

QUICK START

Code Refactoring Workshop

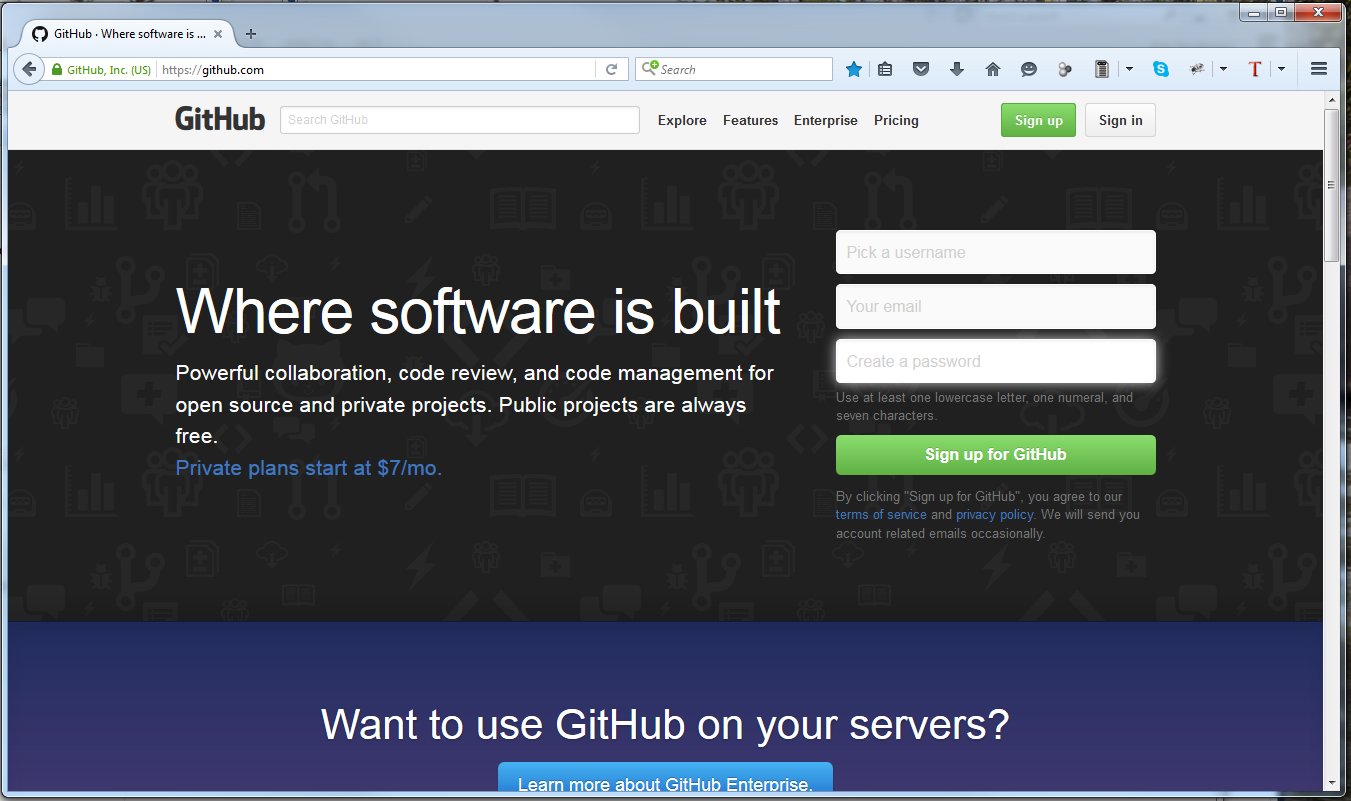
What is [GitHub](https://github.com/)?

GitHub is a web based platform for hosting and collaborating on projects.

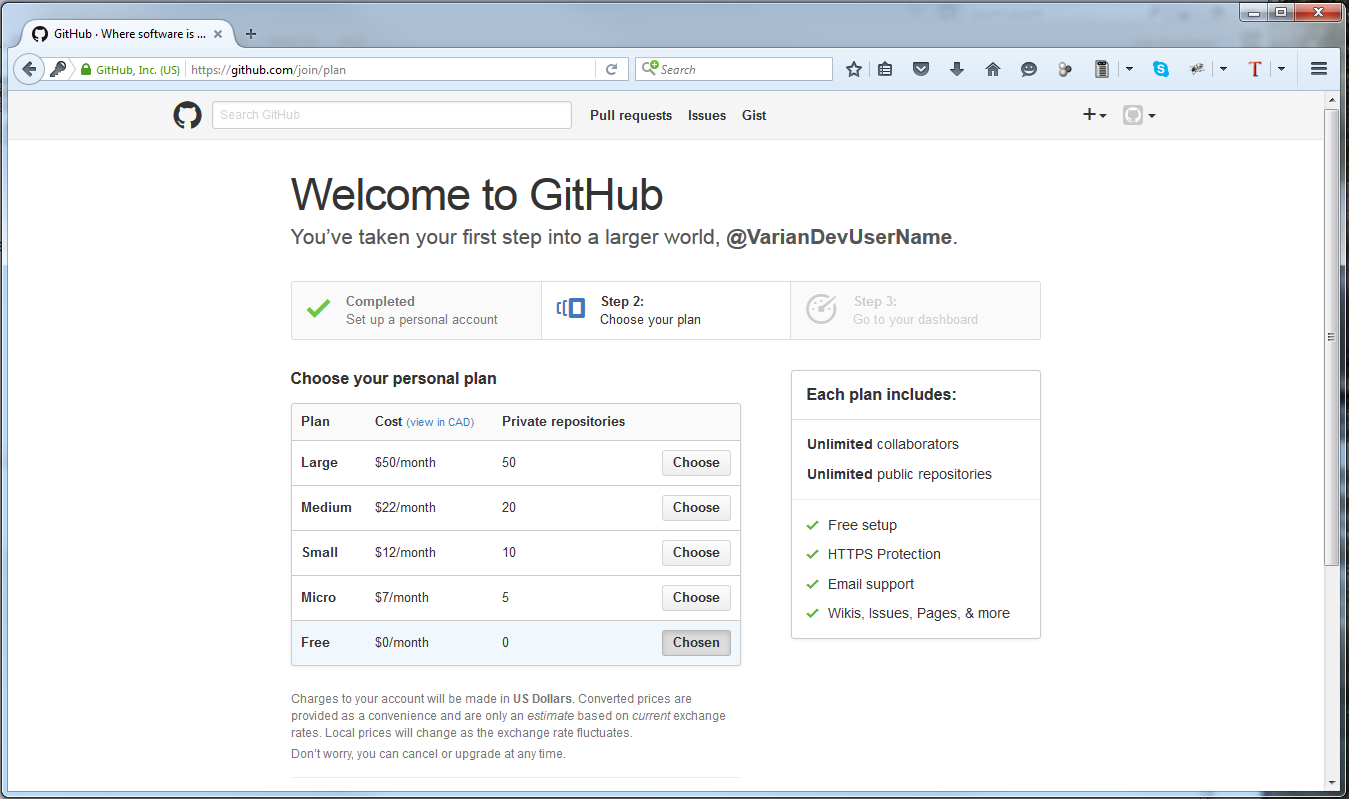
*“You don’t have to worry about losing data on your hard drive or managing a project across multiple computers — sync from anywhere. Most importantly, GitHub is a collaborative and asynchronous workflow for building software better, together”.*

