Special Topics in Data Science

Introduction to Git and GitHub

Outline

- Introduction to GitHub and Git Bash
- Git fundamentals and version control basics
- Installing Git and configuring user settings
- Working with remote repositories and GitHub use cases
- Collaborative workflows, branching, and pull requests

Outline Breakdown

1. Introduction to GitHub and Git Bash
2. Version Control Basics
3. **Git and Git Bash**
4. Installing Git
5. GitHub and GitHub use cases
6. Create GitHub Account
7. Using GitHub
8. Create GitHub Account
9. **Using GitHub and creating a GitHub Profile**
10. Markdown
11. Navigating GitHub
12. Git Configuration
13. Getting Started with Git Bash
14. **Git and GitHub Activity**

Introduction

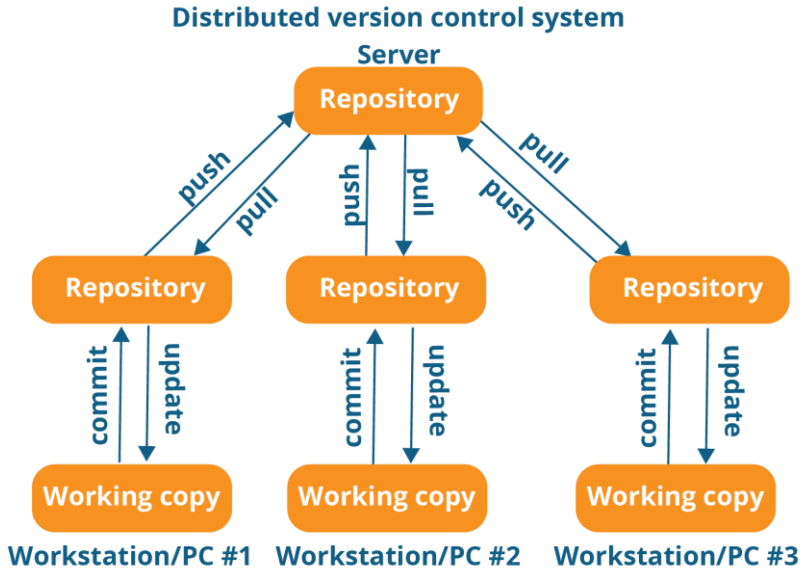
- How do you keep track of data projects or code?
- Where do you store your data projects or code?
- How do you collaborate on data projects or code projects?

Version Control Basics

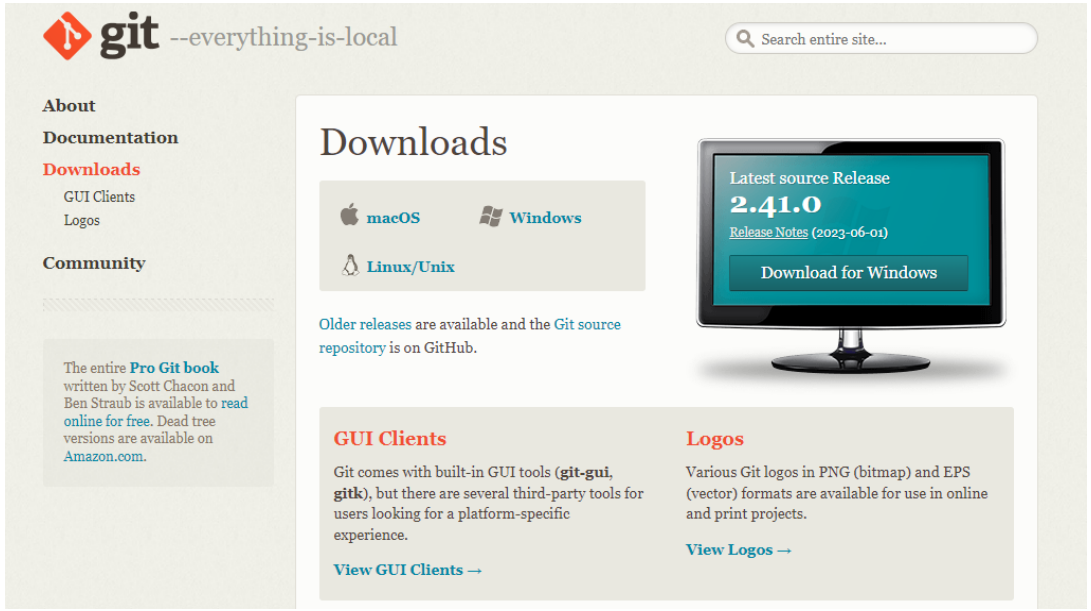
- **Version control:** Tracks changes in software development, enables collaboration, and provides a history of modifications
- **Challenges**
  - Conflicting code changes
  - Difficulty tracking changes
  - Potential loss of data (code)
- **Solution**
  - Git and GitHub

