SOFTWARE ENGINEERING & GITHUB:

COURSE OBJECTIVE:	● Mastering GitLab interface for efficient project management and collaboration ● Implementing core GitLab features for automation and code quality control. ● Enhancing version control skills for tracking changes and maintaining code integrity. ● Proficiency in testing and debugging with GitLab CI/CD pipelines. ● Competency in deployment and distribution using GitLab's deployment features.
COURSE OUTCOME:	● Develop proficient version control skills by utilizing GitLab for tasks such as creating repositories, branching, merging, and resolving conflicts. ● Implement effective collaboration strategies through GitLab's features like issue tracking, merge requests, and commenting, fostering teamwork on software projects. ● Build automated CI/CD pipelines in GitLab to streamline the building, testing, and deployment processes of software applications. ● Develop project management skills by organizing tasks, setting milestones, and tracking progress within GitLab's project management interface. ● Implement code quality assurance measures using GitLab's code review tools, ensuring high-quality code through peer reviews and analysis. ● Deploy and distribute software applications proficiently by leveraging GitLab's deployment features, which include environments, releases, and container registries.

Course Duration: 45 Hours

Course Content:

Unit I: Introduction to Version Control

What is version control? - Various version control tools - Why GIT is popular among others? - Importance and benefits of GIT - Installation of GIT

UNIT II: Git Commands & Portfolio Development

Overview of GIT commands - Working with local GIT commands (add, commit, diff) - Reactjs essentials to build portfolio - Creating a responsive portfolio - Adding files to GIT - View local commit and graphs

UNIT III: GitHub – a Distributed Platform

Introduction to GitHub a distributed Platform - Creating repo with GitHub - Need of Readme file - Configuring created repo with local git folder - Add & commit files to GitHub - Tracking file changes with GitHub - Working with merge, rebase, diff, rest, logs, stash etc. - Overview of GitHub Desktop

UNIT IV: GitHub Actions

Introduction to CI (continuous integrations) - CI principles - Overview of yaml scripts - create yaml script and setup configuration - Event triggering workflow jobs and steps - Custom Workflow scripts - Building artifacts and packaging

UNIT V: Deployment

Setting up your GitHub pages - Check actions log about errors - Enable GitHub pages with GitHub actions

Test Projects:

Use Cases

OVERALL COURSE LEARNING OUTCOME ASSESSMENT CRITERIA AND USECASES		
LEARNING OUTCOME	**ASSESSMENT CRITERIA**	**USECASES**
Exhibit the fundamentals of GIT	Evaluation: Programming and MCQ	**Usecase:1 Creating local folder for GIT** • Task 1: Cerate a local folder and initiate GIT. • Task 2: Create necessary files. • Task 3: Try git add command for local folder differences. • Task 4: Create one more file and try git diff to understand the file tracking. • Task 5: add all the files to git and commit the files
Design and develop a simple web page	Evaluation: Programming Assessment and project	**Usecase:2 Create a sample webpage** • Task 1: Set up a new project with frontend. • Task2: implement necessary design styles to keep you page working. • Task 3: Test your sample web page with the browser for design effectiveness. • Task 4: Ensure you sample page meets the standards. • Task 5: Commit all your files to local git and track changes.

Implement frontend interfaces using modern React library	Evaluation: Programming assignments	**Usecase:3 Creating an application with React** ● Task 1: Set up a React project using Create React App. ● Task 2: Design and implement reusable UI components. ● Task 3: Integrate a CSS framework (e.g., Bootstrap) for responsive design. ● Task 4: Connect your local folder with GitHub and commit the files. ● Task 5: Push your files with GitHub.
Set Up the Initial Project Structure	Evaluation: Programming and MCQ	**Use case: 4 Create the initial structure of the portfolio website including essential HTML pages and folders for assets** ● Task 1: Create a new repository on GitHub named portfolio- website. ● Task 2: Set up the initial project structure with below 4 steps: 1. Create an index.html file for the homepage. 2. Create an about.html file for the about page. 3. Create a projects.html file for the projects page. 4. Create CSS, JS, and images folders. ● Task 3: Add basic HTML boilerplate code to each HTML file. ● Task 4: Commit and push the initial project structure to the repository.