Create a GitHub Account

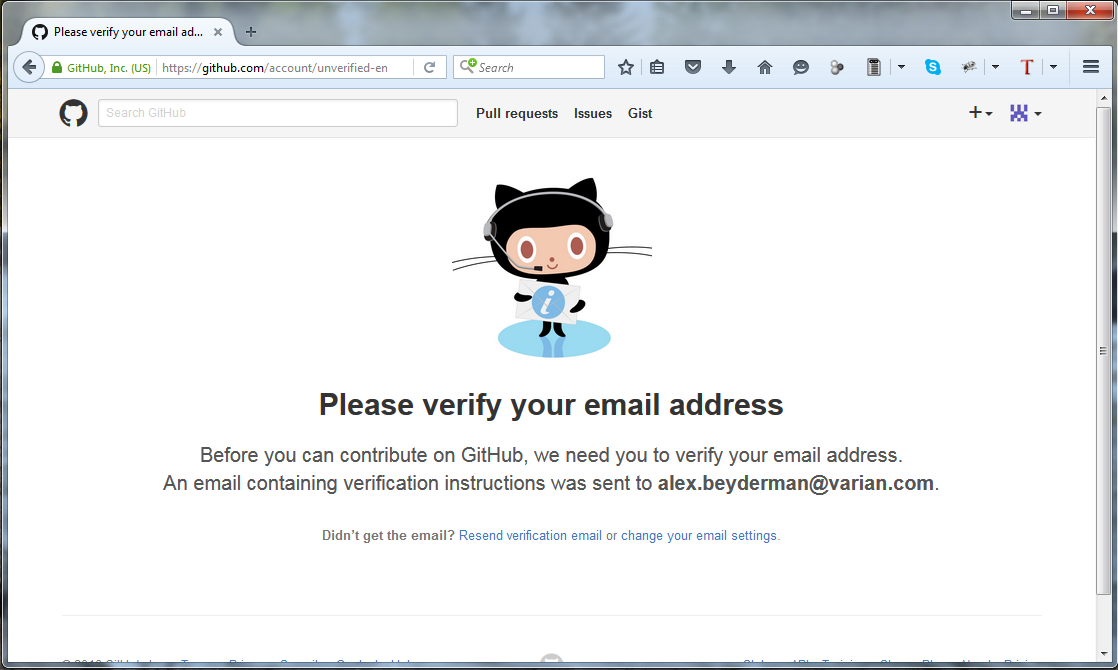
1. Browse to <https://github.com/>.
2. Pick a username, and enter your Varian email and a password.
   1. If you already have a GitHub account, you may use it for this exercise.
   2. Note: You can pick any username you like (AwesomeCoder is already taken), but you will have to let me know what your username is.
3. Click the **Sign up for GitHub** button.



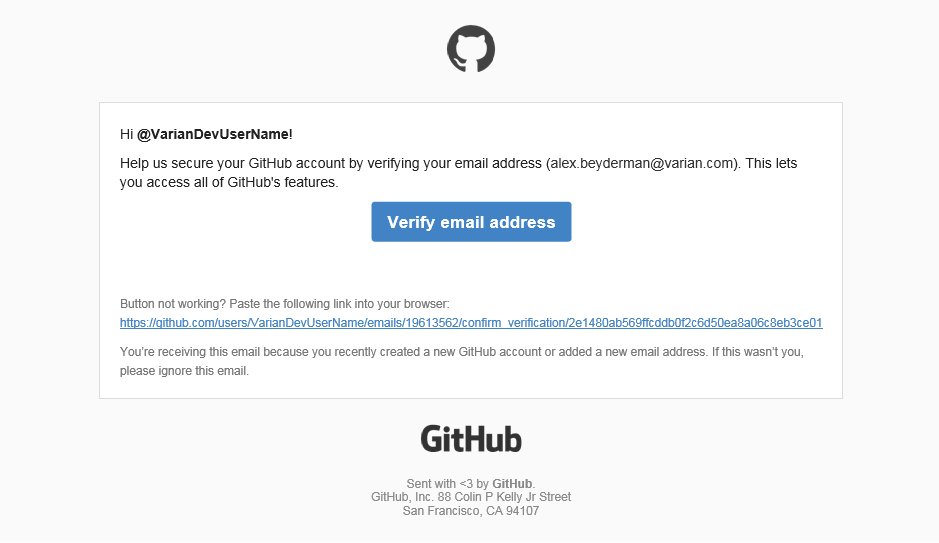
1. You will be forwarded to the **Welcome to GitHub** page with the **Free** plan chosen.
2. Click the **Finish sign up button** to create your GitHub account.



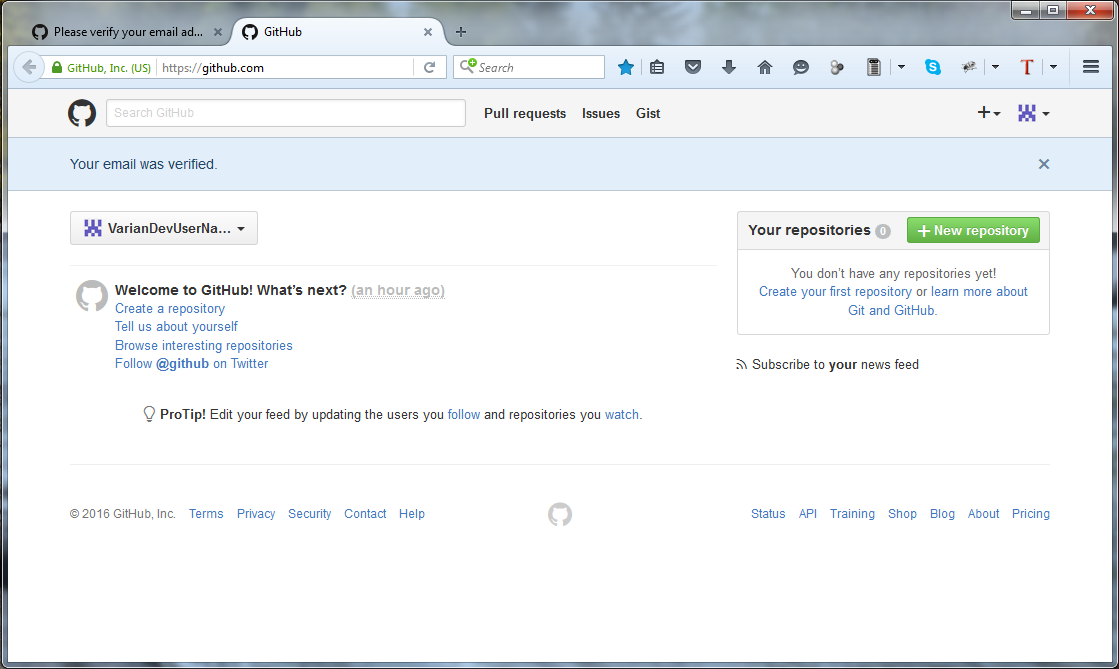
1. You cannot proceed until your email address is verified.



1. You will receive a **Please verify your e-mail address** message in the mailbox you specified.
2. Click the **Verify email address** button in the email.

