

# DevOps Lab - Mukesh T P

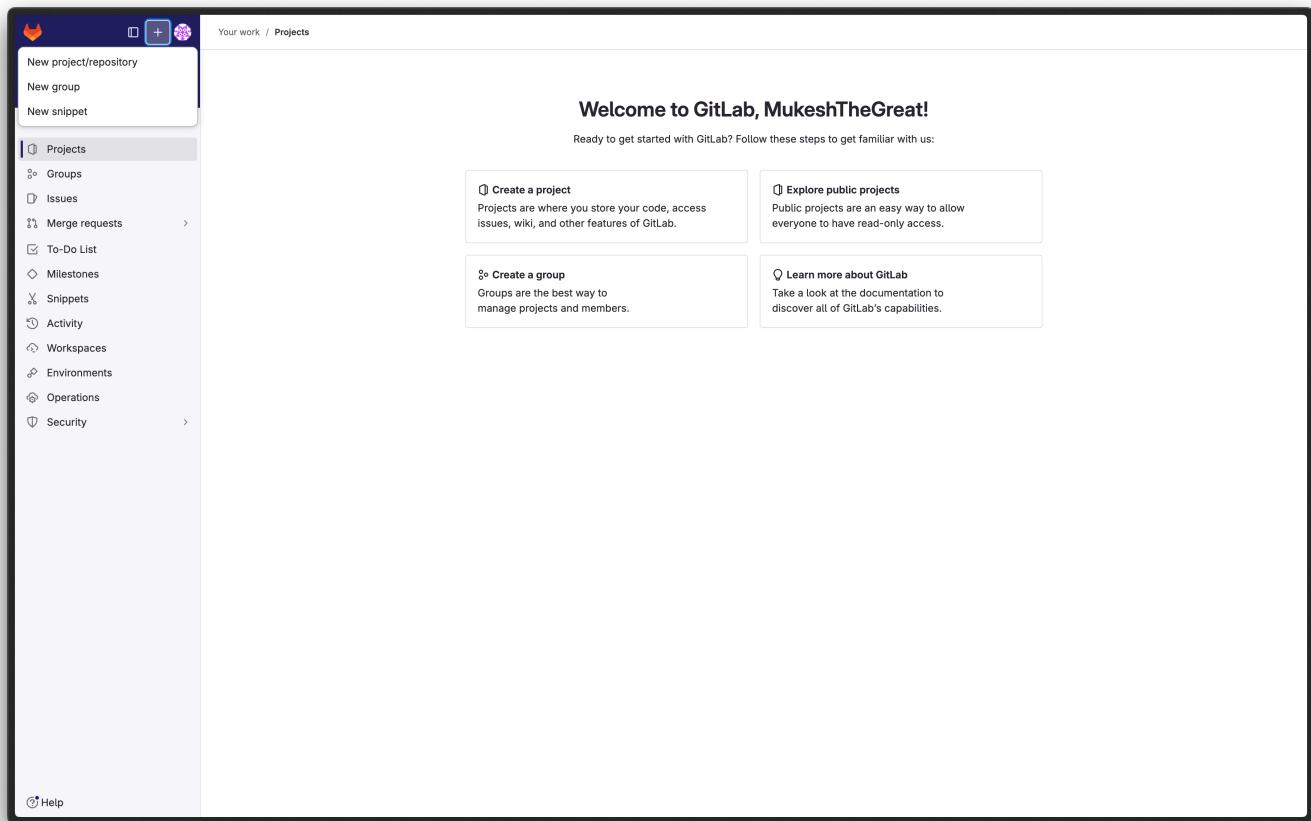
## Exercise 2

Ex. No. 1	Simple Linear Regression
03.01.2024	

## Implement GitLab Operations using Git

### 1. Creating a Repository

1. Go to your GitLab account.



2. Click on the New project button.

The screenshot shows the 'Create new project' page in GitLab. On the left, there's a sidebar titled 'Your work' with various project management options like Projects, Groups, Issues, etc. The main area has a title 'Create new project' and four cards:

- Create blank project**: Create a blank project to store your files, plan your work, and collaborate on code, among other things.
- Create from template**: Create a project pre-populated with the necessary files to get you started quickly.
- Import project**: Migrate your data from an external source like GitHub, Bitbucket, or another instance of GitLab.
- Run CI/CD for external repository**: Connect your external repository to GitLab CI/CD.

At the bottom, it says "You can also create a project from the command line. [Show command](#)". A note at the bottom left says "Open [https://gitlab.com/projects/new#import\\_project](https://gitlab.com/projects/new#import_project) in a new tab and focus it".

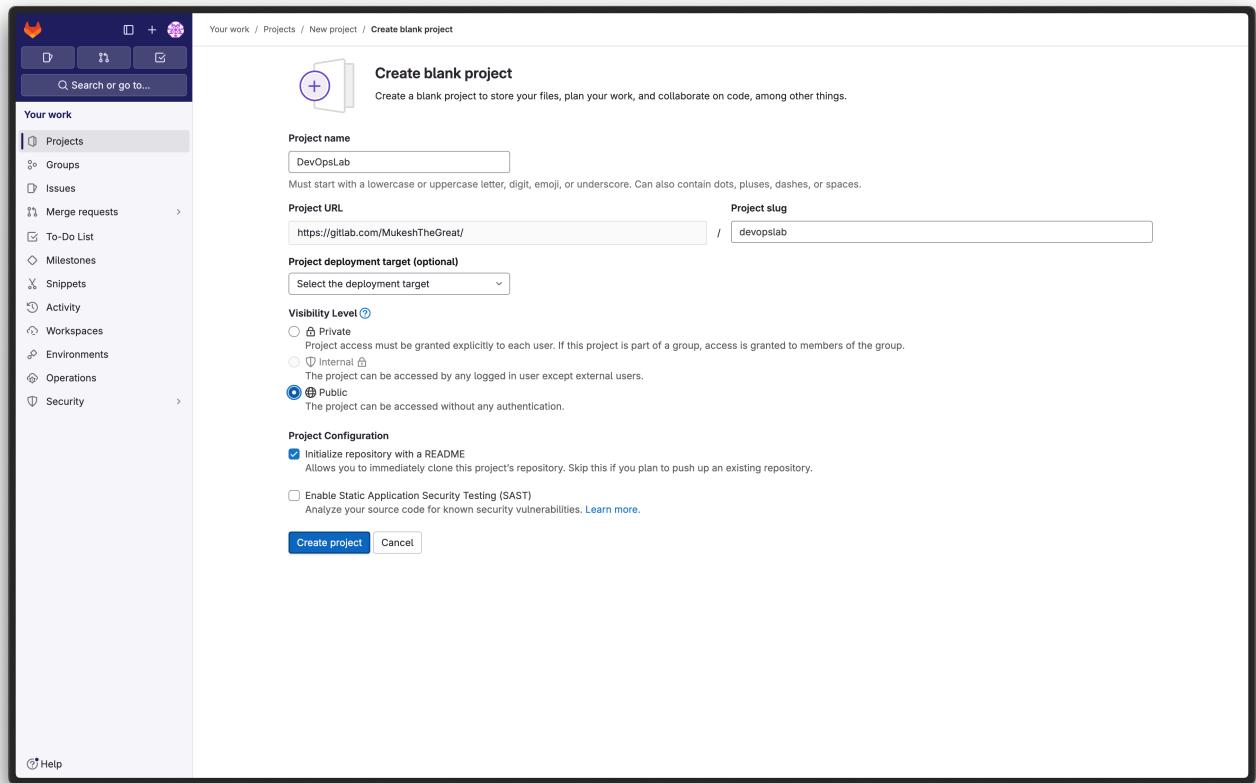
### 3. Select Create blank project .

The screenshot shows the 'Create blank project' form. The 'Project name' field contains 'My awesome project'. The 'Project URL' field contains 'https://gitlab.com/MukeshTheGreat/'. The 'Project slug' field contains 'my-awesome-project'. Under 'Project deployment target (optional)', it says 'Select the deployment target'. Under 'Visibility Level', the 'Private' radio button is selected. Under 'Project Configuration', the 'Initialize repository with a README' checkbox is checked. At the bottom are 'Create project' and 'Cancel' buttons.

### 4. Fill in the project details:

- Project name: DevOpsLab

- Project slug: devopslab (auto-filled)
- Visibility level: Choose between Private, Internal, or Public.



## 5. Click on the Create project button.

MukeshTheGreat / DevOpsLab

You can't push or pull repositories using SSH until you add an SSH key to your profile.

Add SSH key Don't show again

**Auto DevOps**

Automatically build, test, and deploy your application based on a predefined CI/CD configuration. Learn more in the Auto DevOps documentation.

Enable in settings

Project 'DevOpsLab' was successfully created.

