CS673 Software Engineering
Team 3 - Study Buddy
Project Proposal and Planning

Your project Logo here if any

Team Member	Role(s)	Signature	Date
Bohan Lin	Design & Implementation Leader / QA Leader	*Bohan Lin*	9/16/2025
Junzhe Chen	Security Leader	*Junzhe Chen*	9/22/2025
Dexiao Zhang	Design & Implementation Leader / QA Leader	*Dexiao Zhang*	9/22/2025
Qiuting Zhao	Design & Implementation Leader / QA Leader	*Qiuting Zhao*	9/23/2025
Ittoop Shinu Shibu	Configurations Management Leader	*Ittoop Shinu Shibu*	0/23/2025
Melissa Cron	Project Leader and Requirements Leader	*Melissa Cron*	9/24/2025

Revision history

Version	**Author**	**Date**	**Change**
0.1	**Bohan Lin**	**9/15**	**Part 1, 2**
0.2	**Junzhe Chen**	**9/22**	**Part 6**
0.3	**Dexiao Zhang**	**9/22**	**Part 6**
0.4	**Qiuting Zhao**	**9/23**	**Part 3**

0.5	**Ittoop Shinu Shibu**	**9/23**	**Part 5**

1. Overview - Bohan

Study Buddy is a mobile application designed to motivate users to study more effectively while reducing distractions from excessive use of smartphones. The motivation behind this project comes from the increasing challenge students face in maintaining their focus, as smartphones often lead to procrastination and reduced productivity. By combining gamification with study tracking, Study Buddy creates a fun and engaging experience to encourage consistent learning habits.

The purpose of this application is to provide users with both accountability and entertainment. When a user starts studying, they need to keep the app open, which will record the study time. The recorded duration contributes to the growth and development of their virtual companion, the “buddy” in the app. The longer and more consistently the user studies, the more advanced the buddy becomes. If one day of no study time is recorded, the buddy may die. Outside of the study sessions, users can interact with their buddy, play mini-games, and customize its appearance, making the app not only a productivity tool but also a source of enjoyment.

The target users of Study Buddy include students of all ages, self-learners, and anyone who

wants to improve their productivity while reducing screen time on distracting applications. The core functionalities of the app include study session tracking, time-based buddy growth mechanics, interactive mini-games with the buddy, and progress visualization through statistics and achievements. Additional features may include reminders and social sharing.

For the technology stack, the mobile app front end could be developed using React Native to ensure cross-platform compatibility on both iOS and Android. The back end can be implemented using Java or Node.js for handling user authentication, progress data storage and synchronization. A cloud-based database such as MySQL can store user progress and buddy states. For gamified elements, built-in React Native animation libraries will be used.

In summary, Study Buddy integrates productivity tracking with gamification, aiming to transform study time into an enjoyable and rewarding experience. It not only helps users stay focused but also creates a playful environment that reduces the negative impact of smartphone distractions.

2. Related Work - Bohan

Several applications and tools have been developed with the goal of improving focus and reducing distractions such as Forest allow users to plant virtual trees that grow when they stay focused. Similarly, apps like Habitica gamify daily routines and tasks by turning them into role-playing game quests, rewarding users with points and character progression.

While these applications successfully motivate users through rewards and gamified elements, they often lack a more personalized, interactive component that creates emotional attachment. Study Buddy addresses this gap by introducing a virtual companion that grows and evolves directly based on the user's study habits. Unlike existing apps that primarily track time or provide abstract rewards, Study Buddy emphasizes building a relationship with the buddy. The user can not only monitor their productivity but also enjoy playful interactions during breaks. This combination of accountability, entertainment, and companionship distinguishes Study Buddy from existing focus apps.

