Introduction to GitHub:

What is GitHub and what are its primary functions and features? Explain how it supports collaborative software development.

GitHub is an online site that hosts Git repositories and provides all of the capabilities and distributed revision control (SCM) of Git, along with additional features. This website is used to host remote git repositories.

<u>Some of the primary functions and features include:</u>
1.Version control that is repositories where you store and manage code, commit and changes where you track and work on different versions of a project simultaneously.
2.Collaboration that is code review where you have a peer view of the code through pull requests.
3.Social features like forking where you copy and bookmark repositories.

GitHub offers tools for version control, code review, bug tracking and project management to facilitate collaborative software development. Developers are able to independently work on branches, submit pull requests for changes and perform code reviews to ensure accuracy. While GitHub Actions simplify testing and deployment, Issues and project boards aid in keeping track of chores and progress. Collaboration and information sharing are further encouraged by community engagement and integration with additional tools.

Repositories on GitHub:

What is a GitHub repository? Describe how to create a new repository and the essential elements that should be included in it.

A GitHub repository is a central location on GitHub where you may manage and store files, project data and code. It functions similarly to a virtual storage area where you may arrange and easily access all of your project-related materials.

Please visit https://github.com. Open your GitHub account and log in. Click the plus sign (+) in the upper right corner and choose "New repository". Give your repository a name. This is the name of the GitHub folder where your repository is stored. You might choose to include a description of your repository. This will make it clearer to others what the purpose of your repository is. Decide if you want your repository to be private or public. Public repositories are visible to anybody with an internet connection. Only those you have invited to participate on the repository with you can view private repositories. Go to "Create repository" and click.

Version control with Git:
Explain the concept of version control in the context of Git. How does GitHub enhance version control developers?
Many developers can work on a project at once without running into problems if they use a version control system like Git, which helps developers in tracking and managing source code changes over time. Git uses branches to enable continuous work, repositories to hold project files and their history and commits to document snapshots of changes. While developers can examine changes in the staging area before committing them, merging combines changes from many branches. Git's distributed design ensures that every developer has a complete copy of the repository, enabling redundancy and offline work.

GitHub improves on Git's version control features by giving users access to a web-based repository hosting platform and extra project management and collaboration tools. In order to preserve code quality, pull requests allow for code review and discussion without merging changes. GitHub's project boards and issue tracker facilitate work organization and progress monitoring. Code changes are verified through automation of testing and deployment by GitHub Actions. Markdown support improves documentation while social coding tools like forking and following promote community involvement.

Branching and merging in GitHub:
What are branches in GitHub and why are they important? Describe the process of creating a branch, making changes and merging it back into the main branch.
GitHub branches are duplicate versions of a repository that let developers work on separate projects, experiments or bug fixes apart from the main source. The default branch where the project's stable version is kept is called the main branch often referred to as master or main.
They are important in that they one can experiment on new ideas without disrupting the main project. One can maintain a stable branch while development occurs in other branches.

To create a branch, go to your repository in GitHub, click on the branch dropdown menu, type a branch name in the "find or create a branch" and click "create branch". To make changes, modify your files either editing, adding or deleting files, add your changes by typing the command 'git add .' , save your changes by typing the command 'git commit -m "The commit message"', push your branch to your repository using the command 'git push origin branch-name'. To merge your changes, create a pull request and then go to your pull request and click “merge pull request” and confirm the merge by clicking on the confirm merge.

Pull requests and code reviews:
What is a pull request in GitHub and how does it facilitate code reviews and collaboration? Outline the steps to create and review a pull request.
Pull requests are a feature that helps with code review and teamwork. Developers can suggest changes to a repository and ask for another person to check and accept those changes before they are merged into the main branch.

Facilitates code reviews: Team members will review the suggested changes on GitHub. They can debate the usefulness and quality of the code, post questions and comments and make suggestions for changes.

Facilitate collaboration: By promoting communication among team members and ensuring code integrity throughout the software development cycle.

<u>Steps on creating a pull request;</u>

- Create a new branch for your changes.
- Push the branch to GitHub.
- Go to your repository, click "Pull requests".
- Click "New pull request".
- Choose base and compare branches.
- Write a title and description.
- Click "Create pull request".