**DevOpsLab**

main devopslab / +

History Find file Edit Code

Project information

1 Commit 1 Branch 0 Tags 3 KB Project Storage

README

+ Add LICENSE + Add CHANGELOG + Add CONTRIBUTING + Add Kubernetes cluster + Set up CI/CD + Add Wiki + Configure Integrations

Created on August 06, 2024

**DevOpsLab**

**Getting started**

To make it easy for you to get started with GitLab, here's a list of recommended next steps.

Already a pro? Just edit this README.md and make it your own. Want to make it easy? Use the template at the bottom!

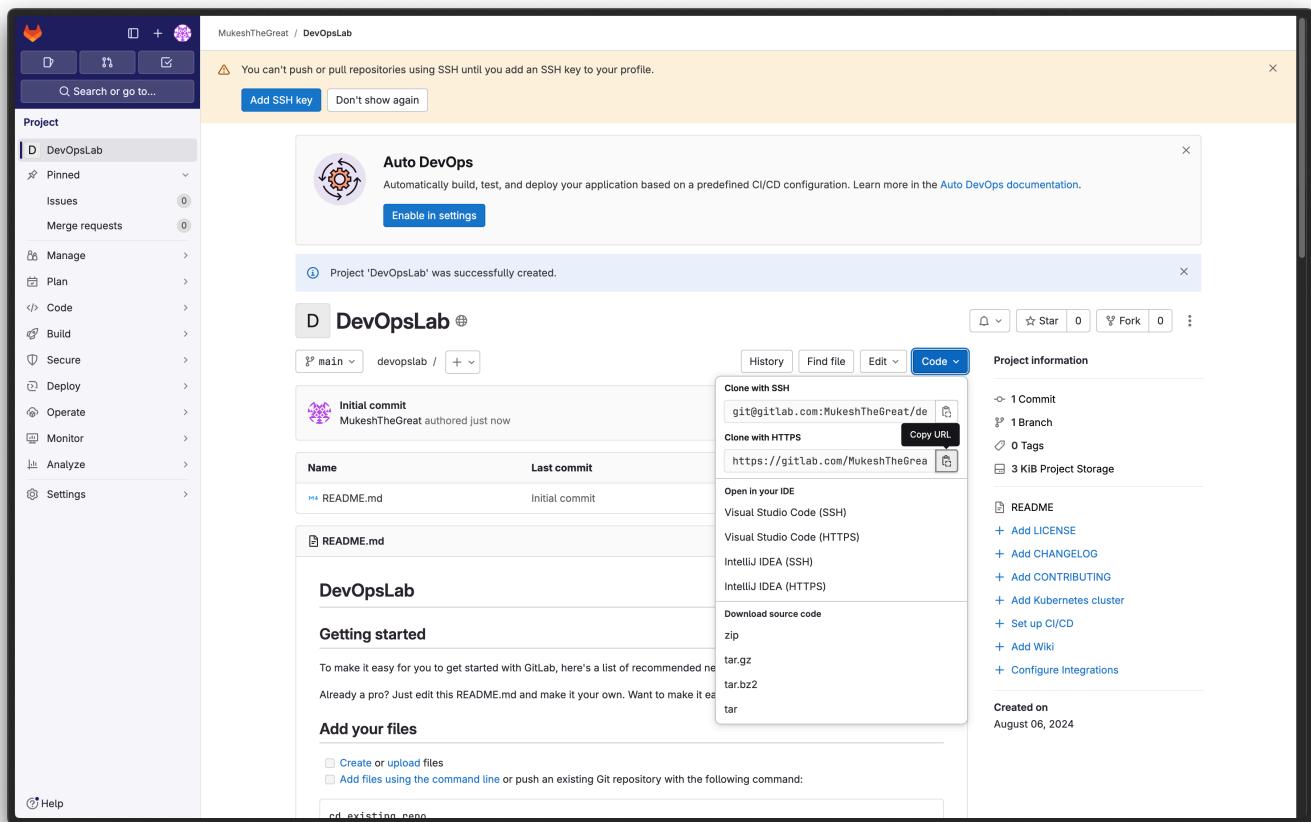
**Add your files**

Create or upload files  
 Add files using the command line or push an existing Git repository with the following command:

