Introduction to GitHub

What exactly is GitHub and what are its main purposes and functions?
GitHub is an online service that undertakes the use of Git which is a version control system for managing code repositories. It creates an environment for the cooperation of developers in relation to the projects they are working on, version control, and support of the software development process.

Primary Functions and Features:
- Repositories: Provide storage and management of files of the project and files with historic context regarding other projects.
- Branches: Permit the working of several versions of the same project at the same time.
- Pull Requests: Encourage reviewers and the authors of change to discuss the code prior to merging it.
- Issues and Project Boards: Monitor task, bug and project status.
- GitHub Actions: Implement application automation relating to work flows and include continuous integration/continuous deployment (CI/CD).
- Collaboration: Permitted work with teams, and connected services that allow working in teams.

Support for Collaborative Software Development:Support for Collaborative Software Development:
GitHub helps in collaboration by offering possibilities for code review, project management, and the management of the workflow. Branching allows developers to build feature branches, conduct and review working with changes separately from the main code, and then merge them into the project.

Repositories on GitHub

What is actually a GitHub Repository?
A GitHub repository can be defined as virtual space where all the files of a specific project are lodged, as well as the past history of the changes in the files. Repositories can either be public or private and may include documentation, issues tracking as well as the collaboration.

How to Create a New Repository:How to Create a New Repository:
1. Sign In: Now go to GitHub https://github. com/ and sign in with the same account you used to sign up for the class.
2. New Repository: Go to the repository page and click on the button called “New”.
3. Repository Details: Enter the name for the new repository and description (Optional), also Set the new repository to public or private.
4. Initialize Repository: It may be started with a README file, . gitignore file or even select a license to start with.
5. Create Repository Here, locate and click the button labeled “Create repository. ”

Essential Elements to Include:
- README. md: Introduction that states what the project is, its intended use, as well as the steps to follow.
- LICENSE: Provides the legal term that governs the project in terms of license.
- . gitignore: Configures the file system and directories that will be excluded from the repository.
- Contributing Guidelines: Includes guidance to help the contributor participate in it.

Rewriting the Text

Git and Version Control Concepts:

Version control is a discipline that keeps track of changes made to code, allowing developers to pinpoint changes, reverse them and work together as well as keep a record of the projects history. Git is an example of DVC that allows many programmers to work on one project without interfering with each other.

Making Use of GitHub in the Context of Version Control:

GitHub is cloud based repository hosting service which improves Git by including web interfaces, integration and collaborative features. Collaboration on code, reviews and project management are simplified making it easy to maintain a robust version control system.

Merging and Branching on GitHub

What are branches in GitHub; why do they matter?

Branches are lines of development that represent separate versions of a repository enabling developers to maintain new features or bug fixes isolated from mainline code. It permits experimentation and development in isolation until merging time comes.

How to Create a New Branch, Make Some Changes, Merge It Back

Making New Branch:
Through Command line: git checkout -b new-branch
Through Github: Go to the repository, click “Branch: Main” then input the name for your new branch.
Do revisions: Keep doing work on the branch, and commit your changes as you go.

Push Changes: Push the branch to GitHub: git push origin new-branch

Create a Pull Request: Open a pull request to merge changes into the main branch.

Review and Merge: Verify the modifications made, respond to any recommendations made by reviewers and then combine the two branches using GitHub interface in getting them together.

Pull Requests and Code Reviews

What is a Pull Request in GitHub, and How Does It Facilitate Code Reviews and Collaboration?

A pull request is when you ask for changes from one set of code to be merged with another set of code. This helps members of the group go through code on their own thus encouraging high level of teamwork among peers before it can be put into the main code base.

Steps to Create and Review a Pull Request:

Create a Pull Request: Go to the repository, click "New pull request," select branches, and provide a title and description.

Review the Pull Request: Team members review the changes, add comments, and request modifications if needed.
Merge the pull request: Once approved, the pull request can be merged into the target branch by clicking the “merge pull request” button.

GitHub Actions

What are GitHub Actions and how can they be used to automate workflows?

GitHub actions is an automation tool for developers to build, test, and deploy code in a workflow. These workflows will be defined within a repository through YAML files.

Example of Simple CI/CD Pipeline Using GitHub Actions:

Create a Workflow File: Create the ci.yml file under .github/workflows in your repository.

Define the Workflow:

```
name: CI Pipeline

on: [push]

jobs:

  build:

    runs-on: ubuntu-latest

    steps:
```

```
- uses: actions/checkout@v2
- name: Set up Node.js
  uses: actions/setup-node@v2
  with:
    node-version: '14'
- name: Install dependencies
  run: npm install
- name: Run tests
  run: npm test
```

Copy

Commit and Push – Commit this workflow file then push it to GitHub, every push will have this workflow running.

Introduction to Visual Studio

What is Visual Studio and What are its Main Features?

Visual studio refers to an integrated development environment (IDE) developed by Microsoft for software development applications across various platforms. It offers features like code editing, debugging, version control, and project management.

How It Differs from Visual Studio Code:
- Visual Studio: A full-featured IDE for large-scale development with advanced debugging, testing, and deployment tools.
- Visual Studio Code: A lightweight, open-source code editor focused on speed and flexibility, ideal for quick edits and lightweight development tasks.

Integrating GitHub with Visual Studio

Steps to Integrate a GitHub Repository with Visual Studio:
1. Install GitHub Extension: Install the GitHub extension for Visual Studio from the Extensions menu.

2. Sign In to GitHub: Use the GitHub extension to sign in to your GitHub account.
3. Clone Repository: Use the `Clone a repository` option to clone a GitHub repository into Visual Studio.
4. Manage Repository: Use the Team Explorer window to manage commits, branches, and pull requests.

How This Integration Enhances Development Workflow:
- Seamless Workflow: Integrate source control directly into your development environment.
- Efficiency: Quickly clone repositories, manage branches, and handle pull requests.
- Collaboration: Easily collaborate with team members and track project progress within Visual Studio.

Debugging in Visual Studio

Debugging Tools Available in Visual Studio:
- Breakpoints: Pause execution at specific lines of code.
- Watch Window: Monitor variables and expressions.
- Call Stack: View the function call hierarchy.
- Immediate Window: Execute code and evaluate expressions on the fly.
- Exception Handling: Manage exceptions and inspect thrown exceptions.

How Developers Use These Tools to Identify and Fix Issues:
Developers use breakpoints to pause execution and inspect the state of the application. They can evaluate variables, monitor changes, and step through code to understand the flow and identify bugs. The call stack helps trace the sequence of function calls, and the immediate window allows testing and evaluating expressions in real-time.

Collaborative Development using GitHub and Visual Studio

How GitHub and Visual Studio Can Be Used Together to Support Collaborative Development:
- Version Control: Use GitHub for version control and collaboration.
- Code Reviews: Create pull requests and conduct code reviews directly from Visual Studio.
- Automated Workflows: Use GitHub Actions to automate testing and deployment.
- Integrated Development: Visual Studio’s integration with GitHub streamlines the development workflow, allowing developers to manage code, branches, and pull requests without leaving the IDE.