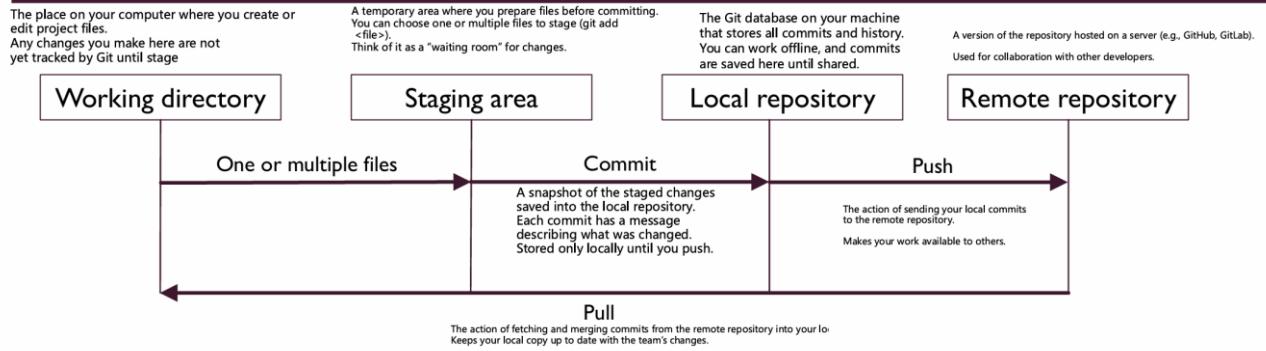


## Database Management

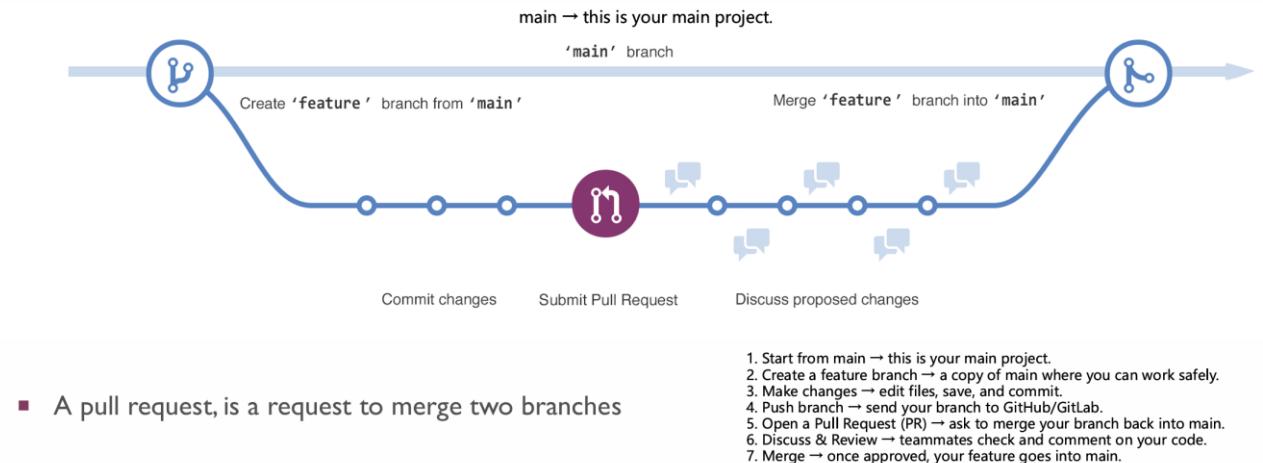
### Gitlab:

- A version control system that records changes to files
- Allows recalling and restoring previous project versions
- A distributed version control system: every client has a full copy of the repository and its history
- Supports collaboration by enabling multiple developers to work simultaneously on the same project
- Provides branching, allowing isolation and experimentation with new features
- Open-source and widely adopted in the software industry

## STAGE, COMMIT, PUSH, AND PULL



## BRANCHING WITH GIT



## REPOSITORY AND BRANCHING

### First create a Repository

The image shows a composite of three screenshots related to GitHub repository management.

**Top Screenshot:** A user's GitHub profile page for "dkgcska". It displays a circular profile picture, the name "Dino Karagic", and the handle "dkgcska". Below the profile are sections for "Achievements" (with one badge shown) and "Contribution activity" (a heatmap showing contributions from September 2024 to September 2023). At the top, there are links for "Overview", "Repositories 2", "Projects", "Packages", and "Stars". A search bar at the top right contains the placeholder "Type / to search".

**Middle Screenshot:** A list of repositories under the "dkgcska" account. Two repositories are listed: "git\_training1" (Public, updated 17 hours ago) and "git\_training" (Public, updated 18 hours ago). Both have a "Star" button next to them. The interface includes a search bar, dropdown menus for "Type", "Language", and "Sort", and a prominent green "New" button.

**Bottom Screenshot:** The process of creating a new repository. The "General" tab is selected, showing fields for "Owner" (set to "dkgcska") and "Repository name" (with a placeholder "/"). Below these is a note about repository names being short and memorable, with a suggestion "urban-garbanzo". The "Description" field is empty. The "Configuration" tab is partially visible below, containing options for visibility ("Public"), README, .gitignore, and license selection. A large green "Create repository" button is at the bottom of the form.

## Create a new branch

- When you create a branch, it starts as an exact copy of main at that moment.

The screenshot shows a GitHub repository named "git\_training3". The repository has one branch, "main", and one commit, "Initial commit". The README file contains the text "git\_training3". On the right side of the repository page, there are sections for "About", "Releases", and "Packages".

Below the repository page, the "Branches" section is visible. It shows the "main" branch under the "Default" tab. A green button labeled "New branch" is located in the top right corner of this section.

A modal window titled "Create a branch" is open in the foreground. It asks for a "New branch name" which is currently "readme\_edits3". Below that is a "Source" dropdown set to "main". At the bottom of the modal are two buttons: "Cancel" and "Create new branch". The "Create new branch" button is highlighted with a red box.

## Edit branch (file)

The screenshot shows the GitHub interface for managing branches and editing files.

**Branches** (Top Left):

- Overview tab selected.
- Search bar: Search branches...
- Default** section:
  - Branch: main (status: Updated 3 minutes ago, Default, Pull request)
- Your branches** section:
  - Branch: readme\_edits3 (status: Updated 1 minute ago, Pull request)
- Active branches** section:
  - Branch: readme\_edits3 (status: Updated 1 minute ago, Pull request)

**git\_training3** (Main Repository View):

- Branch: readme\_edits3 (2 Branches, 0 Tags)
- Go to file search bar.
- Add file, Code, and About tabs.
- This branch is up to date with main.
- Contributors: dkgcsra (Initial commit, 0957ea4 - 5 minutes ago, 1 Commit).
- Files: README.md (Initial commit, 5 minutes ago).
- File editor for README:

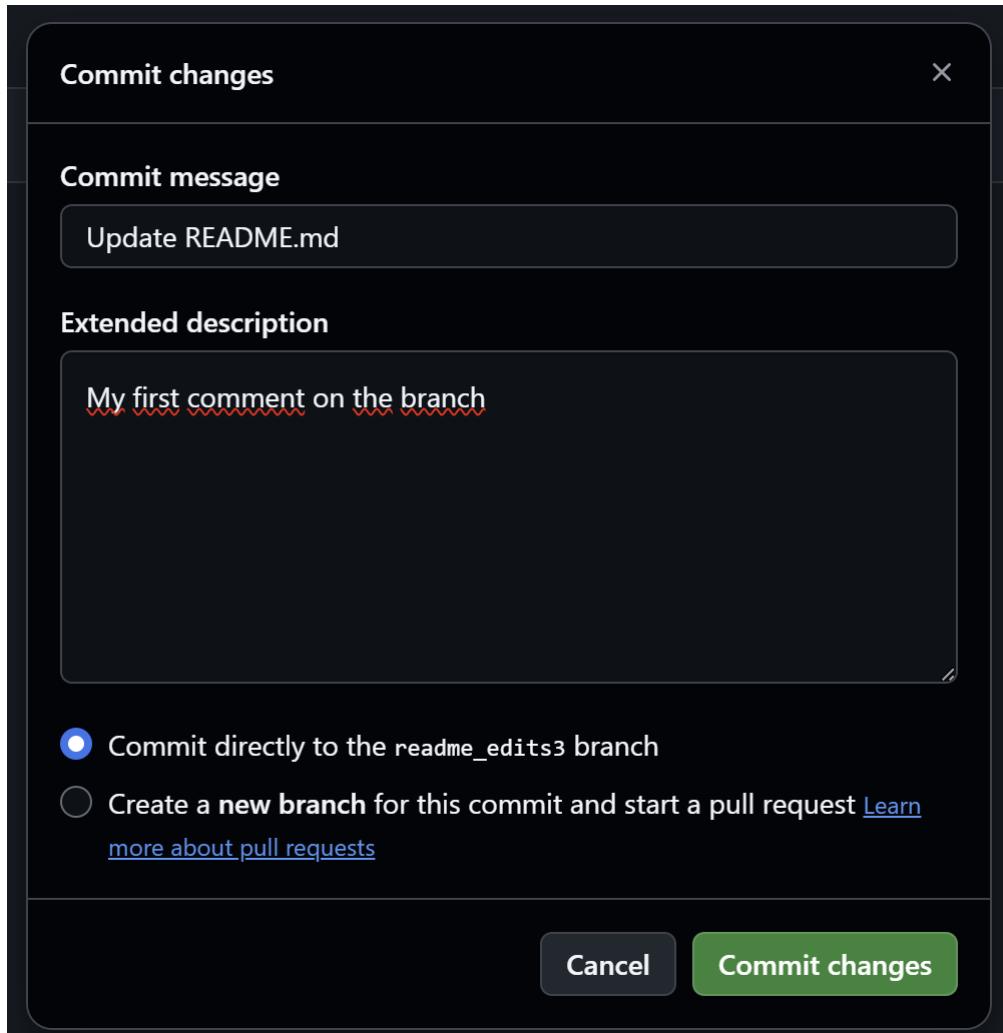
  - Content: # git\_training3  
This is just a Test
  - Save icon highlighted with a red box.

- Right sidebar:
  - About: No description, website, or topics provided.
  - Readme, Activity, Stars, Forks, Watching metrics.
  - Releases: No releases published, Create a new release.
  - Packages: No packages published, Publish your first package.

**File Editor (Bottom):**

- git\_training3 / README.md in readme\_edits3
- Buttons: Edit, Preview, Save, Cancel changes, Commit changes... (highlighted with a red box).
- Text area:

```
1 # git_training3
2 This is just a Test
```
- Editor settings: Spaces, 2, Soft wrap.



#### Option 1: Commit directly to the readme\_edits3 branch

- Your changes will be saved immediately to the existing branch (readme\_edits3).
- No pull request is created.
- Best when:
  - You're working alone on a project, or
  - The branch is already a feature branch meant for experiments.

#### Option 2: Create a new branch for this commit and start a pull request

- A new branch will be created, starting from readme\_edits3.
- Your changes go into this new branch.
- GitHub will suggest creating a Pull Request (PR) so others can review, discuss, and approve before merging.
- Best when:
  - You're collaborating with a team,
  - You want to test something safely without affecting the main branch or even your current feature branch,
  - You want a review process before merging.

## Compare changes

The image consists of three vertically stacked screenshots of a GitHub repository page, specifically for the repository `git_training3`. The top two screenshots show the code view for the `README.md` file, while the bottom one shows the main repository page.

**Screenshot 1 (Top): Code View for README.md**

This screenshot shows the code editor for the `README.md` file. The content is:

```
git_training3
This is just a Test
```

At the top, there are navigation links: Code, Issues, Pull requests, Actions, Projects, Wiki, Security, Insights, Settings. Below the code editor, there are buttons for Preview, Code, Blame, and Raw, along with download and edit options.

**Screenshot 2 (Middle): Repository Overview**

This screenshot shows the main repository page for `git_training3`. It includes the following information:

- Branches:** readme\_edits3 (selected), 2 Branches, 0 Tags
- Commits:** dkgcskra Update README.md (46d5fa7 - 3 minutes ago), 2 Commits
- File List:** README, README.md
- Contributor Information:** This branch is 1 commit ahead of main.
- Actions:** Pin, Watch (0), Fork (0), Star (0)
- About:** No description, website, or topics provided.
- Activity:** Readme, Activity, 0 stars, 0 watching, 0 forks
- Releases:** No releases published, Create a new release
- Packages:** No packages published, Publish your first package

**Screenshot 3 (Bottom): Repository Overview with Comparison Overlay**

This screenshot is similar to the middle one but includes a comparison overlay for the `readme_edits3` branch. The overlay contains the following text:

This branch is 1 commit ahead of main .  
Open a pull request to contribute your changes upstream.

Two buttons are present in the overlay:

- A grey button labeled "Compare" with a red border.
- A green button labeled "Open pull request".

## Comparing changes

Choose two branches to see what's changed or to start a new pull request. If you need to, you can also [compare across forks](#) or [learn more about diff comparisons](#).

base: main < ... compare: readme\_edits3

Discuss and review the changes in this comparison with others. [Learn about pull requests](#)

Create pull request

-o- 1 commit 1 file changed 1 contributor

Commits on Sep 18, 2025

Update README.md ... dkgcokra authored 5 minutes ago

Verified 46dfa7

Showing 1 changed file with 2 additions and 1 deletion.

Split Unified

3 README.md

... ... @@ -1 +1,2 @@

1 - # git\_training3

1 + # git\_training3

2 + This is just a Test

## Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also [compare across forks](#). [Learn more about diff comparisons here](#).

base: main < ... compare: readme\_edits3

Add a title

Update README.md

Add a description

Write Preview

My first comment on the branch

Markdown is supported Paste, drop, or click to add files

Create pull request

Reviewers No reviews

Assignees No one—assign yourself

Labels None yet

Projects None yet

Milestone No milestone

Development Use [Closing keywords](#) in the description to automatically close issues

Helpful resources

As there are no conflicts with the main branch, we can merge the branches

Conversation 0 Commits 1 Checks 0 Files changed 1

dkgcokra commented now

My first comment on the branch

update README.md

No conflicts with base branch Merging can be performed automatically.

Merge pull request You can also merge this with the command line [View command line instructions](#)

Add a comment

Add your comment here...

Markdown is supported Paste, drop, or click to add files

Close pull request Comment

Owner ...

Reviewers No reviews Still in progress? Convert to draft

Assignees No one—assign yourself

Labels None yet

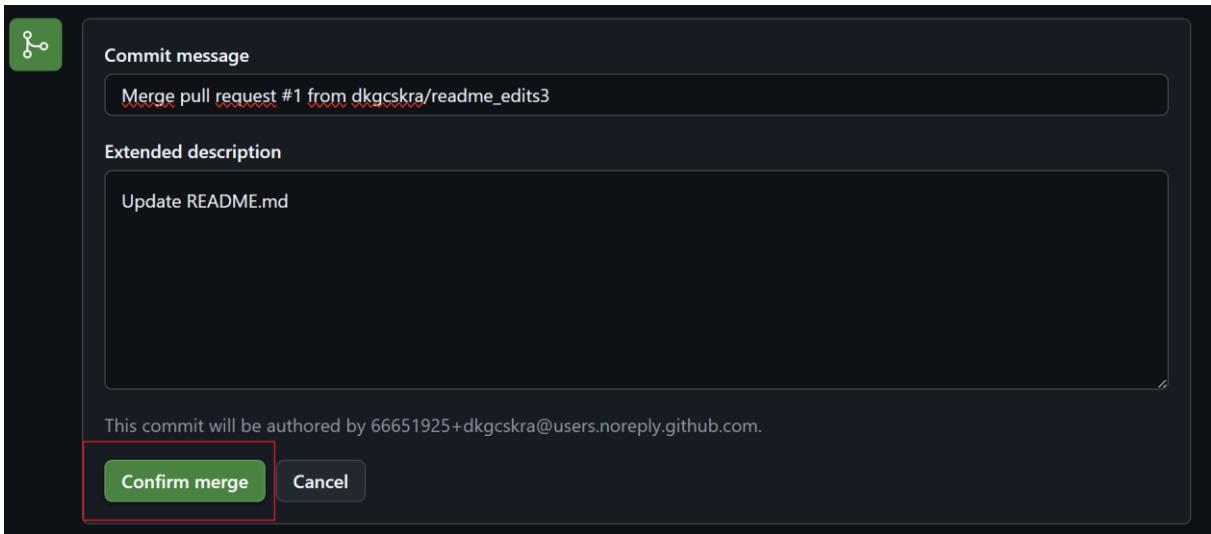
Projects None yet

Milestone No milestone

Development Successfully merging this pull request may close these issues.