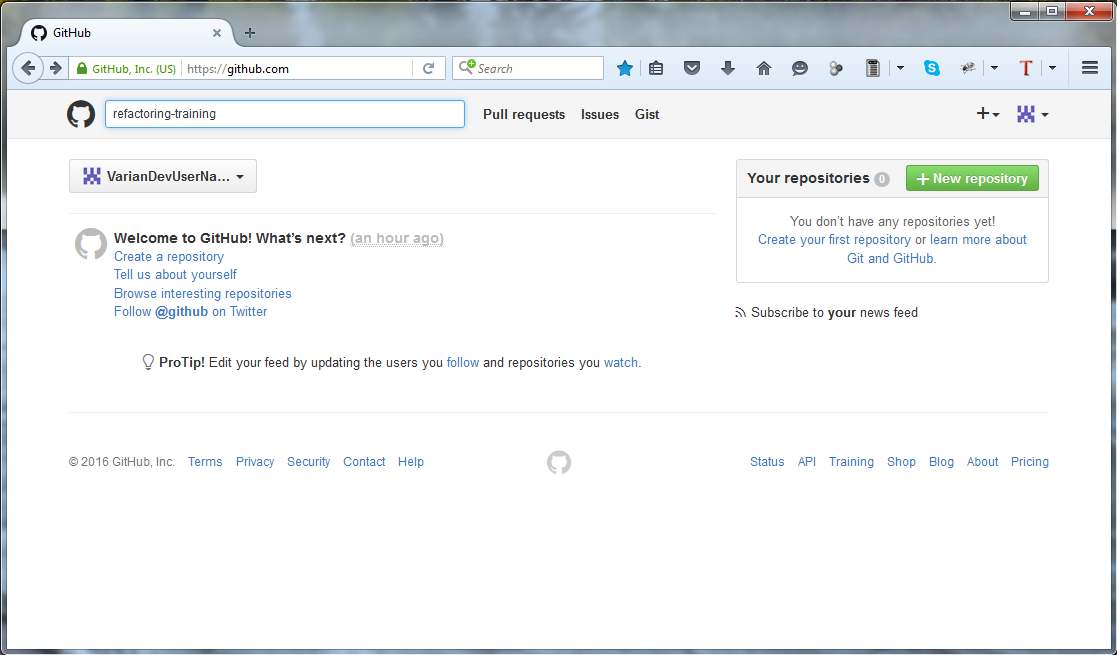


1. Your account was created and verified. You’ll get **Welcome to GitHub** email as well.

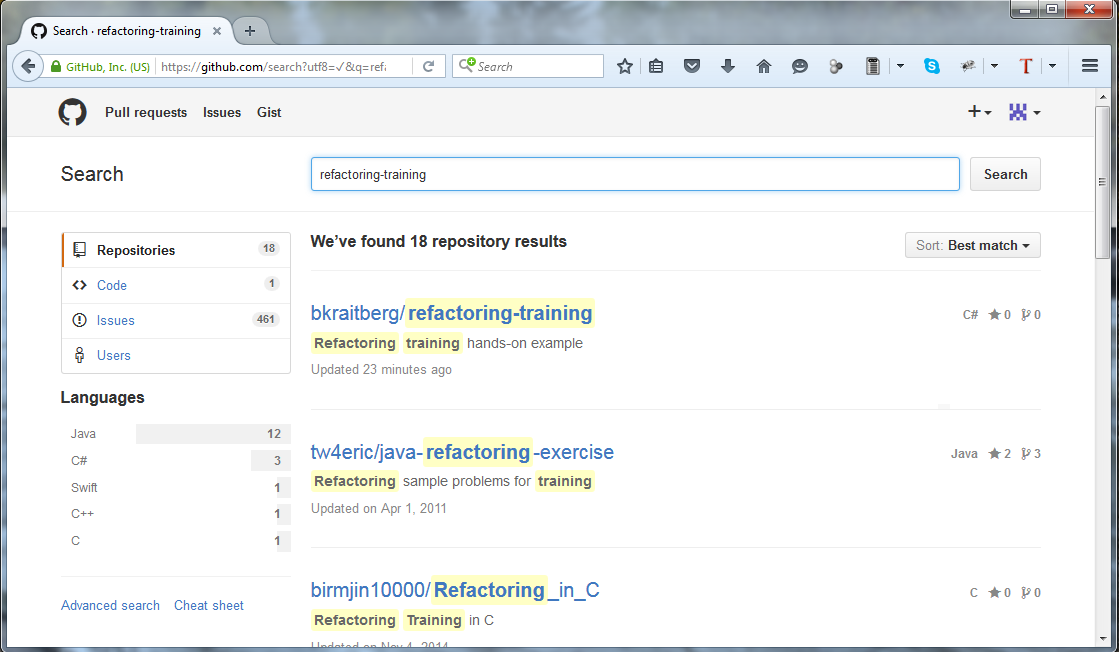


Create your own copy of the refactoring training repository

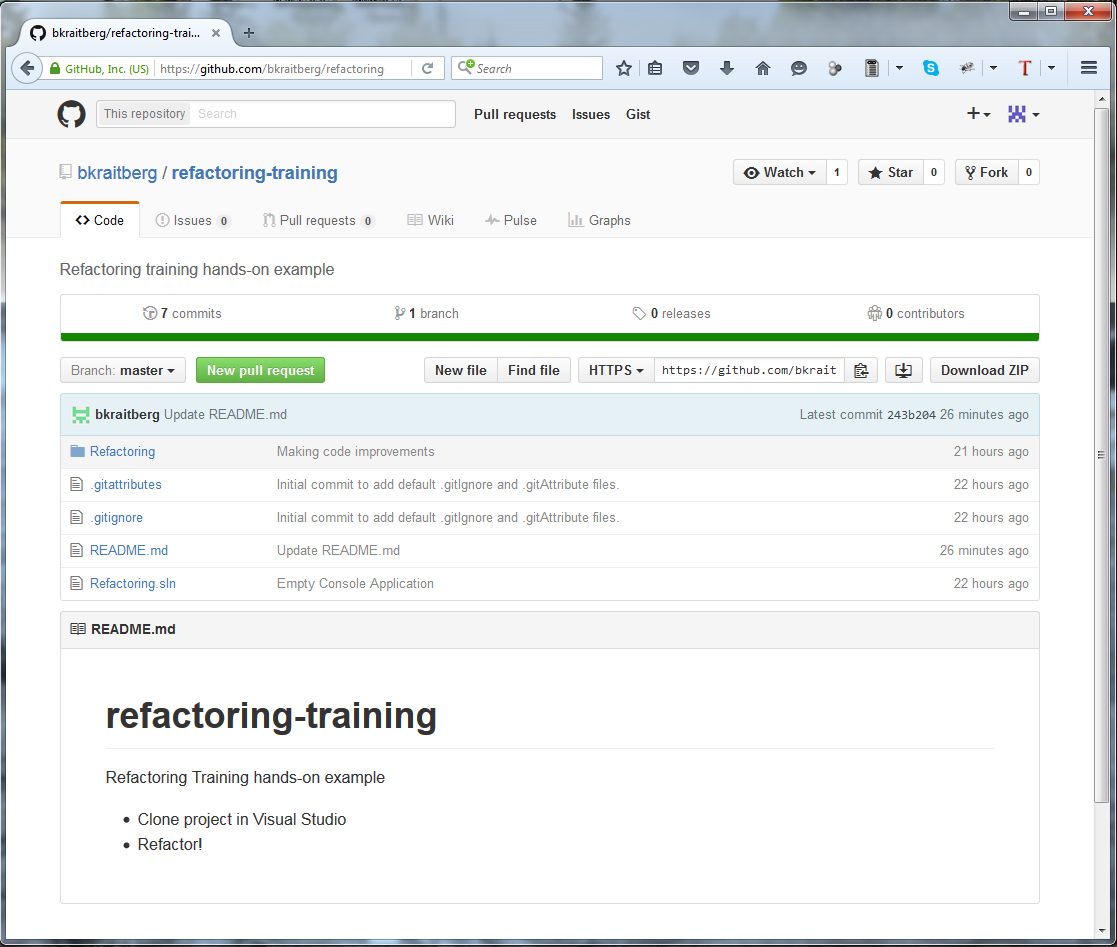
1. Let’s use a repository that was created for this training.
2. Type **refactoring-training** in the search box and hit **Enter**.



