TASK - 2b

**Git commands**

**Submitted by,**

**Manoj A R**

**BE (CSE)**

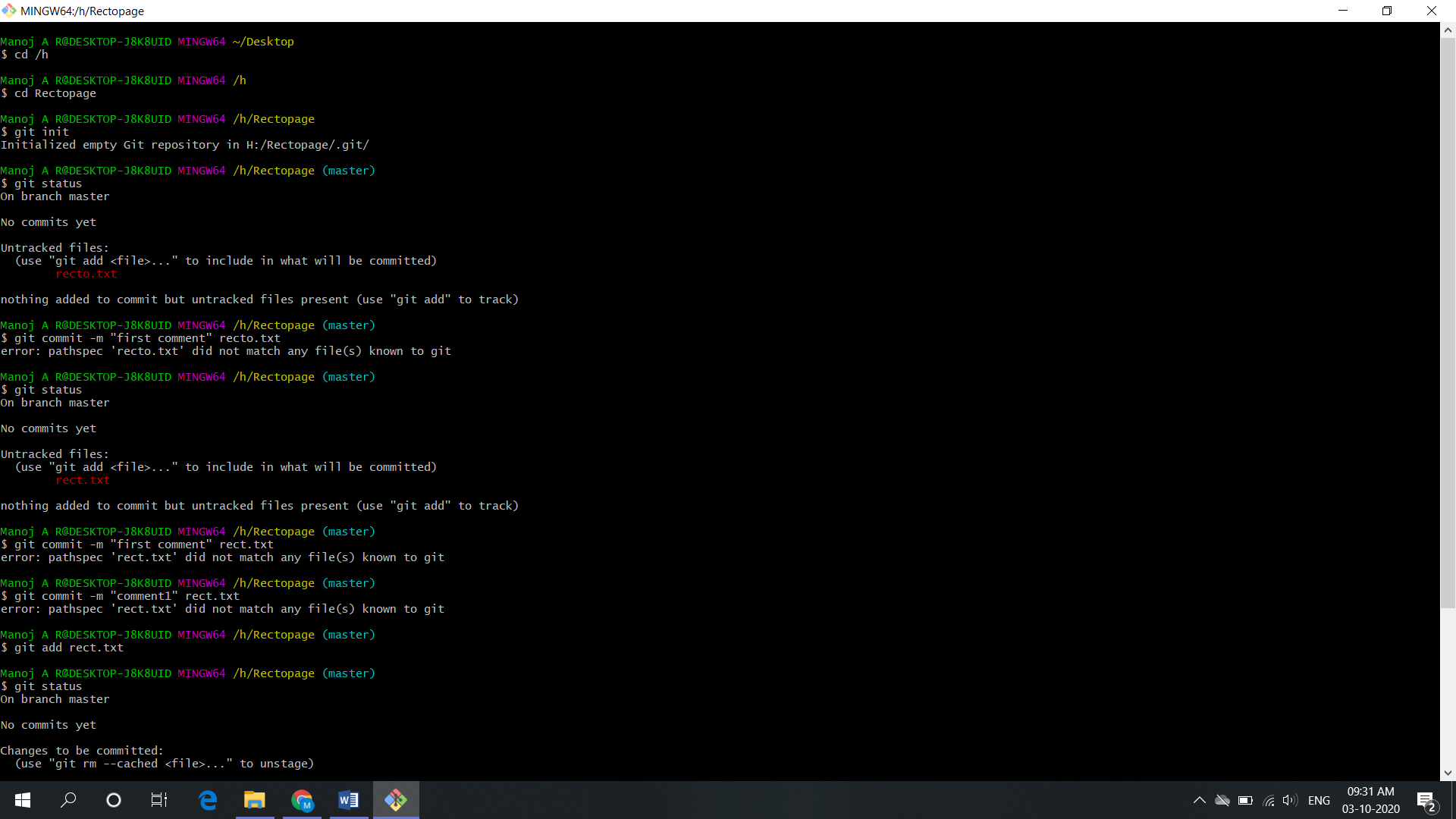
**PESITM**

**Shivamogga**

**Git init**

This command turns a directory into an empty git repository. This is the first step in creating a repository. After running git init, adding and committing files/directories is possible.

