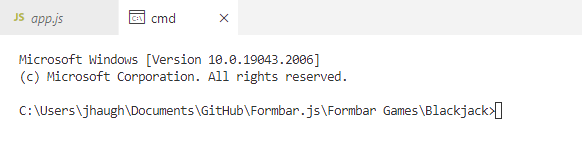
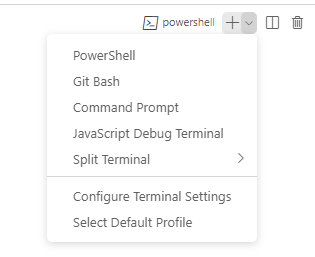
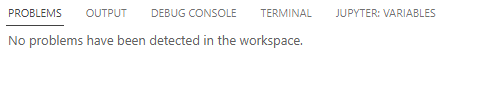
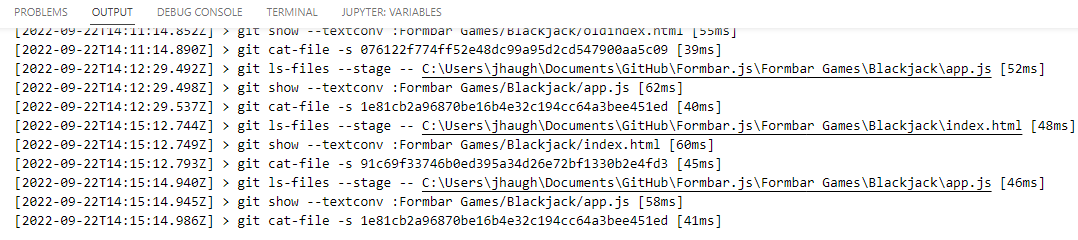
I would say one of the biggest features to my knowledge that Atom didn’t have was the built-in terminal. (Shown Below)

This allows you to test files and websites in the application without having to open a command prompt or terminal and locate the file to run it. It also allows for the option to switch from PowerShell (The Basic Terminal Selected) to either Command Prompt, Git-Bash, and the JavaScript Debug Terminal.

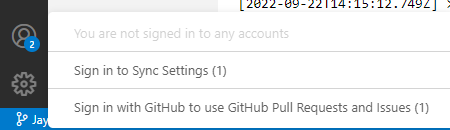
(🡨 Shown Here)

You can also have multiple terminals open at once to test multiple files. It also allows you to test the files and will output

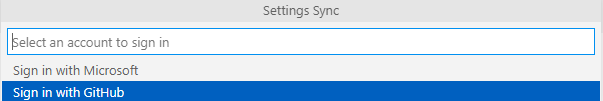
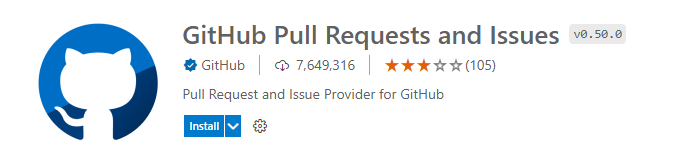
The terminal that visual studio code has will work just like a regular command prompt, PowerShell, or Git Bash. It has all the regular commands and such that the normal prompt would use. It runs locally in the directory so you don’t have to locate the file you want to work on. So long as you select command prompt, PowerShell, or Git Bash, all their local commands will work as if it was the actual thing.

The terminal page in VSCode also has a tab for problems with the code when you run it, They have a output tab as well. The output page constantly updates whenever you do any action at all in VSCode so it works as an active history. The problems tab actively updates whenever a problem is detected in the workspace. It does this so that if there are any problems it will notify you and will tell you exactly where it is so that you can get to work on fixing it. It will also tell you how many milisconds it took to do said action.

A close second I would give to Visual Studio Code would be their GitHub integration. VSCode makes it so that you can sign in with your GitHub account so that it can pick up what repository your in, with help of a extension (Image 4) you can also push and fetch the data for the repository you are currently working in so that you don’t have to load up GitHub Desktop to Fetch, Push, or Pull, new work.

