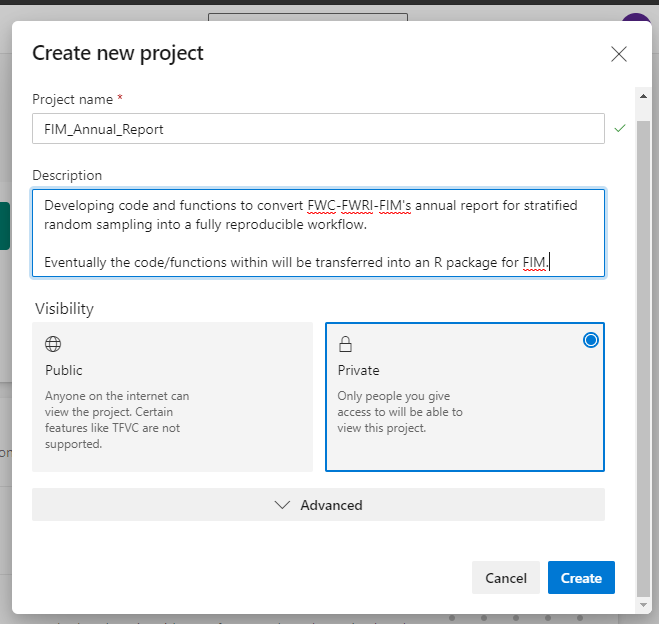
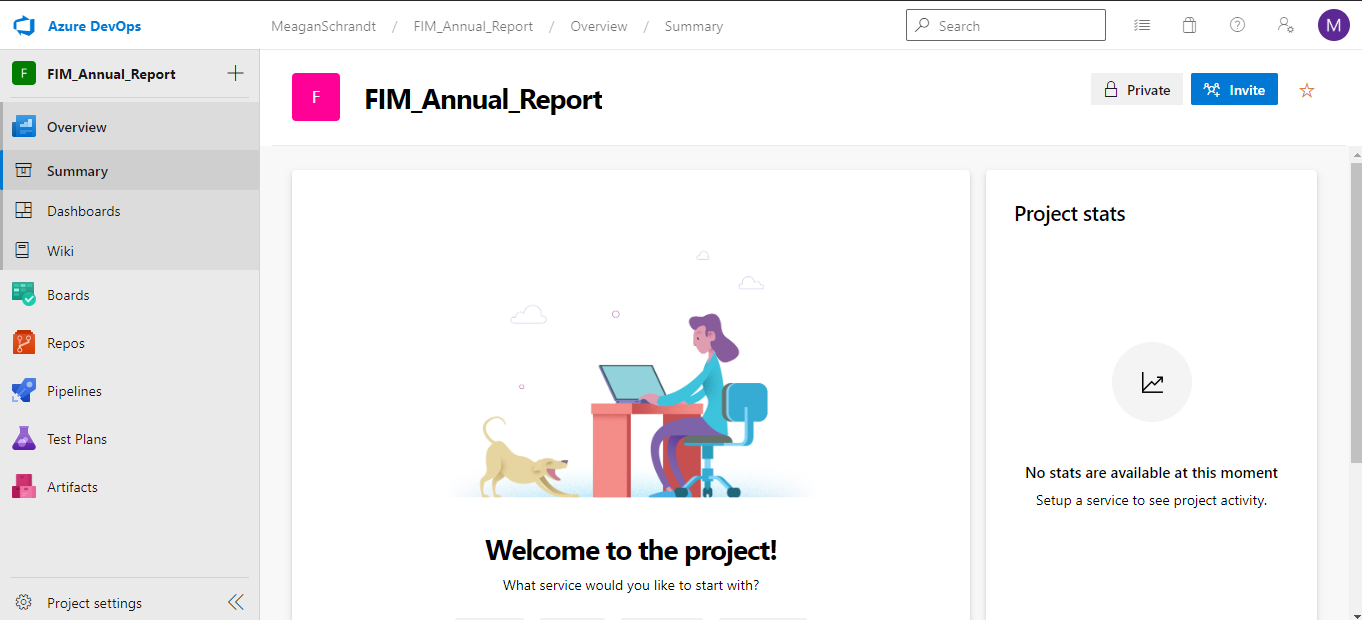
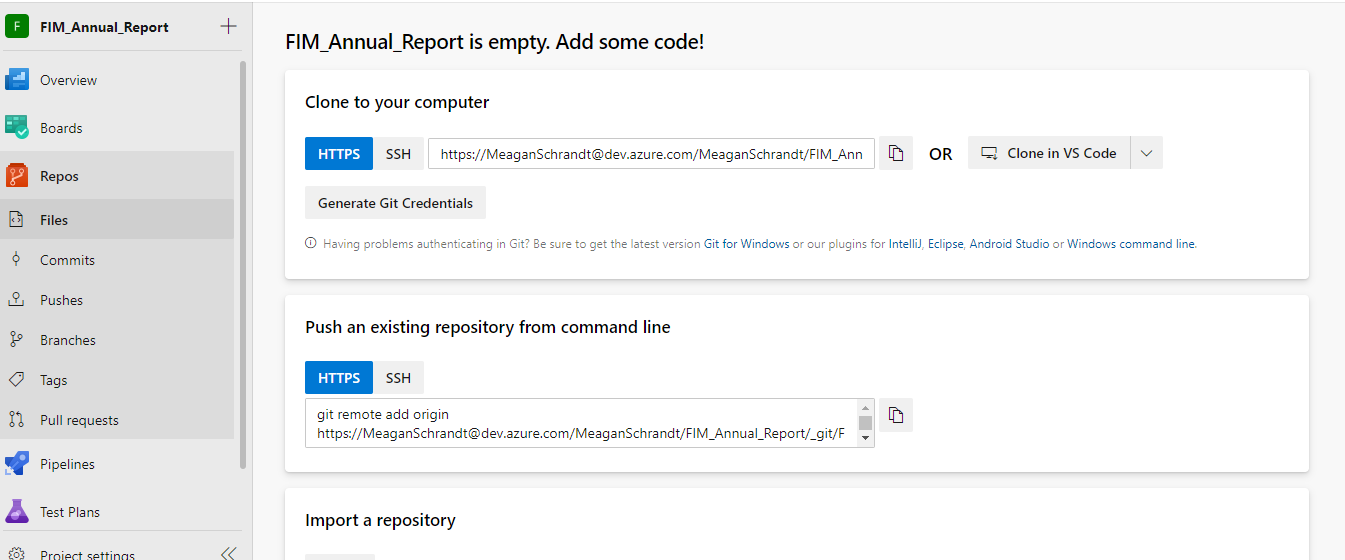
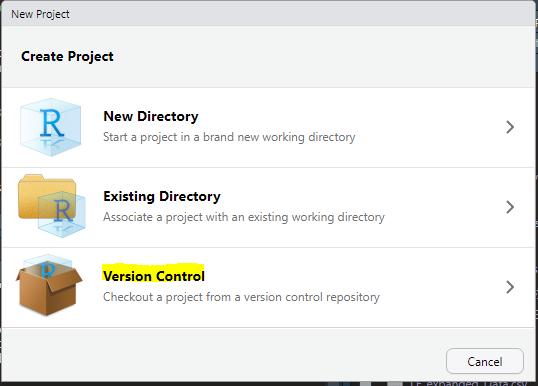
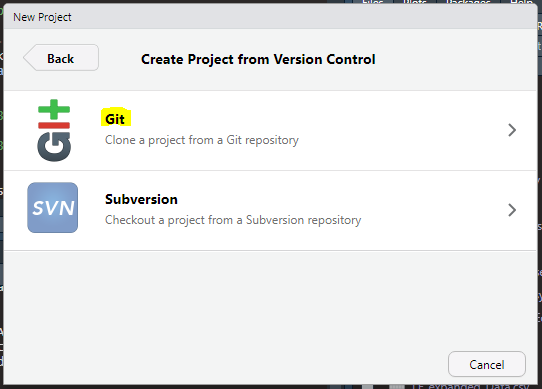