None yet

Notifications Unsubscribe Customize



When creating a new branch `readme_edits3` and pushing the changes the main branch has now the same information as the branch `readme_edits3` and `Test_2` is the old branch

A screenshot of the GitHub "Branches" page. At the top right is a green "New branch" button. The page shows three sections: "Default", "Your branches", and "Active branches".

- Default:** Shows the "main" branch with an "Updated" status of "now", "Check status" as "Default", and "Behind/Ahead" status of "0".
- Your branches:** Shows two branches:
  - "Test\_2" was updated 2 minutes ago, is behind by 1 commit, and has 0 ahead commits. It has a pull request labeled "#2".
  - "readme\_edits3" was updated 22 minutes ago, is behind by 3 commits, and has 0 ahead commits. It has a pull request labeled "#1".
- Active branches:** Shows two branches:
  - "Test\_2" was updated 2 minutes ago, is behind by 1 commit, and has 0 ahead commits. It has a pull request labeled "#2".
  - "readme\_edits3" was updated 22 minutes ago, is behind by 3 commits, and has 0 ahead commits. It has a pull request labeled "#1".

A red arrow points from the text "readme\_edits3" in the "Your branches" section to the "readme\_edits3" entry in the "Active branches" section, indicating they are the same branch.

As we are done with the old readme-edits branch, we can delete it

The screenshot shows the GitHub repository page for 'git\_training3'. The repository has 3 branches and 0 tags. The 'main' branch is selected. The README file contains the text 'This is just a Test This is just a TEst 2'. The repository has 5 commits and 0 forks.

**Branches**

Branch	Updated	Check status	Behind	Ahead	Pull request
main	3 minutes ago	(Default)	0	0	...

**Your branches**

Branch	Updated	Check status	Behind	Ahead	Pull request
Test_2	5 minutes ago	...	1	0	...
readme_edits3	25 minutes ago	...	3	0	...

**Active branches**

Branch	Updated	Check status	Behind	Ahead	Pull request
Test_2	5 minutes ago	...	1	0	...
readme_edits3	25 minutes ago	...	3	0	...

The screenshot shows the GitHub repository page for 'git\_training3'. The repository now has 0 branches and 0 tags. The README file still contains the text 'This is just a Test This is just a TEst 2'. The repository has 5 commits and 0 forks.

**Branches**

Branch	Updated	Check status	Behind	Ahead	Pull request
main	4 minutes ago	(Default)	0	0	...

The update are now on the main branch as we pushed it before

## Resolving Conflicts

Same lines edited differently

- Two people edit the same line in the same file on different branches.
- Git can't automatically decide which change to keep, so it flags a conflict.

File edited vs. file deleted

- One person edits a file while another deletes it in their branch.
- Git can't guess what you want, so it stops for manual resolution.

Other scenarios

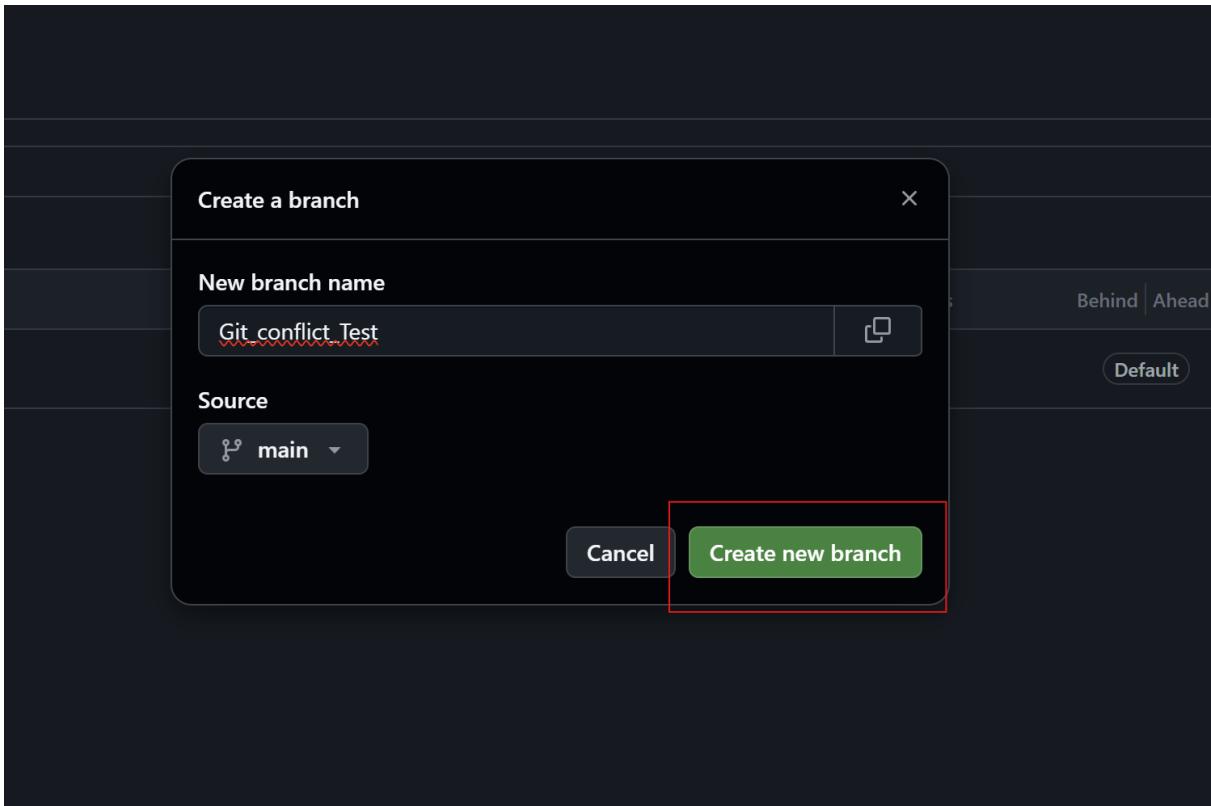
- Renaming a file in one branch while editing it in another.
- Moving a file to a different folder while someone else edits it.

## Conflict example

### Create a new branch

The screenshot shows a GitHub repository page for 'git\_training3'. The repository has 17 commits and 1 branch. The 'About' section indicates no description, website, or topics provided. The 'Branches' section shows a single branch named 'main'. A green button labeled 'New branch' is highlighted with a red box. The 'Default' branch table shows the 'main' branch was updated 1 minute ago.

Branch	Updated	Check status	Behind	Ahead	Pull request
main	1 minute ago	Default			



Edit the new branch:

A screenshot of a GitHub branches page. The page has a dark header with 'Branches' and a 'New branch' button. Below the header are tabs for 'Overview', 'Yours', 'Active', 'Stale', and 'All', with 'Yours' being active. A search bar is present. The main area is divided into three sections: 'Default', 'Your branches', and 'Active branches'. In the 'Your branches' section, there is a table with one row. The first column is 'Branch' with 'Git\_conflict\_Test' listed, which is highlighted with a red box. The second column is 'Updated' with 'now'. The third column is 'Check status' with a green circle icon. The fourth column is 'Behind | Ahead' with '0 | 0'. The fifth column is 'Pull request' with a small icon. The 'Active branches' section below it also shows a single row for 'Git\_conflict\_Test' with identical details. The 'Default' section at the top shows a single row for 'main' with '2 minutes ago' updated, 'now' check status, '0 | 0' status, and a 'Default' pull request.

Branch	Updated	Check status	Behind   Ahead	Pull request
main	2 minutes ago	Default	0   0	...

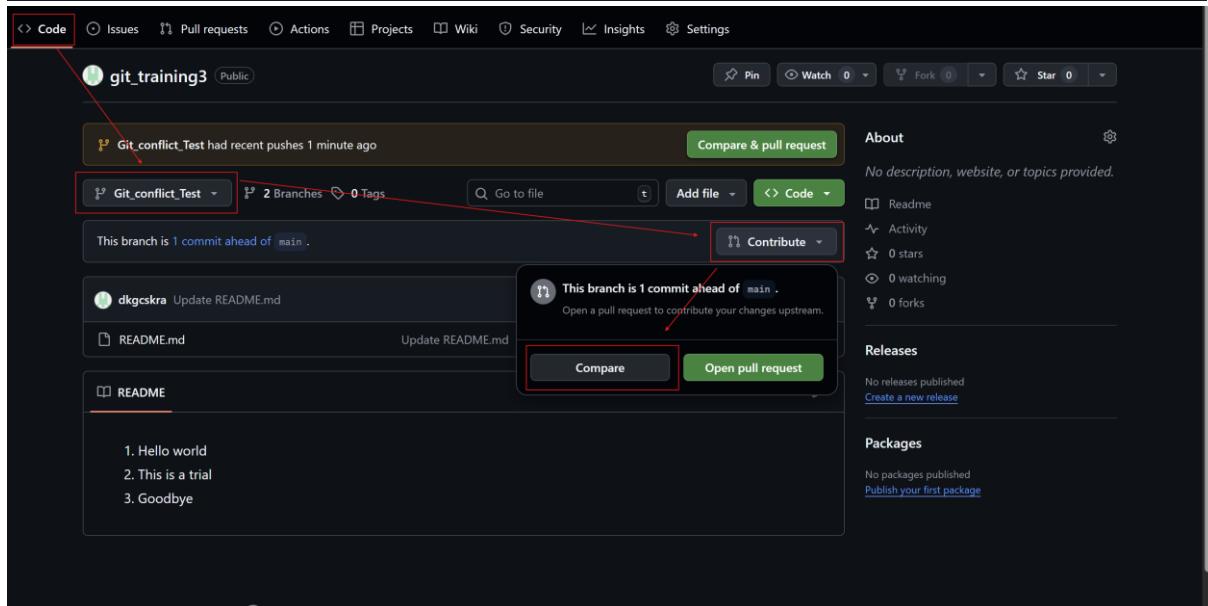
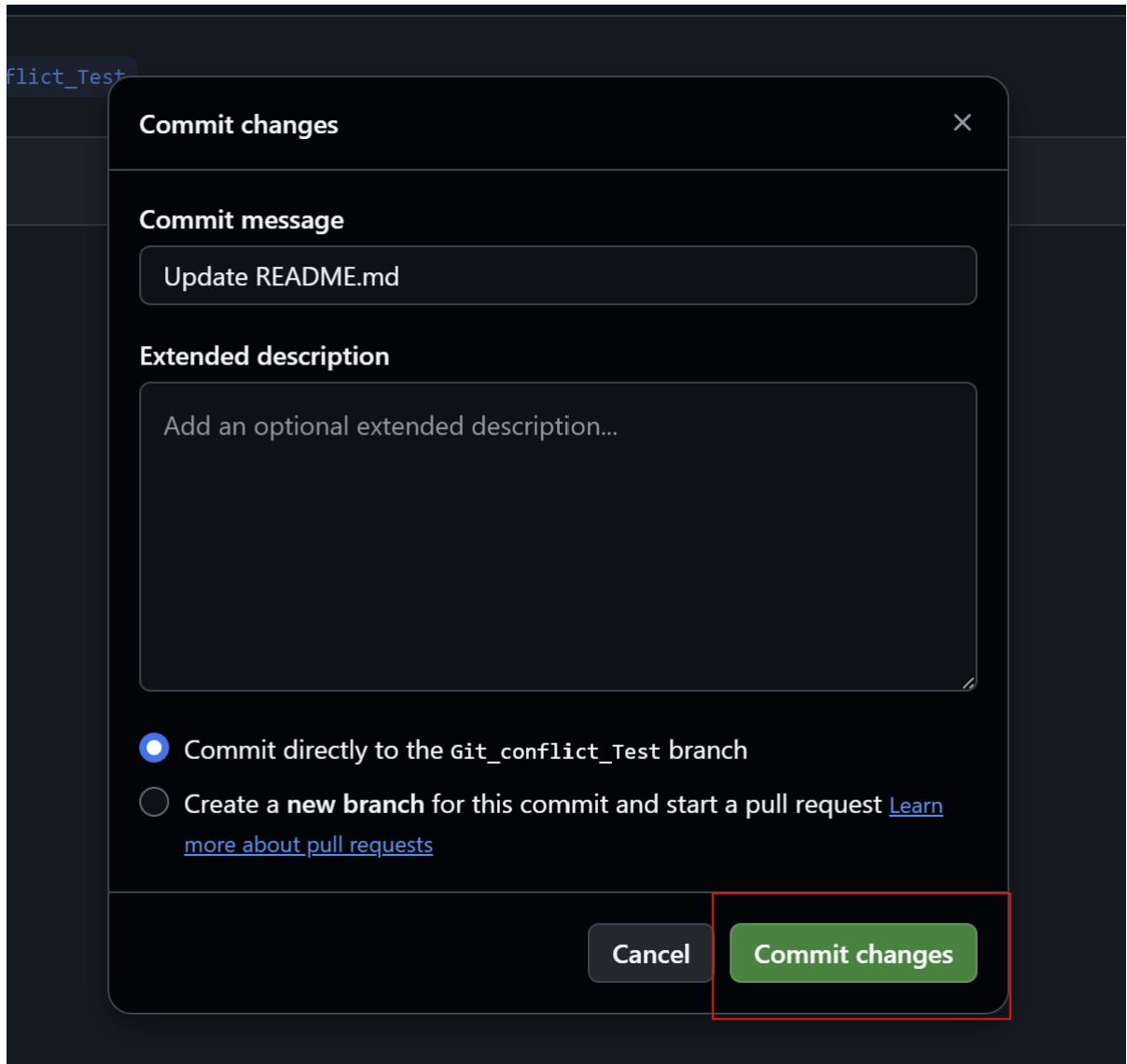
Branch	Updated	Check status	Behind   Ahead	Pull request
Git_conflict_Test	now	Green Circle	0   0	...

Branch	Updated	Check status	Behind   Ahead	Pull request
Git_conflict_Test	now	Green Circle	0   0	...

The screenshot shows a GitHub repository page for 'git\_training3'. The 'Code' tab is selected. The README file contains the following content:

```
1. Hello world
2. This is an experiment
3. Goodbye
```

A red box highlights the 'Commit changes...' button in the top right corner of the code editor.



## Comparing changes

Choose two branches to see what's changed or to start a new pull request. If you need to, you can also [compare across forks](#) or [learn more about diff comparisons](#).

The screenshot shows a GitHub diff comparison interface. At the top, there are dropdown menus for 'base: main' and 'compare: Git\_conflict\_Test'. A red box highlights the 'Create pull request' button. Below the header, a message says 'Discuss and review the changes in this comparison with others. [Learn about pull requests](#)'. The main area shows a summary: '-o 1 commit', '1 file changed', and '1 contributor'. It lists a single commit from 'dkgcsra' on Sep 19, 2025, titled 'Update README.md'. The commit details show a diff of the README.md file with three additions and one deletion. A red box highlights the diff view. The commit message includes the text: '1. Hello world', '2. This is an experiment', and '2. This is a trial'. The commit footer shows 'Verified' and a link to '179677e'. At the bottom, there are 'Split' and 'Unified' options.

## Make another branch

The screenshot shows the GitHub 'Branches' page. At the top, there are tabs for 'Overview', 'Yours', 'Active', 'Stale', and 'All', with 'Yours' selected. A green button labeled 'New branch' is visible. Below the tabs is a search bar with placeholder text 'Search branches...'. The main area is divided into sections: 'Default', 'Your branches', and 'Active branches'. The 'Default' section shows the 'main' branch, which is highlighted with a red box. A note says 'Your main branch isn't protected' with a 'Protect this branch' button. The 'Your branches' section shows two branches: 'Git\_conflict\_Test2' and 'Git\_conflict\_Test', both highlighted with red boxes. The 'Active branches' section shows the same two branches. Each branch row includes columns for 'Branch', 'Updated', 'Check status', 'Behind/Ahead', and 'Pull request'.