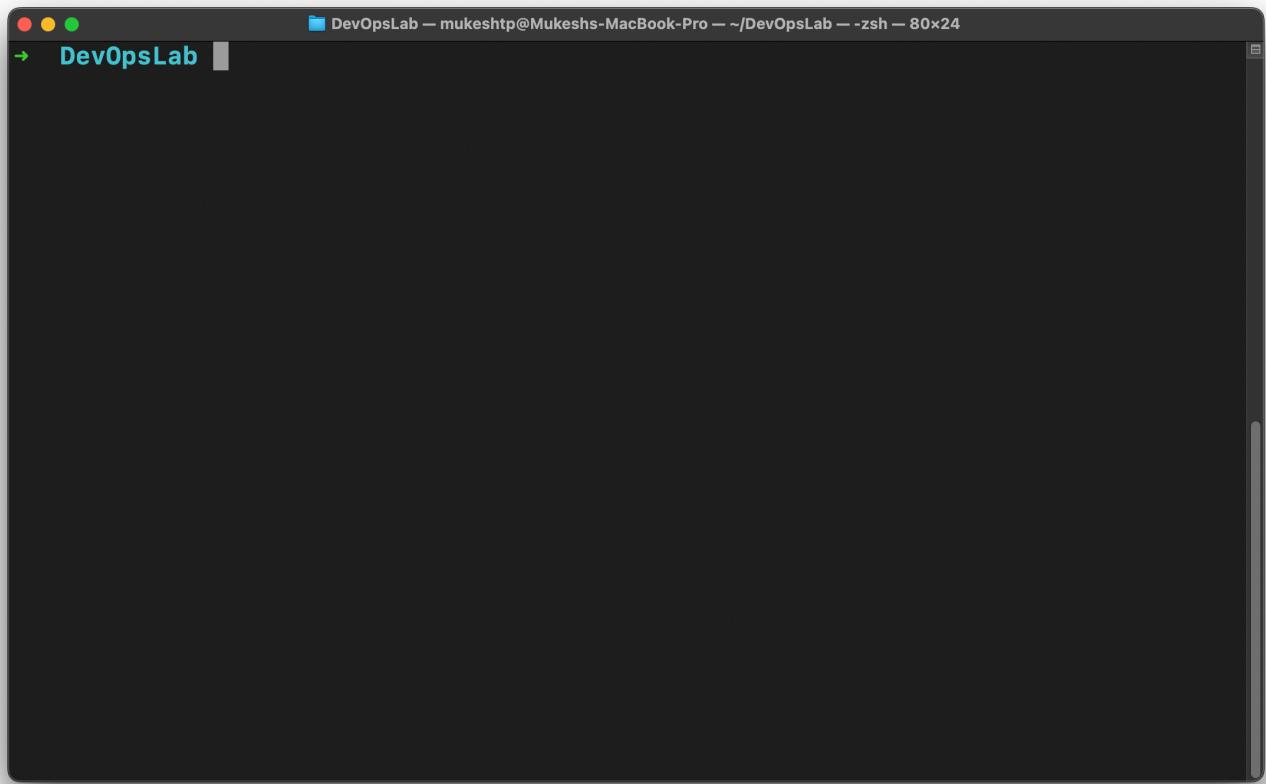
```
cd existing_repos
```

## 2. Cloning a Repository

1. In your GitLab repository, find the `Clone` button and copy the repository URL.

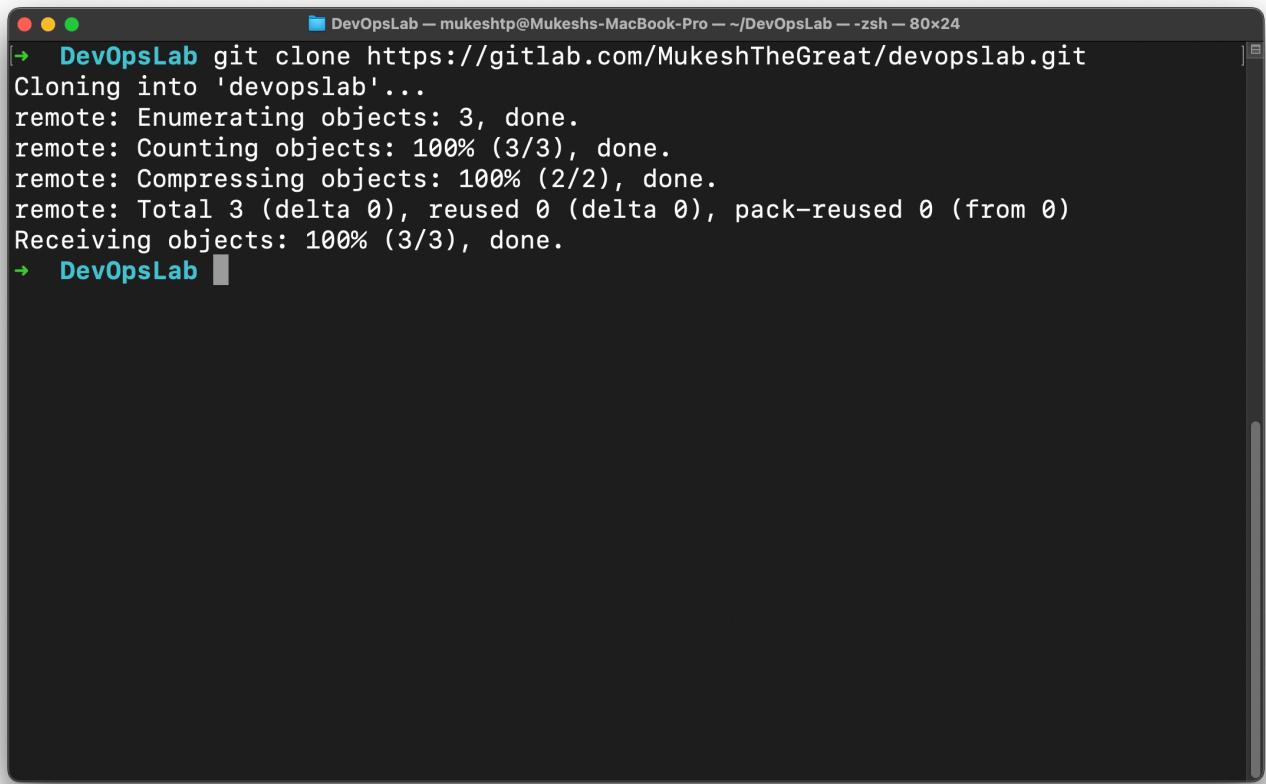


2. Open your terminal.



3. Clone the repository using the following command:

```
git clone https://gitlab.com/MukeshTheGreat/devopslab.git
```

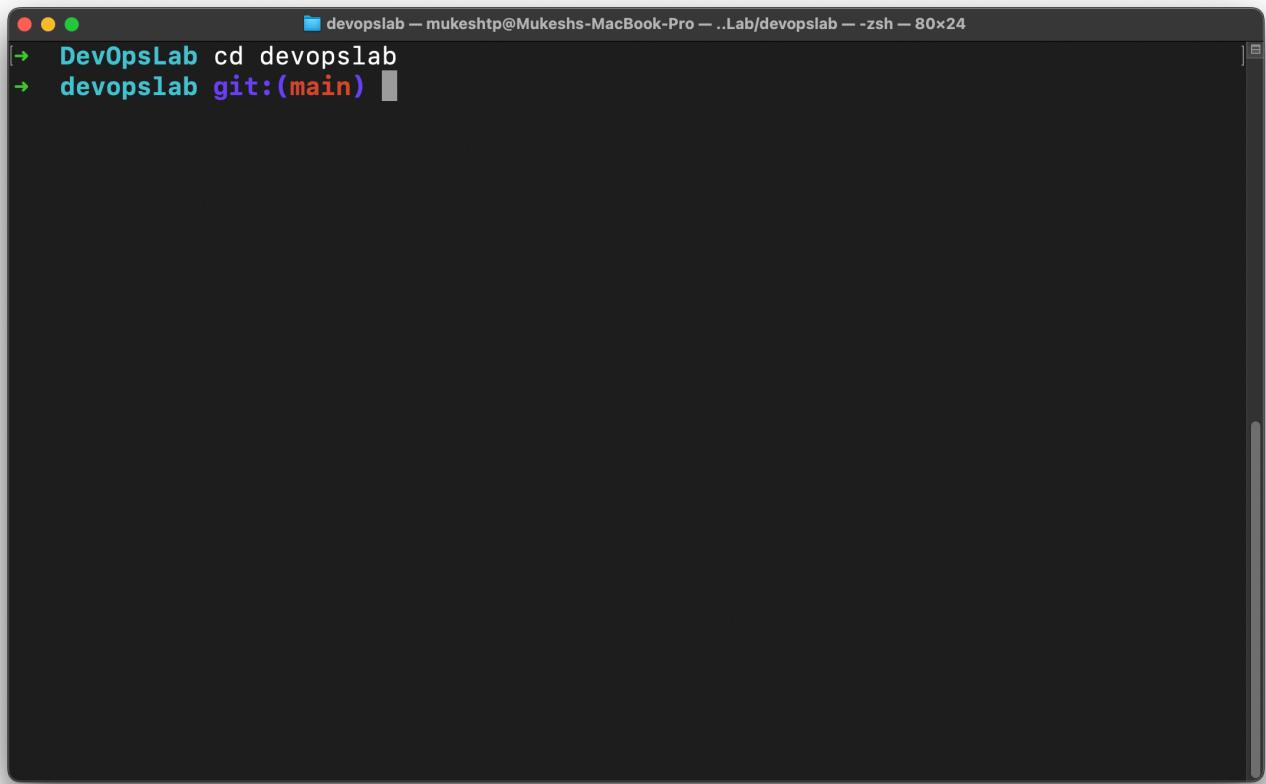


```
DevOpsLab — mukeshtp@Mukeshs-MacBook-Pro — ~/DevOpsLab — zsh — 80x24
→ DevOpsLab git clone https://gitlab.com/MukeshTheGreat/devopslab.git
Cloning into 'devopslab'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (3/3), done.
→ DevOpsLab
```

### 3. Making Changes and Creating a Branch

1. Navigate into the cloned repository:

```
cd example-repo
```

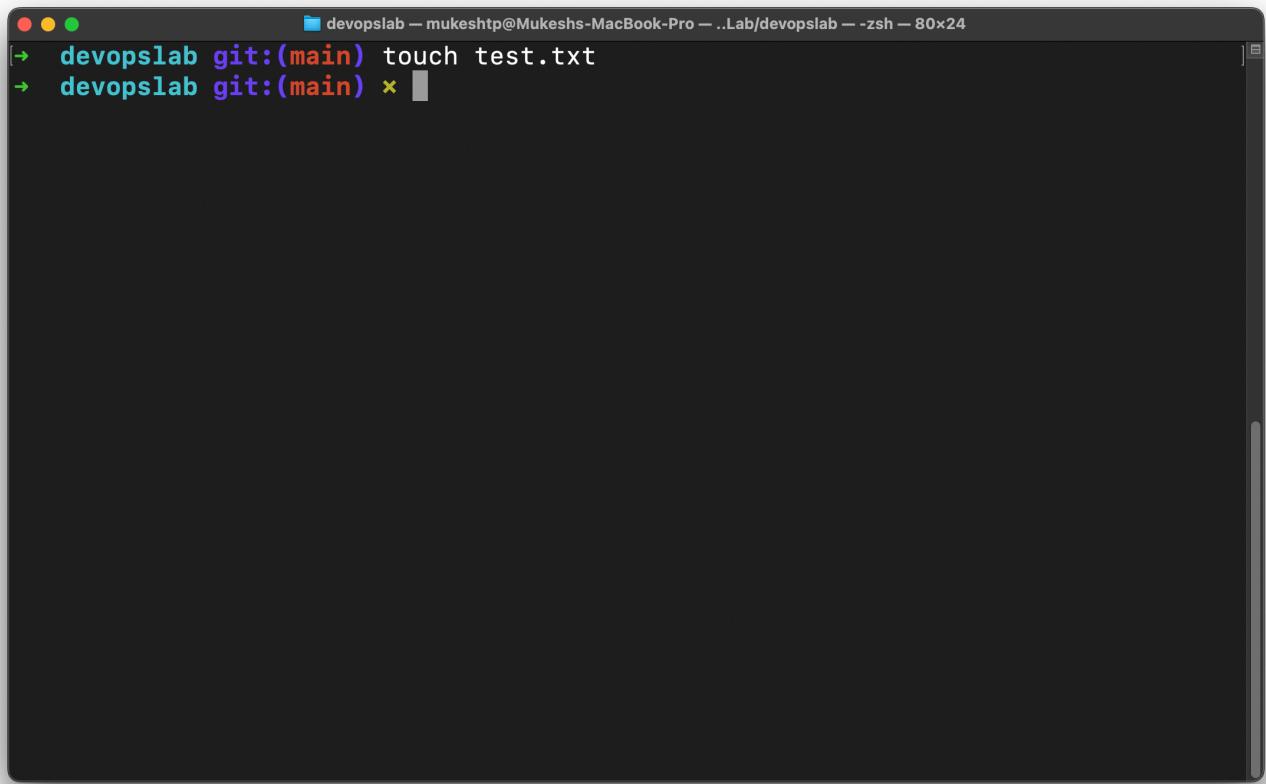


A screenshot of a terminal window titled "devopslab". The title bar also includes the text "mukeshtp@Mukeshs-MacBook-Pro" and "..Lab/devopslab - zsh - 80x24". The terminal shows two commands entered:

```
DevOpsLab cd devopslab
devopslab git:(main)
```

