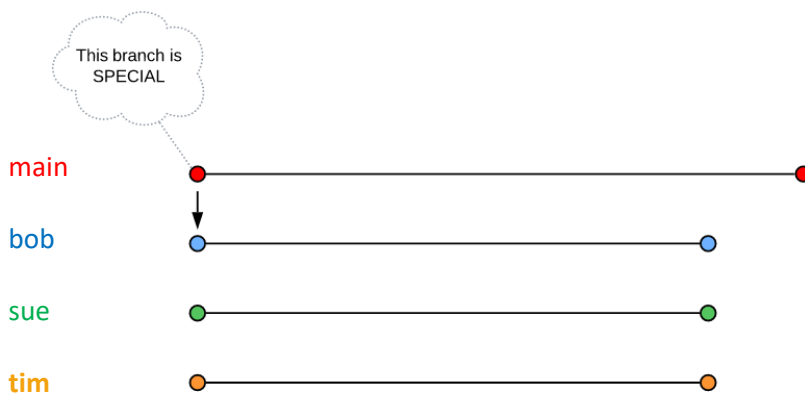


# Capstone Branching Strategies

The `main` branch of your capstone project is special. No member of your team should ever push code that does not compile or is broken to the main branch of the repo. You want to make sure that every member of your team has access to only the latest working version of your project. The only way to guarantee that is to ensure that the code in the `main` branch compiles and is free of errors (reasonably free)

## Everyone Gets a Branch

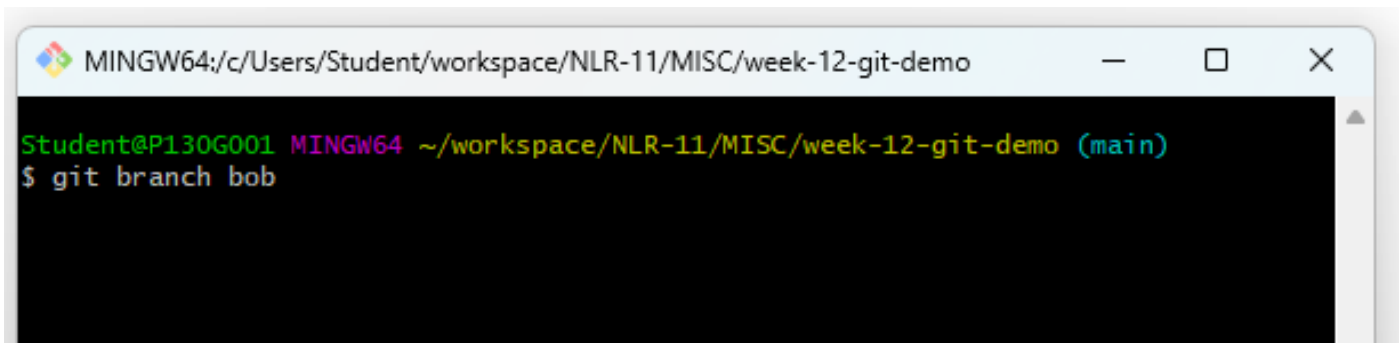
One strategy to protect the `main` branch is to create a development branch for each member of your team. Each developer will then work ONLY in their branch. Once they are ready to push changes to the team, they first merge their changes into main on THEIR computer, and then push. HINT: you should only push when you are certain that your code compiles and runs as expected.



## How to Create a Branch

After you clone your capstone repository, you will only have a single branch on your computer, the `main` branch. The first step is to create your own branch.

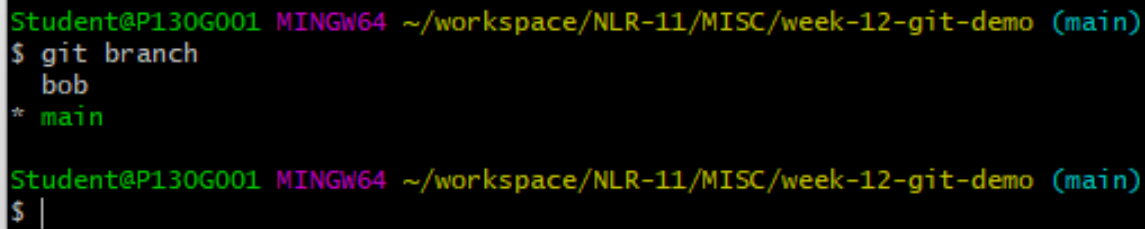
```
git branch bob
```

A terminal window titled 'MINGW64:/c/Users/Student/workspace/NLR-11/MISC/week-12-git-demo'. The prompt is 'Student@P130G001 MINGW64 ~/workspace/NLR-11/MISC/week-12-git-demo (main)'. The command '\$ git branch bob' has been entered.

```
MINGW64:/c/Users/Student/workspace/NLR-11/MISC/week-12-git-demo
Student@P130G001 MINGW64 ~/workspace/NLR-11/MISC/week-12-git-demo (main)
$ git branch bob
```