3. Proposed High level Requirements - Mia

a. Functional Requirements
    i. Essential Features (the core features that you definitely need to finish):
        1. Account & Sync 16-24PH
           As a user, I want to sign up/login and sync my data across devices, so that I can keep the same pet and progress on web, iOS, and Android.
        2. Study Session Engine 28-40PH
           As a user, I want a reasonably planned and scientifically based study schedule and timer, so that I can follow structured study intervals.
        3. Exit Friction 8-12PH

As a user, I want to be prompted by my pet or enter a password (or something else) if I try to quit, so that I resist distractions.

4. Pet Lifecycle 18-28PH
   As a designer, I want users' pets to die if they skip a whole day (or a certain period of time), so that they will feel accountable for daily consistency.
5. Pet Evolution 18-28PH
   As a designer, I want users' pets to evolve after several study streaks, so that they can feel rewarded for long-term discipline.
6. Progress Dashboard & Achievements 18-26PH
   As a user, I want to view my study statistics, streaks, and achievements so that I can understand my progress.
7. Mini games 20-32PH
   As a user, I want to play some mini-games to relax during the rest period, so that I can concentrate more during the later studying period. As a designer, I want users to play some mini games to relax rather than engage in some highly immersive entertainment, so that they won't miss out on learning time due to over-commitment.
8. Cloud Persistence & Anti-Cheat 16-24PH
   As a designer, I want to store session data and validate against server time, so that streaks/evolution cannot be faked by changing device clocks.
9. Reminders & Local Notifications (start/break/return) 10–14 PH
   As a user, I want reminders for session start and break/return, so that I keep a steady rhythm.

ii. Desirable Features (the nice features that you really want to have too):

1. Pet Variations & Cosmetics 14-22PH
   As a user, I want my pets to have different colors/skins, so that I feel attached and motivated.
2. Accessibility & Localization 14-22PH
   As a user, I want to adjust font size, voice mode, theme color, language, etc., so that the app is inclusive.
3. Weekly Summary 8–12 PH
   As a user, I want a weekly summary of time studied and streaks, so that I see my progress.
4. Onboarding tutorial 2–4 PH
   As a user, I want to have a first-run walkthrough of the study/break cycle, so that I can get started faster and understand the various functions.

iii. Optional Features (additional cool features that you want to have if there is time):

1. Leaderboards & Social Sharing 16-26PH
   As a user, I want to share streaks or compete with friends, so that I stay motivated.

2. AI-Driven Pet Personality 20-36PH
   As a user, I want to upload my own pet in the real world to generate my own pet character, so that I can feel more responsible.
3. Advanced Anti-Cheat 10–18 PH
   As a designer, I want anomaly scoring for time jumps and overlapping sessions, so that rewards reflect real effort.

iv. ~~Existing Features (delete this item if your project starts from scratch)~~

b. Nonfunctional Requirements
   i. Security requirements
      1. Authentication & Sessions
      2. Authorization & Data Isolation
      3. Platform Permissions
      4. Privacy & Data Retention

4. Management Plan - Melissa

a. Objectives and Priorities

Our highest priority is creating a working app. This means tracking and storing data on the app, having a simple number-input for the time studied, and creating an algorithm to determine if the user is achieving their goals based on the time studied.

Our second highest priority is making the app welcoming. This means creating a pretty and intuitive interface, adding the visual for the pet, testing for accessibility, and creating a login system that allows the user to share their data across devices.

Our next priority is making the app customizable. Allowing users to set their own goals, setting goals for different classes, and choosing between a variety of pets. This will allow users to use the app better for their specific needs.

Our lowest priority is making the app interactive. This means animating the pet, creating minigames to play with the pet, and allowing AI to make custom pets. This will encourage the user to use the app more and thus study more.

See Requirments Document for more details.

b. Risk Management (need to be updated constantly)

One of the major risks we've identified is settling disputes over design choices. The method we have determined for avoiding this is using a voting system. If the votes are tied (as we have an even amount of people) the deciding vote will be the relevant leader, or if there is no relevant leader the project leader.