<u>Steps on reviewing a pull request;</u>

- Access the pull request from the repository.
- Review changes in files.
- Add comments directly on lines of code.
- Discuss changes with the author and team.
- Approve or request changes.
- Ensure tests pass and code meets standards.
- Merge the pull request when ready.

GitHub actions:

Explain what GitHub actions are and how they can be used to automate workflows. Provide an example of a simple CI/CD pipeline using GitHub actions.

GitHub Actions are created to automate actions within GitHub repositories. They allow you use your GitHub repository to build, test and publish your code directly.

With GitHub Actions, developers may create custom tasks and actions that are generated by events such as pull requests thereby automating workflows.
<u>Example:</u>
Create a GitHub repository on GitHub. Create a workflow directory named '.github workflows'. Create a workflow file and define the workflow. Add the necessary scripts needed for testing, building and deploying. Commit your changes and push them to your repository. Monitor the workflow by going to the actions on GitHub repository.

Introduction to Visual studio:
What is Visual studio and what are its key features? How does it differ from Visual studio code?
Microsoft has developed an integrated development environment (IDE) called Visual Studio (VS). Building a variety of software applications, including as desktop, web, mobile, and gaming apps, is made easier with its extensive capabilities and toolkit.

Some of its key features include code editing, debugging and it is customizable with extensions and themes.

Visual Studio Code is a lightweight, flexible code editor ideal for rapid coding jobs, customization and cross-platform development while Visual Studio is a comprehensive integrated development environment (IDE) focused on building and managing software projects across multiple platforms and languages.

Integrating GitHub with Visual Studio:

Describe the steps to integrate a GitHub repository with Visual studio. How does this integration enhance the development workflow?

To integrate a GitHub repository with Visual Studio, first ensure you have Git installed, a GitHub account and the latest Visual Studio. Begin by signing into GitHub within Visual Studio via `File` then `Account Settings`. To clone a repository, use `File` then `Clone Repository`, enter the repository URL and select a local path. For creating a new repository, navigate to `File` then `New` then `Repository`, fill in the details and push it to GitHub. To add an existing project, open the project, go to `File` then `Add to Source Control`, initialize the repository and use the `Team Explorer` pane to sync and publish it. Manage changes by committing and syncing through the `Team Explorer` and collaborate by managing pull requests and code reviews within Visual Studio.

This integration centralizes version control, streamlines collaboration, automates CI/CD processes, improves code quality, simplifies workflows and enhances visibility and traceability, significantly boosting development productivity.

Debugging in Visual Studio:

Explain the debugging tools available in Visual Studio. How can developers use these tools to identify and fix issues in their code?

Some of the debugging tools and their usage;

1.Breakpoints- they pause your code at specific lines. Used to set breakpoints that is clicking on the margin next to the line of code.

2.Call stacks- they show the sequence of function calls that lead to the current point in the program Used to navigate stack frames that is clicking on different frames in the call stack to view the state of the code at various points.

3.Step commands- They step through your code line by line. Used to step over (F10) that is to execute the next line of code but skip over function calls.

4.Memory Windows- Allow low-level inspection of memory. Used to view raw memory that is go to debug then windows then memory and select a memory window to inspect memory at specific addresses.

Collaborative development using GitHub and Visual Studio:

Discuss how GitHub and Visual Studio can be used together to support collaborative development. Provide a real-world example of a project that benefits from this integration.

Version control and project management functionalities are integrated into an effective development environment by GitHub and Visual Studio to improve collaborative development. Code collaboration is facilitated for developers by Visual Studio's ability to clone repositories, commit changes, manage branches and generate pull requests. Issue tracking on GitHub easily integrates with Visual Studio work items and continuous integration and deployment are ensured through automated processes utilizing GitHub Actions and Azure Pipelines. Additionally facilitating real-time collaboration is Visual Studio Live Share, which enables developers to work together on editing and debugging while integrated tools support the upkeep of documentation and code quality. Through this connection, the entire development process is streamlined, resulting in more productive and efficient teamwork.

References:

Pdf’s from the LMS in Power Learning Project Academy.

Better software and stronger team by Matt Butler and Paige Paquette published in 2016.