**IBM Data Science Coursera Certificate – Tools for Data Science (Course 2)**

**Week 5 – R and GitHub**

**Notes:**

**RStudio**

**GitHub**

1. GitHub is one of the popular environments among developers and data scientists for performing version control of source code files and projects and collaborating with others.
2. GitHub is an Internet hosting service for software development and version control using Git. It provides the **distributed version control** of Git plus access control, bug tracking, software feature requests, task management, **continuous integration**, and wikis for every project.
3. **Version Control :** A version control system allows you to keep track of changes to your documents. This makes it easy for you to recover older versions of your document if you make a mistake, and it makes collaboration with others much easier.
4. Git is free and open source software distributed under the GNU General Public License.
5. Git is a distributed version control system, which means that users anywhere in the world can have a copy of your project on their own computer. When they’ve made changes, they can sync their version to a remote server to share it with you.
6. Git isn’t the only version control system out there, but the **distributed aspect** is one of the main reasons it’s become one of the most common version control systems available.
7. **You can use Git without a web interface by using your command line interface, but GitHub is one of the most popular web-hosted services for Git repositories.**
8. Others include GitLab, BitBucket, and Beanstalk.

**Important Terms needed for GitHub Stuff:**

1. **The SSH protocol** is a method for secure remote login from one computer to another. SSH, also known as Secure Shell or Secure Socket Shell, is a network protocol that gives users, particularly system administrators, a secure way to access a computer over an unsecured network
2. A **repository** contains your project folders that are set up for version control.
3. **A fork** is a copy of a repository.
4. A **pull request** is the way you request that someone reviews and approves your changes before they become final.
5. A **working directory** contains the files and subdirectories on your computer that are associated with a Git repository.

**Important Commands for GitHub Stuffs:**

1. **“git init”**: It helps in cloning an existing repository by using the command "git init" or helps you in creating new repositories for only once.
2. **“git add”:** moves changes from the working directories to the staging data area.
3. **"git status":** allows you to see the state of your working directory and the staged snapshot of your changes.
4. **"git commit":** takes your staged snapshot of changes and commits them to the project.
5. **"git reset":** undo the changes that you’ve made to the files in your working directory.
6. **"git log":** enables you to browse previous changes to a project.
7. **"git branch":** lets you create an isolated environment within your repository to make changes.
8. **"git checkout":** lets you see and change existing branches.
9. **"git merge":** lets you put everything back together again.

**Fundamental Introduction to GitHub**

1. Large software projects need a way to track and control source code updates.
2. Git is a distributed version-control system that is used to **track changes to content**.
3. It serves as a **central point for collaboration** with a particular focus on agile development methodologies.
4. It allows for a centralized point for collaboration.
5. In a central version control system, every developer needs to check out code from the central system and commit back into it.
6. As Git is a distributed version control, each developer has a local copy of the full development history, and changes are copied from one such repository to another.
7. It allows teams to have controlled access scope.
8. The main branch should always correspond to deployable code.
9. Git can co-exist locally such as through the GitHub Desktop client or it can be used directly through a browser connected to the GitHub web interface.
10. IBM Cloud is based on sound and established open-source tools including **Git repositories**, often called **repos**.
11. GitHub is an online hosting service for Git repositories. GitHub is hosted by a subsidiary of Microsoft. GitHub offers free, professional and enterprise accounts.

**What is a Repository?**

***A repository is a data structure for storing documents including application source code.***

**GitLab** provides access to Git repositories, controlled by **source code management**.

**GitHub Repositories**

The heart of a Git-based project is the repository.

This contains all your code and the related artifacts, including things like: A README file to describe the purpose of the project. A license to express the ways in which people can use your code, Etc.

You can also make your repository private (only available to people with accounts that have permission to see it) or public (searchable and seen by everyone).

**There are 8 parts of a GitHub repositories:**

1. **Code:** this is where all the source files reside. Git was initially created as a source code repository and now all sorts of files end up in here.
2. **Issues:** you can track and plan with tools such as “Issues” that lists all open items against your project base.
3. **Pull Requests** – this is part of the mechanism for collaborating with other users. Pull requests define changes that are committed and ready for review before being merged into the main branch.
4. **Projects:** all the tools for managing, sorting, planning, etc. your various projects.
5. **Wiki:** this tool provides a communication base to the external user community.
6. **Security:** this tool provides a communication base to the external user community.
7. **Insights:** this tool provides a communication base to the external user community.
8. **Settings:** GitHub allows for a lot of personalization, including changing the name of your repository and controlling access.

**GitHub Branches**

**Just for my knowledge** : **GitHub Actions** uses YAML syntax to define the workflow. Each workflow is stored as a separate YAML file in your code repository, in a directory named .github/workflows . You can create an example workflow in your repository that automatically triggers a series of commands whenever code is pushed.

**GitHub Branch :** A branch is a snapshot of your repository to which you can make changes. It is a copy of the master branch and can be used to develop and test workflow changes before merging it into the master branch.

**Master Branch:** In Git and GitHub, there is a main branch called **master.** It has the deployable code and is the official working version of your project. It is meant to be stable, thus, it is advisable not to push any code that has not been tested in the master.

If you want to change the code and the workflow in the master branch, you can create a copy of the master branch. This can be the child branch that will be a copy of the workflow.

In the child branch, changes and experiments are done. You can build, make edits, test the changes, and when you are satisfied with them, you can merge them back to the master branch, where you can prepare the model for deployment.

Child branch creates isolation from the master branch which won’t be affected unless you commit the changes directly into master. Child branch helps in creating commits, changes, edits and such without affecting master branch. Once you are satisfied with your work in child branch, you can merge your commit/changes into the master.

To ensure that changes done by one member, do not impede or affect the workflow of other members, multiple branches can be created and merged appropriately with the master after the workflow is properly tested and approved.

**Pull Requests :** Pull requests are a way of proposing changes to the main branch (master).

It can notify other team members of the changes and edits to the main branch. Ideally, another team member reviews the changes and approves them to be merged with the Master branch. Pull requests are a means of collaboration on GitHub. When you open a pull request, you propose your changes.

**Important Stuffs about GitHub Branches**

Github Branch: All files in GitHub are stored on a branch.

Commit indicates that the developer is convinced that the code represents a stable platform for the feature or set of features being developed.

When a developer commits changed source to their path, they are required to write a comment that describes the changes.

The comment should be meaningful and descriptive. The developer can choose to commit to the current branch or create a new branch.

Some best practices : Don’t end the commit message with a period. Keep commit messages under 50 characters – use the extended window for the details. Always write in an “active” voice.

Pull is used to initiate the merging of branches in a way to capture changes.