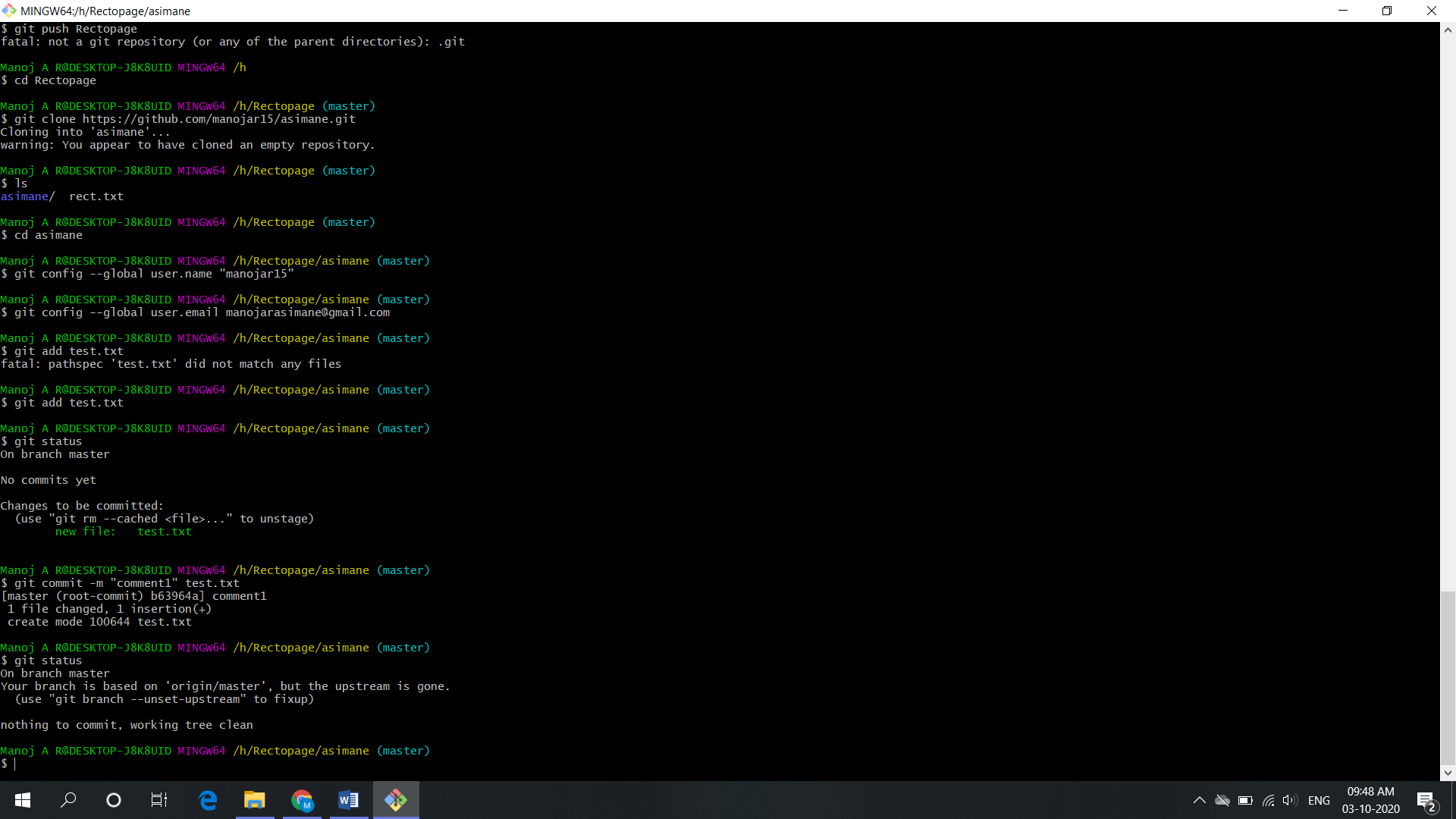
Usage: screenshot



**Git add**

Adds files in the to the staging area for git. Before a file is available to commit to a repository, the file needs to be added to the git index (staging area). There are a few different ways to use git add, by adding entire directories, specific files, or all unstaged files.