2. Create a new text file named `test.txt` :

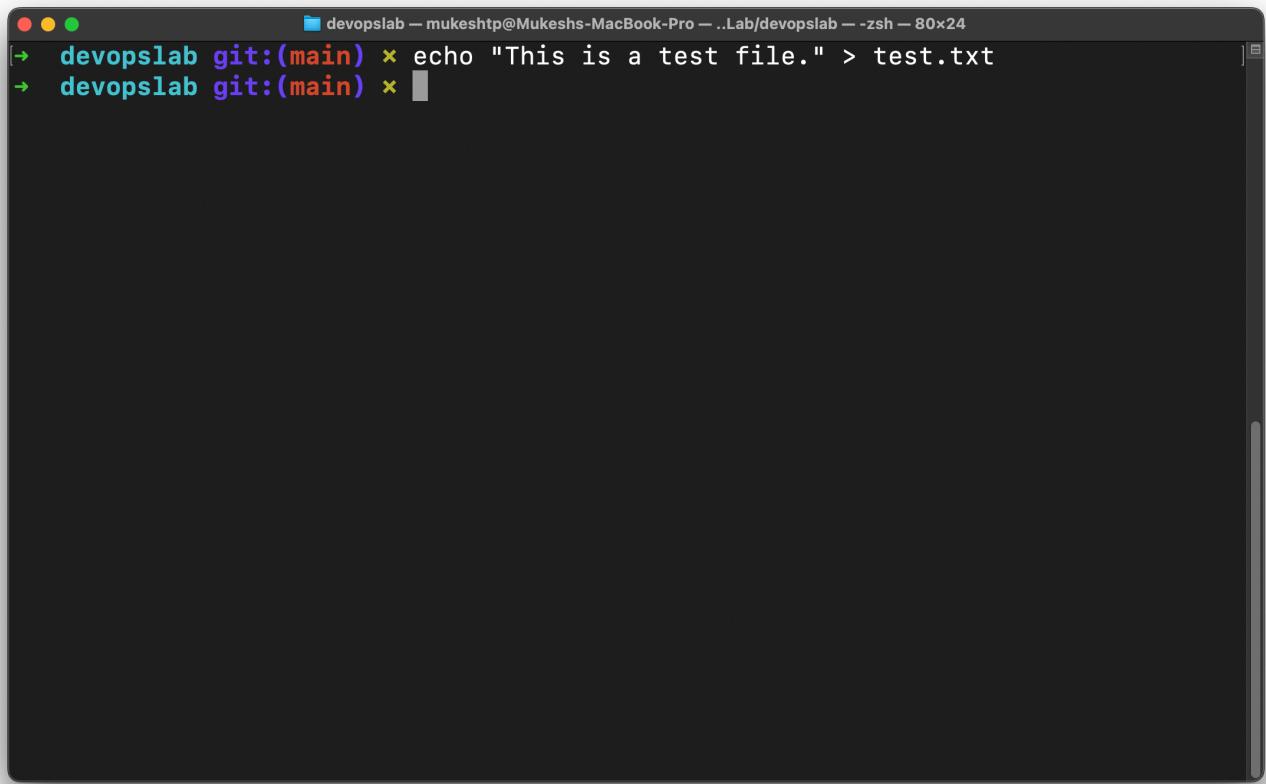
```
touch test.txt
```



```
devopslab — mukeshtp@Mukeshs-MacBook-Pro — ..Lab/devopslab — -zsh — 80x24
↳ devopslab git:(main) touch test.txt
↳ devopslab git:(main) x
```

3. Add some content to the `test.txt` file.

```
echo "This is a test file." > test.txt
```

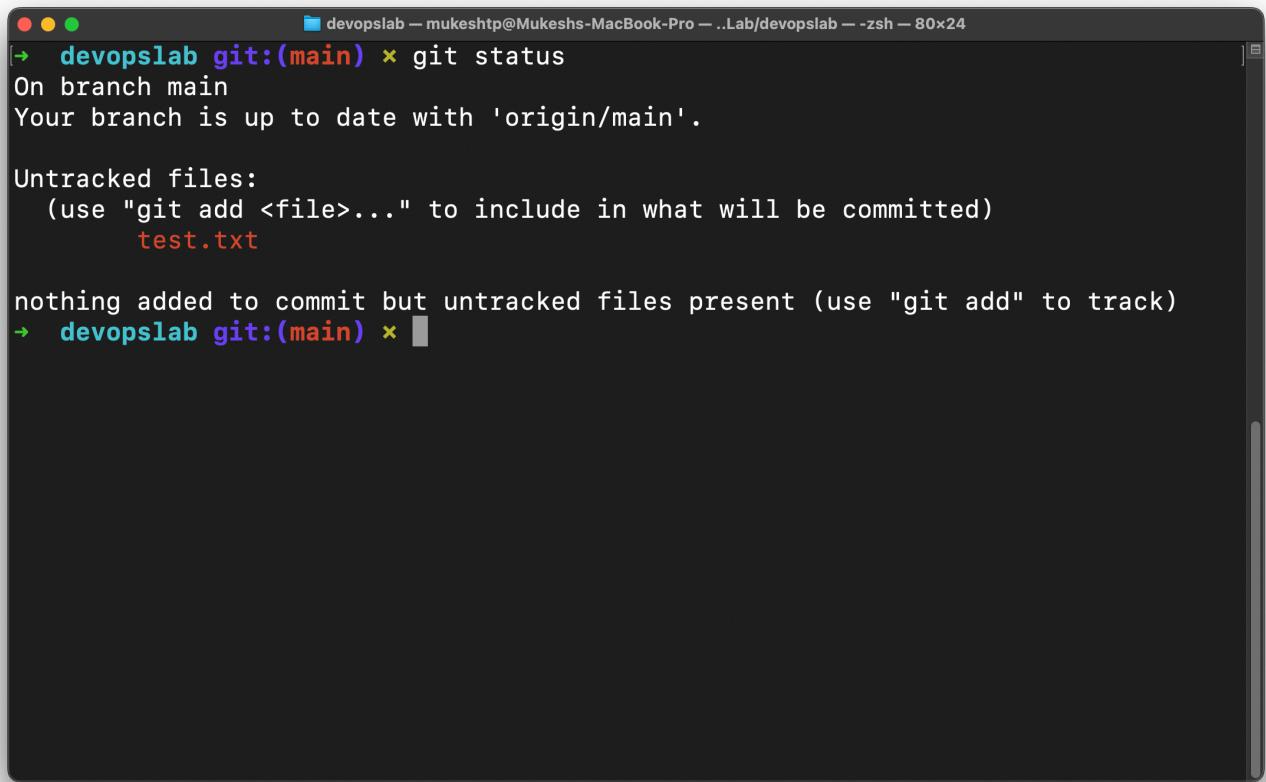


A screenshot of a macOS terminal window. The title bar says "devopslab — mukeshtp@Mukeshs-MacBook-Pro — ..Lab/devopslab — -zsh — 80x24". The main pane shows a command-line session:

```
devopslab git:(main) ✘ echo "This is a test file." > test.txt
devopslab git:(main) ✘
```