1. Multiple results are found, but we’ll pick Bryan’s Repository - [bkraitberg/refactoring-training](https://github.com/bkraitberg/refactoring-training)

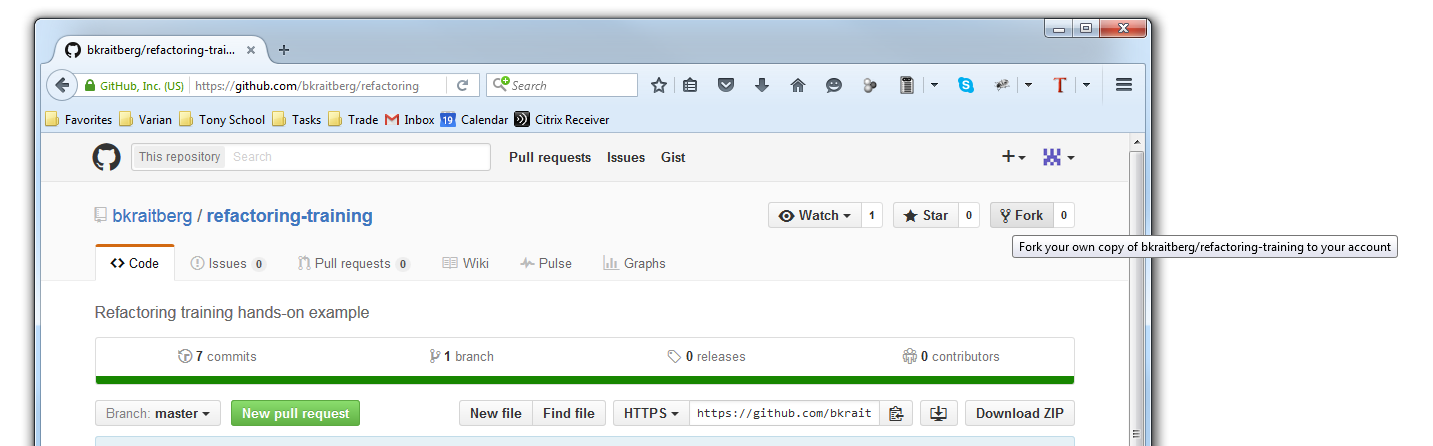


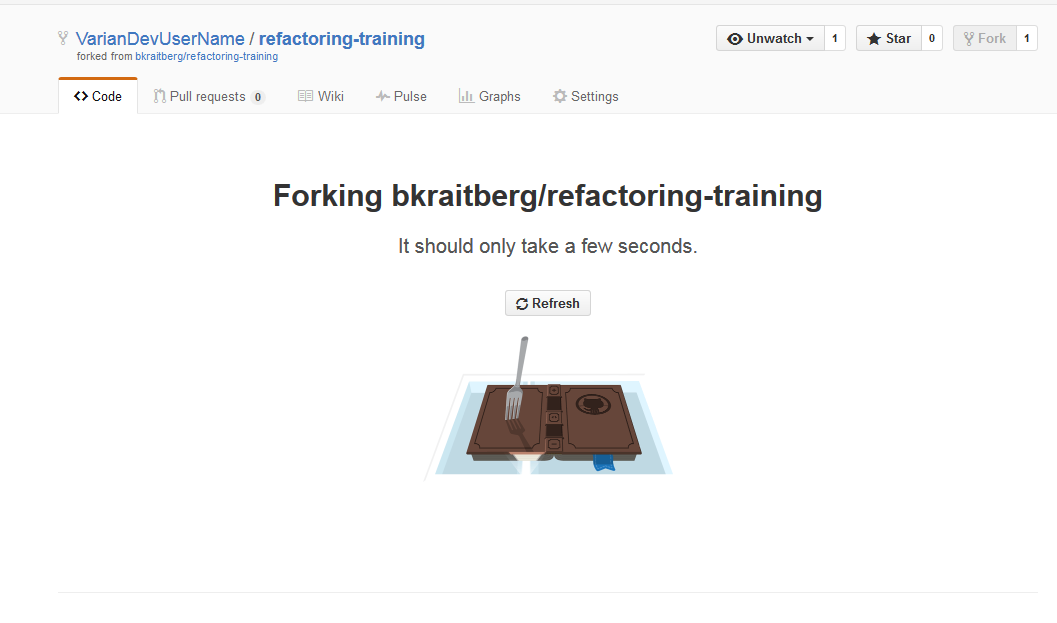
1. It will look like this



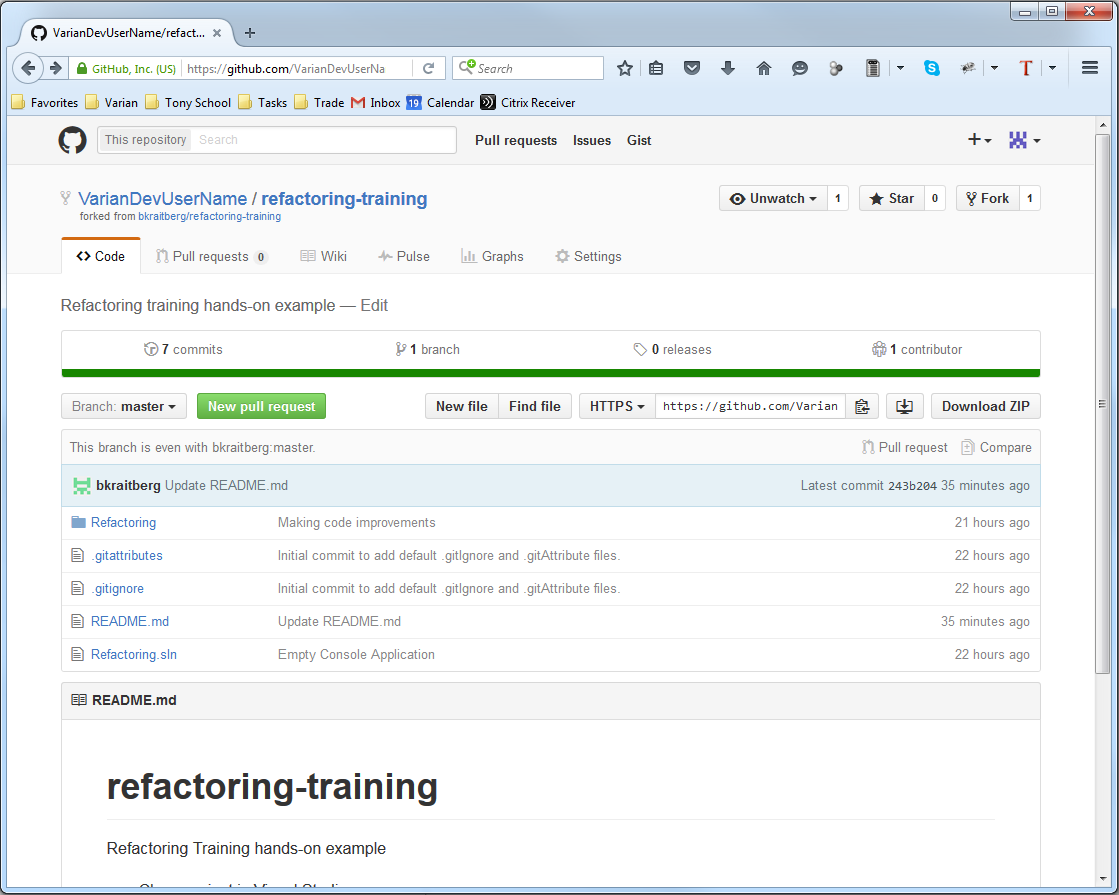
Fork It

1. Let’s create private copy of this repository under your account
2. Click the **Fork** button





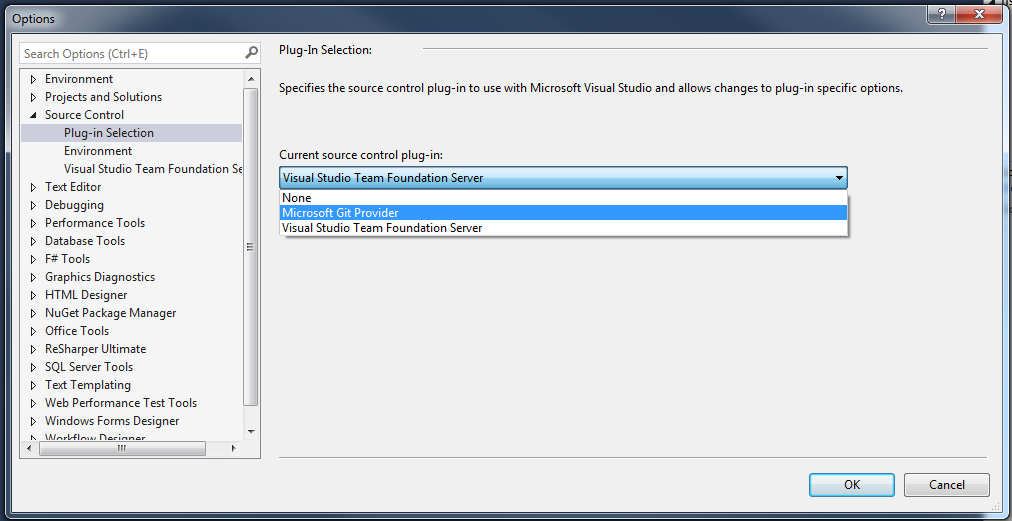
1. This is what it will look like after the Fork.
   1. Note that this repository has the same name and content, but it’s under your account now.
   2. It also specifies where it was Forked from.