Another common concern is with members not finishing their work, either due to mis-assigned work, unclear deadlines, or unclear requirements. This falls on the

individuals to self-motivate and reach out when instructions are unclear. However, failing that, the team leader will give reminders and assign specific tasks to specific people.

Risk Management Sheet Link: **CS673_SPPP_RiskManagement.xlsx**

c. Timeline (this section should be filled in iteration 0 and updated at the end of each later iteration)

Iteration	Functional Requirements(Essential/ Disable/Option)	Tasks (Cross requirements tasks)	Estimated/real total person hours
1	Interface: Displays goal Visual timer to track study time Add Buddy Backend: Store time studied Login using API Create standard for passing information from frontend to backend Stretch: Custom names for pets Custom goal times	Frontend: Rename pages Display goal on home page Polish mind-map Implement mind-map Add timer to studying page Create Buddy Add Buddy to required pages Backend: Store data on device Connect API Create standard for passing information from frontend to backend Stretch: Input for custom pet names Input for custom goal times Store both	90
2	Frontend: Pet visually changes based on goal progress 5 pet options Test platform compatibility Backend:	Frontend: Create many pet visuals Change image based on information from backend Test compatibility and flexible visuals	90

	Algorithm for turning study-time into progress towards goal Connect front-end and back-end Database Stretch: Track multiple goals	for android, apple, and website of multiple sizes Backend: Design algorithm for calculating progress towards goal from study-time Implement algorithm Connect backend to frontend Create database Link database Stretch: Change database to track multiple goals	
3	Frontend: Accessibility Variable app styles Statistics page Backend: Track multiple goals Temporary goals with enddates Stretch: Lock phone option Minigame Animate pets Studying music AI created pets	Frontend: Test for accessibility on all platforms Implement settings for style changes Create page for user to change settings Create page to display user statistics Backend: Change database to track multiple goals Change database to include goal end dates Stretch: Lock phone while on studying page Animate pets Create music Add music Create AI for creating custom pets Create interface for AI	90

5. Configuration Management Plan - Shinu

a. Tools

(In this project, we will use Git and Github as the version control tools. Please also specify any other tools to be used, e.g. IDE tools, CI/CD tools, container tools, SAST or DAST tools, and any other DevOps tools and **AI tools**)

These are the tools we are planning to use for our project:

Version control:

We will be using github to manage the version control for this project.

CI/CD tools:

We will be using github actions for our CI/CD. It is available in github, hence, it can be easily integrated into our project.

Container tools:

We will be using docker to containerise our application for consistent development and production.

SAST/DAST tools:

We will be using github advanced security to scan our code and ensure there are no secrets accidentally kept in our code.

AI tools:

We will use ChatGPT to conduct research and potentially develop our code. We will also Midjourney to develop the little study buddy that we will be coding out for our application.

b. Code Commit Guideline and Git Branching Strategy

(Please briefly describe criteria for the code commitment and the branching strategy used, e.g. what are the branches to be used, how the pull request will be used etc. Here is an article to give you some basic knowledge about different git branching strategies: https://www.flagship.io/git-branching-strategies/)

Our branching strategy will be as follows:

Master: This will contain the code that will be used for production

Develop: This will contain code that will be staging any new changes that occur in the features branch. This branch will later be merged into release.

Feature: This will branch of the development branch. This branch will be used to develop new features.

Hotfix: This branch will be created whenever there are bugs identified in the master and need to be fixed immediately. The branch will merge to master.

Release: This will be branched from the develop branch when development is ready for release.

All code will be reviewed via a pull request before merging to any branch. There will be 2 reviewers for each pull request and the submitter of the pull request must submit a screenshot of test cases that have passed for the PR and have clear

commit messages.

c. CI/CD Plan

(Briefly describe how you plan to continuously integrate and deploy your application).