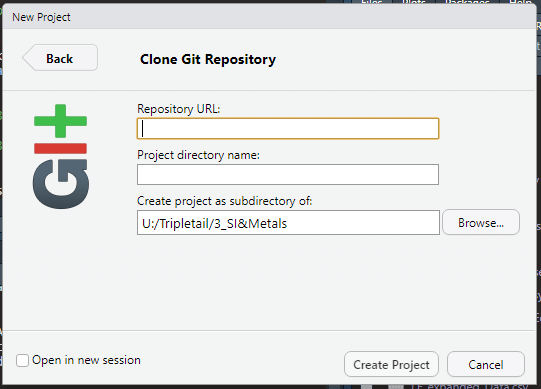
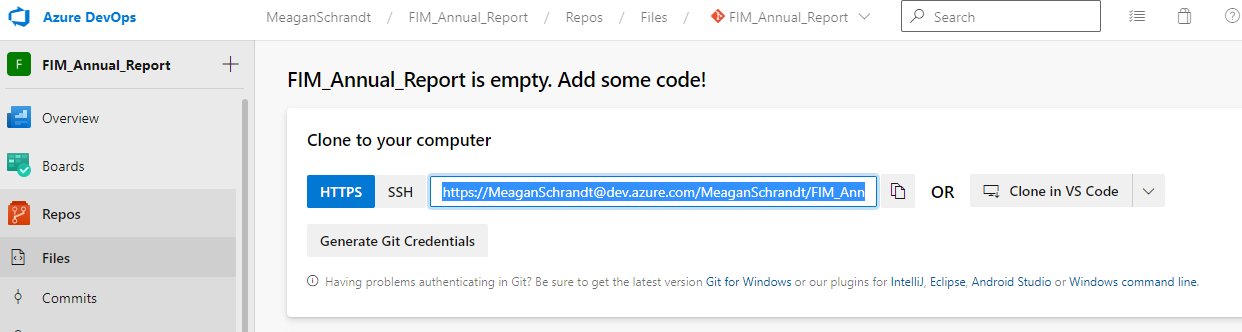
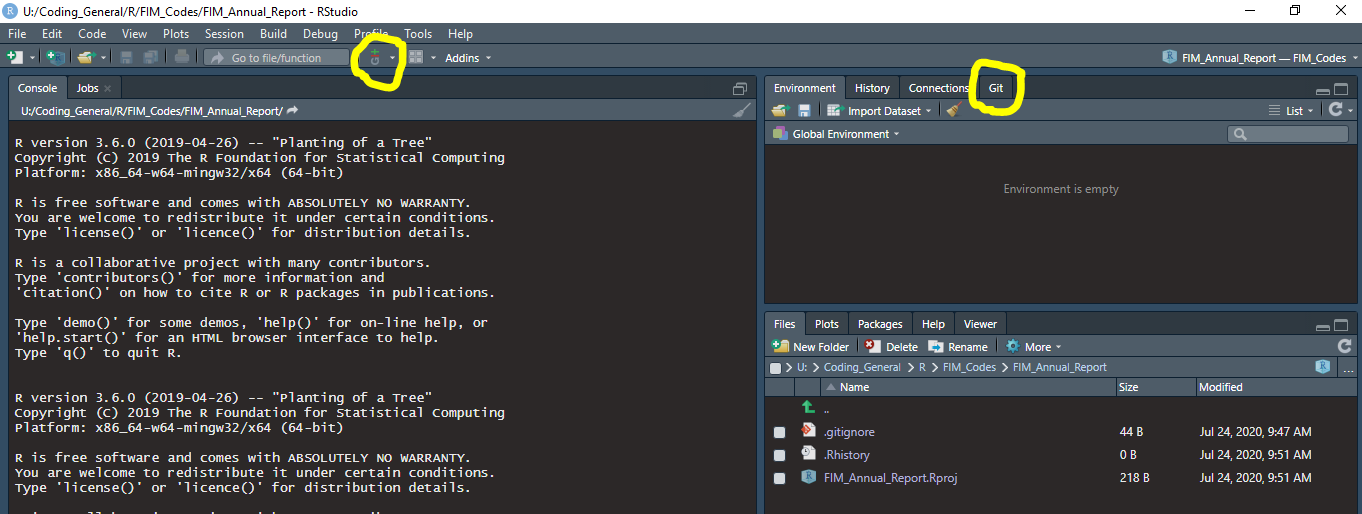
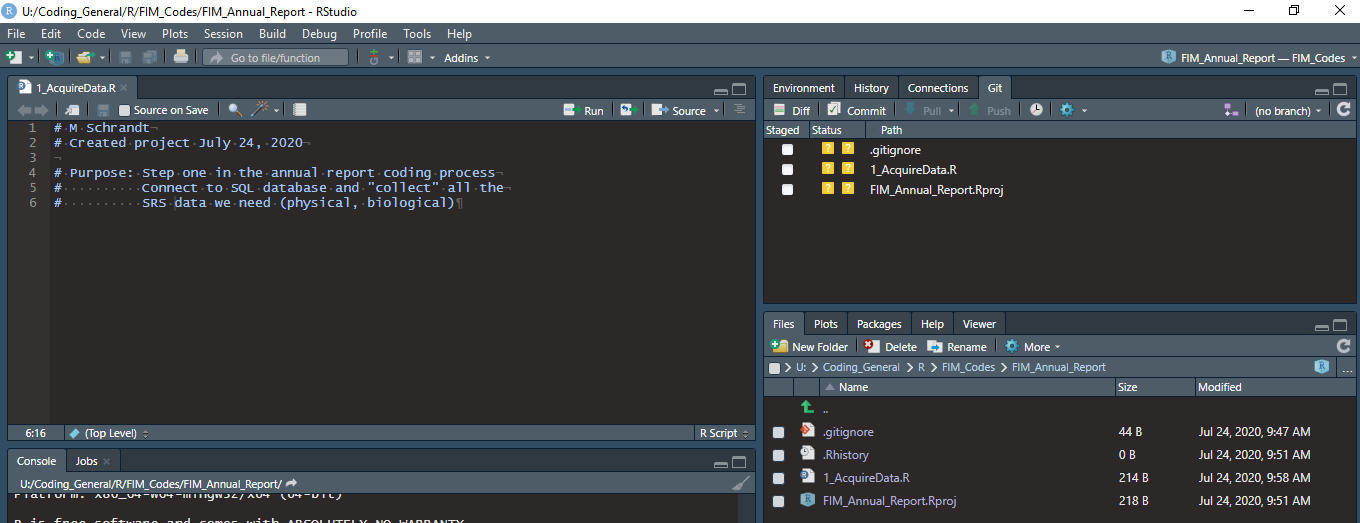
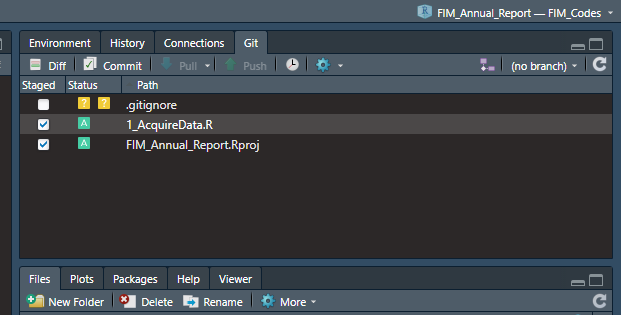
