IDE: Visual Studio Code

Introduction to Visual Studio Code

Overview of Visual Studio Code

- Visual Studio Code (VS Code) is a powerful, open-source code editor developed by Microsoft.

- Highly customizable with a user-friendly interface and extensive settings for personalized development environments.

- Supports multiple programming languages including JavaScript, Python, HTML, CSS, and many more through built-in features and extensions.

- Vast ecosystem of extensions available to enhance development workflow, including integrations for debugging, linting, and version control.
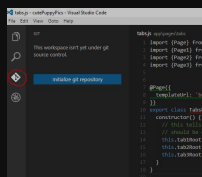
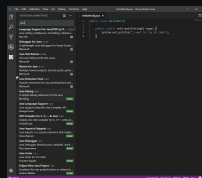# Key Features of Visual Studio Code
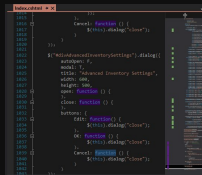
Code Editor

Syntax highlighting, intelligent code completion, integrated terminal, and support for JSX and React components.

Version Control

Built-in Git support, GitLens extension for advanced Git features like blame annotations and repository insights.
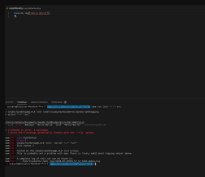

Extensions

Vast library of extensions like Python, ESLint, Prettier, Django, MongoDB, and Docker to enhance the development workflow.

Popular Extensions

Python

Provides support for linting, debugging, IntelliSense, and code navigation for Python development.

ESLint

Integrates ESLint for linting JavaScript/React code, offering auto-fixing of issues.

Prettier

Ensures consistent code style by formatting code automatically.

Setup Project Workspace

Clone the repository and open it in VS Code. Configure workspace settings and install necessary extensions to tailor the environment to your project's needs.

Repository clone
Workspace settings
Necessary extensions installed

Code Development

Write and organize your code using VS Code's powerful editor features like syntax highlighting, code completion, and integrated terminal. Utilize Git integration for efficient version control.

Structured codebase
Version control with Git
Integrated terminal usage

Testing and Debugging

Run unit tests using integrated test runners. Debug both frontend and backend code with VS Code's robust debugging tools, ensuring code quality and functionality.

Unit test results
Debugging configurations
Error-free codebase

Deployment

Manage Docker containers and configurations directly within VS Code. Use terminal commands to interact with deployment pipelines and deploy applications smoothly.

Docker container setup
Deployment scripts
Successful application deployment

Project Management with Jira

Features
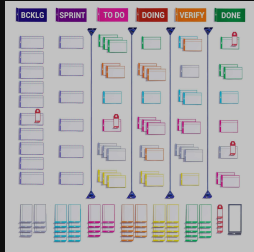
Issue Tracking: Create and manage issues, stories, tasks, and bugs. Sprint Planning: Organize tasks into sprints, set priorities, and track progress. Agile Boards: Kanban and Scrum boards for visual task management.
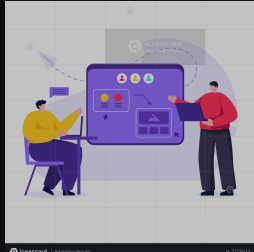

Workflow Integration

Backlog Management: Create a product backlog with detailed user stories and tasks. Sprint Execution: Plan sprints, assign tasks to team members, and track progress. Daily Standups: Use boards to conduct daily standup meetings and track task updates.

Features Overview

Boards, Lists, and Cards allow for visual organization. Customizable workflows adapt to project needs. Due Dates and Checklists ensure task completion.

Workflow Integration

Create boards for different project areas. Track progress by moving cards. Collaborate in real-time with comments and file attachments.

CI/CD with Jenkins

Jenkins Features

- Pipeline Automation: Define CI/CD pipelines using Jenkinsfile.
- Plugins: Extensive plugin ecosystem for integration.
- Automated Testing: Run unit tests, integration tests, and security scans.

Workflow Integration

- Setup Jenkins: Install and configure necessary plugins.
- Create Jenkinsfile: Define build and deployment pipelines.
- Integrate with Source Control: Link Jenkins with GitHub for triggering builds.
- Deployment: Automate deployment to staging or production environments.

CI/CD with GitHub Actions

Features of GitHub Actions

- Workflow automation with YAML files for defining CI/CD processes.

- Access to pre-built actions in the Actions Marketplace for integration.

- Event triggers initiate workflows based on repository events.

- Matrix builds for testing across multiple environments.

- Securely manage secrets and environment variables.

Workflow Integration

- Define workflows in .github/workflows directory.

- Automatically trigger workflows based on repository events.

- Run frontend (JavaScript) and backend (Python) tests in workflows.

- Deploy applications to cloud services or servers post-build and test.

- Easily integrate with other GitHub services and third-party tools.

Thank you.