The repo you’ve just created is located on a remote server. In order to modify it, you’ll need to create a local copy and open it inside the Visual Studio IDE, the same as when you work with TFS.

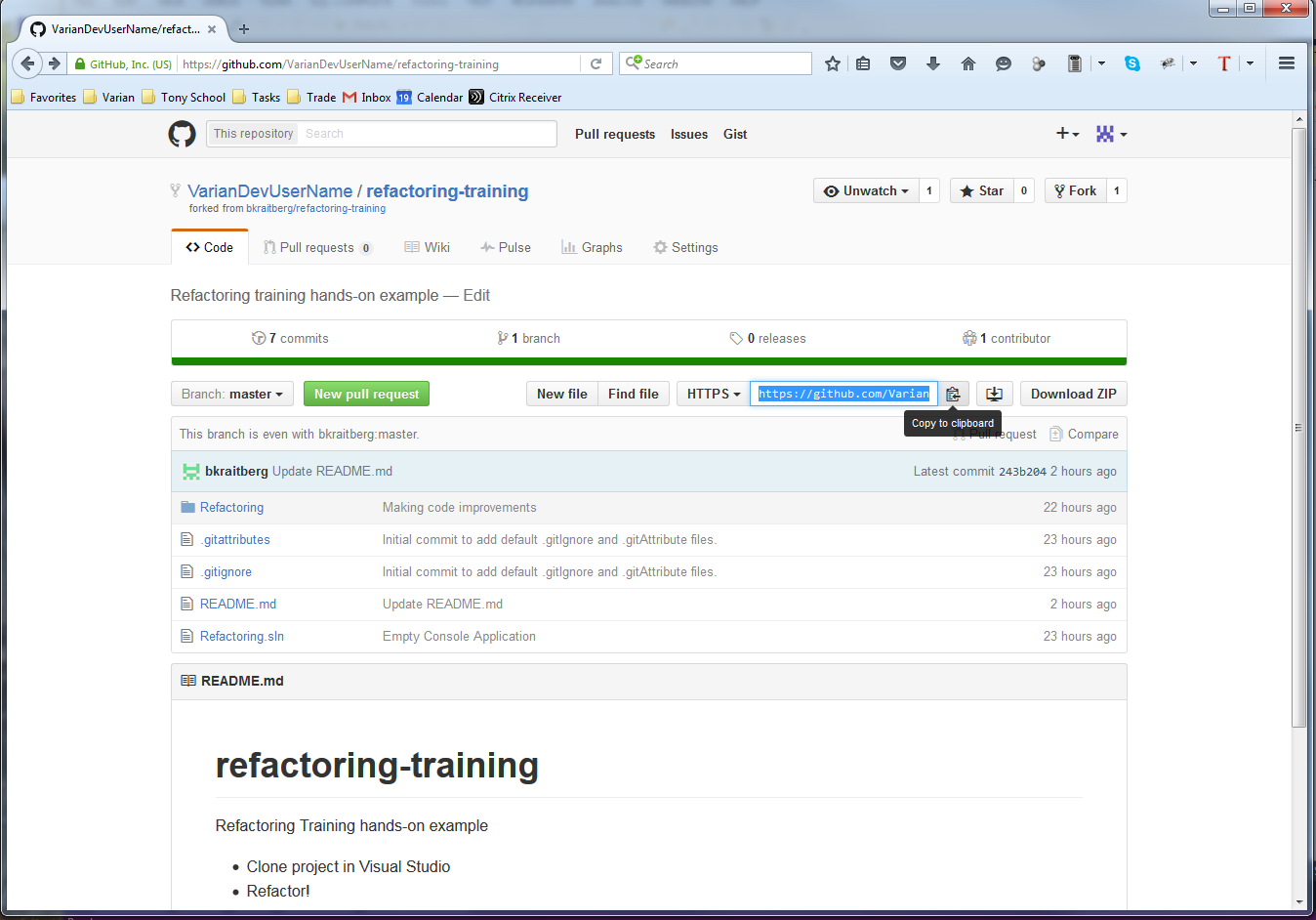
Clone the GitHub repository to your computer

1. You need to configure Visual Studio for Git instead of TFS.
2. Open Visual Studio (2013 in this example)
3. Click **Tools 🡪 Options**
4. Click **Source Control** inside dialog box
5. Select **Microsoft Git Provider**
6. Click **OK**