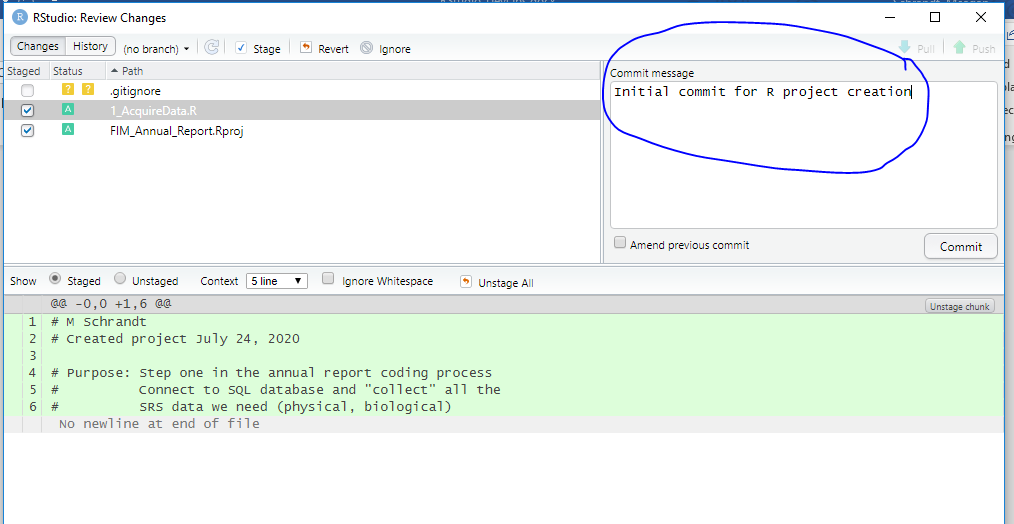
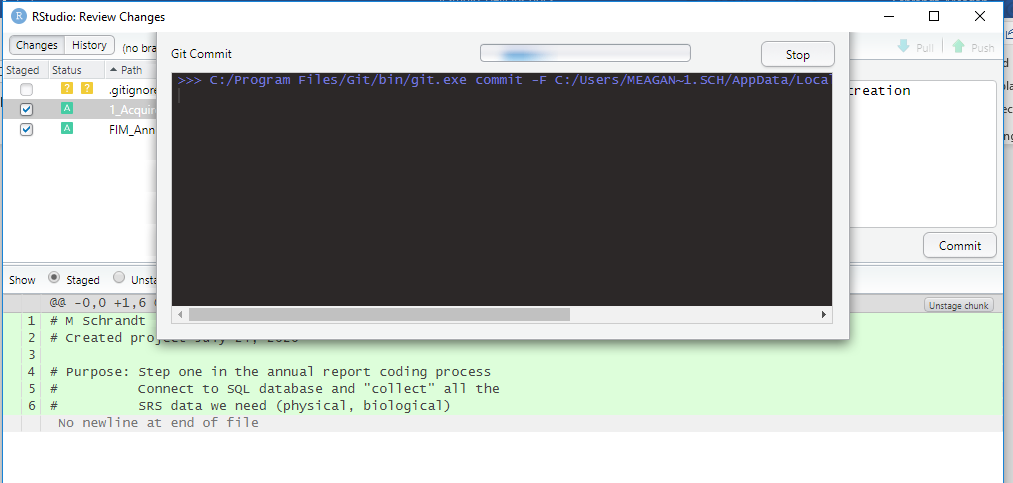
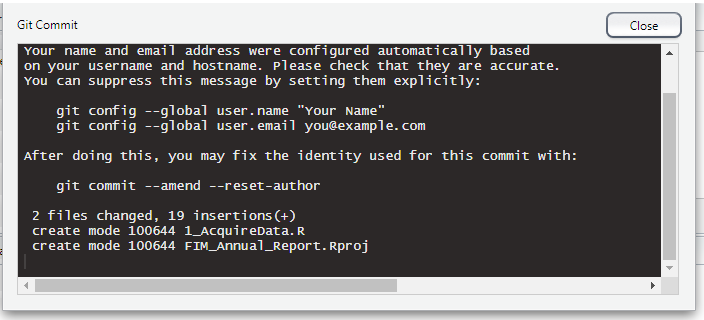
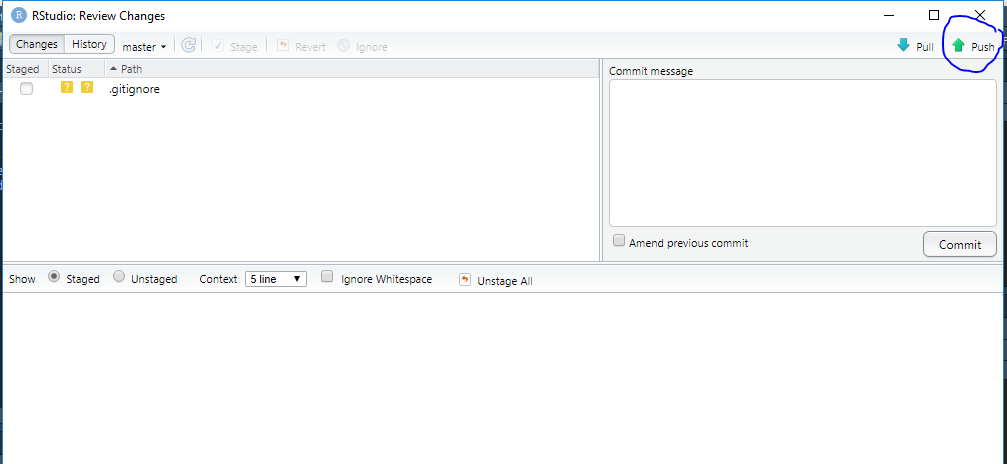
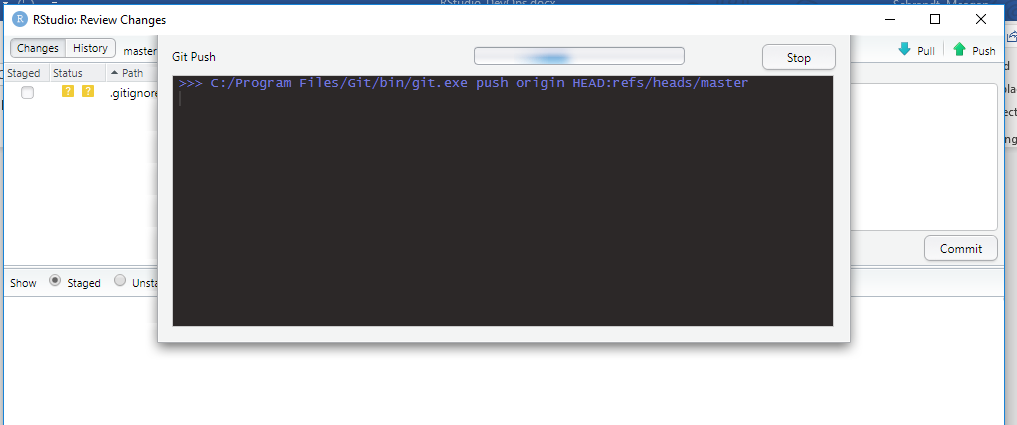