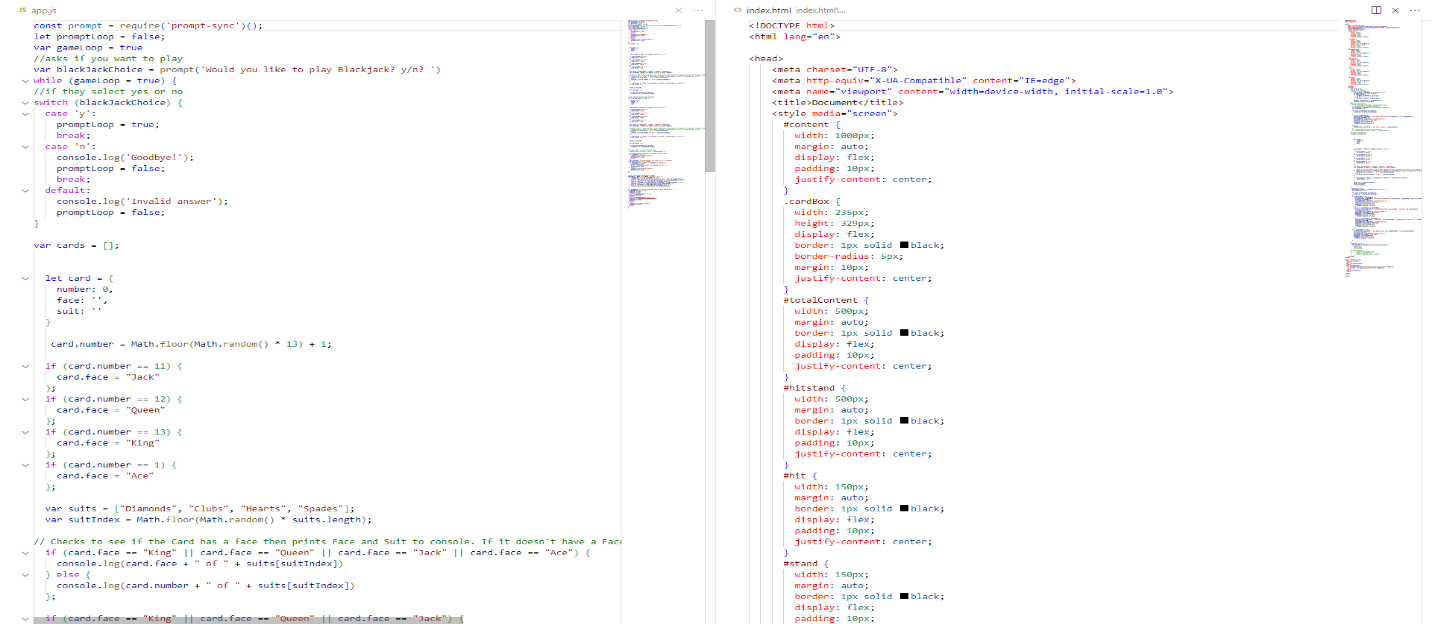
In the bottom left hand side of the screen is a profile icon that if you click on will show this.

(🡨Shown Here)

When you click sign in to settings sync it will come up with an option to sign in with either Microsoft or GitHub. For this you would want to click the sign in with GitHub but you can sign in with Microsoft. This will make it so that whenever you sign in with GitHub or Microsoft the settings you set on one computer for VSCode will sync to all other computers that use VSCode that are signed in with your Microsoft or GitHub account. Once you are signed in with GitHub you can go to the Extensions tab on the left that will look like this. (^ Shown Here). Once in that tab you will search for “GitHub Pull Requests and Issues”, It will look like this (Shown to the Left) Once installed and you have re-launched VSCode you will see the “Sign in with GitHub to use GitHub Pull Requests and Issues (1)” (Shown in the 1st Image). Once signed in the GitHub logo will appear on the left bar and will look like this 🡪

Once you click on that, it may have you sign in again if you haven’t already, but once you have it will show all Pull Requests and Issues that are currently open for the repository that you are currently working in on GitHub. If you are working on a fork it will show all the Issues open on the main repository since GitHub Forks can not have their own issues. But when working on a repository it will show all the issues you have created as long as you are working in the right folder/repository.

If you select a pull request in the tab it will also allow you to view what was changed. If any file was changed in the repository it will show where and what was changed. This will allow you if you are a repository owner to view the code and make changes as you see fit before you merge it into the main repository.

The third feature I like to personally use sometimes is Zen Mode. Zen Mode lets you focus on your code by hiding all UI except the editor (no Activity Bar, Status Bar, Sidebar and Panel) and going to full screen. Zen Mode can be accessed by pressing Ctrl-K then pressing Z. You can then exit Zen Mode by pressing the Esc key twice. Basically what Zen Mode allows you to do is focus on the code as it full screens the code editor and hides every other thing on the screen. This makes it so that it limits the clutter of everything else as it can be distracting to some. The only thing that is kept open when you start Zen Mode is the code editor, the file you had open, the scroll bar and the Mini-Map, this makes it so that the only thing that is open is the bare necessities. Entering Zen Mode also hides the taskbar as it has essentially fullscreened the application. What they also do is that you can have 2 files open at the same time which a good code editor can normally do but with Zen Mode it makes it so that its just those 2 files and the Mini-Maps and the scrollbars open and nothing else to clutter the screen. Which as a base will look like this.