Usage: screenshot

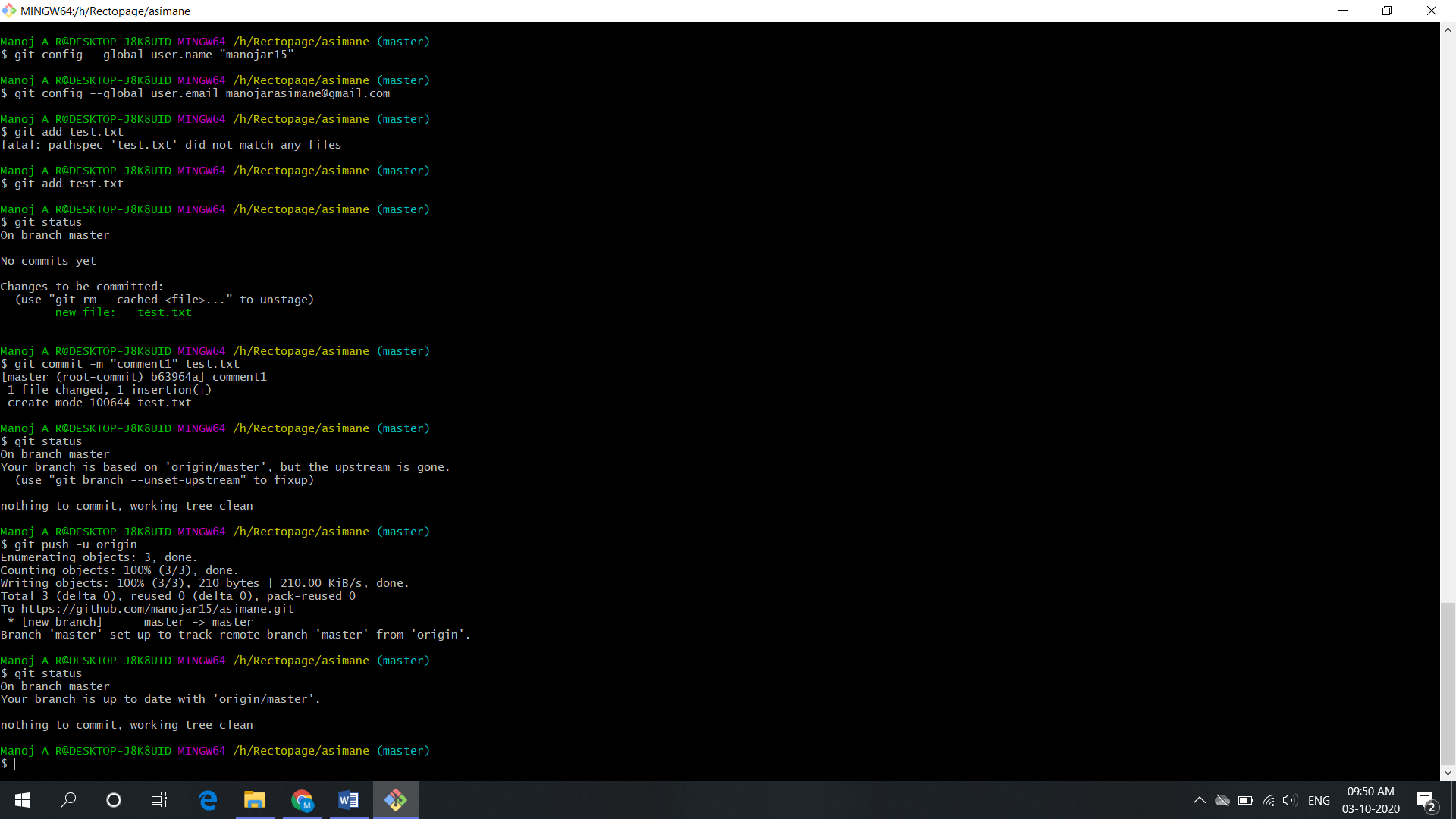
****

**Git commit**

Record the changes made to the files to a local repository. For easy reference, each commit has a unique id.

It’s best practice to include a message with each commit explaining the changes made in a commit. Adding a commit message helps to find a particular change or understanding the changes.