Implement a Responsive Navigation Bar	Evaluation: Programming and MCQ	**Use case: 5 Develop a responsive navigation bar that adjusts to different screen sizes using HTML and CSS** ● Task 1: In the index.html, about.html, and projects.html files, add a <nav> element with links to the homepage, about page, and projects page. ● Task 2: Style the navigation bar in a CSS file (CSS/styles.css) to be horizontal on desktop screens and convert to a vertical or collapsible menu on smaller screens using media queries. ● Task 3: Test the navigation bar on different screen sizes to ensure it adjusts appropriately. ● Task 4: Commit files to your GitHub repo. ● Task 5: Enable the GitHub repo to pages to see in Browser
Enhance the Visual Appeal of a Blog Website with CSS	Evaluation: Programming Assessment and project	**Use case: 6 Creating Visual page using CSS** ● Task 1: Style the Header and Navigation Bar ● Task 2: Implement a Responsive Grid Layout for Blog Posts ● Task 3: Add Hover Effects to Links and Buttons Task 4: Use CSS Variables for Consistent Theming

Add Interactivity to a Website with JavaScript	Evaluation: Programming and MCQ	**Use case 7: Working with JavaScript for interactive website.** • Task 1: Task 1: Implement a Mobile Navigation Toggle • Task 2: Create a Slideshow for Project Images • Task 3: Validate a Contact Form • Task 4: Display a Dynamic Greeting Based on the Time of Day
Enhance Website Responsiveness with Media Queries	Evaluation: Programming and MCQ	**Use case 8: Creating a responsive page with media query** • Task 1: Task 1: Adjust Layout for Mobile Devices • Task 2: Change Font Sizes for Better Readability on Tablets • Task 3: Hide Sidebar on Small Screen • Task 4: Adjust Image Sizes for Different Screen Resolutions • Task 5: Test your page in anyone browser
Build a Responsive Web Application with CSS framework	Evaluation: Programming Assessment and project	**Use case: 9 Building a Responsive Web Application with Bootstrap** • Task 1: Implement a Responsive Grid Layout • Task 2: Style a Navigation Bar for Different Screen Sizes • Task 3: Adjust Typography for Better Readability • Task 4: Customize Button Styles for Different States • Task 5: Test your responsive page in anyone browser

Enhance Web page Interactivity with JavaScript DOM Manipulation	Evaluation: Programming assignments	**Use case: 10 Working with DOM manipulation to understand the page interactivity** • Task 1: Create necessary files to work on dynamic content Update • Task 2: Add New Elements to the DOM • Task 3: Toggle CSS Classes for Styling • Task 4: Remove Elements from the DOM • Task 5: Test your webpage with browser.
Create Dynamic User Interfaces with React JSX Elements	Evaluation: Programming Assessment and project	**Use case: 11 Create a react project to work on JSX elements** • Task 1: Render Dynamic Content • Task 2: Conditionally Render Components • Task 3: Implement Event Handling • Task 4: Pass Props to Child Components
Enhance React Components with Hooks	Evaluation: Programming and MCQ	**Use case: 12 Working on react hooks** • Task 1: Manage State with useState Hook • Task 2: Effectively Use useEffect Hook for Lifecycle Events • Task 3: Implement Custom Hooks for Reusable Logic • Task 4: Optimize Performance with useRef Hook

Manage Git Repositories	Evaluation: Programming Assessment and project	**Use case: 13 Managing Git Repositories** • Task 1: Initialize a New Local Repository • Task 2: Create a Remote Repository on Git Hosting Service • Task 3: Connect Local Repository to Remote Repository • Task 4: Set Up Branches and Manage Branching Strategy • Task 5: Initialize a Git Repository for an Existing Project
Margin GIT commands	Evaluation: Programming assignments and MCQ	**Use case 14: Managing GIT commands** • Task 1: Stage Files for Commit • Task 2: Commit Changes with a Descriptive Message • Task 3: Amend the Last Commit • Task 4: Push Commits to a Remote Repository • Task 5: Push a Specific Branch to Remote