This branch is 1 commit ahead of `main`.

**dkgcskra** Update README.md

`README.md` Update README.md

**README**

1. Hello world
2. This is a test Showing with 18 additions and 15 deletions.
3. Goodbye
4. Hello world
5. This is a test Showing with 18 additions and 15 deletions.
6. Goodbye
7. i. Hello world
8. This is a test Showing with 18 additions and 15 deletions.
9. Goodbye
10. i. Hello world

Create pull request from one branch (Git\_conflict\_Test2) to the other (Git\_conflict\_Test)

Comparing changes

Choose two branches to see what's changed or to start a new pull request. If you need to, you can also [compare across forks](#) or [learn more about diff comparisons](#).

base: `Git_conflict_Test` compare: `Git_conflict_Test2`

**Update README.md #10**  
No description available **View pull request**

**Update README.md #9**  
No description available **View pull request**

Create another pull request to discuss and review the changes again. [Learn about pull requests](#) **Create pull request**

→ 1 commit **1 file changed** ↗ 1 contributor

Commits on Sep 19, 2025

**Update README.md**  
dkgcskra authored 3 minutes ago **Verified** **f41c97e** **Split** **Unified**

Showing 1 changed file with 3 additions and 3 deletions.

## Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also compare across forks. [Learn more about diff comparisons here.](#)

The screenshot shows the GitHub interface for creating a pull request. At the top, there are two comparison cards for 'README.md' changes between 'Git\_conflict\_Test' and 'Git\_conflict\_Test2'. Each card has a 'View pull request' button. Below the cards, there's a section to 'Add a title' with a placeholder 'Update README.md' and a 'View pull request' button. A rich text editor is available for 'Add a description' with a 'Create pull request' button highlighted with a red box. To the right, there are sections for 'Reviewers', 'Assignees', 'Labels', 'Projects', 'Milestone', 'Development' (with a note about closing keywords), and 'Helpful resources'. At the bottom, a note says 'Remember, contributions to this repository should follow our [GitHub Community Guidelines](#)'.

## Resolve conflict manually

### Update README.md #11

[Open](#) dkgcska wants to merge 1 commit into `Git_conflict_Test` from `Git_conflict_Test2`

The screenshot shows a GitHub pull request page for 'Update README.md' (PR #11). It displays a message from 'dkgcska' stating 'No description provided.' and a warning that 'This branch has conflicts that must be resolved'. A 'Resolve conflicts' button is highlighted with a red box. The pull request summary includes metrics: +3 -3. The right sidebar contains sections for 'Reviewers', 'Assignees', 'Labels', 'Projects', 'Milestone', 'Development' (with a note about merging), and 'Notifications' (with an 'Unsubscribe' button). A note at the bottom says 'You're receiving notifications because you authored the thread.'

Update README.md #11

Resolving conflicts between `Git_conflict_Test2` and `Git_conflict_Test` and committing changes → `Git_conflict_Test2`

1 conflicting file

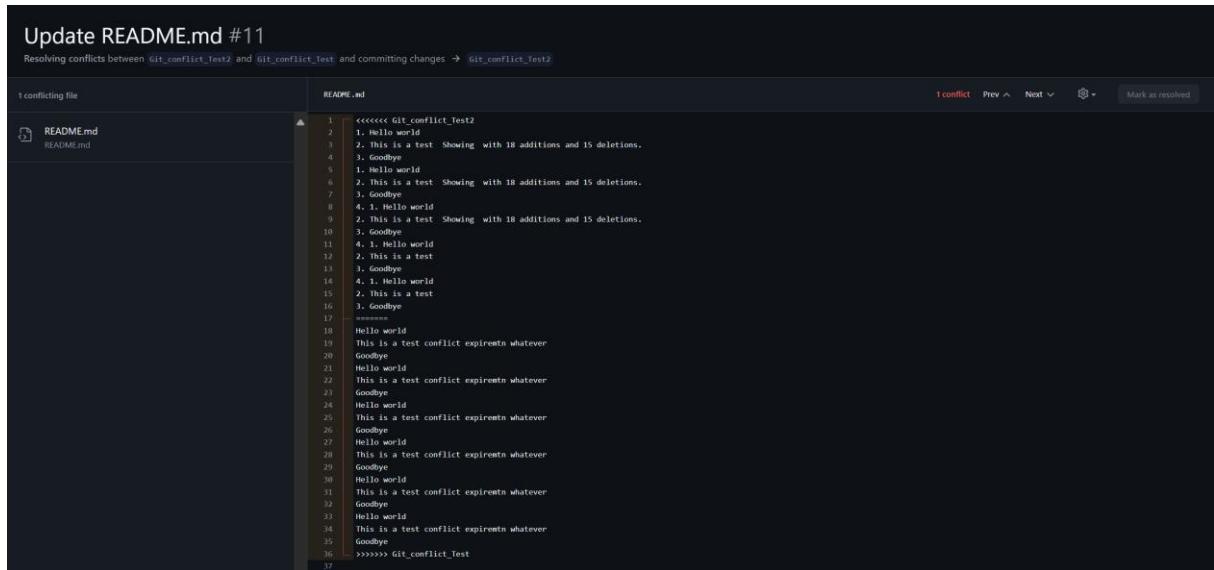
README.md

1    <<<< Git\_conflict\_Test2  
2    1. Hello world  
3    2. This is a test Showing with 18 additions and 15 deletions.  
4    3. Goodbye  
5    1. Hello world  
6    2. This is a test Showing with 18 additions and 15 deletions.  
7    3. Goodbye  
8    4. 1. Hello world  
9    2. This is a test Showing with 18 additions and 15 deletions.  
10   3. Goodbye  
11   4. 1. Hello world  
12   2. This is a test  
13   3. Goodbye  
14   4. 1. Hello world  
15   2. This is a test  
16   3. Goodbye  
17   ======  
18   Hello world  
19   This is a test conflict expirents whatever  
20   Goodbye  
21   Hello world  
22   This is a test conflict expirents whatever  
23   Goodbye  
24   Hello world  
25   This is a test conflict expirents whatever  
26   Goodbye  
27   Hello world  
28   This is a test conflict expirents whatever  
29   Goodbye  
30   Hello world  
31   This is a test conflict expirents whatever  
32   Goodbye  
33   Hello world  
34   This is a test conflict expirents whatever  
35   Goodbye  
36   >>>> Git\_conflict\_Test  
37

1 conflict

Prev ⌂ Next ⌂

Mark as resolved



Conflict because the same line has different values

→ Resolve the conflict manually

Update README.md #11

Resolving conflicts between `Git_conflict_Test2` and `Git_conflict_Test` and committing changes → `Git_conflict_Test2`

1 conflicting file

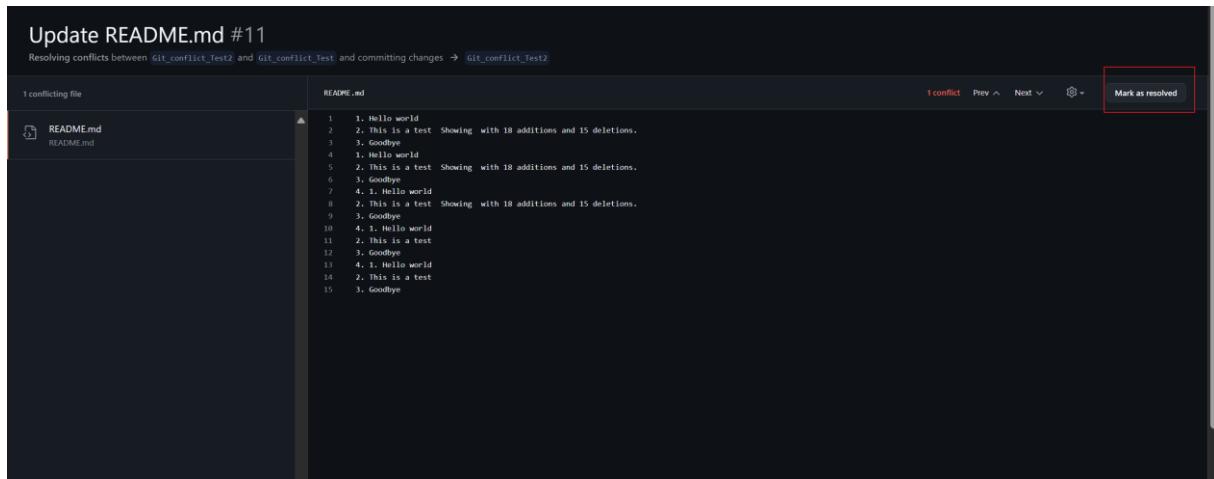
README.md

1    1. Hello world  
2    2. This is a test Showing with 18 additions and 15 deletions.  
3    3. Goodbye  
4    1. Hello world  
5    2. This is a test Showing with 18 additions and 15 deletions.  
6    3. Goodbye  
7    4. 1. Hello world  
8    2. This is a test Showing with 18 additions and 15 deletions.  
9    3. Goodbye  
10   4. 1. Hello world  
11   2. This is a test  
12   3. Goodbye  
13   4. 1. Hello world  
14   2. This is a test  
15   3. Goodbye

1 conflict

Prev ⌂ Next ⌂

Mark as resolved



Update README.md #11

Resolving conflicts between `Git_conflict_Test2` and `Git_conflict_Test` and committing changes → `Git_conflict_Test2`

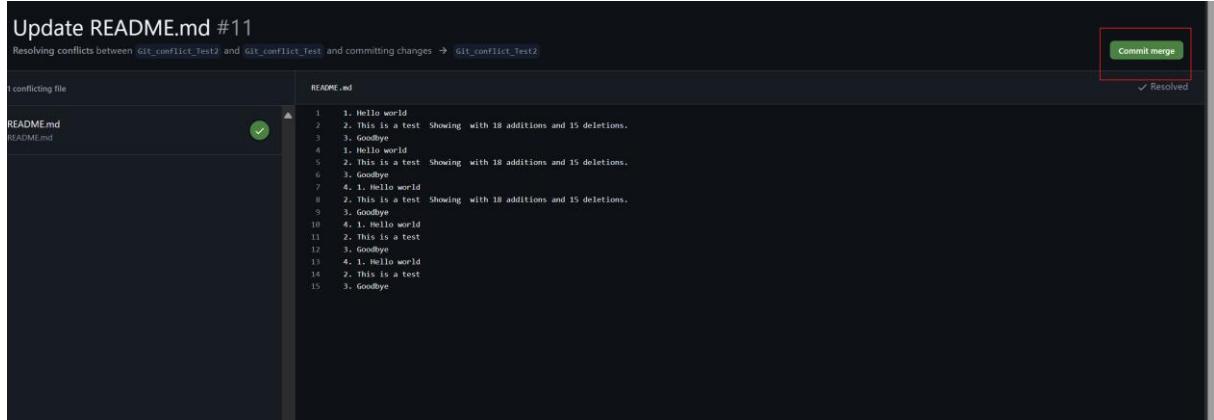
1 conflicting file

README.md

1    1. Hello world  
2    2. This is a test Showing with 18 additions and 15 deletions.  
3    3. Goodbye  
4    1. Hello world  
5    2. This is a test Showing with 18 additions and 15 deletions.  
6    3. Goodbye  
7    4. 1. Hello world  
8    2. This is a test Showing with 18 additions and 15 deletions.  
9    3. Goodbye  
10   4. 1. Hello world  
11   2. This is a test  
12   3. Goodbye  
13   4. 1. Hello world  
14   2. This is a test  
15   3. Goodbye

Commit merge

✓ Resolved



**Update README.md #11**

[Edit](#) [Code](#)

[Open](#) dkgskra wants to merge 1 commit into `Git_conflict_Test` from `Git_conflict_Test2`

Conversation 0 · Commits 1 · Checks 0 · Files changed 1

**dkgskra** commented 2 minutes ago

No description provided.

Update README.md

Verified f41c97e

**No conflicts with base branch**  
Merging can be performed automatically.

Merge pull request ▾ You can also merge this with the command line. [View command line instructions](#).

Add a comment

Write Preview

Add your comment here...

Markdown is supported Paste, drop, or click to add files

[Close pull request](#) [Comment](#)

Remember, contributions to this repository should follow our [GitHub Community Guidelines](#).

ProTip! Add `.patch` or `.diff` to the end of URLs for Git's plaintext views.

Reviewers  
No reviews  
Still in progress? Convert to draft

Assignees  
No one—assign yourself

Labels  
None yet

Projects  
None yet

Milestone  
None

Development  
Successfully merging this pull request may close these issues.  
None yet

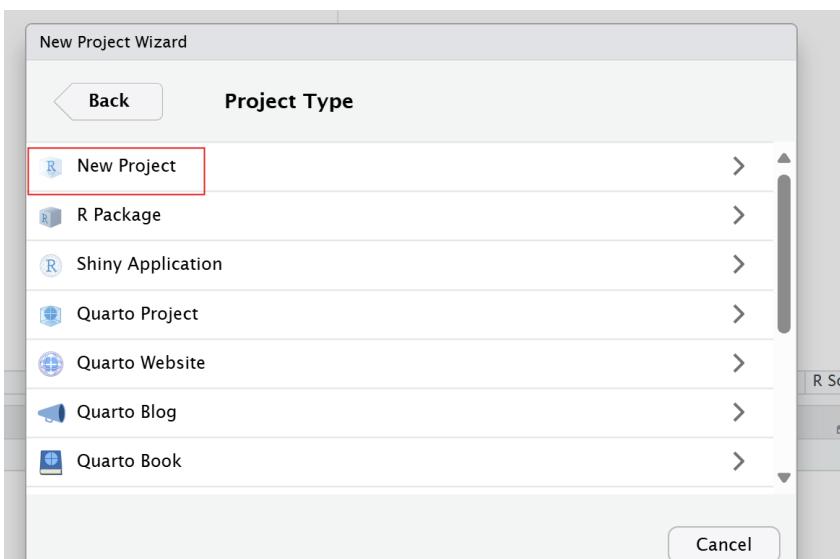
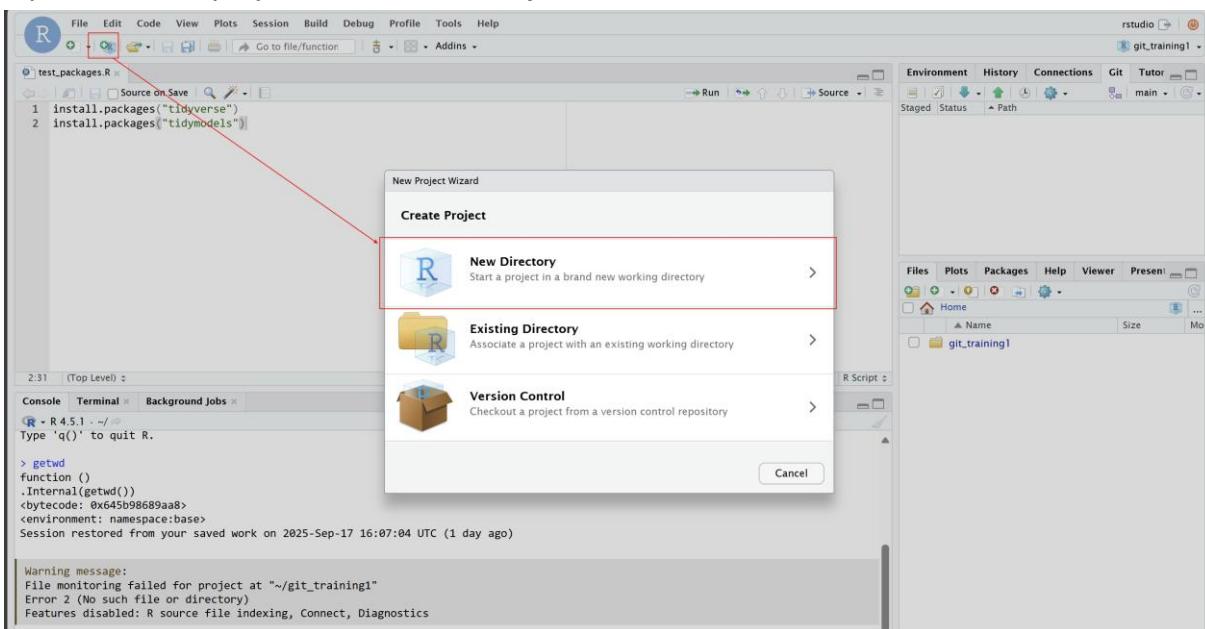
Notifications [Customize](#)

[Unsubscribe](#)  
You're receiving notifications because you authored the thread.