Usage: screenshot

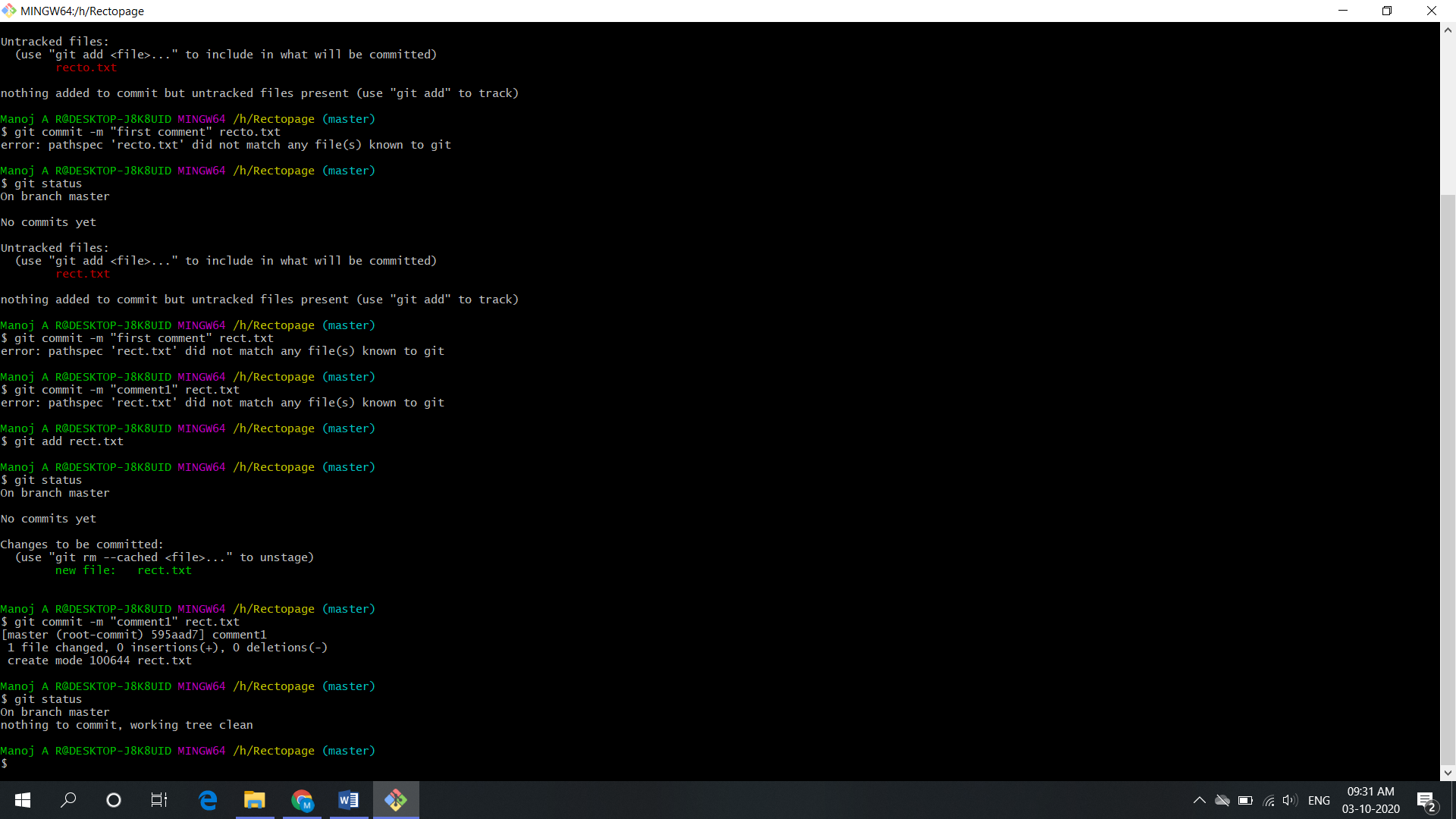


**Git status**

This command returns the current state of the repository.

Git status will return the current working branch. If a file is in the staging area, but not committed, it shows with git status. Or, if there are no changes it’ll return nothing to commit, working directory clean.