Continuous Integration (CI):

- Triggered when there is a push or PR to the main or develop branch
- Run linting, black and ruff
- Scan code and run checks for secrets
- Build docker images

Continuous Deployment (CD):

- Triggered when code is merged into the main branch
- Build docker images
- Deploy the application

6. Quality Assurance Plan - Kimi/Elijah

a. Metrics

(Describe the metrics to be used in the project to measure the quality of your software. Each metric should be measurable and quantifiable. Examples of metrics include product complexity (LOC, # of files, # of classes, # methods, cyclomatic complexity, etc.) , defect rate (# of defect per KLOC), # of test cases, test case pass rate, cost (# of person hours used), # of user stories completed, etc. **The result of these metrics should be reported in the progress report/ iteration summary sheet.**)

Metric Name	Description
Test Pass Rate	JUnit unit test pass rate, target > 90%
Code Style Compliance	Checkstyle validation pass rate, target 100%
Core Test Cases	Minimum 10 test cases for core functionality
Cyclomatic Complexity	Static analysis of method complexity to prevent overly complex code.
Defect Density	Number of defects per KLOC, measuring code robustness.

b. Coding Standard

(Describe any coding standard to be used)

Class names: PascalCase (e.g., StudentManager).

Variable names: camelCase (e.g., studentName).

Method names: camelCase starting with a verb (e.g., calculateGrade())

Constant names: UPPER_CASE with underscores (e.g., MAX_SCORE).

Comments: Each class and method must have a brief comment describing its purpose, parameters, and return values.

Commit messages: Must be descriptive, e.g., fix: corrected grade calculation logic.

c. Code Review Process

(Everyone should review all documents to be submitted. Here you will mainly describe how the code review will be done. Who will review the code, e.g. design or implementation leader will review all code or team members review each other's code. Do you use pull requests for the code review? Is there a checklist to help review? What feedback should the reviewer provide?)

All new features and major changes must be submitted through GitHub Pull Requests (PRs).

Each PR requires approval from at least two team members.

Review scope includes:

Correctness against requirements.

Adherence to coding standards.

Adequate test coverage and passing tests.

Absence of potential performance or security issues.

d. Testing

(Both manual testing and automated testing should be considered. Both unit testing and integration testing should be considered. Briefly describe the testing tools/framework to be used, the personnel involved (e.g. the QA leader will focus on

the integration testing and each developer will unit test their own code), when and what types of testing will be performed, the testing objectives, etc)

Tools: JUnit 5

Requirement:1-2 test cases for each core class

e. Defect Management

(Describe the tool to be used to manage the defect (e.g github issues). The types of defects to look at. The actions or personnel for defect management.)

Management Tool:

GitHub Issues will be used for defect tracking.

Defect Types:

Functional defects: Features not working as expected.

Performance issues: Code efficiency not meeting requirements.

Security vulnerabilities: Potential risks to data or access control.

UI/UX issues: Visual or usability-related problems.

Compatibility issues: Problems across different environments

Deal with the issues:

Developer: Fixes reported defects.

QA Leader: Verifies fixes and runs regression tests.

7. AI usage Log

You are allowed and even encouraged to use AI tools to help you generate the project idea, plan it and build it, but you need to clearly describe 1) What tools were used? 2) for what specific tasks and 3) Is it helpful? 4) how did you evaluate or modify AI-generated content? Additionally, you should submit the exported AI chat history as an appendix or share that

with the instructor and facilitators.

Tools	Who	Tasks	helpful	Evaluation/modification	links
ChatGPT	Bohan	Research of related works	yes	ChatGPT is helpful for searching related works. It helps integrate data that we feed and output the most relevant information we need.	https://chatgpt.com/share/68c9a308-97e8-8004-90aa-3afbfbc38edb
Midjourney	Junzhe	Find a good digit pet	yes	Midjourney can create good character.	

8. References

(Any references/citations that you have used)

9. Glossary

(Any acronym used in the document should be explained here)