What is Git?

- **Git:** A **distributed** version control system that:
  - Track changes to files and code
  - Manage changes to files and code
  - Record modifications to projects
  - Compare different versions of files and code
  - Revert to previous states
  - Allow collaborations
- **Benefits**
  - Enables offline work, fast branching, and merging, and robustness



Download and Install Git



- Git link: https://git-scm.com/downloads

What is Git Bash

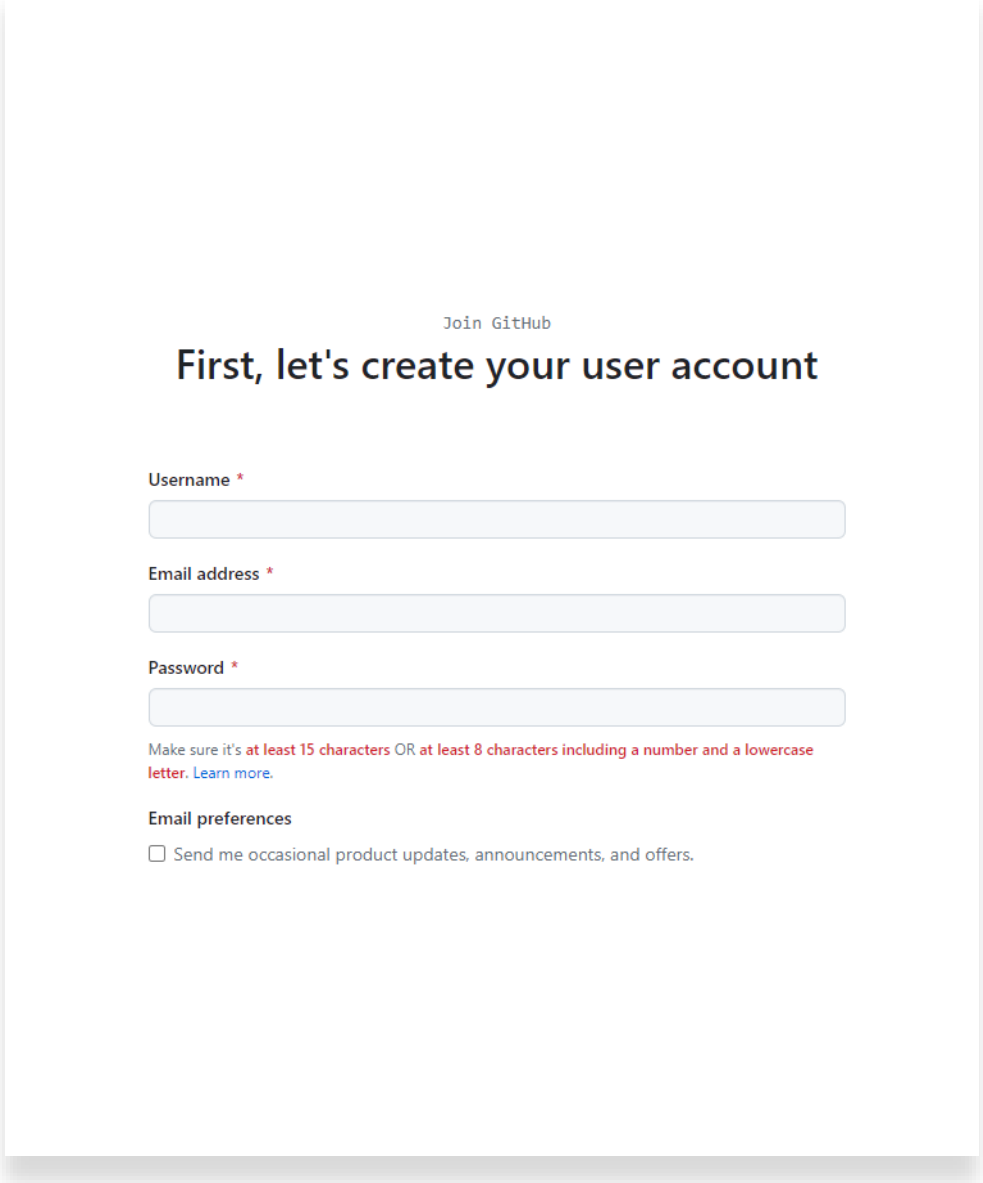
- **Git Bash:** A command-line interface for Git on Windows
- **Uses:**
  - Git operations – Interacting with git
  - Command-line Operations – Unix-like commands: cd, ls, mkdir, rm
  - Integration with other tools – used alongside other tools and utilities in the software development ecosystem.