Usage: screenshot

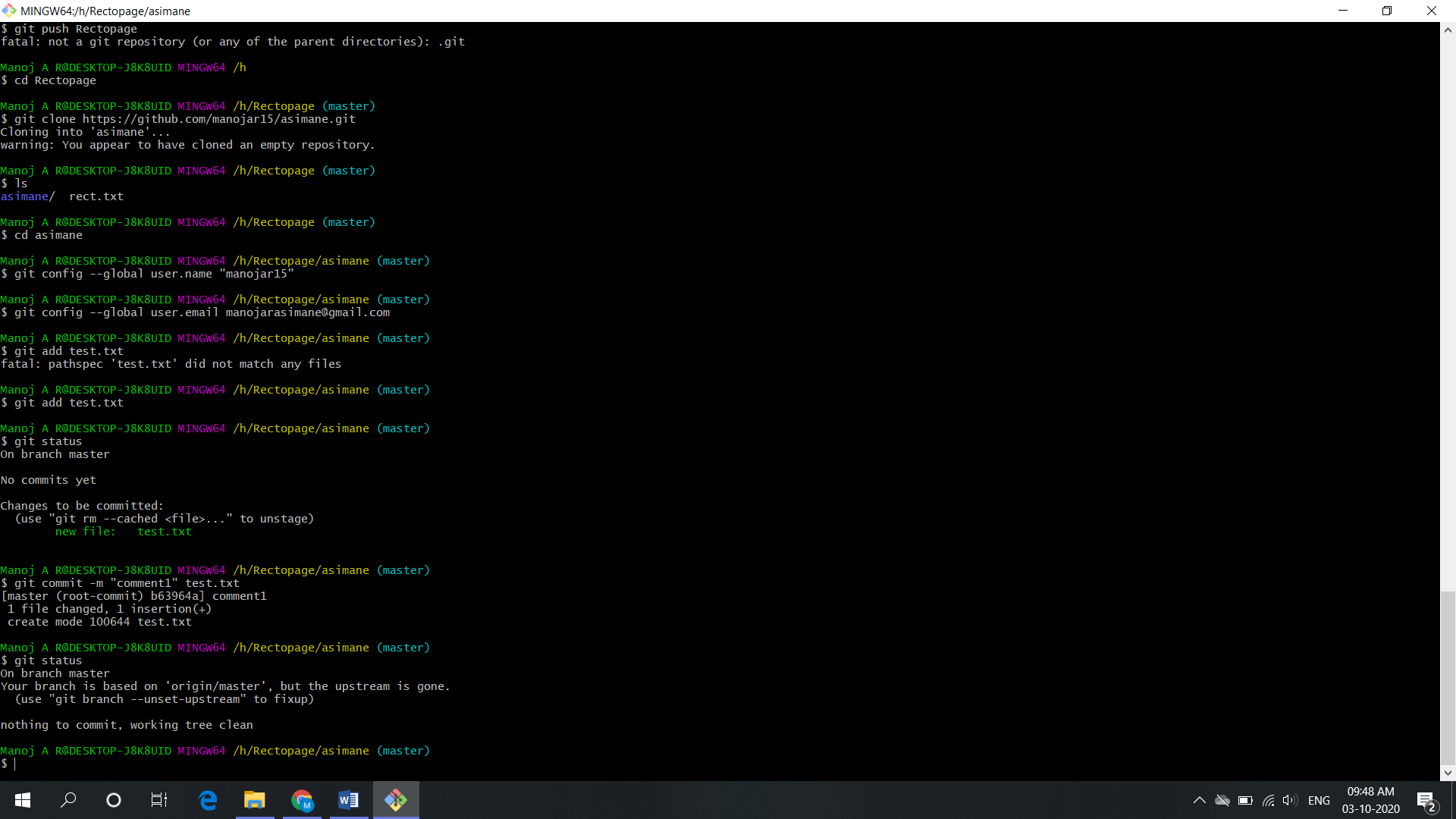


**Git config**

With git, there are many configurations and settings possible. Git config is how to assign these settings. Two important settings are user user.name and user.email. These values set what email address and name commits will be from on a local computer. With git config, a --global flag is used to write the settings to all repositories on a computer. Without a --global flag settings will only apply to the current repository that you are currently in.