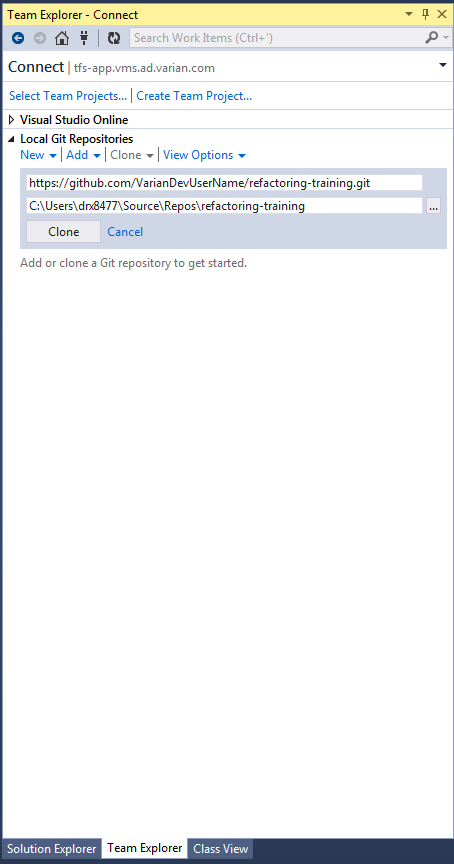


Note: Remember to set this back when you need to work with TFS again

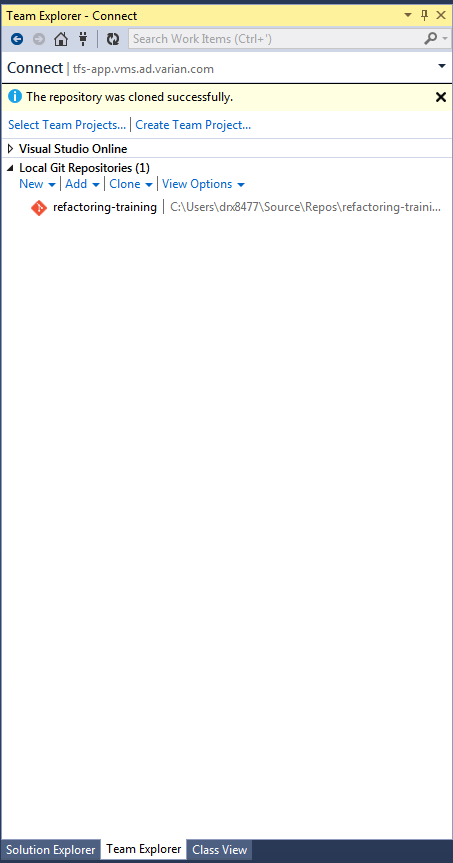
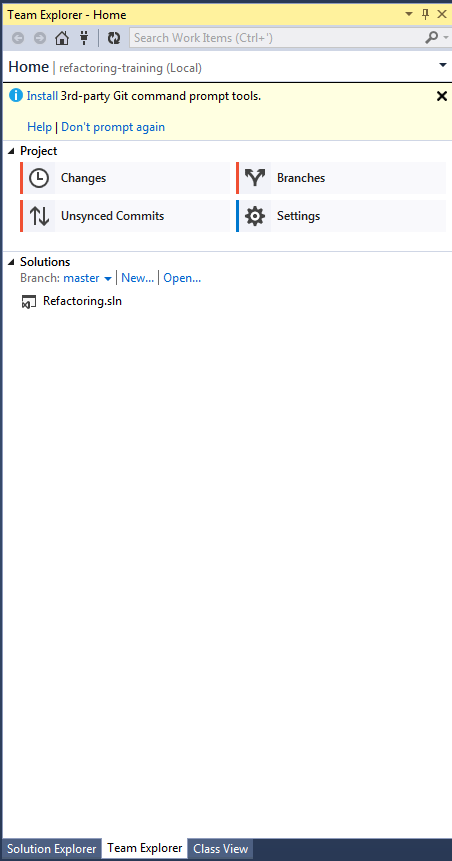
1. Go to your repository on **GitHub** and copy repository URL to clipboard.



1. In Visual Studio, open **Team Explorer**
2. Click **Connect**
3. Under **Local Git Repositories** click **Clone**
4. Paste the URL for you repo in the URL field
5. Enter a local folder for your repo
6. Click the **Clone** button

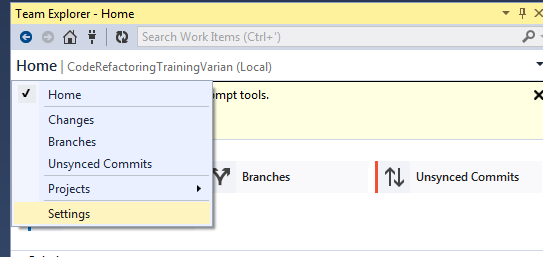


1. After successful cloning the project, you can open the project by following the below screenshots in order from left to right.
   1. Double click the repository to open it
   2. Double click the .sln file to open it
2. You’re now ready to work locally.

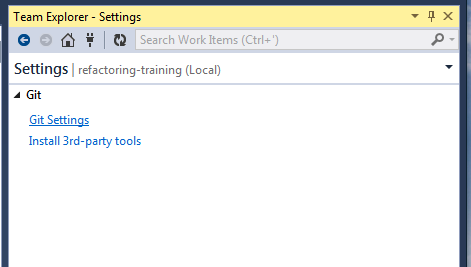


Modifying and committing local code to your remote repository.

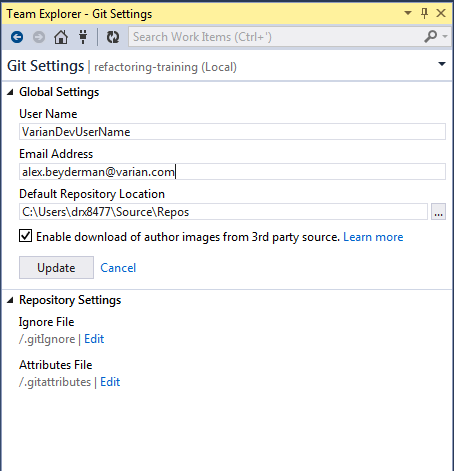
1. You need to configure Visual Studio to be able to push, pull and sync your local changes.
2. Under **Team Explorer** go to **Settings**



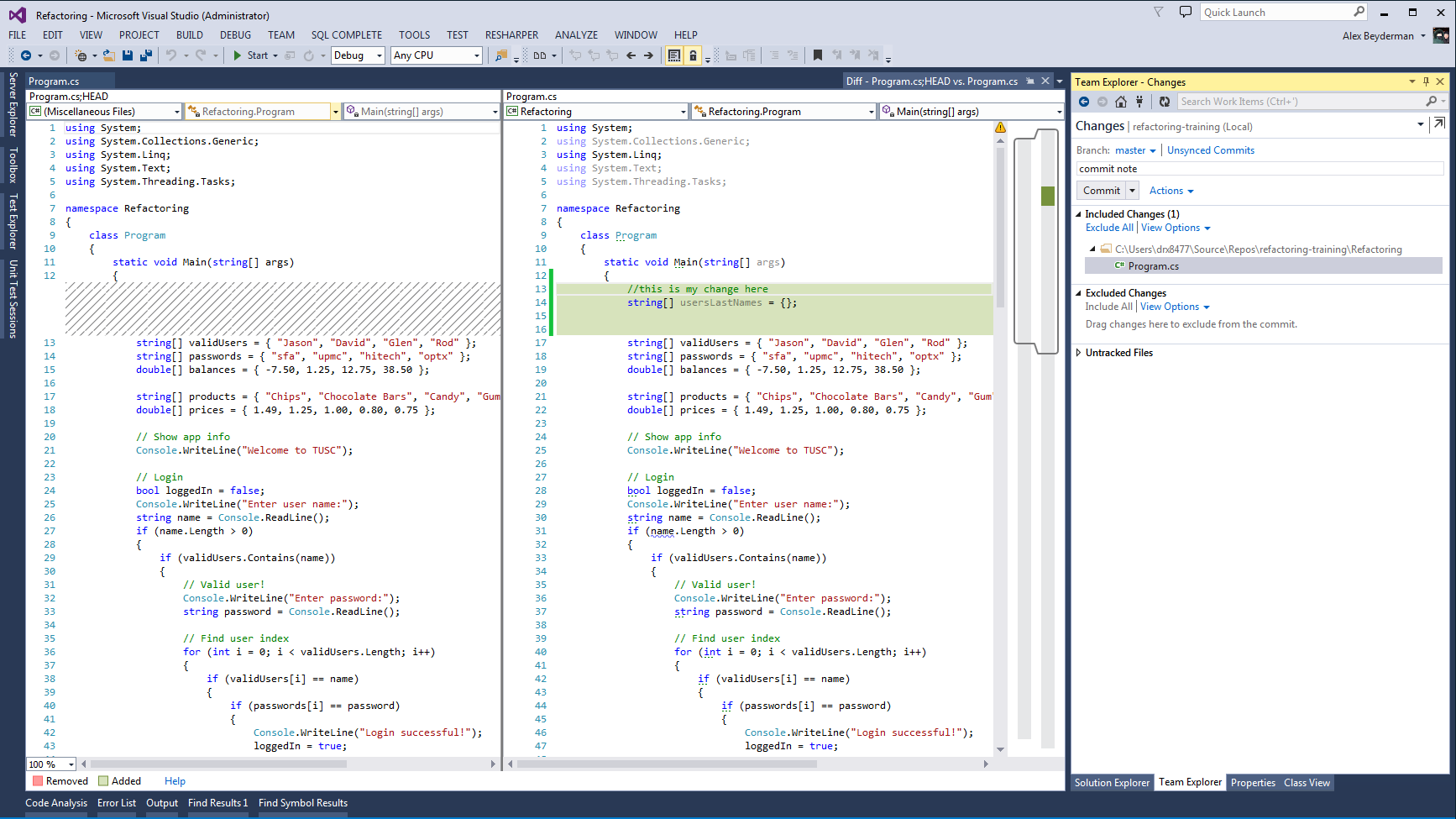
1. Click **Git Settings**



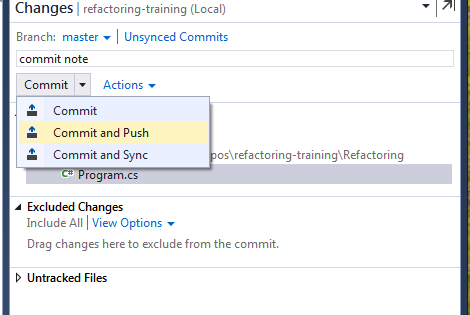
1. Enter your GitHub username and email address
2. Click **Update**
3. You will be prompted for your username and password during your first interaction with GitHub. The credentials will be saved under Windows Credentials Manager.