4. Check the status of the repository:

```
git status
```



devopslab — mukeshtp@Mukeshs-MacBook-Pro — ..Lab/devopslab — -zsh — 80x24

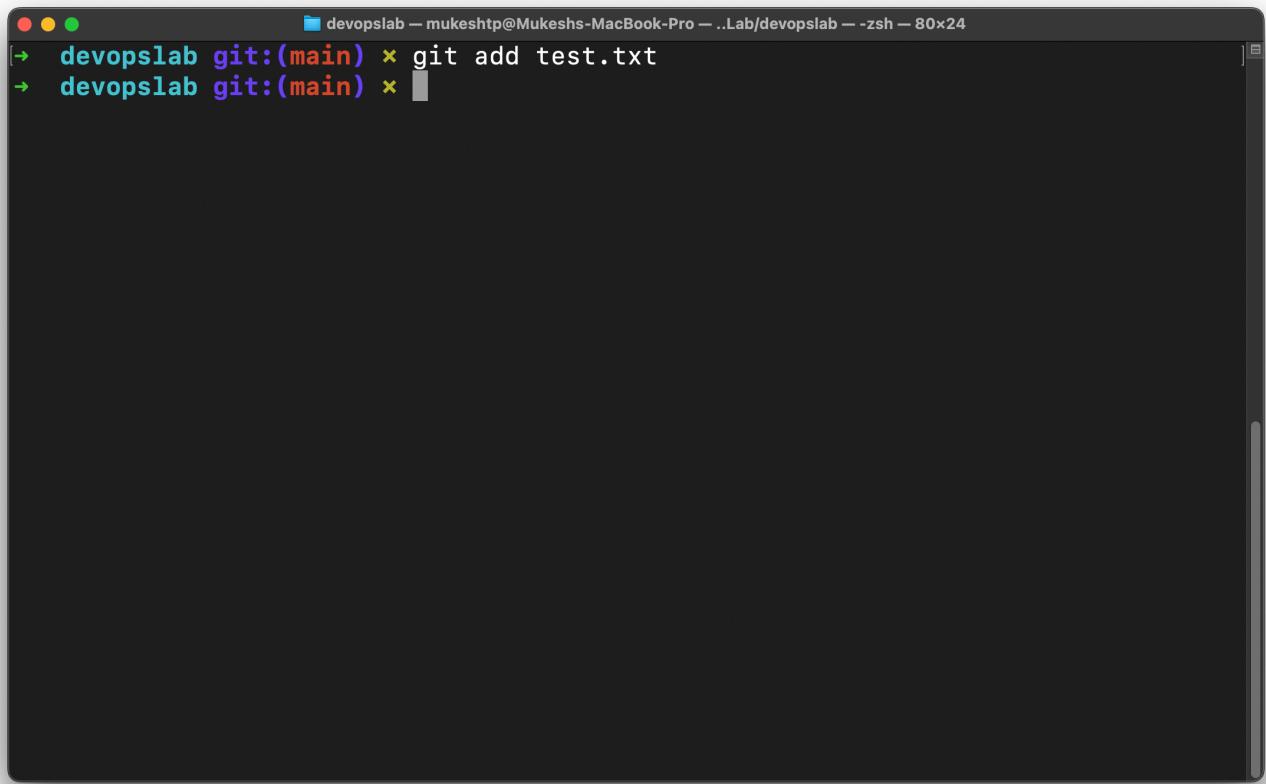
```
→ devopslab git:(main) ✘ git status
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    test.txt

nothing added to commit but untracked files present (use "git add" to track)
→ devopslab git:(main) ✘
```

5. Stage the changes for commit:

```
git add test.txt
```

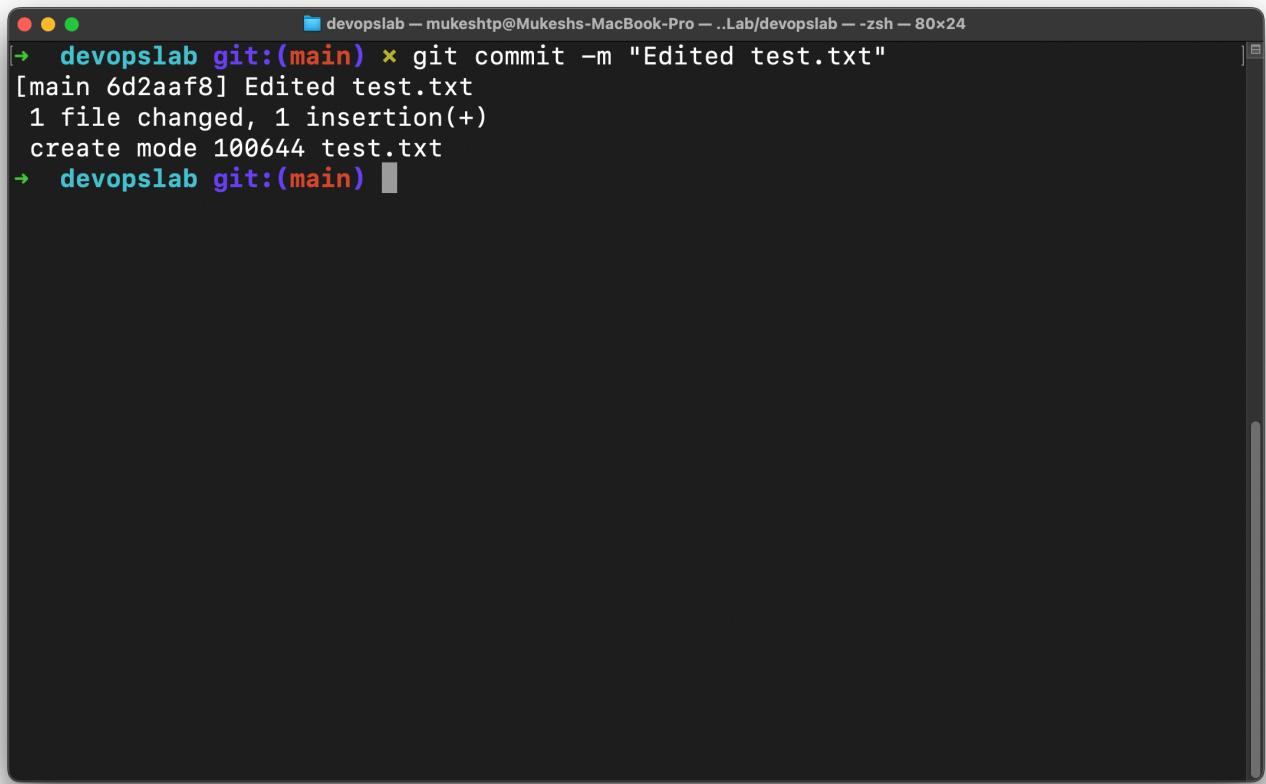


A screenshot of a macOS terminal window titled "devopslab". The window shows the command "git add test.txt" being run. The terminal has a dark background with light-colored text. The title bar includes the name of the terminal, the user's name, the host name, the path, the shell type, and the window size.

```
devopslab — mukeshtp@Mukeshs-MacBook-Pro — ..Lab/devopslab — -zsh — 80x24
↳ devopslab git:(main) ✘ git add test.txt
↳ devopslab git:(main) ✘
```

6. Commit the changes with a descriptive message:

```
git commit -m "Edited test.txt"
```

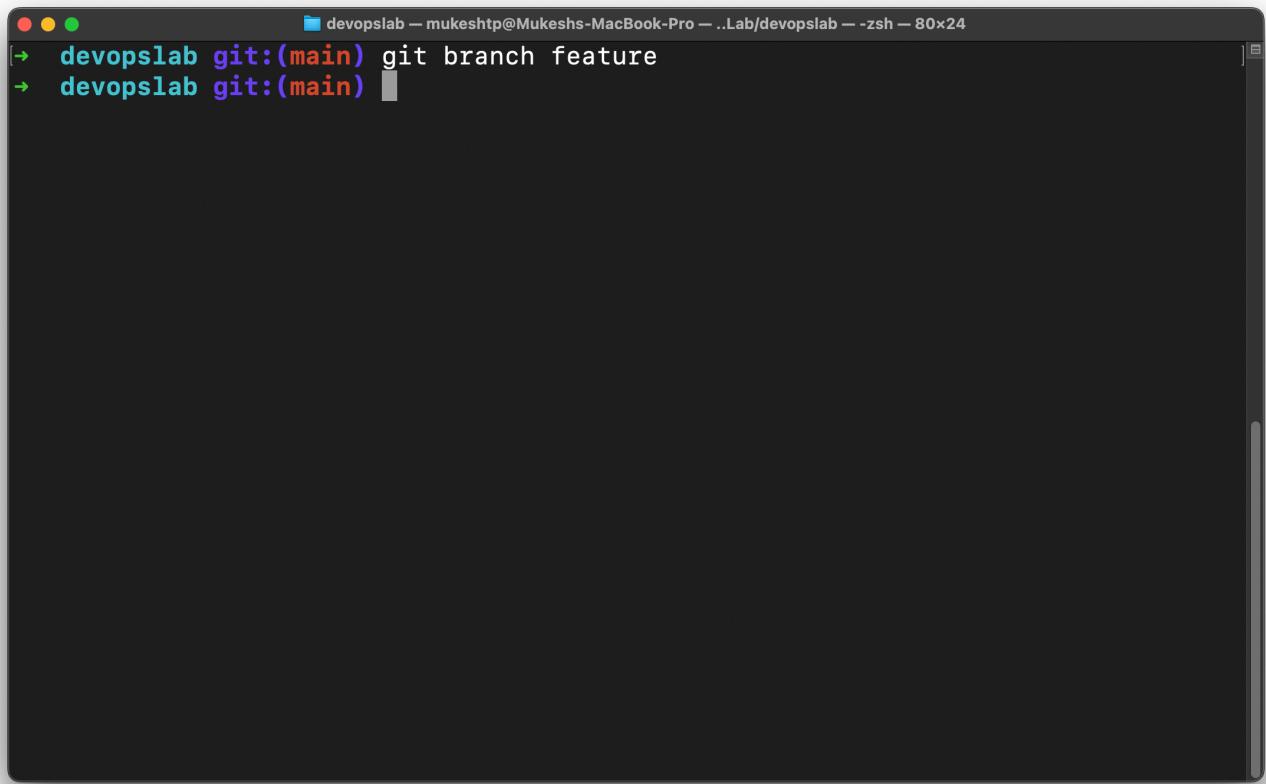


A screenshot of a macOS terminal window titled "devopslab". The window shows a command-line session where a user has committed changes to a file named "test.txt". The commit message is "Edited test.txt". The terminal window has a dark background with light-colored text. The title bar includes the window name, the user's name "mukeshtp@Mukeshs-MacBook-Pro", the path "..Lab/devopslab", the shell "-zsh", and the dimensions "80x24".

```
devopslab git:(main) ✘ git commit -m "Edited test.txt"
[main 6d2aaf8] Edited test.txt
 1 file changed, 1 insertion(+)
  create mode 100644 test.txt
devopslab git:(main)
```