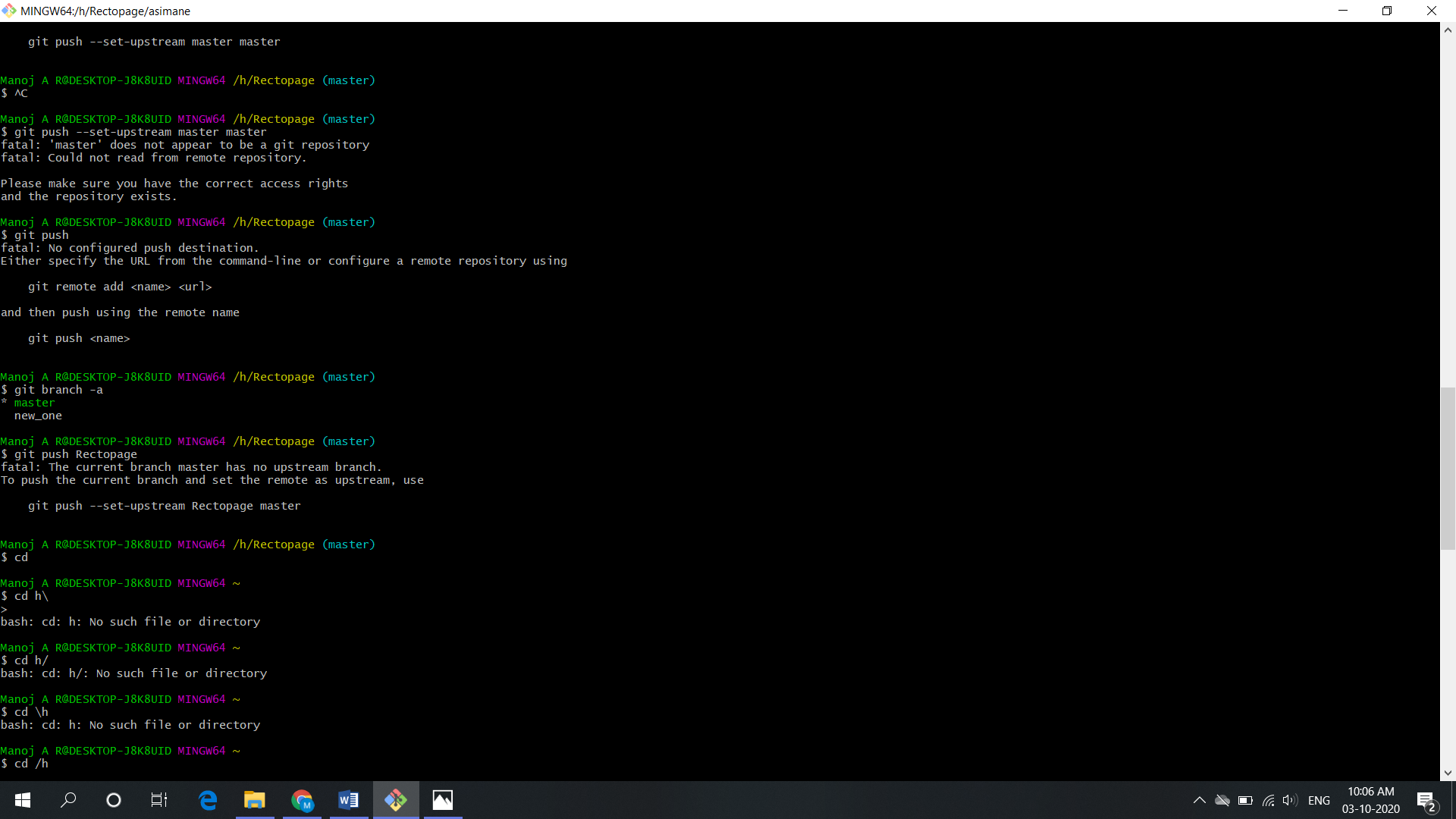
There are many other variables available to edit in git config. From editing color outputs to changing the behavior of git status. Learn about git config settings in the official [git documentation](https://git-scm.com/docs/git-config).

Usage: screenshot



**Git branch**

To determine what branch the local repository is on, add a new branch, or delete a branch.



**Git merge**

Integrate branches together. Git merge combines the changes from one branch to another branch. For example, merge the changes made in a staging branch into the stable branch.