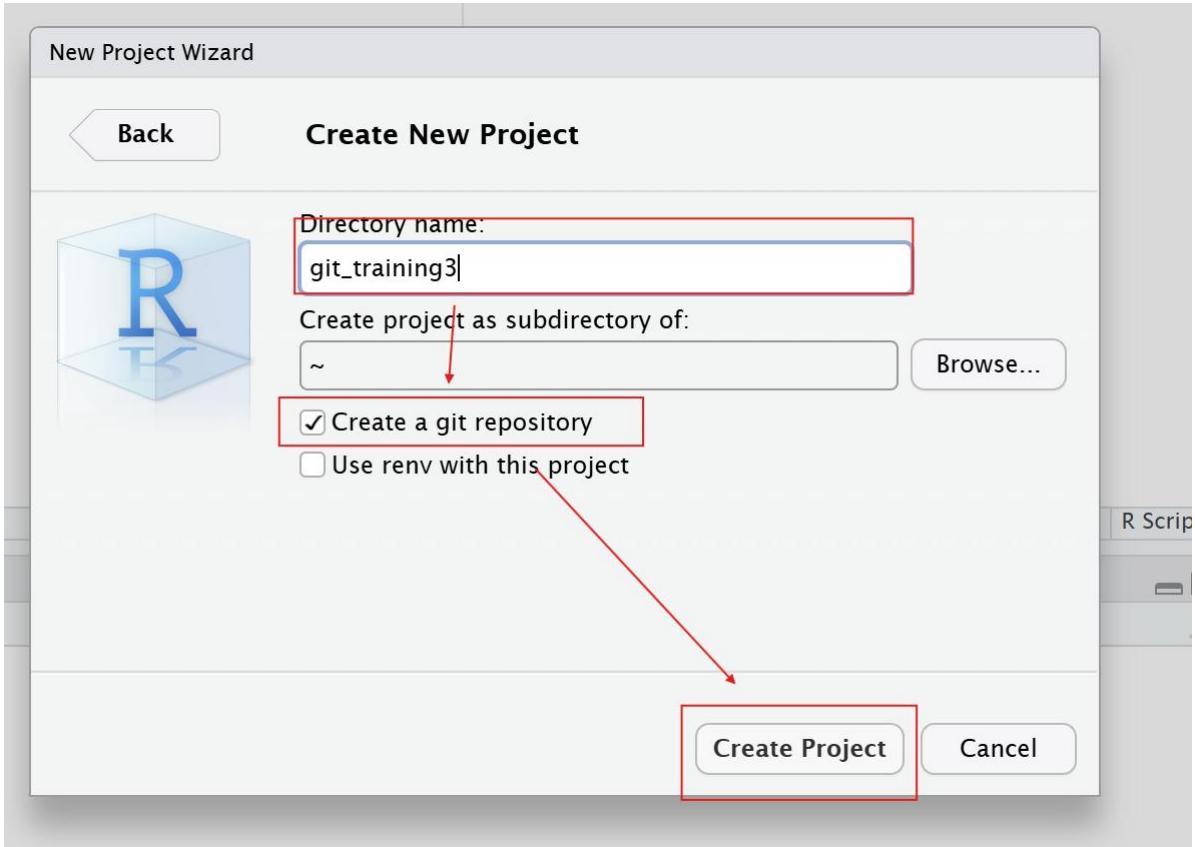
1 participant

## Git on R studio

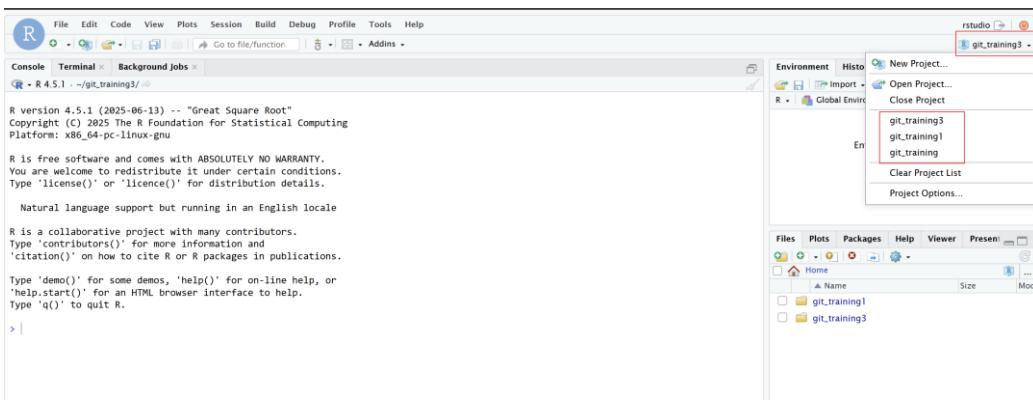
Open R -> new project -> new Directory



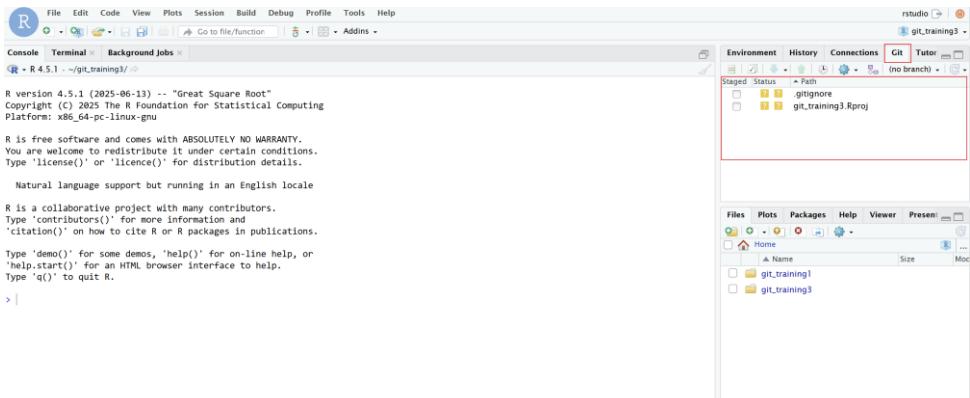
## Select Create a git repository



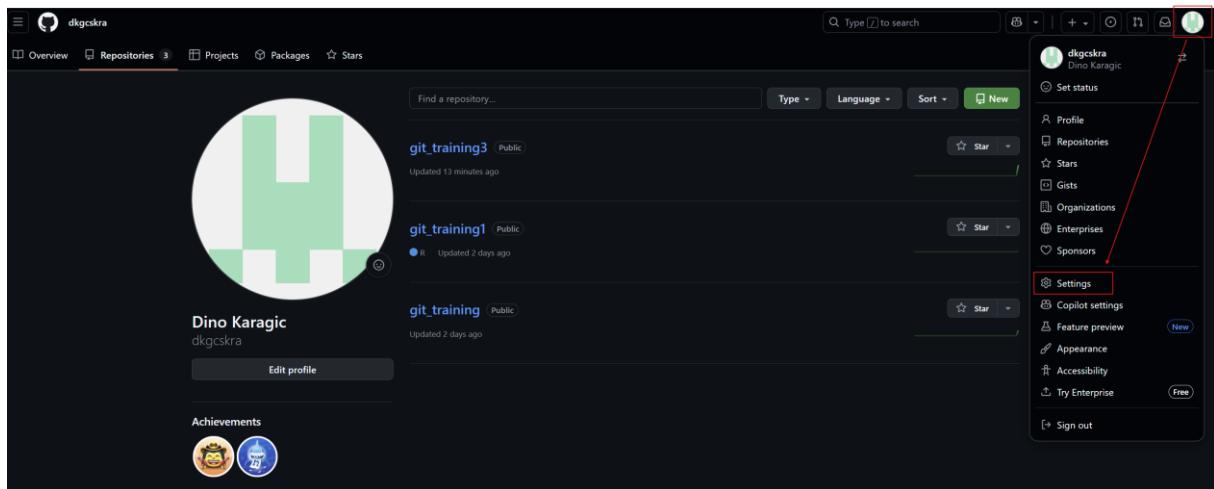
## To switch between projects



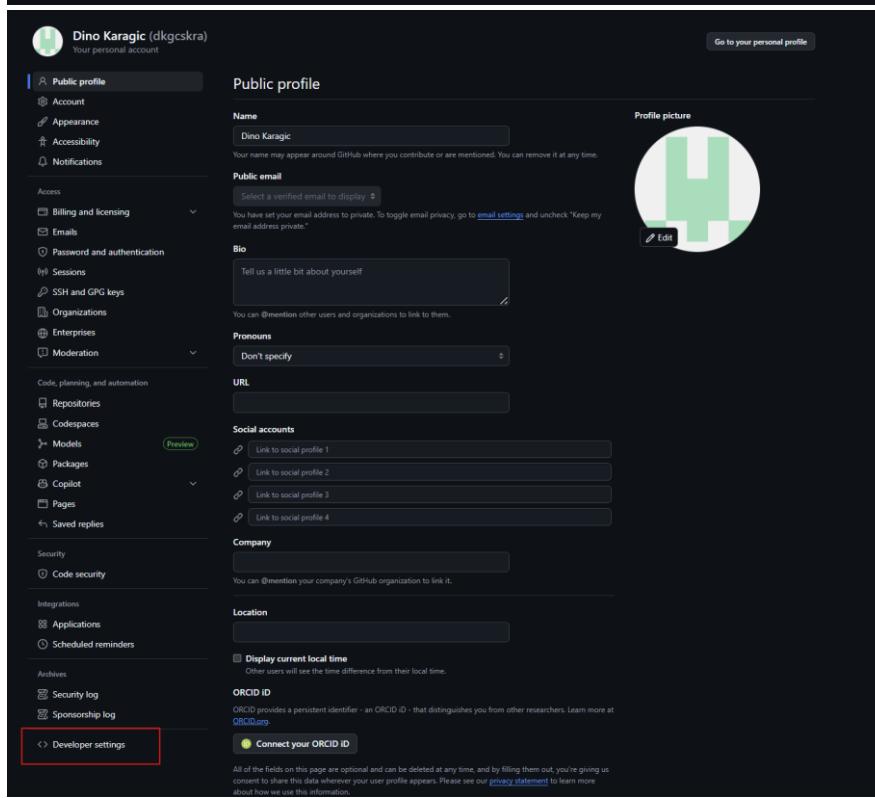
## Visual interface for git in R



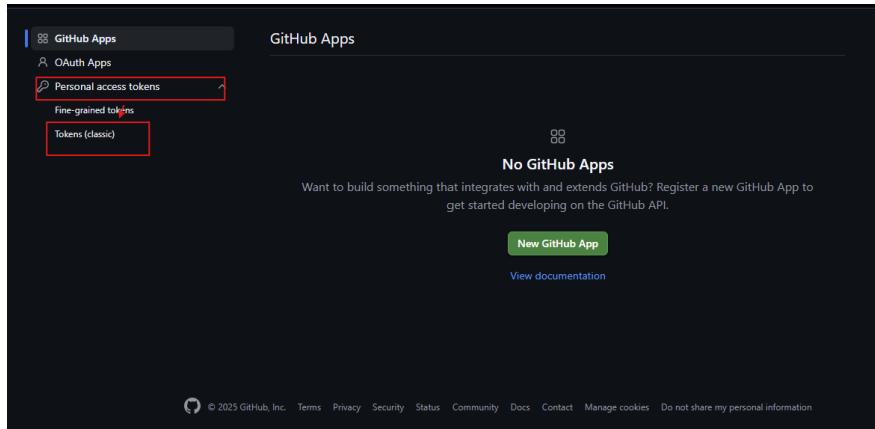
## To authenticate Communication we need to create a Personal Access Token (PAT) on GIT



The screenshot shows the GitHub profile page for user 'dkgcskra'. The 'Settings' menu is open on the right side, with the 'Personal access tokens' option highlighted by a red box.



The screenshot shows the 'Public profile' settings page. The 'Personal access tokens' section is visible on the left sidebar under the 'Access' category. A red box highlights the 'Developer settings' link at the bottom of the sidebar.



The screenshot shows the 'GitHub Apps' page. The 'Personal access tokens' section is highlighted by a red box. A red arrow points from the 'Personal access tokens' link on the GitHub profile page to this section. The 'Tokens (classic)' button is also highlighted by a red box.

The screenshot shows the GitHub 'Personal access tokens (classic)' page. On the left, there's a sidebar with 'GitHub Apps', 'OAuth Apps', and 'Personal access tokens' (which is expanded). Under 'Personal access tokens', there are 'Fine-grained tokens' and 'Tokens (classic)'. The main area displays two tokens:

- R-Test1** — *notifications, project, repo, write:discussion*  
Expires on Fri, Oct 17 2025.
- R-Test** — *public access*  
⚠️ This token has no expiration date.  
Never used Delete

A note at the bottom states: "Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to [authenticate to the API over Basic Authentication](#)".

At the top right, there are two buttons: 'Generate new token' and 'Generate new token (classic)'. A red arrow points from the 'Generate new token (classic)' button to the 'Generate new token' button.

Select scope (what you are allowed to do when using the token): repo, notifications, write:discussion, project

The screenshot shows the 'Select scopes' dialog. It lists various GitHub API scopes with their descriptions. Scopes are grouped into sections:

- repo**:
  - repo**: Full control of private repositories
  - repostatus**: Access commit status
  - repo\_deployment**: Access deployment status
  - public\_repo**: Access public repositories
  - repoinvite**: Access repository invitations
  - security\_events**: Read and write security events
- workflow**:
  - workflow**: Update GitHub Action workflows
- write:packages**:
  - readpackages**: Upload packages to GitHub Package Registry
  - deletepackages**: Download packages from GitHub Package Registry
- admin:org**:
  - writeorg**: Delete packages from GitHub Package Registry
  - readorg**: Full control of orgs and teams, read and write org projects
  - manage\_runner:org**: Read and write org and team membership, read and write org projects
- admin:public\_key**:
  - writepublic\_key**: Read org and team membership, read org projects
  - readpublic\_key**: Manage org runners and runner groups
- admin:repo\_hook**:
  - writerepo\_hook**: Full control of repository hooks
  - readrepo\_hook**: Write repository hooks
- admin:org\_hook**:
  - adminorg\_hook**: Read repository hooks
- gist**:
  - notifications**: Create gists
- user**:
  - notifications**: Access notifications
  - readuser**: Update ALL user data
  - useremail**: Read ALL user profile data
  - userfollow**: Access user email addresses (read-only)
  - userfollow**: Follow and unfollow users
- delete\_repo**:
  - delete\_repo**: Delete repositories
- write:discussion**:
  - read:discussion**: Read and write team discussions
  - read:discussion**: Read team discussions
- admin:enterprise**:
  - manage\_runner:enterprise**: Full control of enterprises
  - manage\_billing:enterprise**: Manage enterprise runners and runner groups
  - readenterprise**: Read and write enterprise billing data
  - scim:enterprise**: Read enterprise profile data
  - provisioning:enterprise**: Provisioning of users and groups via SCIM
- audit\_log**:
  - readaudit\_log**: Full control of audit log
  - readaudit\_log**: Read access of audit log
- codespace**:
  - codespace\_secrets**: Full control of codespaces
  - codespace\_secrets**: Ability to create, read, update, and delete codespace secrets
- copilot**:
  - manage\_billing:copilot**: Full control of GitHub Copilot settings and seat assignments
  - manage\_billing:copilot**: View and edit Copilot Business seat assignments
- write:network\_configurations**:
  - readnetwork\_configurations**: Write org hosted compute network configurations
  - readnetwork\_configurations**: Read org hosted compute network configurations
- project**:
  - readproject**: Full control of projects
  - readproject**: Read access of projects
- admin:gpg\_key**:
  - writegpg\_key**: Full control of public user GPG keys
  - readgpg\_key**: Write public user GPG keys
  - readgpg\_key**: Read public user GPG keys
- admin:ssh\_signing\_key**:
  - writessh\_signing\_key**: Full control of public user SSH signing keys
  - writessh\_signing\_key**: Write public user SSH signing keys
  - readssh\_signing\_key**: Read public user SSH signing keys

At the bottom left are 'Generate token' and 'Cancel' buttons. A red arrow points from the 'Generate token' button to the 'Generate new token' button in the previous screenshot.