Integrate Features Using Merge Capability	Evaluation: Programming Assessment and project	**Use case 15: Multiple developers working on different features and integrating the feature using merge capability** • Task1: Create multiple feature branches from main to work independently. • Task2: Push feature branch changes to github and create pull requests. • Task3: After code review changes had to be merged into main using merge capability. • Task4: Identify and resolve conflicts that arise during integration
Automate deployment using Github actions	Evaluation: Programming Assessment and project	**Use case 16: Automating deployment using GitHub Actions involves setting up workflows that execute predefined tasks whenever specific events** • Task1: Create a workflow file (deploy.yml) in the .github/workflows directory • Task2. Specify the event (push) and branch (main) in the workflow file to trigger deployment whenever changes are pushed to the repository. • Task3: Use package managers like npm or yarn to install dependencies • Taks4: Use build tools, frameworks, or libraries needed for compiling and preparing the static site assets. • Task5: Build the static site and deploy it to GitHub Pages

Enable Github pages and choose deployment source	Evaluation: Programming Assessment and project	**Use case 17: Configure GitHub Pages settings to specify the source branch (main or master) from which GitHub Pages will deploy your static site** • Task1: Access the settings page of your GitHub repository. • Task2: Locate the GitHub Pages settings area where you can enable GitHub Pages and configure deployment options. • Task3: Activate GitHub Pages to host your static website directly from your GitHub repository. • Task4: Ensure to save your changes after enabling GitHub Pages. • Task5: Select the deployment source branch to apply the configuration
Schedule Tasks and Maintenance Jobs	Evaluation: Programming assignments	**Use case 18: Schedule recurring tasks and maintenance jobs using GitHub Actions.** • Task1: Automatically backup database or file systems at regular intervals. • Task2: Perform database maintenance tasks (e.g., cleanup, index optimization) on a schedule. • Task3: Run batch processing jobs or pipelines periodically. • Task4: Monitor system health by running diagnostic checks and generating reports. • Task5: Send notifications or alerts based on specific events or thresholds

Collaborative Note-Taking Application	Evaluation: Programming Assessment and project	**Use case 19: Develop a collaborative note- taking application where multiple users can create and edit notes in real-time** • Task1: Initialize a Git repository for the note-taking application and push it to GitHub. • Task2: Create a README file with project setup instructions. • Task3: Set up GitHub Actions to run build the project automatically on every push. • Task4: Configure workflows to ensure code quality and functionality • Task5: Automate the deployment process to ensure the latest version is always live.
Collaborate on Feature Development with Pull Requests	Evaluation: Programming Assessment and project	**Use case 20: Collaborating on Feature Development with Pull Requests** • Task 1: Create a Pull Request • Task 2: Review and Comment on Pull Requests • Task 3: Merge a Pull Request • Task 4: Resolve Merge Conflicts in Pull Requests • Task 5: Close and Manage Pull Requests

LIST OF FINAL PROJECTS (20 PROJECTS THAT COMPREHENSIVELY COVER ALL THE LEARNING OUTCOME)
FINAL PROJECT
1. Create a website showcasing your resume, projects, and skills. 2. Implement a task management system with features like adding tasks, marking as completed, and filtering. 3. Develop a quiz app where users can answer multiple-choice questions with immediate feedback. 4. Create a basic calculator with arithmetic operations and a responsive layout. 5. Design and code a reusable portfolio theme with multiple sections and smooth scrolling navigation. 6. Display current weather conditions and forecast using a weather API. 7. Build a currency converter that fetches exchange rates and performs conversions. 8. Implement a classic Tic-Tac-Toe game with two-player mode. 9. Manage and version control your personal website's codebase using Git, allowing for easy rollback and collaboration. 10. Collaborate with a team on a coding project, using Git to manage branches, pull requests, and code reviews. 11. Simulate a typical Git workflow with branches for feature development, bug fixes, and releases for a hypothetical project. 12. Set up a Git repository with CI/CD pipelines (using GitHub Actions) to automate testing and deployment processes. 13. Develop a task management application with React where users can create, update, delete tasks using react 14. Create a simple counter application using React where users can increment, decrement, and reset the counter value 15. Develop a markdown editor application with React where users can input markdown text and see the formatted preview in real-time. 16. Create a GitHub Actions workflow to build and deploy a static website (e.g., built with React, Vue, or plain HTML/CSS) to GitHub Pages or another hosting service whenever changes are pushed to the main branch. 17. Develop a GitHub Actions workflow to run scheduled tasks (e.g., daily backups, data cleanup) 18. Develop a Responsive User Interface with React. 19. Create a shopping cart using React that allows users to add and remove items from the cart 20. Build an event countdown timer using HTML, CSS, and JavaScript to countdown to a specific date.