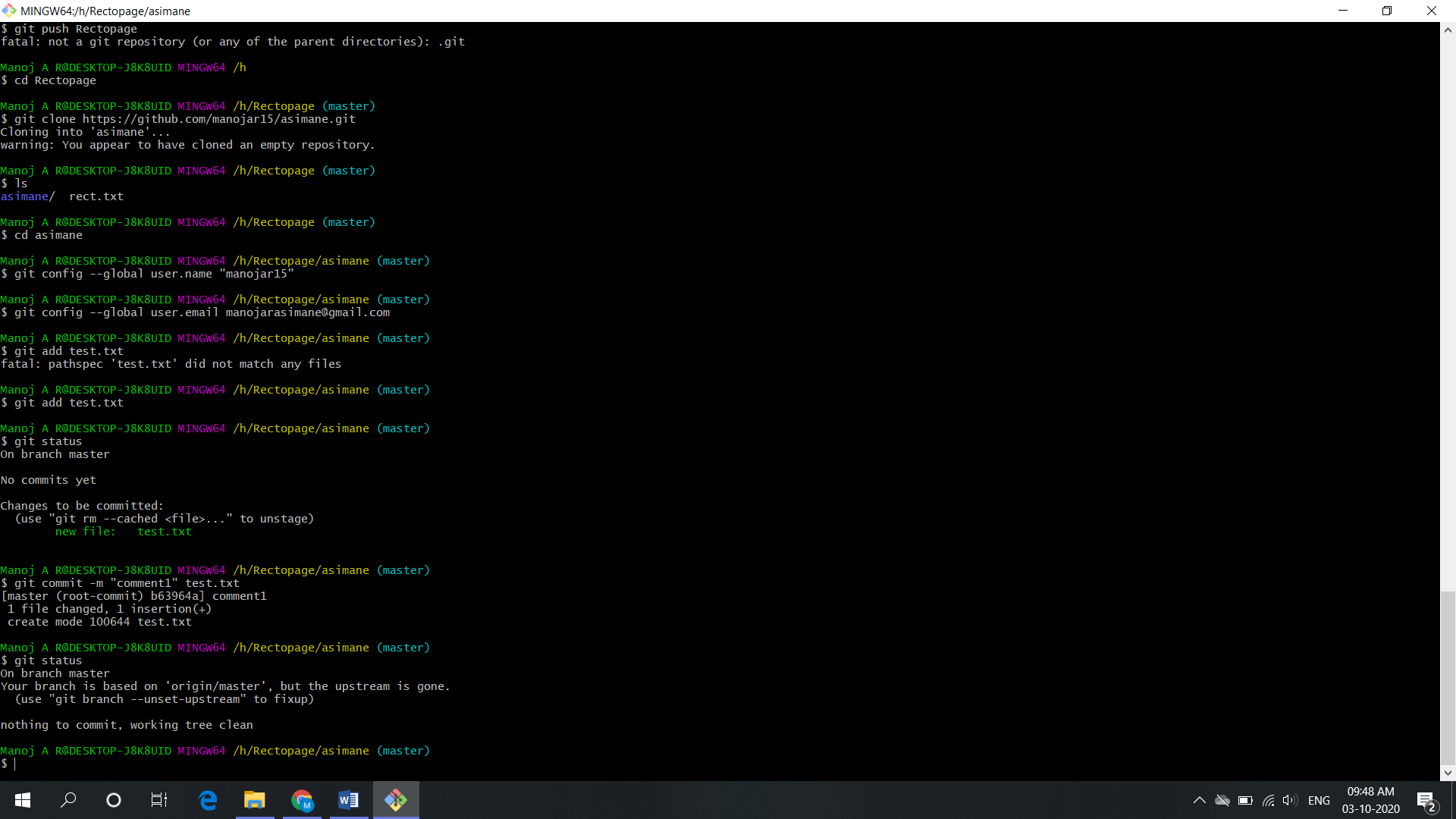
# merge changes into current branch

$ git merge <branch\_name>

**Git clone**

To create a local working copy of an existing remote repository, use git clone to copy and download the repository to a computer. Cloning is the equivalent of git init when working with a remote repository. Git will create a directory locally with all files and repository history.