What is GitHub

- **GitHub** – A code hosting platform for version control and collaboration.
- **Uses**
  - Version Control
  - Code Collaboration
  - Code Review
  - Documentation
  - Sharing Code and Open Source
  - Community Engagement
  - Portfolio and showcase
  - Education and Learning
  - Data Hosting

Create GitHub Account

- GitHub Website: https://github.com/



GitHub Portfolio and Showcase

- Developers can use GitHub to showcase their projects, skills, and contributions to potential employers or collaborators.
- **Sources**
  - YouTube tutorial (link)
  - Profile generator (link)
  - Profile repository (link)
  - Profile examples (link)
- **Activity:** Create a GitHub profile

GitHub Project Hosting and Documentation

- Documentation, sharing code and open-source, and hosting examples:
  - https://github.com/EddieHubCommunity/awesome-github-profiles
  - https://github.com/MAIF/shapash
  - https://github.com/pandas-dev/pandas
  - https://github.com/Ellie190/BCNN-for-Ocular-Disease-Classification
  - https://github.com/Ellie190/Database_Systems_Tutor
  - https://github.com/Ellie190/Google-Trends-Dashboard
  - https://github.com/valeman/awesome-conformal-prediction

Git Configuration

- Set up your Git identity with:
  - git config --global user.name "Your Name“
  - git config --global user.email "youremail@example.com"

Navigation GitHub

- How to create a repository:
  - Repository naming convention
  - Description
  - Repository visibility
  - README file (markdown cheat sheet)
  - .gitignore file (gitignore cheat sheet)
  - License (The MIT License link)
  - How to add collaborators

Getting Started with Git Bash

- Initialize a new Git repository with `git init`
- Clone an existing repository with `git clone <repository URL>`.
- Check the status of your repository with `git status`.
- Stage changes for a commit using `git add <file>` or `git add .` to include all changes.
- Commit changes with `git commit -m "Your commit message here“`.
- Push commits to a remote repository using `git push origin <branch>`.

General Activity

- Project Workflow
- Creating a repository
- Repository basics – visibility, readme, gitignore, adding files, code, deleting
- Adding collaborators
- Pushing a local repository
- Creating branches, branch naming convention (`git pull` and `git status` is key)
- Making pull requests
- All students to create branches and make pull requests
- Create a gitignore file and explain its working with examples

Version Control and Collaboration Activity

- Create repository named: simple-python-functions
- Specify visibility, add README, add license
- Clone repository
- Add python script with incorrect functions for students to debug
- Add students as collaborators
- Let each student create a branch named functionfix/function<func_num>
- Let each student commit with –m “I debugged function <func_num>”
- Each student should submit a pull request
- Each student should provide a clear description of what was fixed in the pull request
- Show pull request changes

Overview: GitHub and Git

- Overview
  - A version control system
  - A Publishing Tool
  - A Collaboration Platform
- Benefits
  - Great way to keep track of our codebase
  - A way to share and collaborate with others
  - Ability to save our process and go back to earlier points in our projects
- Major Benefit
  - GitHub makes it difficult to “mess up”
- Overview source (YouTube)