Personal access tokens (classic)

[Generate new token ▾](#)

Tokens you have generated that can be used to access the [GitHub API](#).

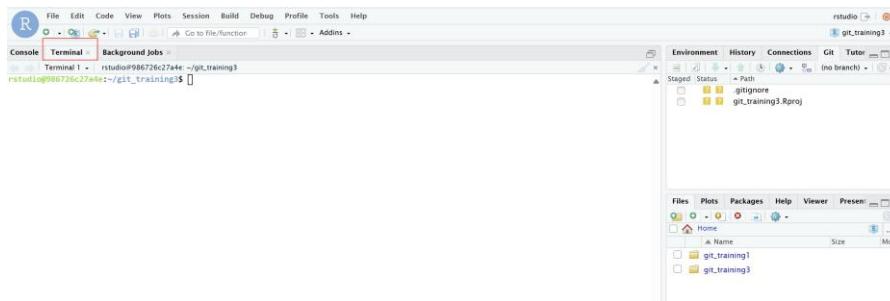
⚠ Make sure to copy your personal access token now. You won't be able to see it again!

✓ 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 <a href="#">Copy</a>	<a href="#">Delete</a>
R_Test1 — notifications, project, repo, write:discussion Expires on Fri, Oct 17 2025.	Never used <a href="#">Delete</a>
R_Test — public access ⚠ This token has no expiration date.	Never used <a href="#">Delete</a>

Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to [authenticate to the API over Basic Authentication](#).

Save the token as it will be invincible after

In R studio, you can access the terminal of your docker container. This terminal functions largely as your host terminal but is restricted to operating within the container. By default, you are logged in as the rstudio user and, therefore do not have full root privileges



We can configure git with our user information from the terminal by running a set of git commands

- `git config --global user.name [username_from_github]`
  - Add your github username as the global Git username in your container
- `git config --global user.email [email_from_github]`
  - Add the Email associated with your github account as the global Git Email in your container
- `git config --global credential.helper store`
  - Stores credentials in plain text. From your Rstudio file pane the stored credentials can be found at "Home/.gitconfig" (it is a hidden file)
  - This approach is for convenience in this course. Both the PAT, your email, and your username are stored in the .git-credentials file, which is again stored in a shared volume. Hence, you do not need to retype this information as you stop and start the container. However, for security reasons, it is likely not the approach you will use in a company
- `git config --list`
  - Check your Git settings

This only needs to be done once the first time  
You can check by running `git config --list`

Console Terminal x Background Jobs x

Terminal 1 | rstudio@986726c27a4e: ~/git\_training3

```
rstudio@986726c27a4e:~/git_training3$ git config ls
error: key does not contain a section: ls
rstudio@986726c27a4e:~/git_training3$ git config --list
credential.helper=cache --timeout=3600
push.default=simple
user.name=dkgcskra
user.email=dinokaragic@googlemail.com
credential.helper=store
core.repositoryformatversion=0
core.filemode=true
core.bare=false
core.logallrefupdates=true
rstudio@986726c27a4e:~/git_training3$ []
```

## Create and save R script in the project

The screenshot shows the RStudio interface. On the left, the code editor displays an R script named "Test for git training.R" with the following content:

```
1 install.packages("tidyverse")
2 install.packages("tidymodels")
3
```

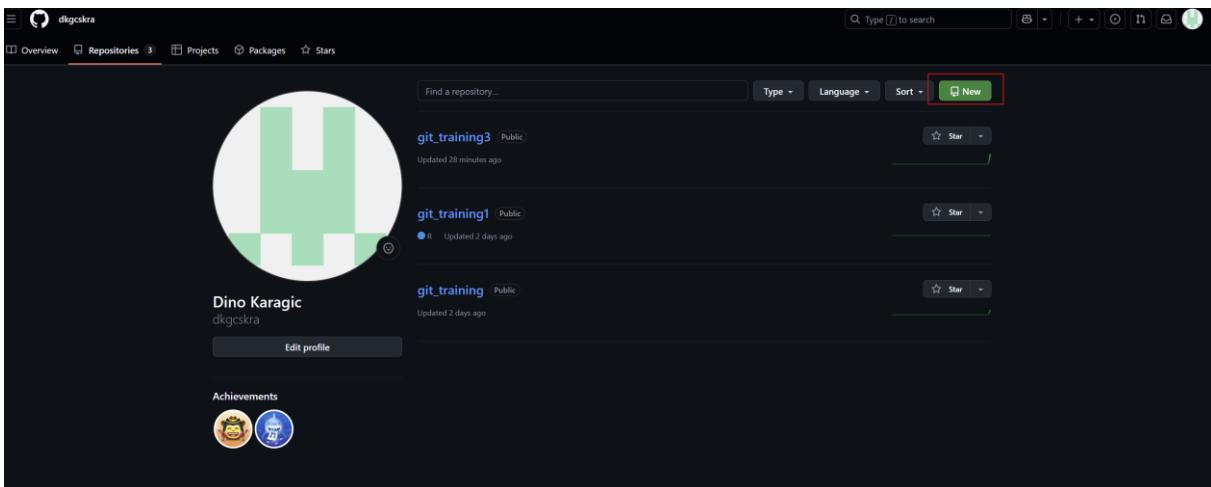
The status bar at the bottom indicates "R Script". On the right, the Project Explorer shows the directory structure of the project "git\_training3":

- Staged: .gitignore
- Status: git\_training3.Rproj
- Path: Test for git training.R
- git\_training3.Rproj

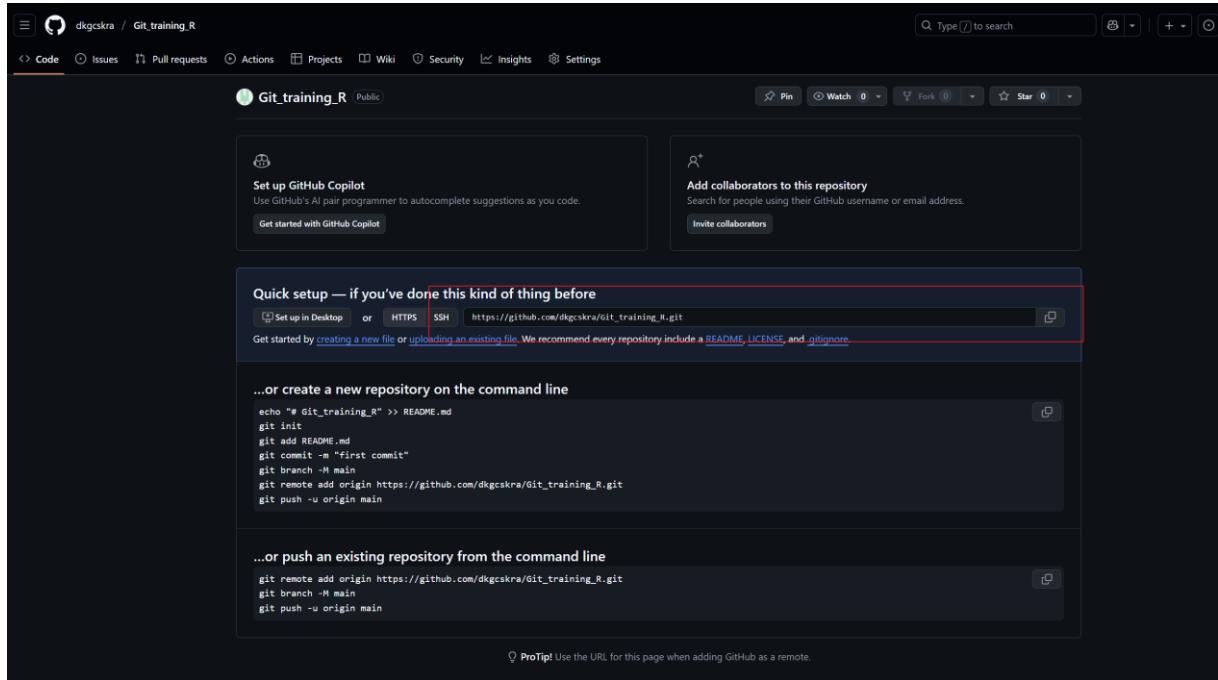
The "Files" tab in the Project Explorer shows the contents of the project:

Name	Size	Last Modified
.gitignore	40 B	Sep 20, 2025
git_training3.Rproj	205 B	Sep 20, 2025
Test for git training.R	61 B	Sep 20, 2025

## Create Repository on Git

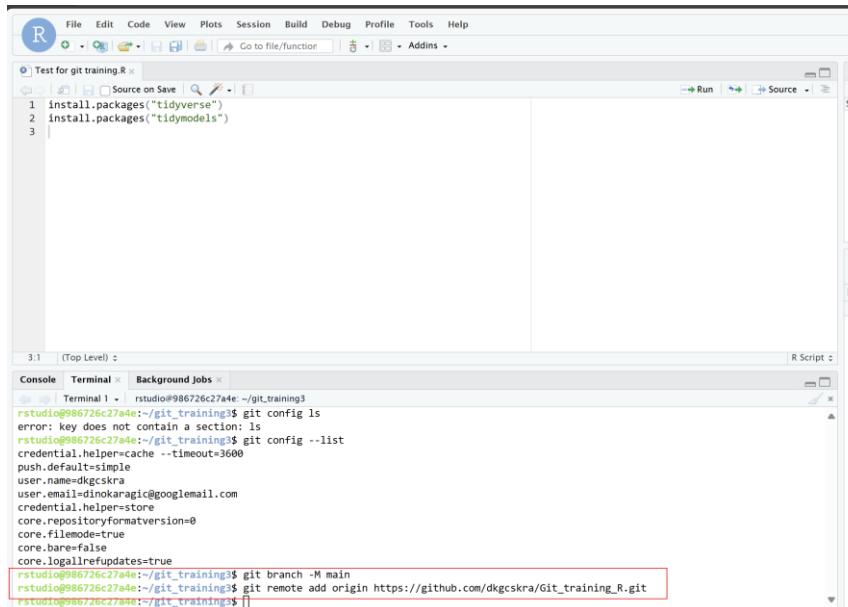


## Copy the URL

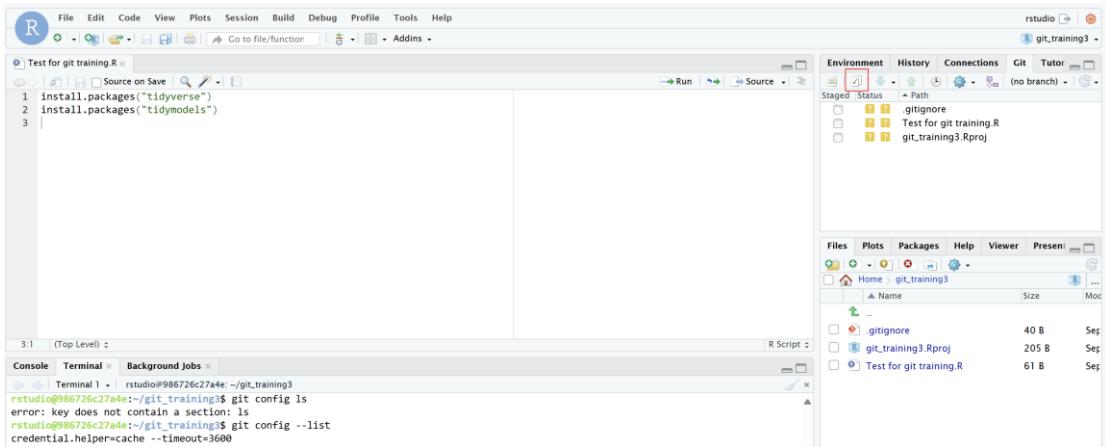


Back in the terminal accessed from Rstudio

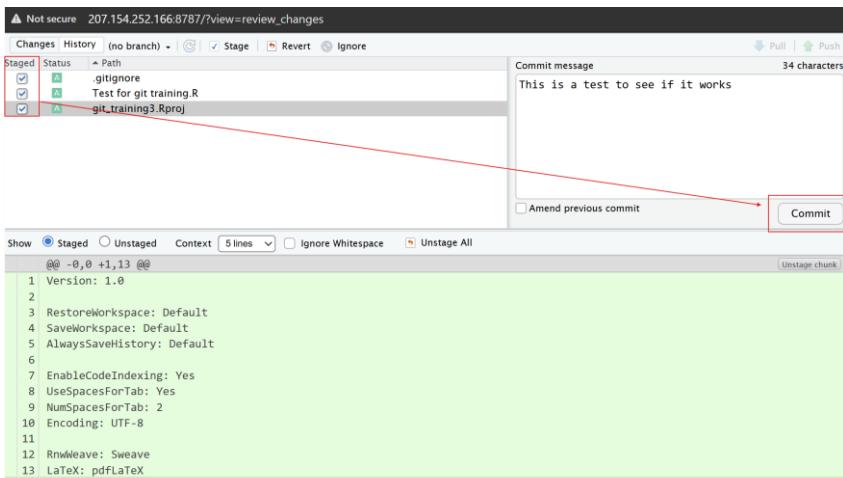
- Change our branch name from “master” to “main” to align with Github terminology  
git branch -M main
  - Connect our local repository to the remote:  
git remote add origin https://github.com/dmdvben/git\_training.git
  - Where “https://github.com/dmdvben/git\_training.git” is replaced with your own repository



## In the Rstudio IDE, choose Git and then Commit



Mark all files as "Staged" and write a small commit message indicating what you committed



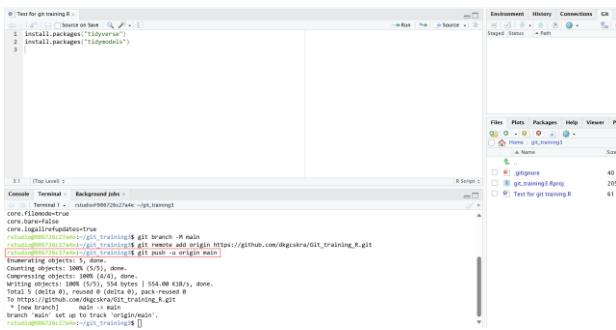
Committing changes will not change anything on Git only pushing it will update it in Git.

First time you push to the new repository, we do it in the container terminal from Rstudio:

```
git branch -M main
```

```
git push -u origin main
```

(if you do it the first time will ask you for mail, username and token you created above?)



It worked:

The screenshot shows the GitHub interface for the user 'dkgcsra'. The top navigation bar includes 'Overview', 'Repositories 4', 'Projects', 'Packages', and 'Stars'. A search bar at the top right says 'Type / to search'. Below the search bar is a button labeled 'New'. The main area displays four repositories: 'Git\_training\_R' (highlighted with a red border), 'git\_training3', 'git\_training1', and 'git\_training'. Each repository card includes a star icon, a 'Star' dropdown menu, and an 'Updated' timestamp. The 'Git\_training\_R' card also shows a green checkmark icon and the message 'Updated 1 minute ago'. On the left side, there is a profile picture of Dino Karagic and a 'Edit profile' button. Below the profile is a section titled 'Achievements' featuring two circular icons.