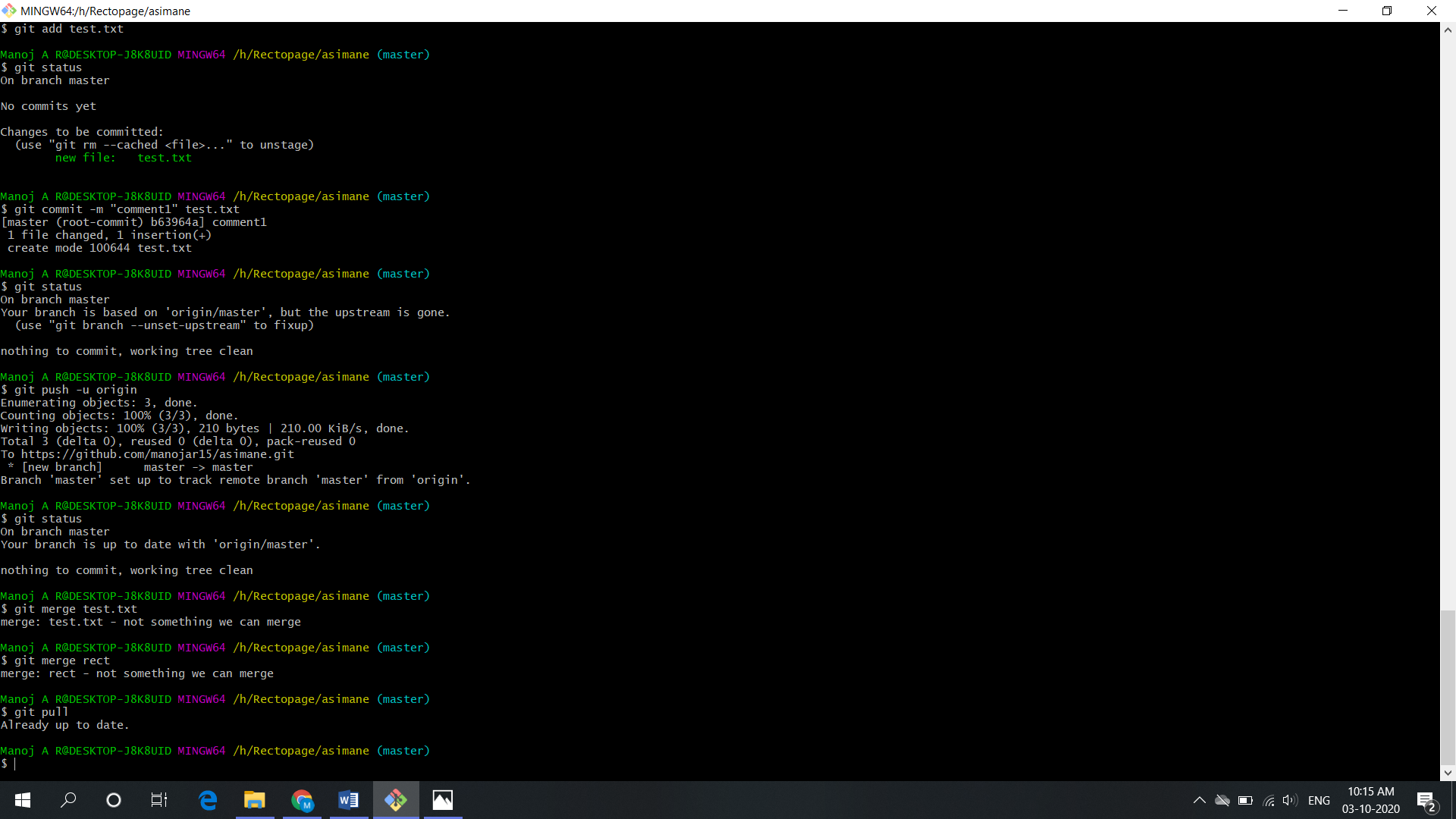
Usage: screenshot



**Git pull**

To get the latest version of a repository run git pull. This pulls the changes from the remote repository to the local computer.

Usage : screenshot



**Git push**

Sends local commits to the remote repository. Git push requires two parameters: the remote repository and the branch that the push is for.

Usage : screenshot