7. Create a new branch named feature :

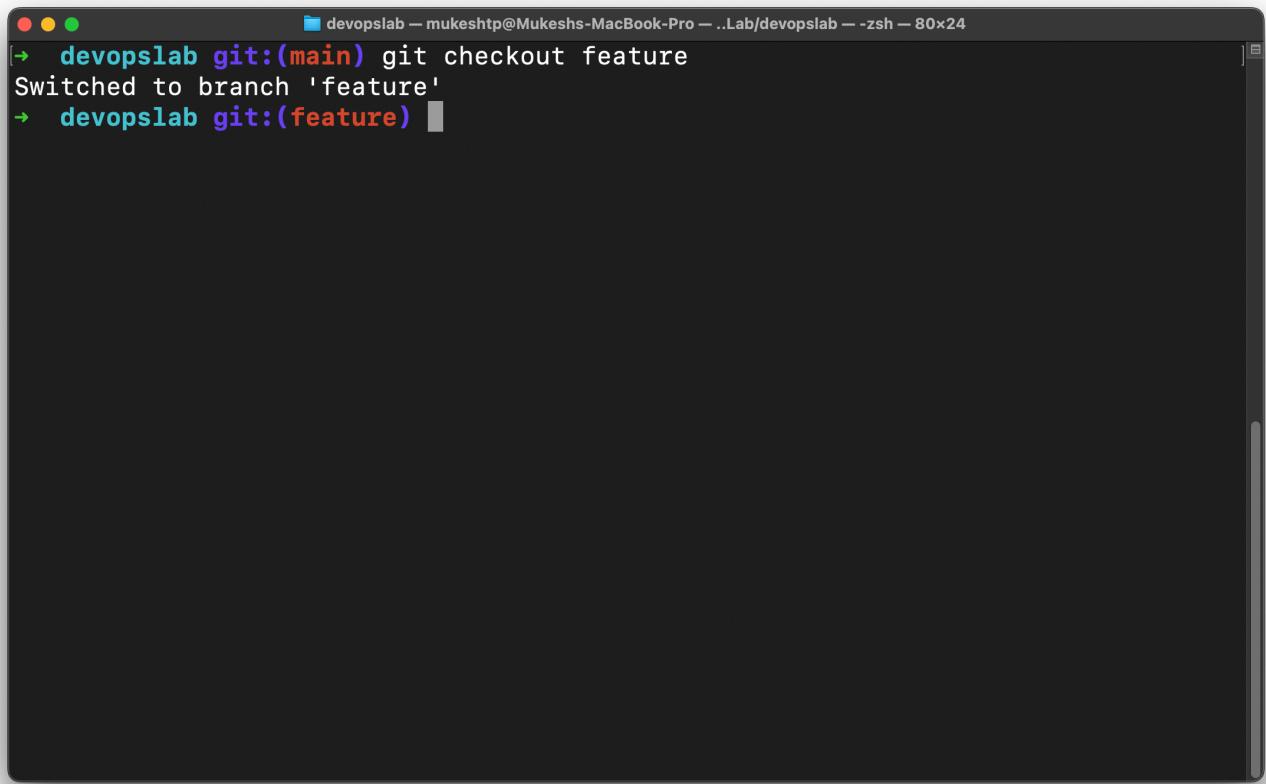
```
git branch feature
```



A screenshot of a macOS terminal window. The title bar reads "devopslab — mukeshtp@Mukeshs-MacBook-Pro — ..Lab/devopslab — -zsh — 80x24". The main pane shows the command "git branch feature" being typed, with the cursor at the end of the line. The background of the terminal is dark.

8. Switch to the `feature` branch:

```
git checkout feature
```



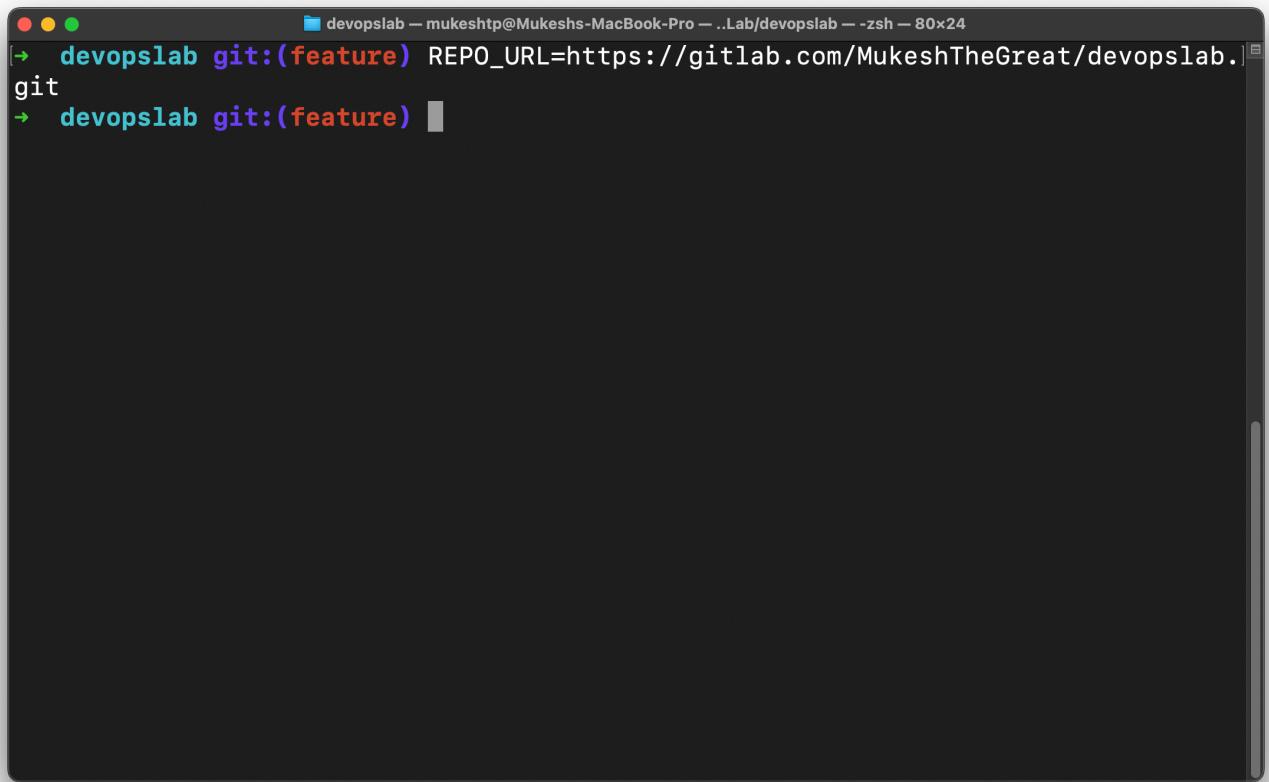
A screenshot of a macOS terminal window titled "devopslab". The window shows the command "git checkout feature" being run, which switched the branch to "feature". The terminal has a dark background with light-colored text.

```
devopslab git:(main) git checkout feature
Switched to branch 'feature'
+ devopslab git:(feature)
```

## 4. Pushing Changes to GitLab

1. Add the repository URL in a variable:

```
REPO_URL=<repository_url>
```



A screenshot of a macOS terminal window titled "devopslab". The window shows a command-line interface with the following text:

```
devopslab git:(feature) REPO_URL=https://gitlab.com/MukeshTheGreat/devopslab.
git
→ devopslab git:(feature)
```

2. Push the `feature` branch to GitLab:

```
git push -u origin feature
```

```

devopslab git:(feature) git push -u origin feature
info: please complete authentication in your browser...
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 10 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 299 bytes | 299.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: To create a merge request for feature, visit:
remote: https://gitlab.com/MukeshTheGreat/devopslab/-/merge_requests/new?merge_request%5Bsource_branch%5D=feature
remote:
To https://gitlab.com/MukeshTheGreat/devopslab.git
 * [new branch]      feature -> feature
branch 'feature' set up to track 'origin/feature'.
→ devopslab git:(feature)

