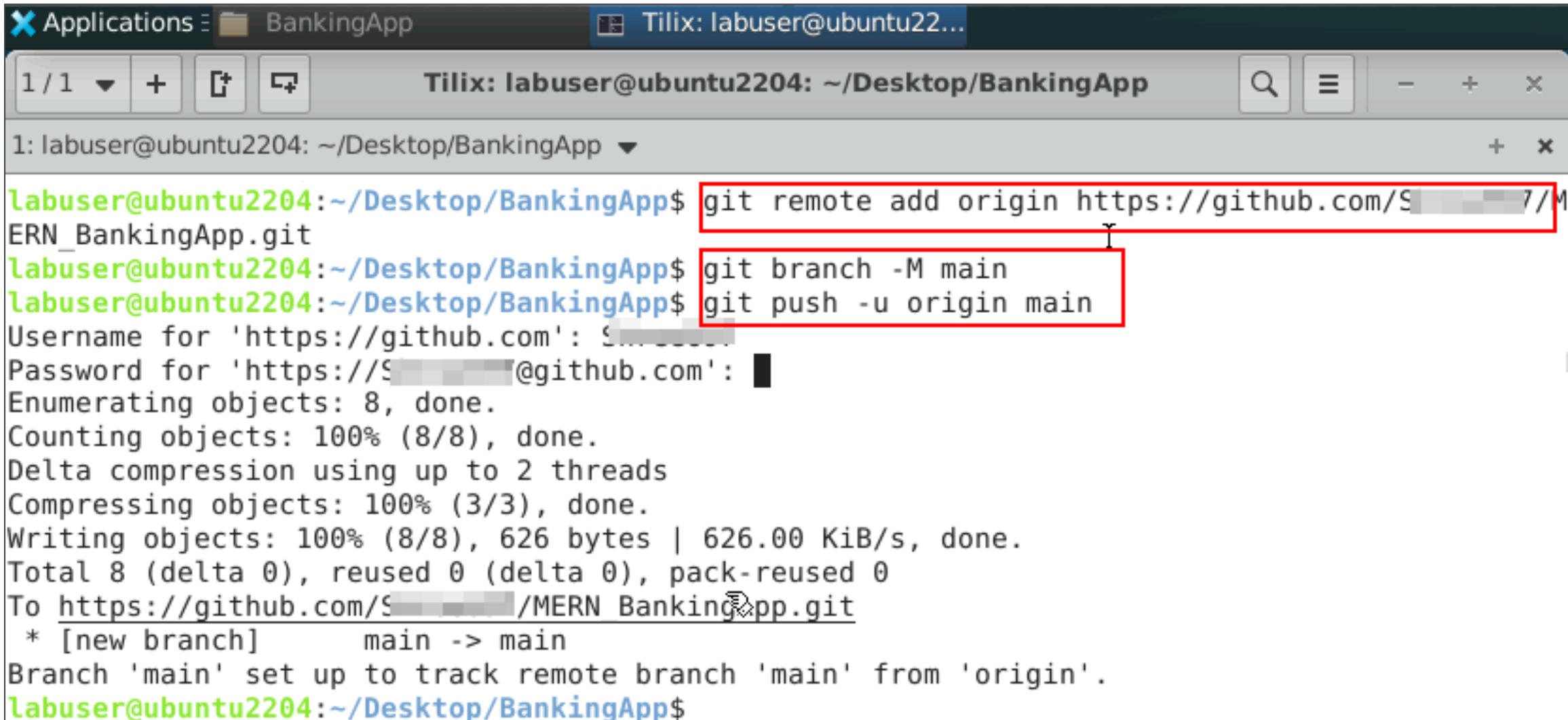
...or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

[Import code](#)