The 3 files we created are visible here now

The screenshot shows the detailed view of the 'Git\_training\_R' repository. At the top, it shows 'main' branch with 1 commit. Below the branches is a list of files: '.gitignore', 'Test for git training.R', and 'git\_training3.Rproj', each with a brief description and timestamp. To the right is the 'About' section, which notes 'No description, website, or topics provided.' and lists activity metrics: 0 stars, 0 watching, and 0 forks. Below the files is a 'README' section with a 'Add a README' button. The right sidebar contains sections for 'Releases' (no releases), 'Packages' (no packages), and 'Languages' (R 100.0%).

The screenshot shows the content of the 'Test for git training.R' file. It contains two lines of R code:

```
install.packages("tidyverse")
install.packages("tidymodels")
```

## Update Branch in Git through R:

### Change something and save the file

The screenshot shows the RStudio interface with several panes:

- File Menu:** A red box highlights the "Save" option under the "File" menu.
- Console:** Displays R command history and output related to cloning a GitHub repository and pushing changes.
- Code Editor:** Shows an R script named "Test for git training.R" with four lines of code: `install.packages("tidyverse")`, `install.packages("tidyML")`, `install.packages("tidymodels")`, and a closing brace `4`.
- Environment:** Shows the project structure: ".gitignore", "git\_training3.Rproj", and "Test for git training.R". The "Test for git training.R" file is selected.
- Files:** Shows the same project structure with file details like size and modification time.
- Terminal:** Displays the command line history for cloning the repository and pushing changes.
- Changes:** A detailed view of the current commit, showing the file "Test for git training.R" with the status "Staged". A red box highlights the "Commit" button.
- Code View:** Shows the diff of the staged changes, which includes the addition of the tidyverse package.

Committing the change will only change on the local repository in this case R it will not change on the git to change it on git press Push

The screenshot shows the RStudio interface with the following details:

- Environment Tab:** Shows the file structure: .gitignore and Conflict.R.
- History Tab:** Shows the commit history for the current branch.
- Console Tab:** Displays the command "git push -u origin main" and its output, indicating divergence between the local and remote branches.
- Terminal Tab:** Shows the command "git pull origin main" and its output.

git push -u origin main

The screenshot shows the GitHub repository page for 'dkgcska / Git\_training\_R' with the following details:

- Repository Overview:** Shows 2 commits, 1 branch, and 0 tags.
- Commits:** Lists three commits:
  - .gitignore: "This is a test to see if it works" (20 minutes ago)
  - Test for git training.R: "Update" (2 minutes ago)
  - git\_training3.Rproj: "This is a test to see if it works" (20 minutes ago)
- About Section:** Shows 0 stars, 0 forks, and 0 watching.
- Releases:** No releases published.
- Packages:** No packages published.
- Languages:** R 100.0%
- Code View:** Shows the code for 'Test for git training.R'.

## Branches in Git and R

### Create new branch in Git in the repository

Branches

New branch

Overview Yours Active Stale All

Search branches...

Default

Branch	Updated	Check status	Behind	Ahead	Pull request
main	10 minutes ago	Default			

Your branches

Branch	Updated	Check status	Behind	Ahead	Pull request
Test_conflictR_2	now		0	0	
Test_conflictR	1 minute ago		2	1	

Active branches

Branch	Updated	Check status	Behind	Ahead	Pull request
Test_conflictR_2	now		0	0	
Test_conflictR	1 minute ago		2	1	

### Get the branch into R by pulling

```
1 install.packages("tidyverse")
2 install.packages("tidyMLN")
3 install.packages("tidymodels")
4 #test2
```

Environment History Connections Git Tutor

Staged Status Path

Files Plots Packages Help Viewer Present

Local Branches

main  
Test\_conflictR (REMOTE ORIGIN)

main  
Test\_conflictR  
Test\_conflictR\_2

Terminal : rstanstudio@986726c274e: ~/git\_training3

```
1: Enumerating objects: 5, done.
2: Counting objects: 100% (5/5), done.
3: Compressing objects: 100% (3/3), done.
4: Writing objects: 100% (3/3), 374 bytes | 374.00 KIB/s, done.
5: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
6: To https://github.com/dkgcskra/git_training_R.git
7: 510f5c4..2403381 main -> main
8: branch 'main' set up to track 'origin/main'.
9: Everything up-to-date
10: rstanstudio@986726c274e: ~/git_training3$ git pull -u origin main
11: Branch 'main' set up to track 'origin/main'.
12: Your branch is up-to-date with 'origin/main'.
13: You have no unmerged commits.
14: Your branch is fully tracked by its remote.
15: You can merge or rebase your local changes onto the remote branch.
16: rstanstudio@986726c274e: ~/git_training3$ git push -u origin Test_conflictR
```

### Switch between the branches

```
1 install.packages("tidyverse")
2 install.packages("tidyMLN")
3 install.packages("tidymodels")
4 #test2
```

Environment History Connections Git Tutor

Staged Status Path

Files Plots Packages Help Viewer Present

Local Branches

main  
Test\_conflictR (REMOTE ORIGIN)

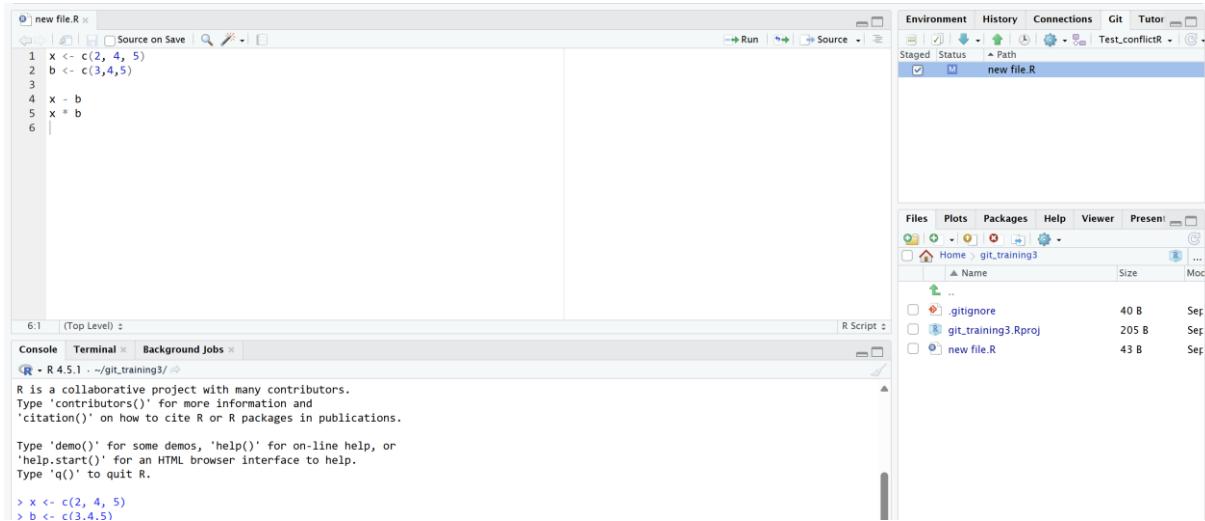
main  
Test\_conflictR  
Test\_conflictR\_2

Terminal : rstanstudio@986726c274e: ~/git\_training3

```
1: Enumerating objects: 5, done.
2: Counting objects: 100% (5/5), done.
3: Compressing objects: 100% (3/3), done.
4: Writing objects: 100% (3/3), 374 bytes | 374.00 KIB/s, done.
5: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
6: To https://github.com/dkgcskra/git_training_R.git
7: 510f5c4..2403381 main -> main
8: branch 'main' set up to track 'origin/main'.
9: Your branch is up-to-date with 'origin/main'.
10: You have no unmerged commits.
11: Your branch is fully tracked by its remote.
12: You can merge or rebase your local changes onto the remote branch.
13: rstanstudio@986726c274e: ~/git_training3$ git checkout Test_conflictR
14: Switched to branch 'Test_conflictR'
15: rstanstudio@986726c274e: ~/git_training3$
```

In files you see than all the files off the current branch

Change the lines of the file and save



The screenshot shows the RStudio interface. In the top-left, there's a code editor window titled "new file.R" with the following content:

```
1 x <- c(2, 4, 5)
2 b <- c(3,4,5)
3
4 x - b
5 x * b
6
```

Below the code editor is the R Console window, which displays the R environment and some initial code execution:

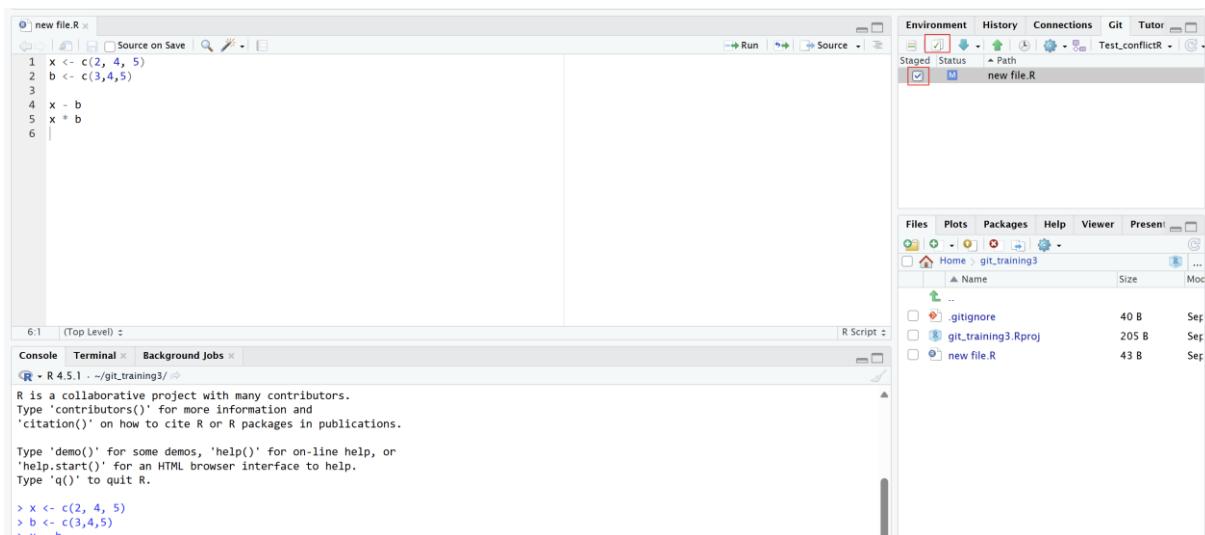
```
R > x <- c(2, 4, 5)
R > b <- c(3,4,5)
```

To the right of the code editor is the "Git" panel, which shows the status of the file "new file.R":

Staged	Status	Path
<input checked="" type="checkbox"/>	M	new file.R

At the bottom right is the "Files" browser, showing the contents of the "git\_training3" directory:

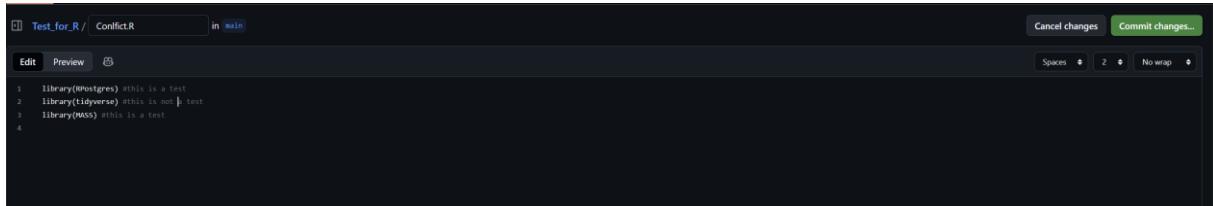
Name	Size	Last Modified
..		
.gitignore	40 B	Sep 10 10:45
git_training3.Rproj	205 B	Sep 10 10:45
new file.R	43 B	Sep 10 10:45



This screenshot is identical to the first one, except the "new file.R" code editor window is now closed. The rest of the interface (Console, Git panel, Files browser) remains the same.

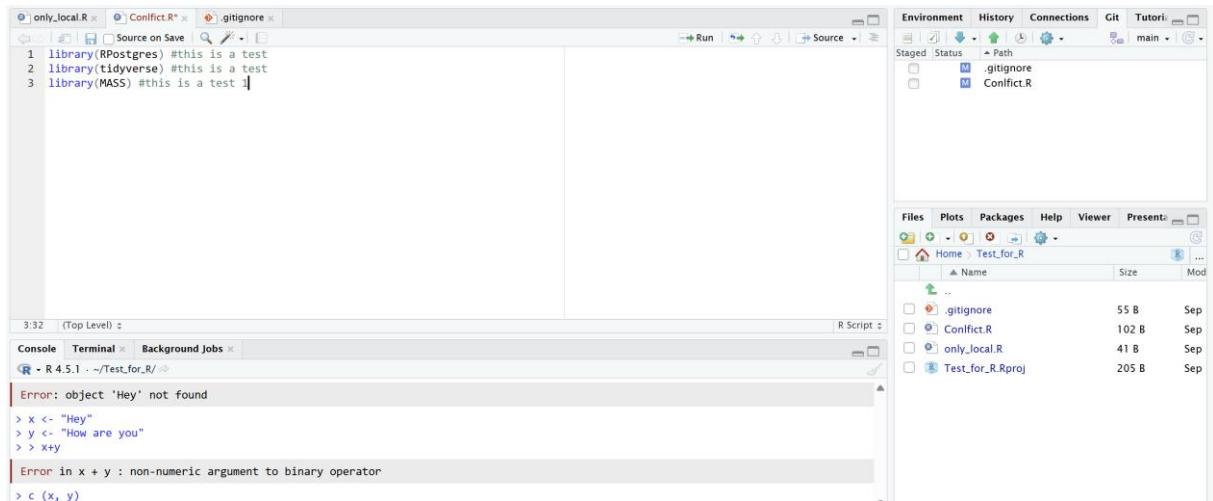
## Conflicts in R

Imagine someone edits in the main branch in git the file



```
1 library(RPostgres) #this is a test
2 library(tidyverse) #this is not a test
3 library(MASS) #this is a test
```

While we edit the file of the main branch in R

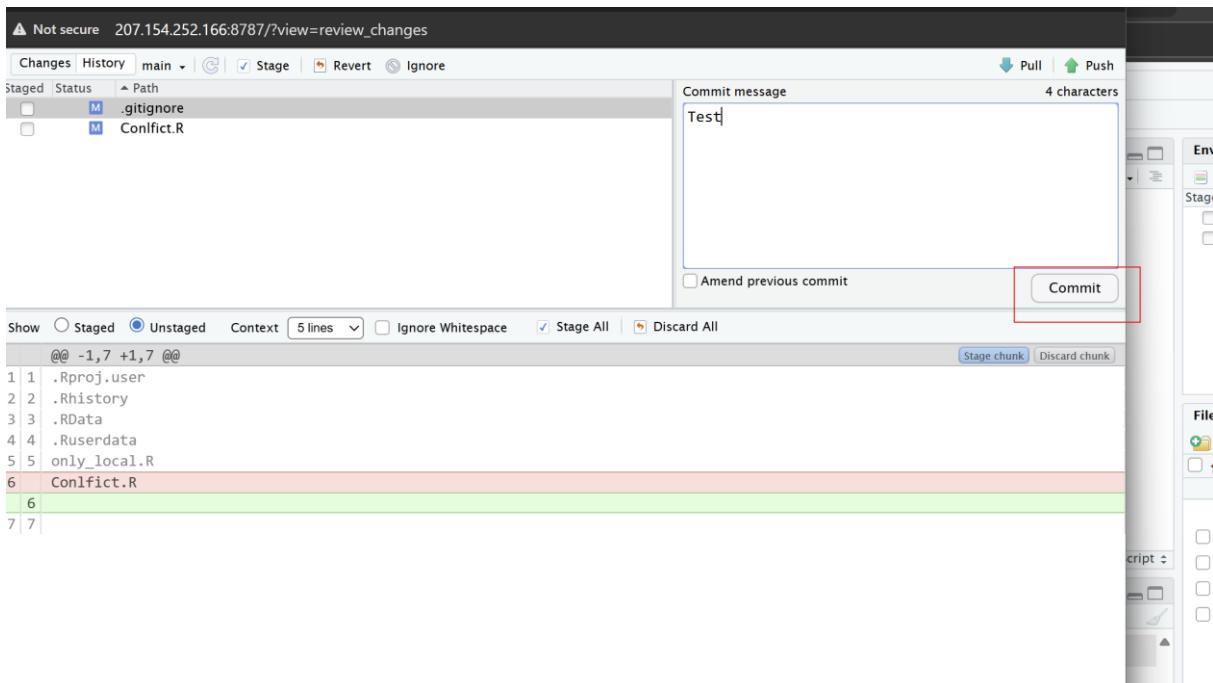


```
1 library(RPostgres) #this is a test
2 library(tidyverse) #this is a test
3 library(MASS) #this is a test
```