Verify that you have created the branch. This will show you what branches are on your machine.

```
git branch
```

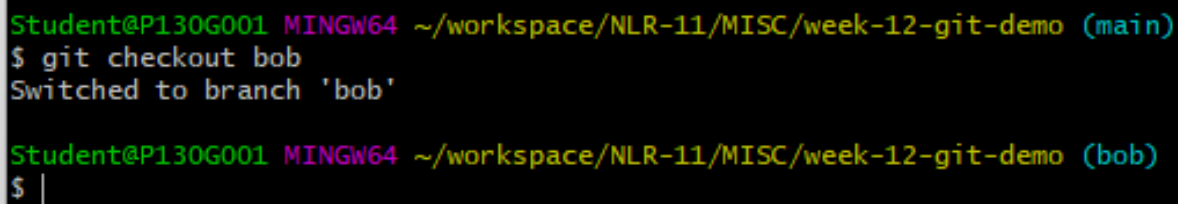
A terminal window showing the output of the 'git branch' command. The prompt is 'Student@P130G001 MINGW64 ~/workspace/NLR-11/MISC/week-12-git-demo (main)'. The command '\$ git branch' has been entered, and the output is 'bob' and '\* main'.

```
Student@P130G001 MINGW64 ~/workspace/NLR-11/MISC/week-12-git-demo (main)
$ git branch
bob
* main

Student@P130G001 MINGW64 ~/workspace/NLR-11/MISC/week-12-git-demo (main)
$ |
```

Switch to your branch:

```
git checkout bob
```

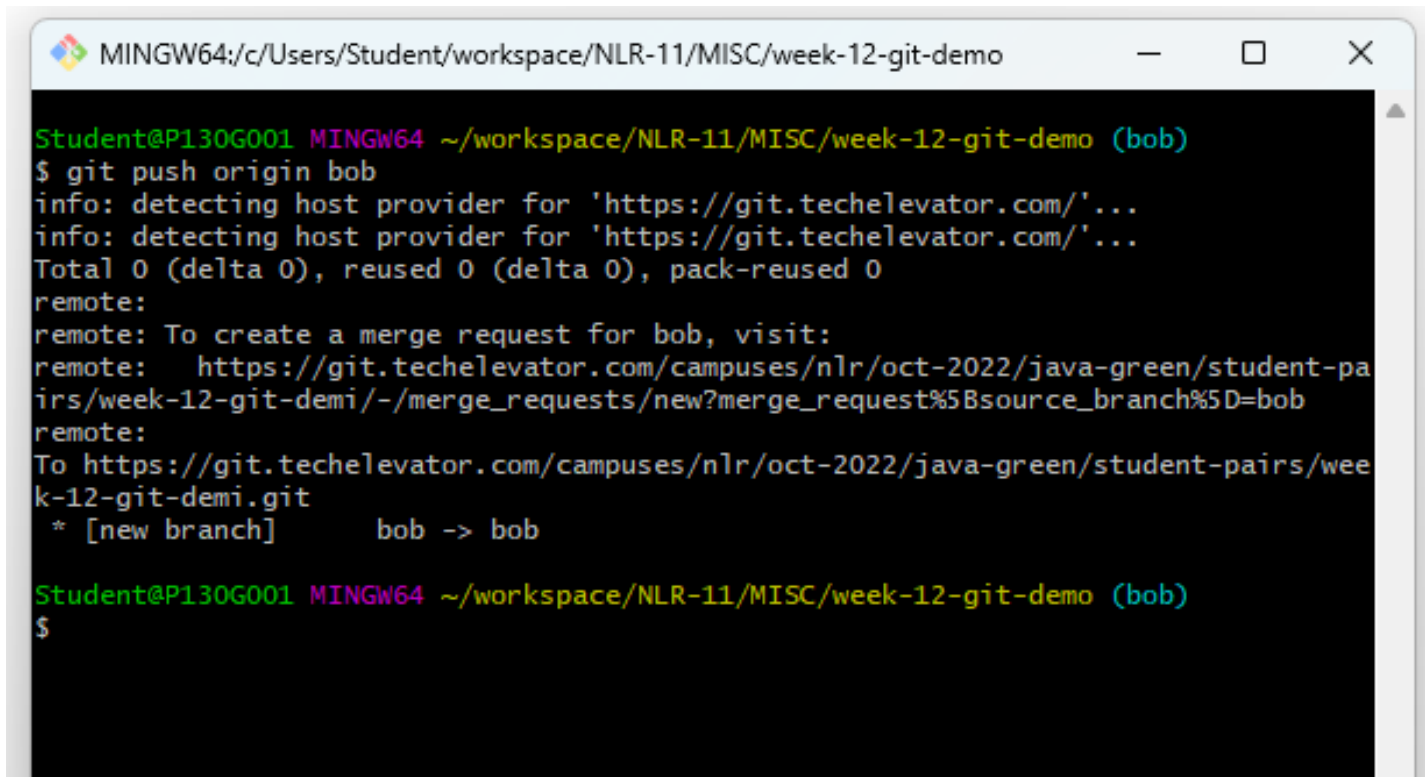
A terminal window showing the output of the 'git checkout bob' command. The prompt is 'Student@P130G001 MINGW64 ~/workspace/NLR-11/MISC/week-12-git-demo (main)'. The command '\$ git checkout bob' has been entered, and the output is 'Switched to branch 'bob''.

```
Student@P130G001 MINGW64 ~/workspace/NLR-11/MISC/week-12-git-demo (main)
$ git checkout bob
Switched to branch 'bob'

Student@P130G001 MINGW64 ~/workspace/NLR-11/MISC/week-12-git-demo (bob)
$ |
```