```

### 3. Check your GitLab repository to confirm that the new branch `feature` is available.

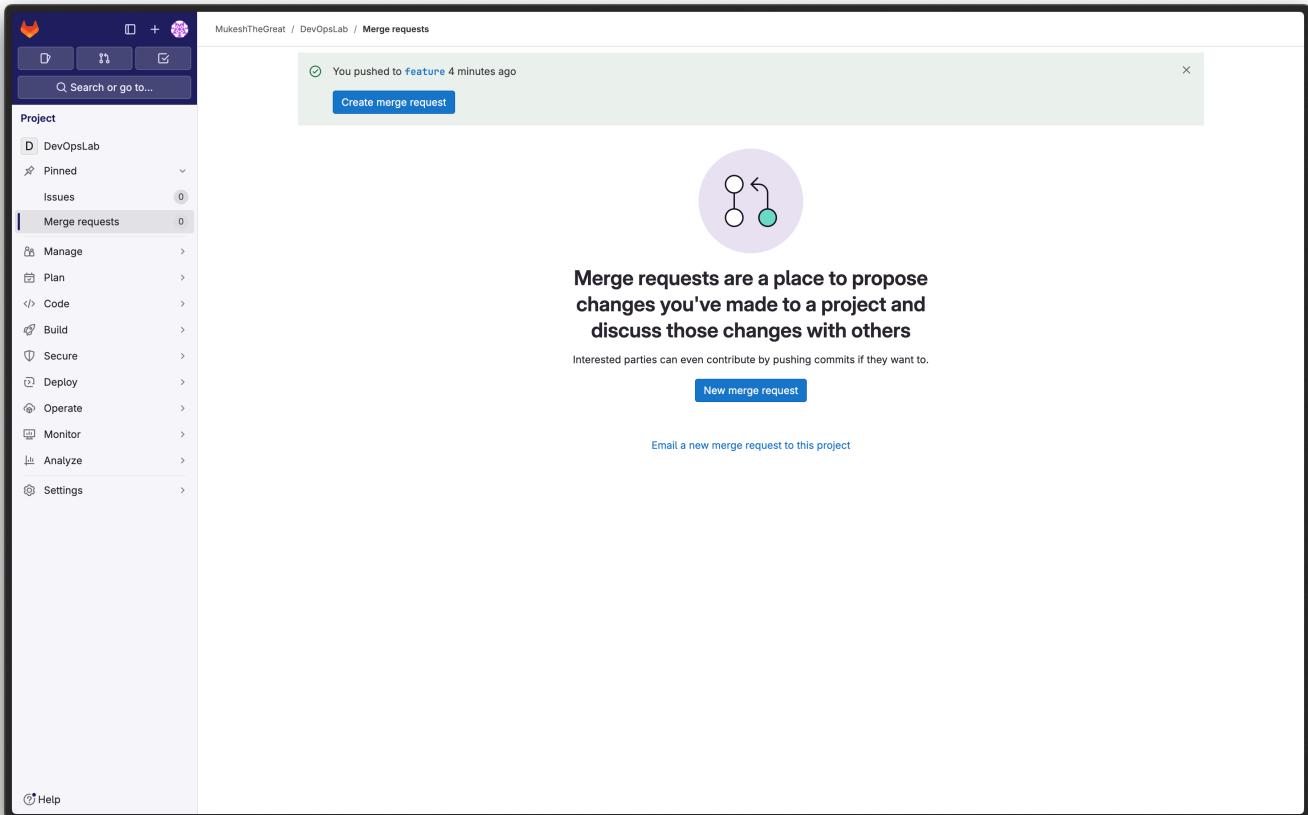
The screenshot shows the GitLab web interface for the project 'MukeshTheGreat / DevOpsLab'. The left sidebar lists various project management sections like 'Project', 'Manage', 'Plan', etc. The main area displays the 'Auto DevOps' configuration, a success message about pushing to the 'Feature' branch, and the 'DevOpsLab' repository details. The repository page shows a single commit named 'Initial commit' by 'MukeshTheGreat' and a file named 'README.md'.

## 5. Collaborating through Merge Requests

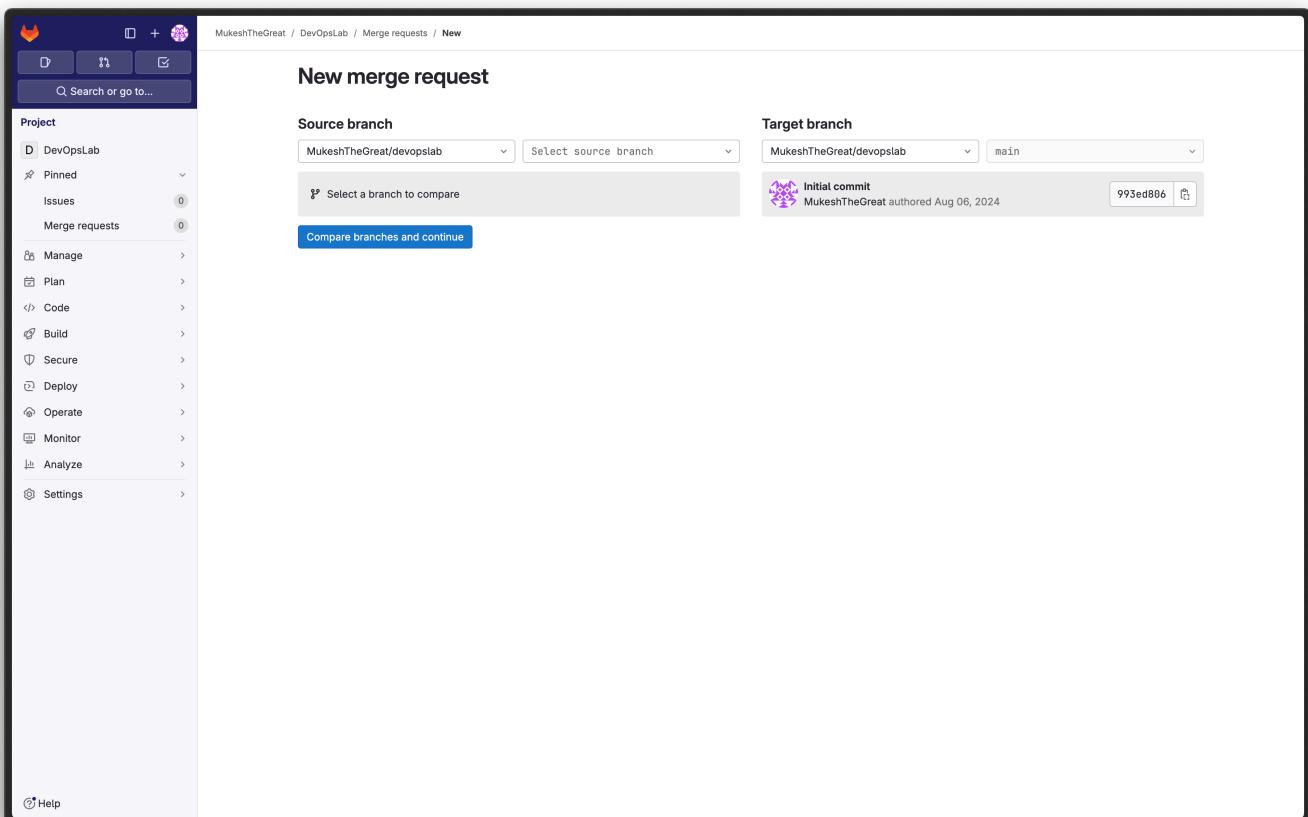
1. Go to your GitLab repository.

The screenshot shows the GitLab web interface for a project named 'DevOpsLab'. The left sidebar contains navigation links for 'Pinned', 'Issues' (0), 'Merge requests' (0), 'Manage', 'Plan', 'Code', 'Build', 'Secure', 'Deploy', 'Operate', 'Monitor', 'Analyze', and 'Settings'. A prominent yellow banner at the top right says 'You can't push or pull repositories using SSH until you add an SSH key to your profile.' with buttons for 'Add SSH key' and 'Don't show again'. Below the banner, there's a 'Auto DevOps' section with a gear icon and a link to 'Enable in settings'. A message indicates 'You pushed to feature at MukeshTheGreat / DevOpsLab just now' with a 'Create merge request' button. The main content area shows a commit history with 'Initial commit' by 'MukeshTheGreat' 30 minutes ago, a file list with 'README.md', and sections for 'Getting started' and 'Add your files'. The right sidebar displays 'Project information' including 1 Commit, 2 Branches, 0 Tags, and 3 KIB Project Storage. It also lists project management options like 'README', 'Add LICENSE', 'Add CHANGELOG', etc., and a 'Created on' section showing 'August 06, 2024'.