Error: object 'Hey' not found  
> x <- "Hey"  
> y <- "How are you"  
> x+y  
Error in x + y : non-numeric argument to binary operator  
> c(x, y)

Name	Size	Mod
.gitignore	55 B	Sep
Conflict.R	102 B	Sep
only_local.R	41 B	Sep
Test_for_R.Rproj	205 B	Sep

If we commit and try to push



Not secure 207.154.252.166:8787/?view=review\_changes

Changes History main ▾ Stage Revert Ignore

Staged Status Path

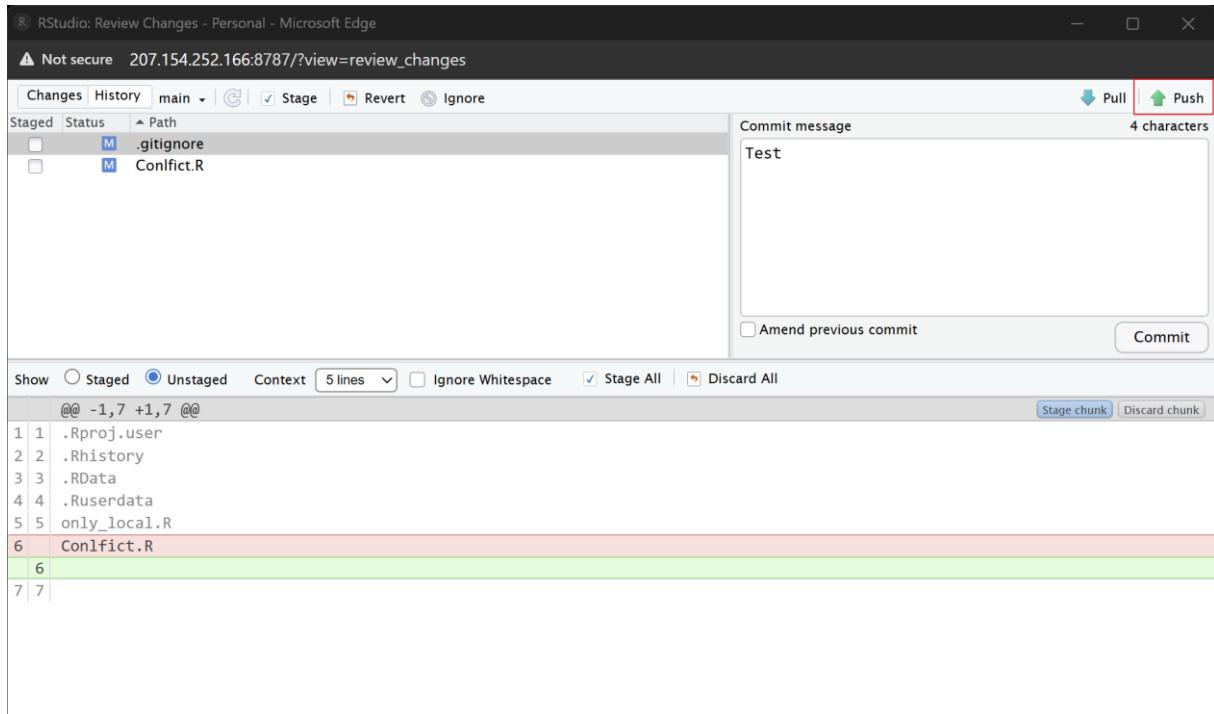
.gitignore Conflict.R

Commit message 4 characters  
Test

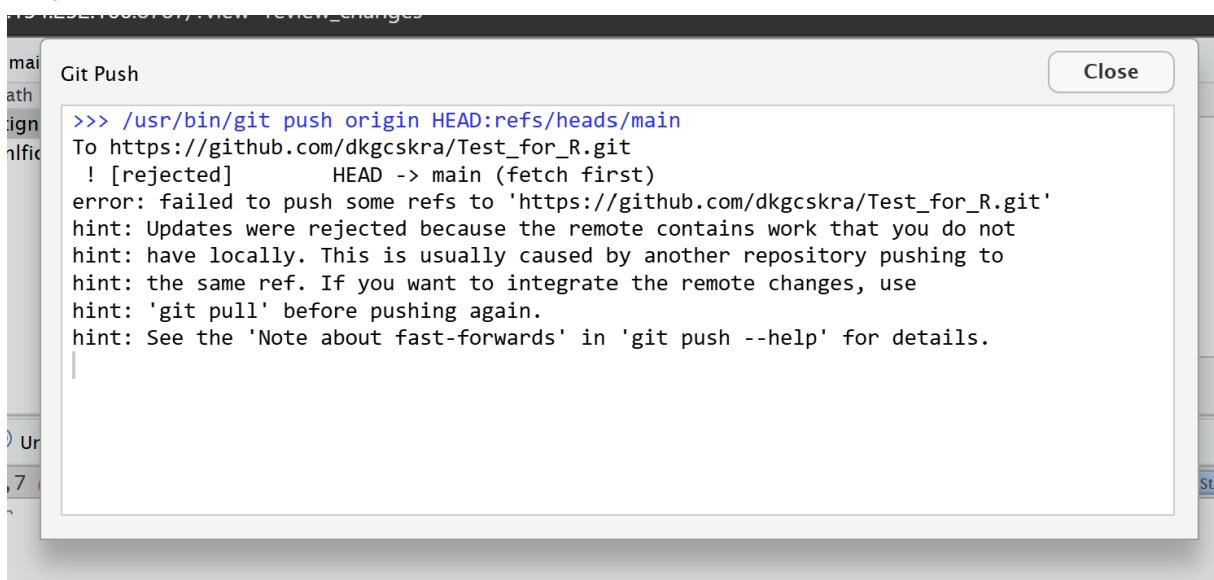
Amend previous commit Commit

Show Staged Unstaged Context 5 lines Ignore Whitespace Stage All Discard All

@@ -1,7 +1,7 @@
1 .Rproj.user
2 .Rhistory
3 .RData
4 .Ruserdata
5 only\_local.R
6 Conflict.R
7 | 7 |



## Error

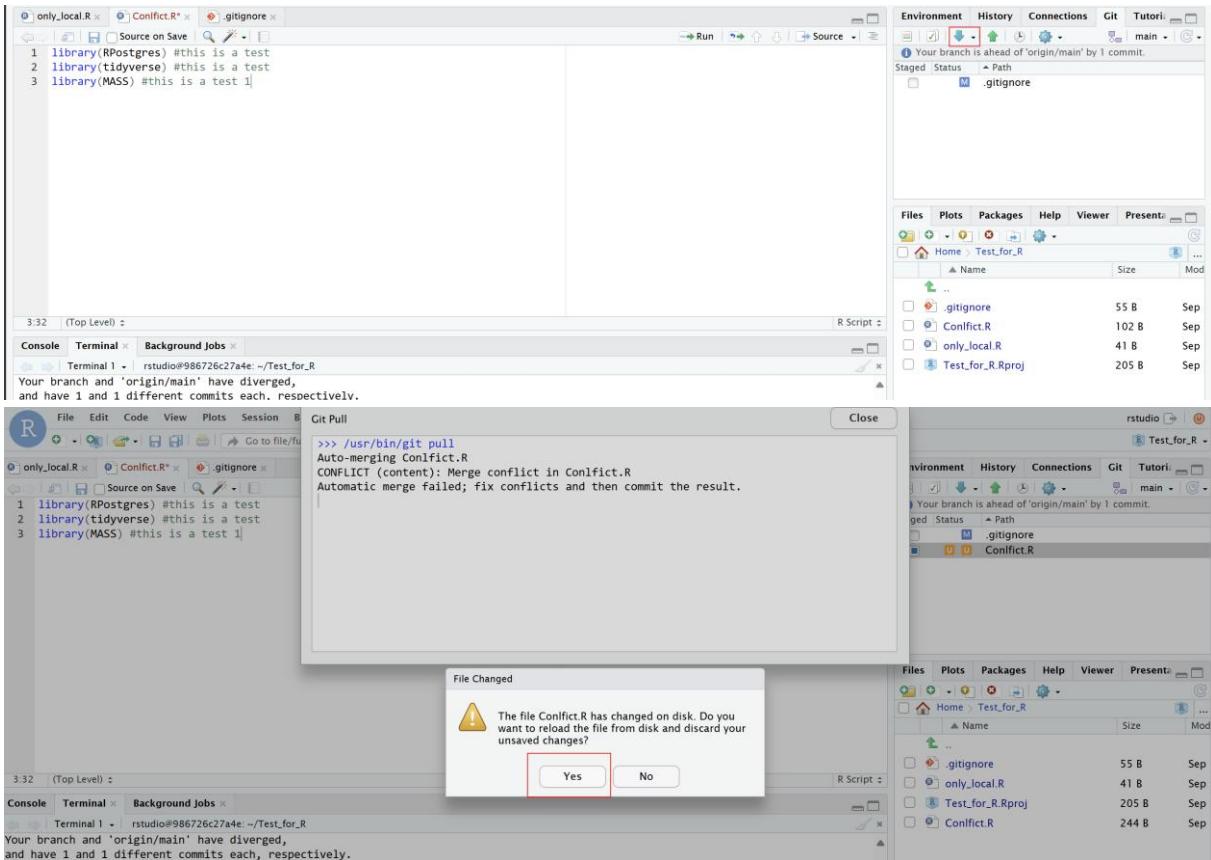


In the terminal type

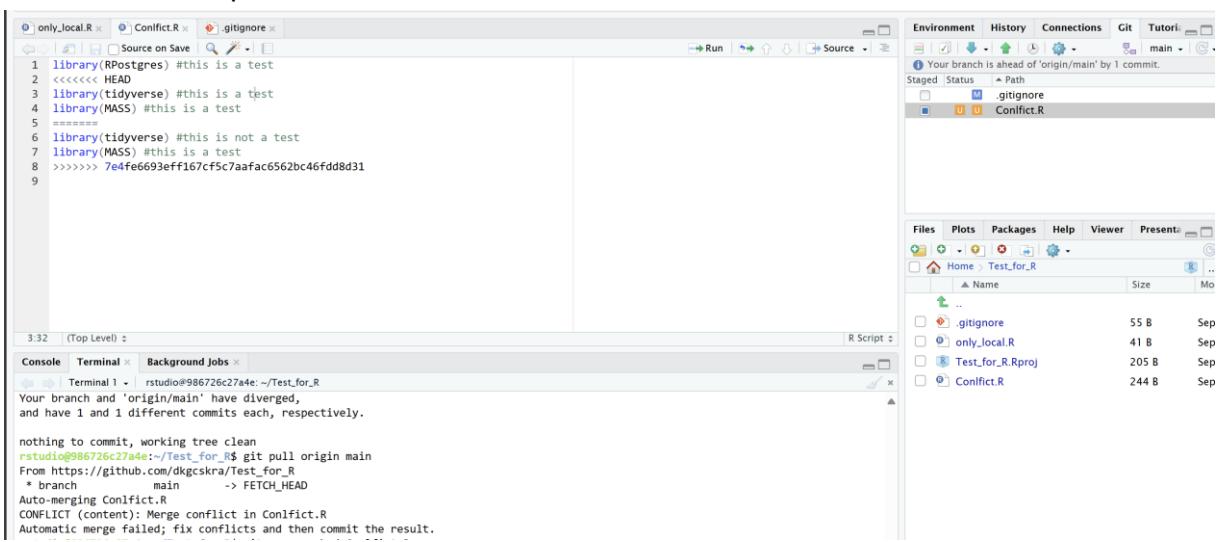
```
git config pull.rebase false
```

```
rstudio@986726c27a4e:~/git_training3$ git config pull.rebase false
```

Then pull data from git to R



Error will show up in R



Fix manually

```

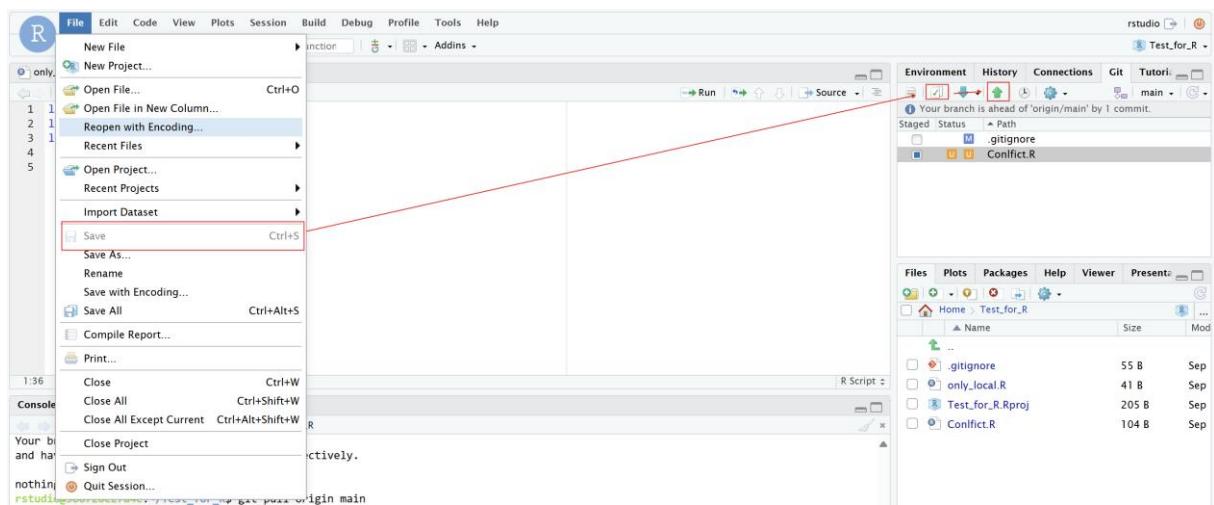
1 library(RPostgres) #this is a test
2 library(tidyverse) #this is a test
3 library(MASS) #this is a test
4
5

```

Your branch and 'origin/main' have diverged,  
and have 1 and 1 different commits each, respectively.

