**NAME: PETER MUHUMUKI**

**EMAIL: muhumukip@gmail.com**

# ASSIGNMENT: GITHUB AND VISUAL STUDIO

## INTRODUCTION TO GITHUB

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

GitHub is a platform that supports collaborative software development by providing tools such as version control, code review, issue tracking, project management, and integration with other development tools.

Has the following features:

1. Version control: Allows developers to track the changes to their code. This enables them to work on projects simultaneously without conflicts.
2. Collaboration: provides tools such as pull requests, branching, forks, and issue tracking to facilitate teamwork.
3. Code hosting: GitHub hosts repositories in the cloud, eliminating the need for local servers and enabling access from anywhere.
4. Code review: Has integrated features for reviewing proposed changes, commenting on specific lines of code, and discussing improvements, ensuring code quality and consistency.
5. Project management: Has features like project boards, milestones and task list to help organize and manage project tasks and progress.
6. **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 centralized space to store project code, documentation, and other related resources.
* How to create a New Repository:
  + Sign in or log in to GitHub.
  + Click on the profile picture at the top right corner of the screen and select your select “Your Repositories” from the drop-down menu.
  + Click the green “New” button at the top right of the screen.
  + Enter the repository details, i.e., Repository Name, Description (Optional), and Visibility (Public / private).
  + Initiate the repository with a README.md file, .gitignore template (Optional), and choose a LICENSE (Optional).
  + Click the “Create Repository” button to create the new repository.
* The essential Elements of a repository are:
  + **Readme.md file** – provides an overview of the project and other relevant information to the project.
  + **.gitignore file** – specifies files that should be ignored by git, preventing them from being tracked and included in the repository.
  + **LICENSE** – Define the legal terms under which the project can be used, modified and shared.
  + **Source code** – the actual code files that make the project.
  + **Documentation** – Detailed information on how to use, configure, and contribute to the project.

1. **Explain the concept of version control in the context of Git. How does GitHub enhance version control for developers?**

Git is a distributed version control system that allows developers to track changes in their work, collaborate with other developers, and manage various versions of their work.

GitHub enhances Version Control for developers through:

* Allows developers to host their projects in the cloud, making it easy for team members to collaborate on projects.
* Allows developers to suggest changes to the system using pull requests.
* Allows developers to track their work, report bugs, request features, and track project progress.
* Provides developers with code review tools that enable them to comment on specific lines of code facilitating detailed and collaborative review processes.
* Supports the creation of multiple branches, allowing developers to work on multiple features or fixes simultaneously.

1. **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.**

Refers to separate lines of development within a repository, allowing developers to work on different features, bug fixes, or experiments independently from the main codebase.

Branches are important in that they:

* Help in isolating changes, preventing unfinished work from affecting the main codebase.
* Enable multiple developers to work on different branches simultaneously without conflicts.

To create a branch, type command `*git checkout -b branch-name*`. Replace the *branch-name* with the real name of the branch you want to create.

To make changes:

* Edit files and make the changes to them as needed.
* Stage and commit the changes:
  + *git add .*
  + *git commit -m “commit message”*
* Push the changes to the branch.
  + *git push -u origin branch-name*

To create a Pull Request:

* On GitHub account, navigate to your repository.
* Click *‘New Pull Request’* and select the branch to merge into the *main* branch,
* Review and submit the pull request,

To Merge a branch:

* Review the pull request, discuss changes if needed.
* Merge the pull request into the main branch
* Delete the branch if no longer needed.

1. **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.**

* A pull request is a method to propose changes to a codebase, facilitating code reviews and collaboration before merging.
* Pull requests facilitate code reviews by:
  + Enabling team members to review and suggest improvements.
  + Acts as a discussion forum to implement detail.
  + Enhance tracking of changes and decisions.
  + Integrate with testing and deployment pipelines.
* To create a pull request:
  + Enter the command **git push origin branch-name** on git bash.
  + Navigate to the repository.
  + Click ‘Pull Requests’ -> ‘New Pull Requests’
  + Select the branches to merge.
  + Add title and description
  + Click ‘Create pull request’.
* To review the pull request:
  + Go to the “Pull Requests” tab and select the pull request.
  + Review the changes, add comments, and use “Review changes” to approve, comment, or request changes.
  + After the approval, click “Merge pull request” and choose the merge method. Confirm the merge.

1. **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 is a CI/CD platform that allow developers to automate workflows directly in their GitHub repositories. For Example: Automate testing and deployment of an application.
* An example of a simple workflow is shown below:

name: CI/CD Pipeline

on: [push, pull\_request]

jobs:

build:

runs-on: ubuntu-latest

steps:

- name: Checkout code

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

- name: Deploy to production

if: github.ref == 'refs/heads/main'

run: |

npm run build

npm run deploy

## INTRODUCTION TO VISUAL STUDIO

1. **What is Visual Studio, and what are its key features? How does it differ from Visual Studio Code?**

* Visual Studio is an IDE designed to build a variety of applications across multiple platforms.
* The main features are:
  + Comprehensive development features, including a powerful code editor, compilers and tools for version control.
  + Support for multiple programming languages.
  + Offers a wide range of project templates for web, mobile, desktop, and cloud applications.
  + Advanced debugging features,
  + Supports team collaboration through tools such as Azure DevOPs and git integration.

Main differences include between Visual Studio and Visual Studio Code are:

* Visual Studio is mainly used for large scale development projects while VS Code is used for lightweight code editor for small scale projects.
* Visual Studio is more resource intensive compared to VS Code.
* Visual Studio uses plugins with many built-in features while VS Code uses extensions from the marketplace.
* Visual Studio is mainly used by professionals working on large, complex applications while VS Code is mainly used for lightweight applications.

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

Integrating GitHub repository to Visual Studio

* Install the GitHub extension
* Sign in to GitHub
* Clone or create repositories
* Commit and sync changes, manage branches, and create pull requests directly from visual studio.

The integration enhances development by:

* Streamlining Collaboration
* Simplifying branch management
* Facilitating commit and sync operations.
* Improving code review processes through pull requests
* Supporting debugging and testing within the IDE.

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

* Breakpoints – Pauses program execution at specific lines of code. Helps to identify problematic areas by pausing execution and inspecting the code behaviour.
* Call stack – Shows the sequence of function calls that led to the current point in the code. This helps understand what actions triggered the issue.
* Watch window – monitors variables and expressions. Helps to monitor changes as well as any unexpected behaviour.
* Edit and continue – helps the user to modify error and continue the debugging process without having to restart the process.

1. **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.**

GitHub and visual studio synchronization supports collaborative development by enabling:

* Seamless code synchronization and version control through GitHub repositories
* Efficient branching, merging and code review processes using GitHub pull requests
* Enhance issue tracking and project management using GitHub issues.
* Automation of workflows and using GitHub actions for continuous integration and development
* Real-time monitoring and feedback on builds and tests directly within Visual Studio