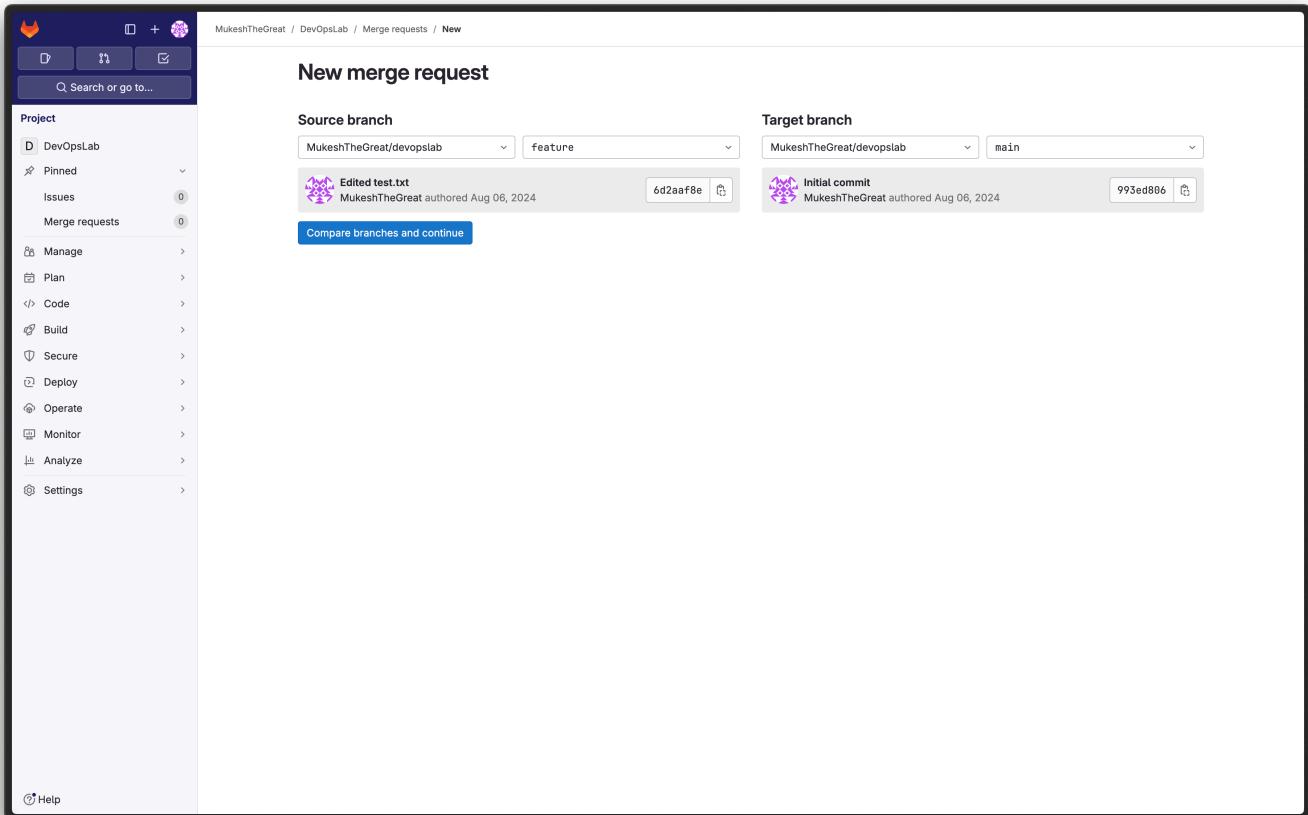
2. Click on the Merge Requests tab.



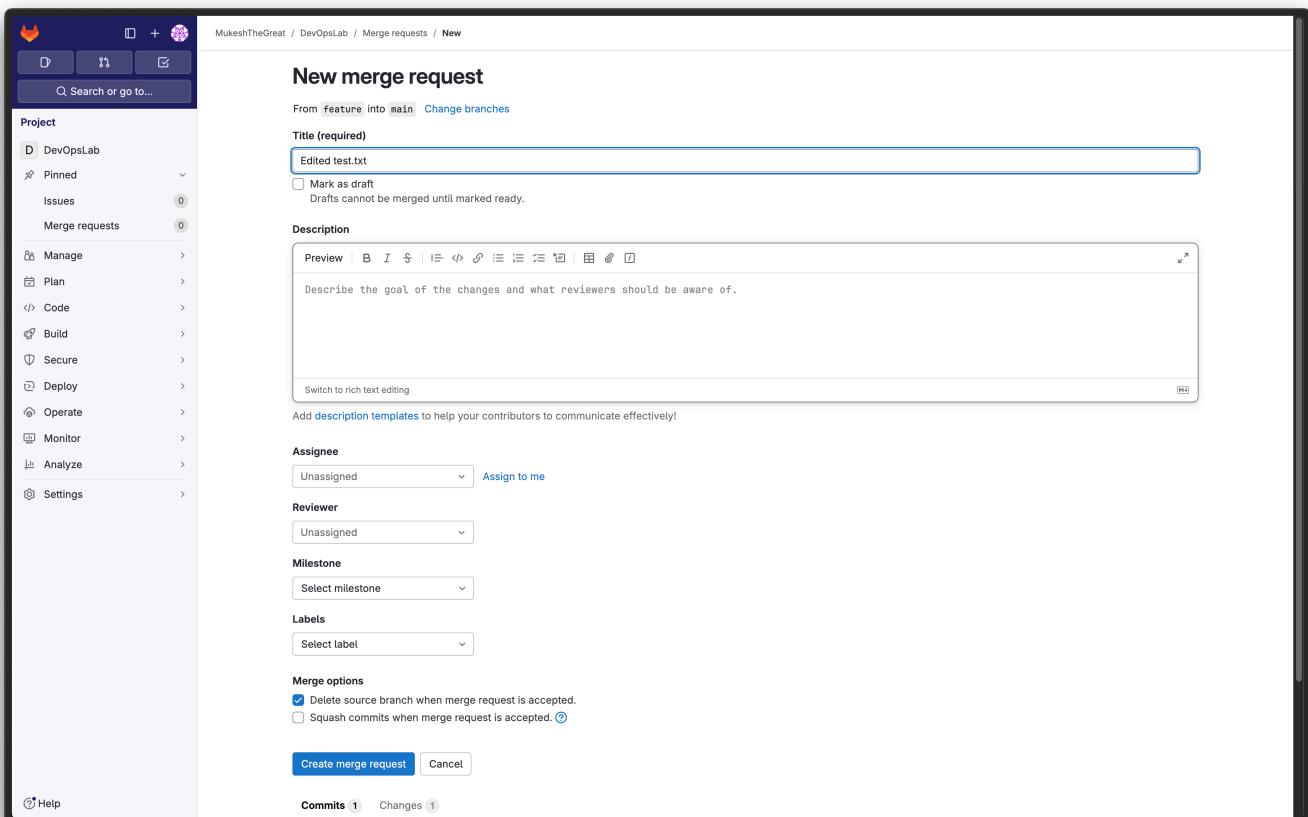
3. Click on the New merge request button.



4. Select the source branch as feature and the target branch as main (or master ).



5. Click on the Compare branches and continue button.



6. Fill in the details of the merge request and click on the Create merge request button.

The screenshot shows a merge request for a file named 'test.txt'. The left sidebar has a 'Merge requests' section selected. The main area displays the content of 'test.txt' with a preview toolbar above it. A message at the top says 'MukeshTheGreat requested to merge feature into main just now'. Below this are sections for 'Overview', 'Merge details', and 'Activity'. The 'Merge details' section indicates '1 commit and 1 merge commit will be added to main.' and 'Source branch will be deleted.' The 'Activity' section shows a recent assignment to MukeshTheGreat. On the right side, there are fields for 'Assignee' (MukeshTheGreat), 'Reviewers' (None), 'Labels' (None), 'Milestone' (None), and 'Time tracking' (No estimate or time spent). A 'Participant' section shows MukeshTheGreat.

7. Review the merge request and click on the **Merge** button to merge it.

This screenshot shows the same merge request page after the merge has been completed. The 'Merge details' section now says 'Merging! The changes are leaving the station...'. The 'Merge' button is now grayed out. The rest of the interface remains the same, including the sidebar, preview toolbar, activity log, and participant list.

The screenshot shows a GitLab interface for a project named 'DevOpsLab'. The left sidebar has a 'Merge requests' section with one item. The main area displays a merge request for 'Edited test.txt' from the 'feature' branch into the 'main' branch. The status is 'Merged' by 'MukeshTheGreat' just now. The merge details show changes merged into 'main' with commit 'dada@bebe' and the source branch deleted. The activity feed shows MukeshTheGreat assigned to the merge, mentioned in the commit, and merged it. A comment section is present at the bottom.

## 6. Syncing Changes

1. After the merge request is merged, update your local repository:

```
git checkout main  
git pull origin main
```

```
devopslab git:(feature) git checkout main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)
devopslab git:(main) git pull origin main
remote: Enumerating objects: 1, done.
remote: Counting objects: 100% (1/1), done.
remote: Total 1 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (1/1), 255 bytes | 127.00 KiB/s, done.
From https://gitlab.com/MukeshTheGreat/devopslab
 * branch            main      -> FETCH_HEAD
   993ed80..dada0eb  main      -> origin/main
Updating 6d2aaf8..dada0eb
Fast-forward
devopslab git:(main)
```