nothing to commit, working tree clean

## Merge, Save, commit and push file



It worked

```

1 library(RPostgres) #this is a test
2 library(tidyverse) #this is a test
3 library(MASS) #this is a test

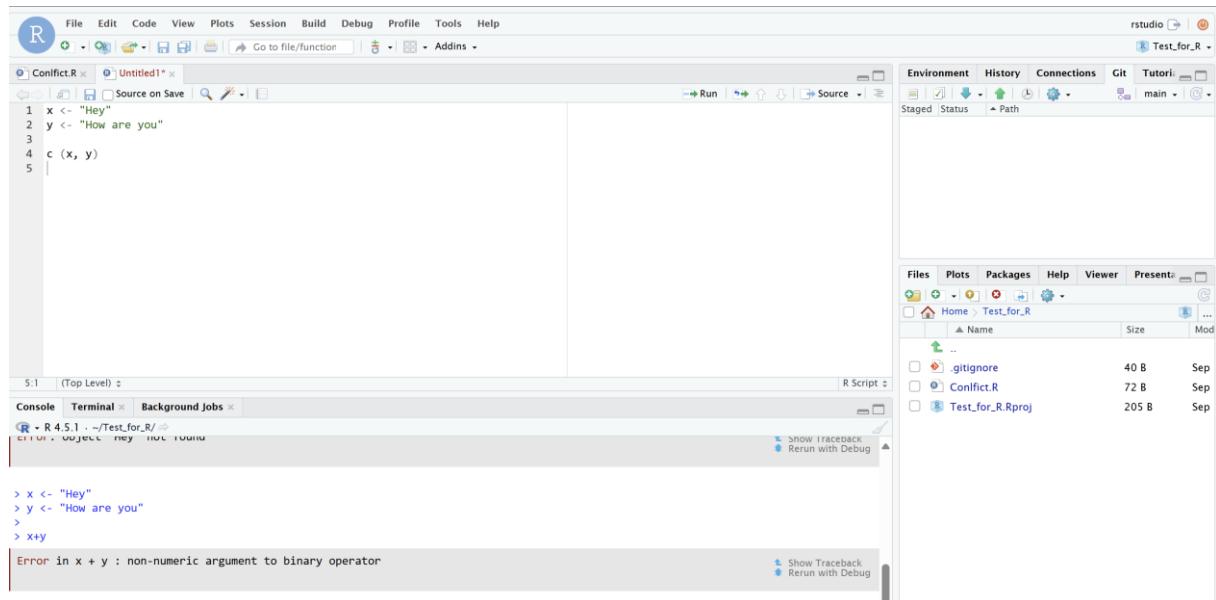
```



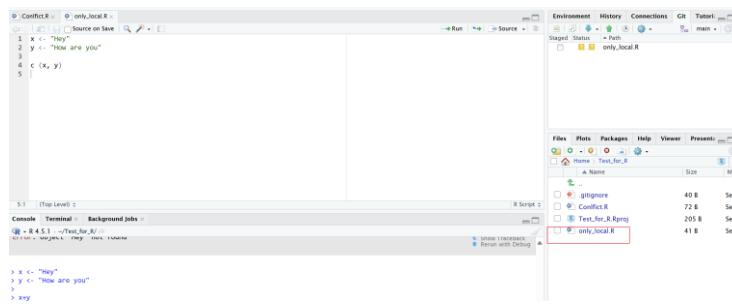
## .Gitignore file

You can edit in the .gitignore file, to specify files or folders you want Git to not version control

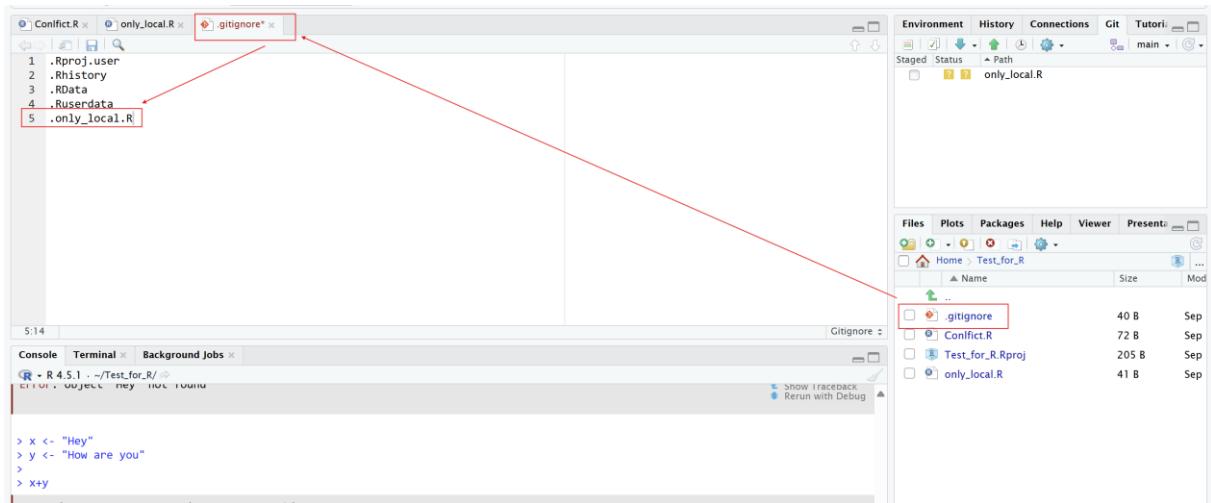
Create a new file we don't want it be tracked



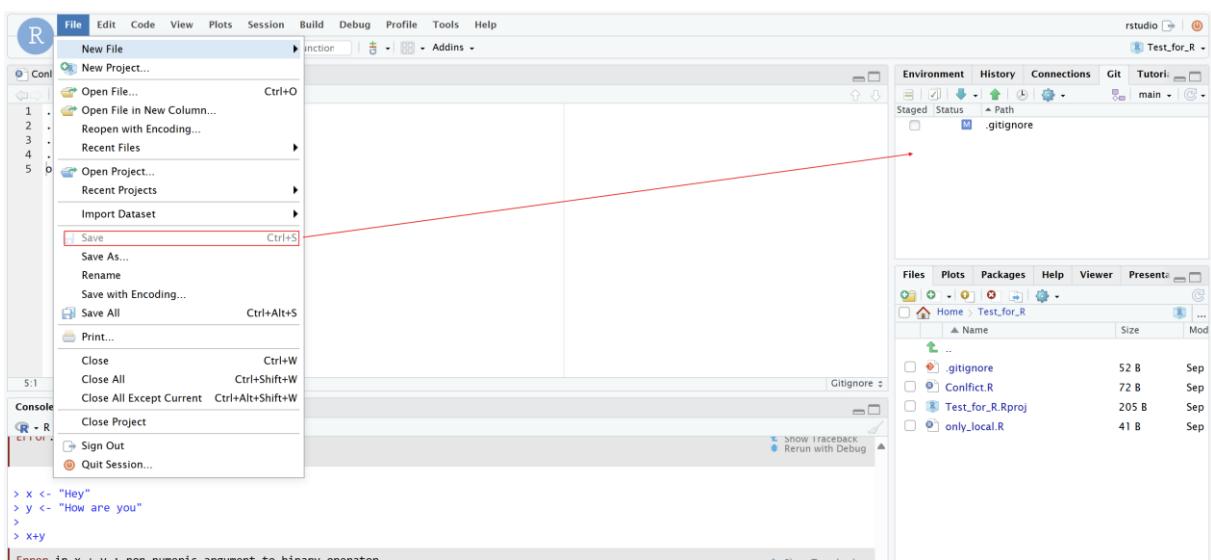
## Save it



Go into gitignore file and add the file you want not to be tracked (only you can see it)



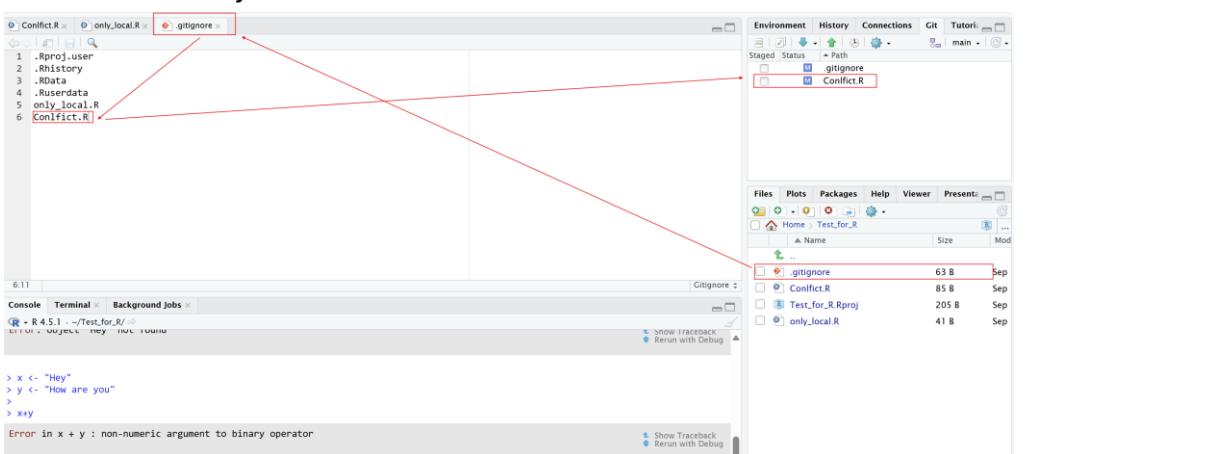
## Save the file



As you can see only\_local.R is gone from “git” and not registered by Git anymore

→ Git is not going to pull or push anything to that file

If we have already made commits to a file this does not work



Instead:

```
git rm --cached Conflict.R
```

The screenshot shows the RStudio interface with the following details:

- Code Editor:** Shows the contents of the `.gitignore` file, which includes entries for `.Rproj.user`, `.Rhistory`, `.RData`, `.Ruserdata`, `only_local.R`, and `Conflict.R`.
- Git Environment:** The `Staged` tab is selected, showing the file `Conflict.R` with a status of `M` (modified).
- Terminal:** The command `git rm --cached Conflict.R` was run, resulting in the message: "Your branch and 'origin/main' have diverged, and have 1 and 1 different commits each, respectively." followed by the output of a `git pull` command.

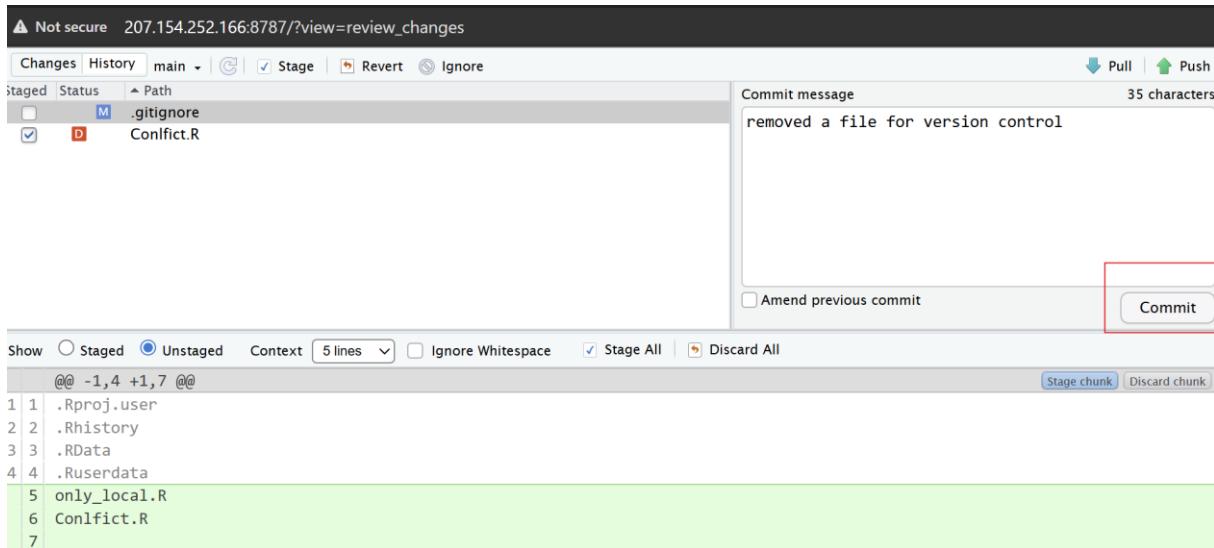
Then save gitignore file again

After saving gitignore file the file will be shown as deleted in “git” but not on local

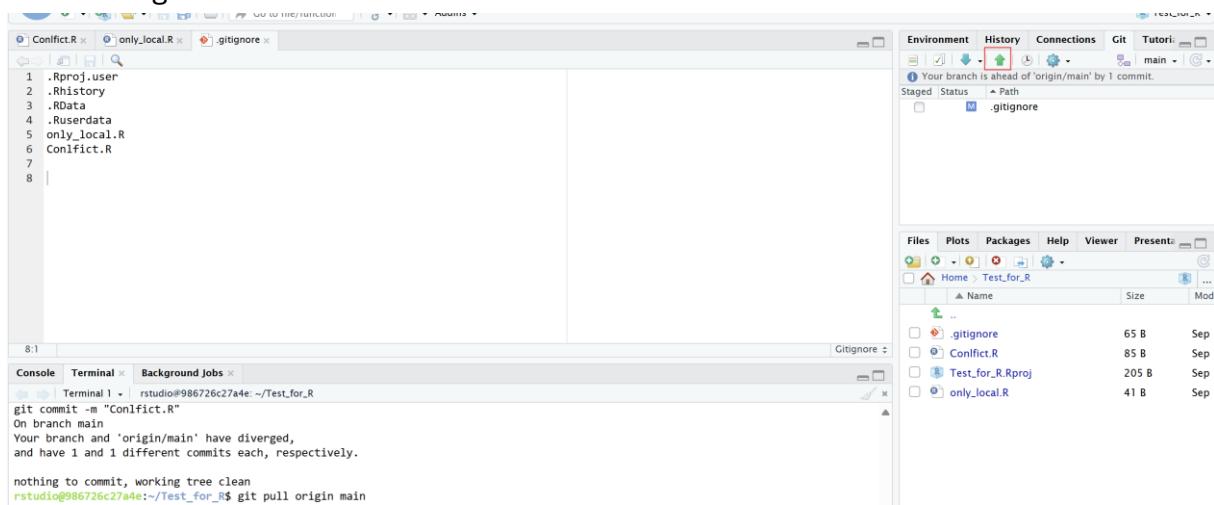
The screenshot shows the RStudio interface with the following details:

- Code Editor:** Shows the contents of the `.gitignore` file, identical to the previous screenshot.
- Git Environment:** The `Staged` tab is selected, showing the file `Conflict.R` with a status of `D` (deleted).
- Terminal:** The command `git rm --cached Conflict.R` was run, resulting in the message: "Your branch and 'origin/main' have diverged, and have 1 and 1 different commits each, respectively." followed by the output of a `git pull` command.

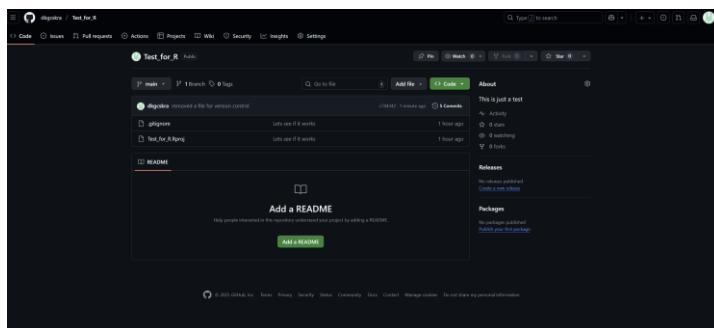
Commit changes



## Push changes



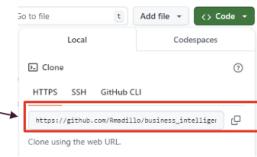
## Its gone from git



## Cloning a repository

### CLONING A REPOSITORY

1. In the terminal (for example, the terminal in Rstudio) navigate to the folder where you wish to locate the new repository
  2. Find the repository HTTP link on Github
  3. In the terminal run:
    1. `git clone [https link to repository]`
    2. For example,  
`git clone https://github.com/Rmadillo/business_intelligence_with_r.git`
  4. The repository will now be downloaded as a folder within the current directory in your terminal.
- Assignment: Clone the repository located at this link:  
<https://github.com/nayelnoorani/-teeny-tiny-machine>  
into a folder you call "test\_cloning"



## COLLABORATING

- If you wish to collaborate with fellow students then you can do the following
  - 1. One person creates the repository on Github
  - 2. The person who creates the repository invites the other students
    - 1. Go to the Repository → Settings → Collaborators → Add people
  - 3. The other students accept the invitation (see the mailbox icon at the top right corner on your Github page) 
  - 4. The other students clone the repository to their local machine (and/or their droplet)
    - 1. Create a new folder, and navigate to it in the terminal. Then, clone the repository into this folder. This prevents cloning a repository named 'git\_training' into a location where a folder with the same name already exists.
- Assignment: Share your git\_training repository with one other student. Then push a <sup>28</sup> change to the repository, and check if your fellow student can pull the change