Push your branch to the remote repository:

```
git push origin bob
```

A screenshot of a Windows terminal window titled "MINGW64:/c/Users/Student/workspace/NLR-11/MISC/week-12-git-demo". The terminal shows the execution of the command "git push origin bob". The output includes information about detecting the host provider, the total number of objects (0), and a remote message to create a merge request. The terminal also shows the current branch is "bob" and the user is "Student@P130G001".

```
Student@P130G001 MINGW64 ~/workspace/NLR-11/MISC/week-12-git-demo (bob)
$ git push origin bob
info: detecting host provider for 'https://git.techelevator.com/'...
info: detecting host provider for 'https://git.techelevator.com/'...
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: To create a merge request for bob, visit:
remote:   https://git.techelevator.com/campuses/nlr/oct-2022/java-green/student-pa
irs/week-12-git-demi/-/merge_requests/new?merge_request%5Bsource_branch%5D=bob
remote:
To https://git.techelevator.com/campuses/nlr/oct-2022/java-green/student-pairs/wee
k-12-git-demi.git
 * [new branch]      bob -> bob

Student@P130G001 MINGW64 ~/workspace/NLR-11/MISC/week-12-git-demo (bob)
$
```

# Coding Workflow

This is the workflow we described in the bonus lecture:

## 1. Switch to **your** branch

```
git checkout bob
```

- Do all your coding here
- Take a task from Trello and do the work
- Test it on your computer
- When the work is complete...

```
git add -A
```

```
git commit -m "task 123 complete"
```

```
git push origin bob
```

## 2. Switch to the **main** branch (so that you can pull the latest work)

```
git checkout main
```

```
git pull origin main
```

## 3. Switch to **your** branch (so that you can integrate the latest code with yours)

```
git checkout bob
```

```
git merge main
```

- verify that your code still works and fix any errors/issues

```
git add -A
```

```
git commit -m "fixed merge issues for task 123"
```

```
git push origin bob
```

## 4. Switch to the **main** branch (merge your changes into main)

**TELL YOUR TEAM YOU ARE ABOUT TO PUSH CHANGES**

```
git checkout main
```

```
git merge bob
```

```
git push origin main
```

**TELL YOUR TEAM THAT YOU HAVE PUSHED YOUR CHANGES AND THEY NEED TO PULL**

## 5. Switch back to your branch

**Repeat steps 1-5 until the cows come home**