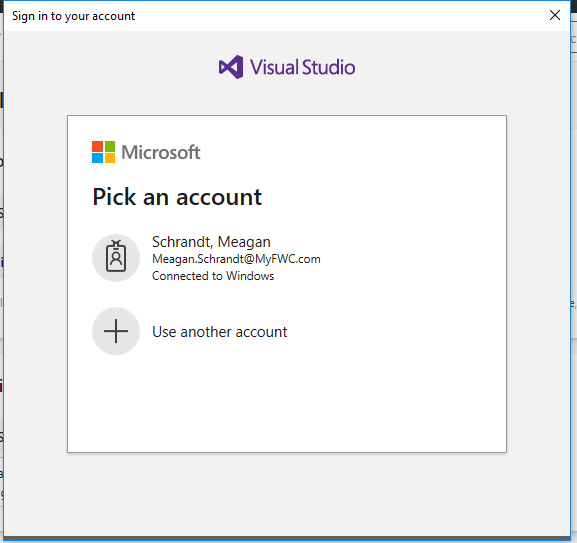
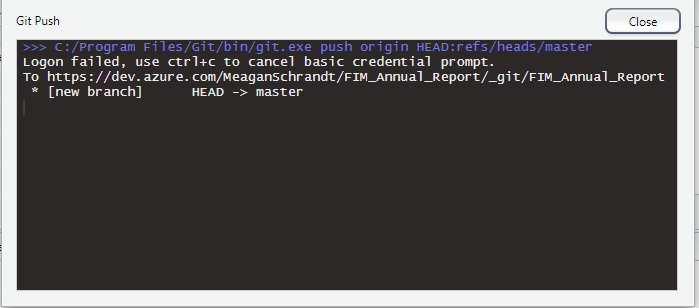
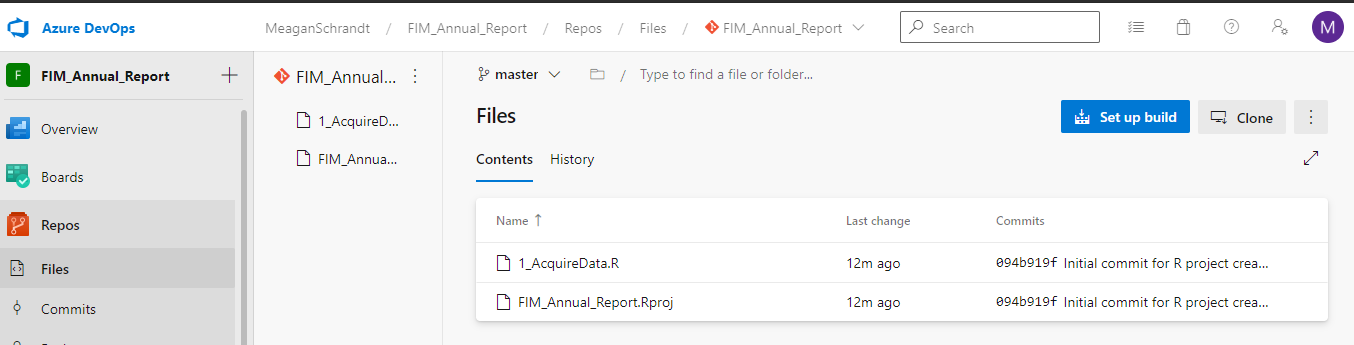
\*Before you can use RStudio and DevOps together, you need to do the following:

* Install R & RStudio
* Install Git (<https://git-scm.com/>)
* Set up your Azure DevOps account with your FWC computer username and password

\*Now you can begin with version control and RStudio:

1. Starting a New Project – start with DevOps first!
   1. Sign into DevOps
   2. Click the “+ Project” button in the top right corner
   3. Fill in a project name (no spaces in names) and a project description
      1. 