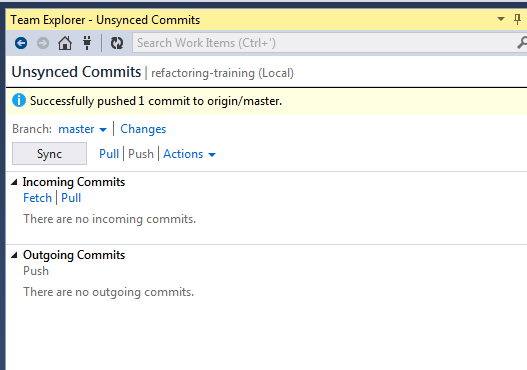
1. Edit the code



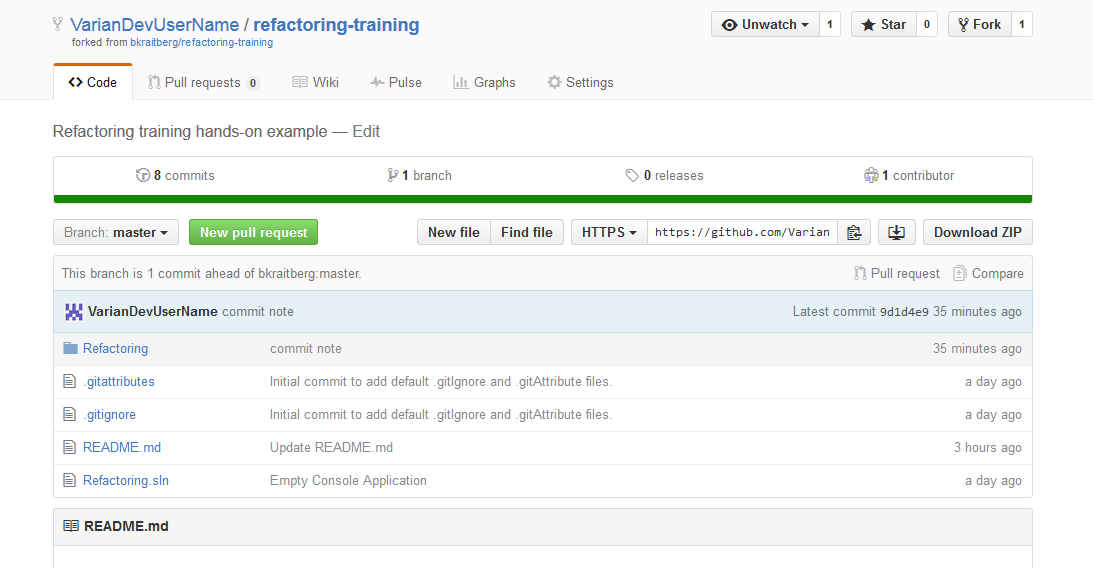
1. Go to **Changes** under Team Explorer
   1. Type your commit message.
   2. You have 3 choices:
      1. Commit - Commit your changes locally. These are not pushed to the server immediately.
      2. Commit and Push – Commit your changes locally and Push to GitHub server (Check In in TFS)
      3. Commit and Sync – Commit your changes locally and Pull all recent updates from GitHub Server (Check In + Get Latest version in TFS)



1. You’ll get a success message if no issues are found



1. You can verify your changes on GitHub by clicking the **commits** link



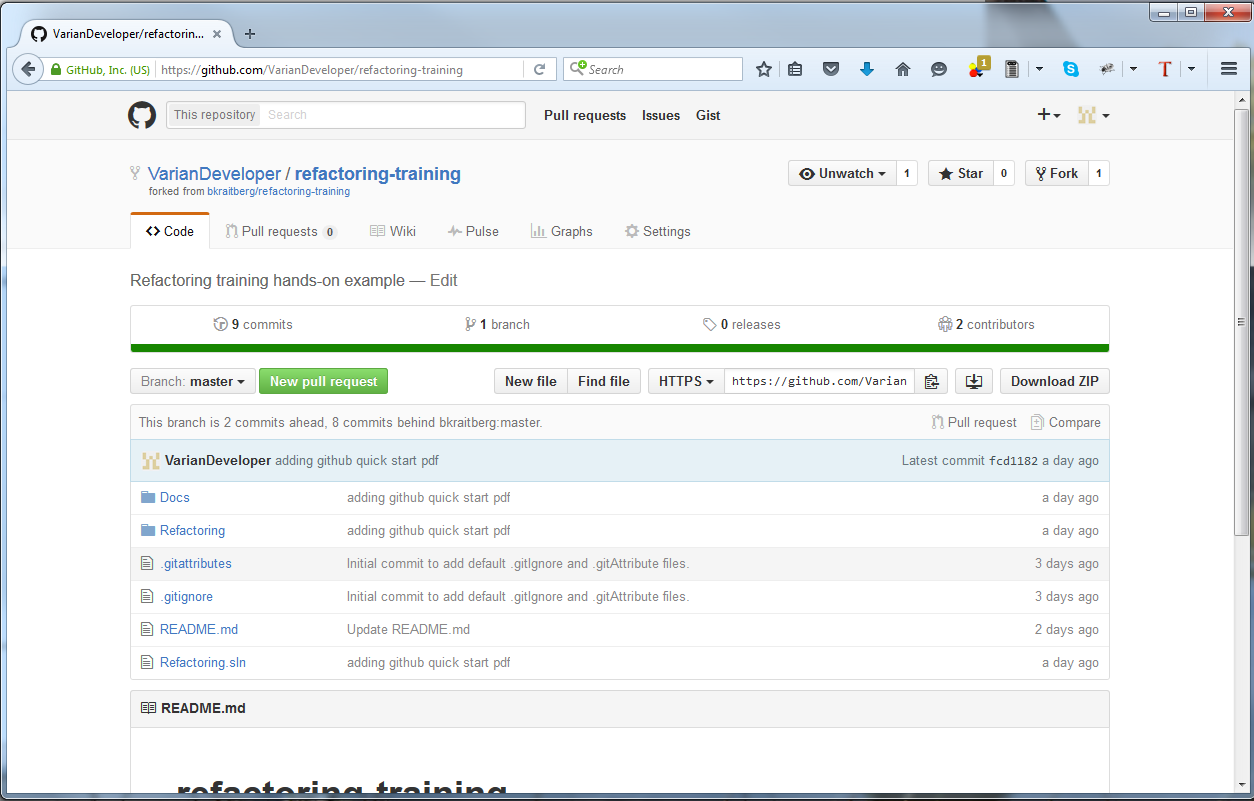
1. Remember, this is your repo. Feel free to commit and push to it as often as you like.

