Using git to push your changes to gitlab:

1. In your project folder on your local machine.

```
$ git init
```

Note:

This command creates a new git repository in your folder.

2. In your project folder on your local machine. $ git remote add origin <your project url (clone with https on gitlab)>

Note:

This tells git where the remote repository is located.

3. In your project folder on your local machine.

```
$ git add .
```

Note:

The dot (.) here stages all your changes in your current folder. If you have files you do NOT want to commit unstage them in the Source tab of vs code (CTRL+SHIFT+G).

4. In your project folder on your local machine.

```
$ git commit -m "Initial commit"
```

Note:

This creates a commit using your staged changes (i.e. the changes you made are stored in one unit). The -m flag stands for Message. The message is used to describe what changes the commit made to the codebase.

As a general guideline for commits, the message should ...

- not be more than 50 characters
- follow the conventional commit format: "[change type]: [explanation]"
- use the present tense only (i.e. "change remoting implementation to use ssh" instead of "changed remoting implementation")
- not contain and

Think of it like this:

"If this commit is applied it will: [explanation]"

Examples:

(1) "If this commit is applied it will: change the remoting implementation to use ssh" (GOOD)

(2) "If this commit is applied it will: restructure files and log output to verbose and change x" (BAD)

5. In your project folder on your local machine.

```
$ git push -u origin
```

Note:

This command pushes (i.e. saves) the code in the commit you just created to the remote origin that was specified in step 2.

If the branch-name is not specified, Git assumes you want to push the current branch. It’s good practice to state it explicitly, especially for beginners.

6. In gitlab.

Now your changes should be visible under Code > Branches in GitLab. Look for the branch you pushed (e.g., main or another branch name you used).

Using git to push your changes to gitlab if you've already been using another platform, you can still push your changes to gitlab. Here's how:

1. In your project folder on your local machine.

```
$ git remote add gitlab <your project url (clone with https on gitlab)>
```

Note:

Using this command an additional remote is added where the same code can be stored.

2. In your project folder on your local machine.

```
$ git push -u gitlab
```

Note:

By using the -u gitlab flag, git pushes the changes to the newly previously configured remote.

Be cautious about which remote you are pushing to, especially if you're working on a project mirrored between platforms.

Suggested reading if you are new to git:

1. https://webtuu.com/blog/04/a-laymans-introduction-to-git (https://webtuu.com/blog/04/a-laymans-introduction-to-git)

2. https://webtuu.com/blog/04/git-basics-branching-merging-push-to-github (https://webtuu.com/blog/04/git-basics-branching-merging-push-to-github)
3. https://rogerdudler.github.io/git-guide/ (https://rogerdudler.github.io/git-guide/)
4. https://www.conventionalcommits.org/en/v1.0.0/ (https://www.conventionalcommits.org/en/v1.0.0/)

Advanced material:

1. https://blog.bespinian.io/posts/git-the-important-parts/ (https://blog.bespinian.io/posts/git-the-important-parts/)
2. https://blog.bespinian.io/posts/infrastructure-as-code/ (https://blog.bespinian.io/posts/infrastructure-as-code/)
3. https://blog.bespinian.io/posts/gitops/ (https://blog.bespinian.io/posts/gitops/)