Chop-Chop Environment and Workflow Report

By Xander Bezanson and Ryan Wetmore

Contents

Development Environment:

Text Editors/IDEs: (e.g., Visual Studio Code, Sublime Text)

We have chosen **Visual Studio Code (VS Code)** as our integrated development environment (IDE). The primary reasons for this selection are:

- Familiarity: Both team members are well-versed in using VS Code, which minimizes the learning curve.

- Cost-effectiveness: VS Code is free and open source, making it an accessible and practical choice.
- Extensive Features: It offers a rich ecosystem of extensions, built-in Git support, and excellent debugging capabilities.

Version Control: (e.g., Git, GitHub)

For version control, we will use **GitHub**, as it provides:

- Reliability: GitHub is a widely used and trusted platform for collaborative development.
- Experience: Both team members have extensive prior experience with Git and GitHub.
- Collaboration: GitHub's features, such as pull requests, issue tracking, and branch management, facilitate smooth collaboration and code review.

Frameworks/Libraries: (e.g., React, Django)

Our primary framework for this project will be **React**, chosen for the following reasons:

- Previous Experience: We completed a React course last semester and have worked on several projects using React.
- Component-Based Architecture: Reacts modular approach enhances code reusability and maintainability.
- Strong Ecosystem: React has a vast library of third-party packages that can enhance development efficiency.

Development Environment: (e.g., local setup, Docker, Vagrant)

We have opted for a **local setup** for our development environment due to its:

- Ease of Use: Setting up a local environment is straightforward and requires minimal configuration.
- Speed: Running the project locally ensures faster development and debugging compared to cloud-based or containerized solutions.
- Flexibility: A local setup provides greater control over dependencies and configurations without additional overhead.

Workflow Process:

Routine and Practices

We follow a structured weekly workflow to ensure efficient development.

- **Task Planning & Management:** Tasks are planned to use our **weekly plan**, which outlines milestones and deadlines.

- **Time Management:** We rely on **weekly check-ins** to assess progress, adjust priorities, and stay on schedule.
- **Collaboration & Communication:** Since our team consists of two developers, we communicate informally but consistently via **GitHub Issues** and direct discussions.

Workflow Stages

- **Planning & Research:** We begin each task by researching solutions and outlining our approach before implementation.
- **Coding & Implementation:** Code is written in **VS Code**, with structured commits to maintain clarity.
- **Testing & Debugging:** We perform manual testing and fix bugs iteratively before merging changes.
- **Version Control:** We use **GitHub** for committing updates, creating branches for major features, and merging changes after review.

Reflection

- Our workflow has been effective in keeping us organized and ensuring steady progress. Using GitHub for version control and tracking progress has streamlined collaboration. One challenge we foresee is maintaining a balance between development and documentation, but we plan to mitigate this by setting aside time for both each week. If needed, we may incorporate **task tracking tools** like Trello to enhance visibility and accountability.