Collaboration in GitHub

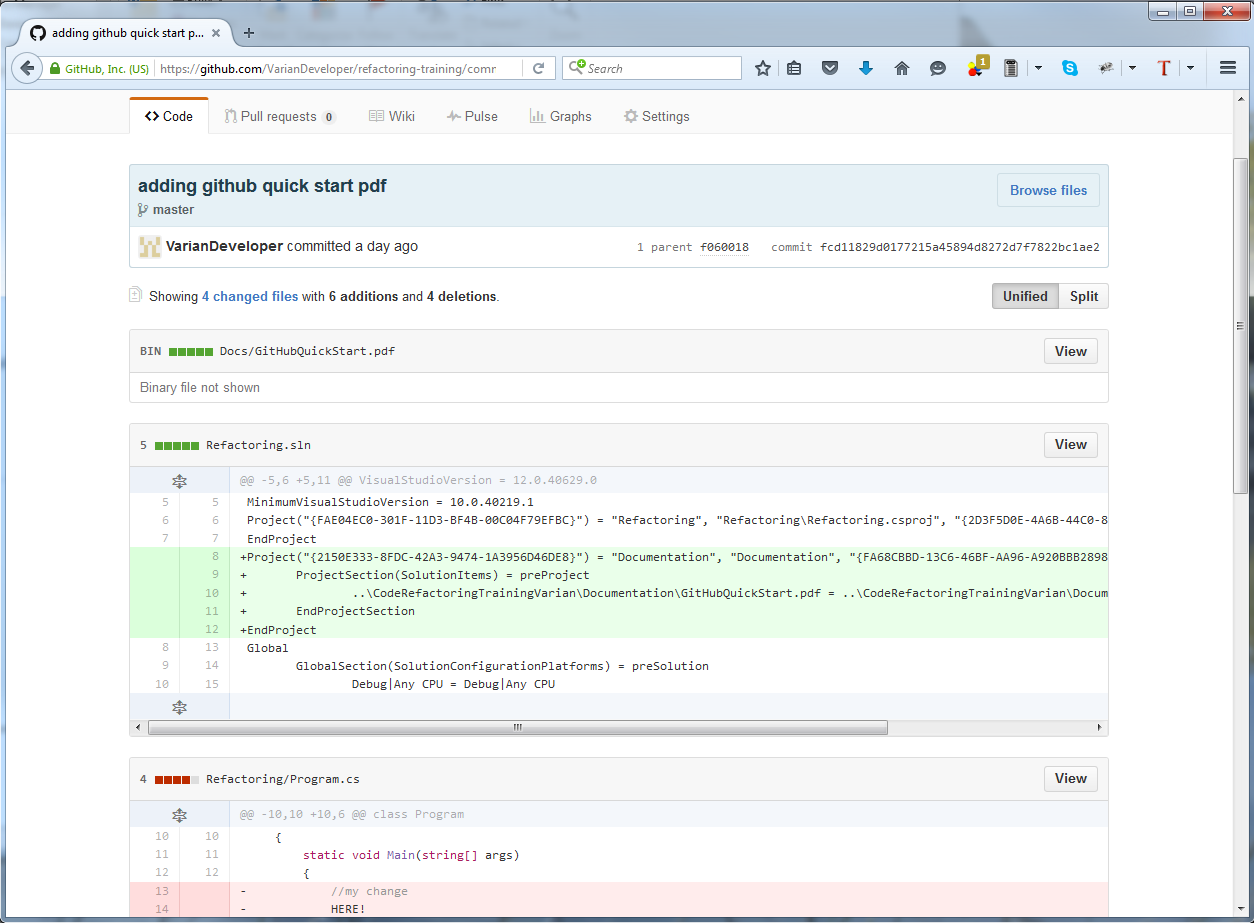
Now after you’ve finished your work, you need to push it back into the **original** repository (Bryan’s).

In GitHub we do this by creating **Pull Request**.

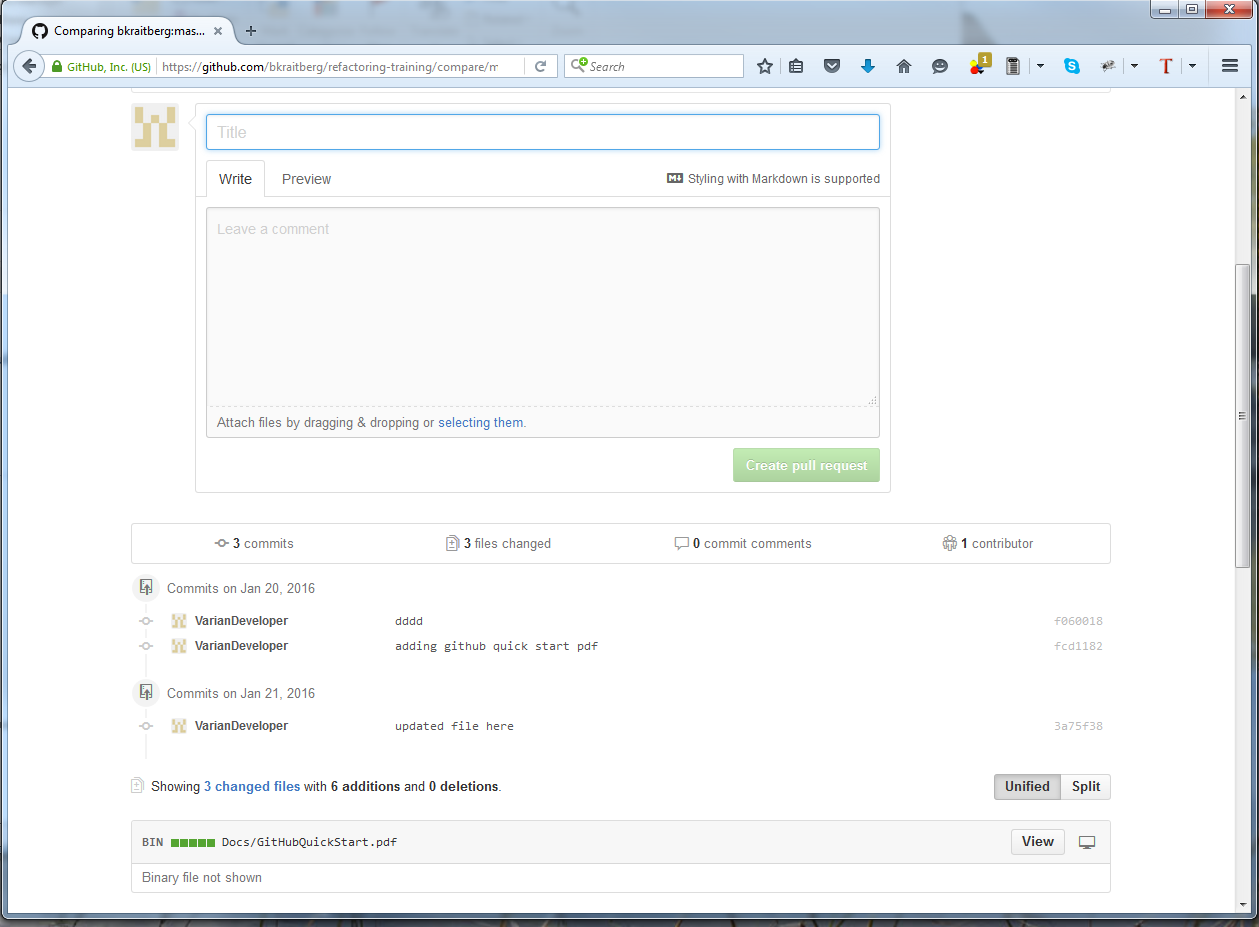
1. Make sure you’ve committed and pushed all of your changes.
2. Go to your repository on GitHub server.



1. Press the **New pull request** button.
2. You should see all of your commits.
3. Try changing the view by pressing the **Split** button. This will show changes in manner similar to TFS.



1. Go back to the previous page.
2. You’ll see that by default your master branch is going to merge with the original master branch.
3. Fill in the details and press the **Create pull request** button.
4. The pull request message will be sent to the original repository owner (Bryan in this case).



Additional resources

1. [GitHub Guides](https://guides.github.com/) – includes examples, tutorials and video guides.
2. [GitHub with VS2013](http://incyclesoftware.com/2014/03/using-git-team-foundation-server-visual-studio-2013/) – working with Git in Visual Studio 2013.