      2. Select whether the project is public or private (you can give others access to it after creation)
      3. Click “Create” button in bottom right corner
   4. This should automatically direct you to the project’s (empty) page:
   5. 
   6. Select “Repos” from the table of contents on the left side of the page (see image above)
   7. This opens the “Repository” for the project – this is your “folder” where you will be adding files … but you’ll do it from RStudio
   8. 
   9. Our next step is to “Clone” the repository to our computer (via RStudio)
   10. Leave your DevOps account open/signed in and Open RStudio
       1. Start a New Project via File > New Project > Version Control:
       2. 
       3. Choose “Git”:
       4. 
       5. Fill in the repository information:
       6. 
          1. The Repository URL is the one from AzureDevOps given under “Clone to your computer” (use the HTTPS one given for your project)
          2. 
          3. Once you paste in the URL, it will automatically populate with the Project directory name from DevOps
          4. Choose what directory you want the Rproject to be housed in on your computer (it will automatically make a folder with the project name so you do not need to create a folder first)
          5. Click “Create Project”
       7. You’ll get a pop up to select your Microsoft Office 365 account that you want to use
          1. You might get a warning that says your logon failed and/or that you cloned an empty repository – both are OK as long as you are still signed into your DevOps account
       8. When your project opens, you’ll know it’s been associated with version control when you can see the Git icon and the Git tab (circled in yellow below):
       9. 
   11. If you are having issues with the Git part within RStudio, and you are sure you installed Git, try going to Tools > Global Options > Git/SVN
       1. Check the box for “Enable version control interface for RStudio projects”
       2. If necessary, browse for where the Git “executable” is located on your computer; for my FWC computer it is C:/Program Files/Git/bin/git.exe
       3. *Restart RStudio*
       4. Re-open your project in RStudio and go to Tools > Version Control > Project Setup
       5. Select “Git” as your version control system
   12. Make your first “Commit” (change/update) and “Push” (save/transfer) it to DevOps
       1. Start a new script and save it with a name of your choice
       2. You should now see this saved script in the Files pane and also in the Git tab of the Environment pane (note: I had to hit the “refresh” button in the Git tab in order to see the new script show up here
       3. For example, I started a script called “1\_AcquireData”:
       4. 
       5. In your Git pane, you will see any “changed” files from the DevOps version of the project. That’s why we also see the .Rproj file.
       6. To make a commit, select the files from the Git tab that you want to commit to DevOps. Do this by checking the box next to them. It might take a little while for this to register, depending on your internet speed.
          1. I’m going to select my .Rproj file and my .R script file to send to DevOps
          2. 
          3. When they have a check mark and their status is now changed to a green box with an “A”, they are ready to be committed.
          4. Click on the “Commit” button in the Git tab
          5. This will open an additional interface to perform the Commit and subsequent Push
             1. Put a short, informative commit text in the box on the right:
             2. 
             3. Once you’ve entered your message and confirmed that the appropriate files are checked in the top left pane, click the “Commit” button
             4. You should then see a command prompt window open:
             5. 
             6. Eventually it will give you the results of your commit:
             7. 
             8. Click the “Close” button
          6. Now you are ready to “Push” that commit to DevOps
             1. When the commit has completed, the window should clear out and the “Push” button should become active.
             2. Click the “Push” button at the top right corner:
             3. 
             4. This might take a minute, but you should see the command prompt for Git push:
             5. 
             6. You might get a pop up to pick your MS 365 account:
             7. 
             8. You might also get a warning that the logon failed; in my experience, everything still works fine as long as a second message appears afterward with your DevOps project URL
             9. 
             10. Now you can click “Close”
   13. Confirm that you Commit & Push went through to DevOps
       1. Go back to your DevOps web page and click the refresh icon.
       2. Now instead of the URL page for your repository, you should see the files that have been added to your repository:
       3. 
   14. Now, every time you make changes/updates to your R code and associated project, you can re-do the commit & push process. You can select which files from your R project that you want to commit and push each time. DevOps will keep track of all these changes, line by line.