Push the Project into the Remote Repository

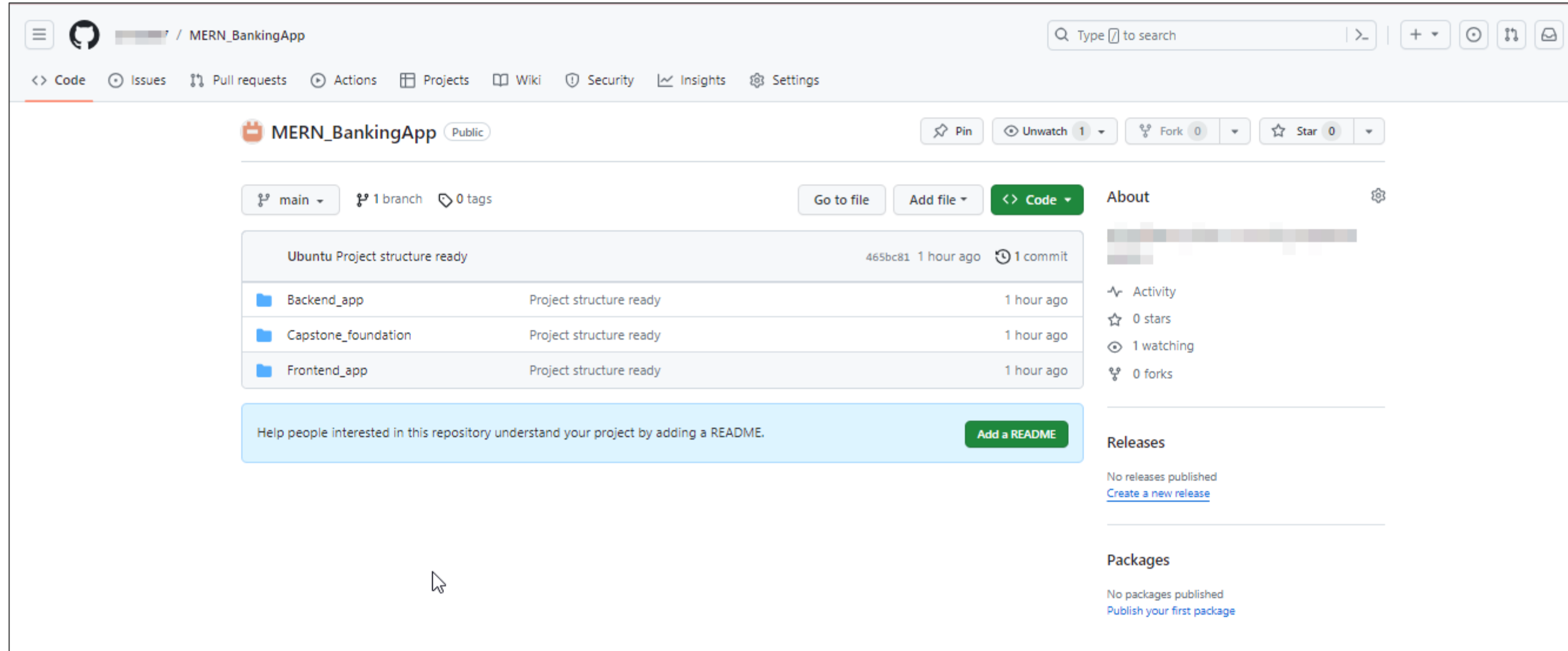
Open the terminal and execute the push commands



```
Applications BankingApp Tilix: labuser@ubuntu22...
1 / 1 + [?] [?] Tilix: labuser@ubuntu2204: ~/Desktop/BankingApp
1: labuser@ubuntu2204: ~/Desktop/BankingApp
labuser@ubuntu2204:~/Desktop/BankingApp$ git remote add origin https://github.com/S[REDACTED]7/M
ERN_BankingApp.git
labuser@ubuntu2204:~/Desktop/BankingApp$ git branch -M main
labuser@ubuntu2204:~/Desktop/BankingApp$ git push -u origin main
Username for 'https://github.com': S[REDACTED]
Password for 'https://S[REDACTED]@github.com': 
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Delta compression using up to 2 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (8/8), 626 bytes | 626.00 KiB/s, done.
Total 8 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/S[REDACTED]/MERN_BankingApp.git
 * [new branch]      main -> main
Branch 'main' set up to track remote branch 'main' from 'origin'.
labuser@ubuntu2204:~/Desktop/BankingApp$
```

Push the Project into the Remote Repository

The project structure is pushed into the remote GitHub repository.



Conclusion

By the end of the session, it is expected that:

- All the required user stories must be created for the admin and customer.
- All these user stories will give a clear picture of the project development process.
- A local Git repository must be created for all phases of the project development.
- A remote Git repository must be created which will be linked with the local Git repository of the project.
- This remote Git repository must be used for version control throughout the development process.





Thank you