




Group 20 - Waves

MILESTONE 2: STUDY BUDDY APPLICATION

Karl Matthew Rodriguez
Tasha Gandevia
Wesley Chen



Links

Git Hub Repository Link

<https://github.com/CMPT-276-FALL-2024/project-20-waves>

Website Link

<https://cmpt-276-fall-2024.github.io/project-20-waves/>

Video Presentation Link

https://www.youtube.com/watch?v=X0hT-C6_ung

The Project

Overview

Study Buddy is a web application that allows users to have multiple study tools at their fingertips in a single application. The user will be able to view their upcoming assignment due dates, use an AI chatbot to ask questions, study with AI generated quizzes and track their progression as they hone their skills and improve their knowledge.

Analysis of the Project's Success

Due to the scope constraints of this project, Study Buddy is not as well-rounded as we had set out to make it. Originally, we had planned to make it an application that reduces outside distractions – view your schedule, read your uploaded notes, test your knowledge, expand your understanding with YouTube videos and chatbot interactions, and manage your time with a pomodoro timer.

While some of these features were not implemented, Study Buddy is still successful in being a useful tool to study – viewing your calendar to see upcoming exams, asking chatbot questions on subjects you're unsure of, and testing your understanding with the quiz generator.

During peer testing, there was comments on the functionality of the calendar being very similar to Google Calendar (which was planned) but that individually, it was questioned as to why not just use Google Calendar. This is a very valid question, which upon explaining the purpose of Study Buddy as a whole to be a single application to reduce external distractions, was well received.

Analysis of our Software Development Life Cycle

We decided to use the iterative software development life cycle in our project as we felt this approach was best suited for the project because of the flexibility it offers. We as a group felt that since we only had 3 members, it would be best that we allow ourselves to be able to go back to certain cycles of the project in the event that something is left out or is wrong.

Having less developers can raise the chances of errors so for us to be able to be flexible and go back to previous phases of the project to fix these errors would be very beneficial and for that reason, we chose to use the iterative software development life cycle for our project.

Features

Implemented

Gemini AI API

1. Quiz Generation
 - Generates quiz based on user's input
 - Creates 10 multiple choice questions with four possible answers
 - Visual feedback after submitting the quiz shows correct answers (and users incorrect answers if applicable)
 - Quiz history tab
 - Allows user to view their previous topics and scores
 - Allows user to monitor their progress for improvement
2. ChatBot
 - User can ask questions to AI and receive a response

Google Calendar API

1. Syncs with the user's calendar
 - View existing events
 - View start time / start date, end time / end date, title, notifications from Google Calendar on the in-application calendar
 - Add, Edit, Delete events
 - To create an event, users can click on the calendar to create an event in various methods
 - Clicking on a date in monthly view (populates event creation with date)
 - Clicking on time in weekly / daily view (populates event creation with date and time)
 - Clicking the floating action button (populates event creation with current date / time)
 - Click and dragging to select time (populates event creation with start and end date accordingly)
 - Users can edit their events
 - Clicking an existing event will populate event sidebar with details that can be changed and then saved
 - Users can delete their events
 - Clicking an existing event will open the event to edit, allowing the user to delete it

2. Provides user with notifications

- Notifications in the application are updated from events on Google Calendar
 - The notifications appear to the user from any page within the application
 - The notifications can be edited within the calendar application

Changed Features

Planned: Gemini AI generated flashcards

Actual: Removed

Reason: As the user wouldn't be creating the flashcards, they may not be aware of what the backside would show. We believed that it would be too similar to the quizzes than an actual study tool. By removing the flashcards, we were able to focus more on other functionality and features.

Planned: Gemini AI generated quizzes with difficulty and quantity selection

Actual: Gemini AI generated quizzes

Reason:

Planned: Google Calendar current date/time with time zone selection to track other time zones around the world

Actual: Removed

Reason: Due to reducing the scope, we removed this feature as it was not a core functionality of Google Calendar API.

Planned: Google Calendar – view only

Actual: Google Calendar – view, edit, delete and create

Reason: This allows more functionality for the user to have control over their events within our application.

Overview of CI/CD Pipeline

Deployment and Monitoring

For the **continuous integration** aspect of our CI/CD pipeline, we have included jobs that will run whenever code is pushed to main, or whenever a pull request is made for main. For the jobs, we implemented a code lint checker that runs on ubuntu-latest. This job first checks out the code from the repository, sets up the Node.js environment, installs the needed dependencies, then runs the linter through the code. We also added an automated testing job which runs on ubuntu-latest. This job first checks out the code from the repository, sets up the Node environment, installs the dependencies required, then runs the test scripts created. This job will run on multiple versions of Node.js.

This workflow is to ensure the code is properly formatted and that it passes its tests before fully being pushed to prevent inconsistency and errors. For the **continuous deployment** part of the pipeline we decided to use a Hugo workflow file that allows you to deploy to build the project, then deploy it to Github pages. The website is deployed each time code is pushed to main which happens after code passes the checks in the continuous interaction section of the pipeline.

When the continuous integration part of the pipeline is finished, the continuous deployment workflow first checkouts the code from the repository, configures the Github pages environment, installs the required dependencies, turns the project into a deployable state, uploads the project to Github pages, then deploys it to Github pages.

Testing

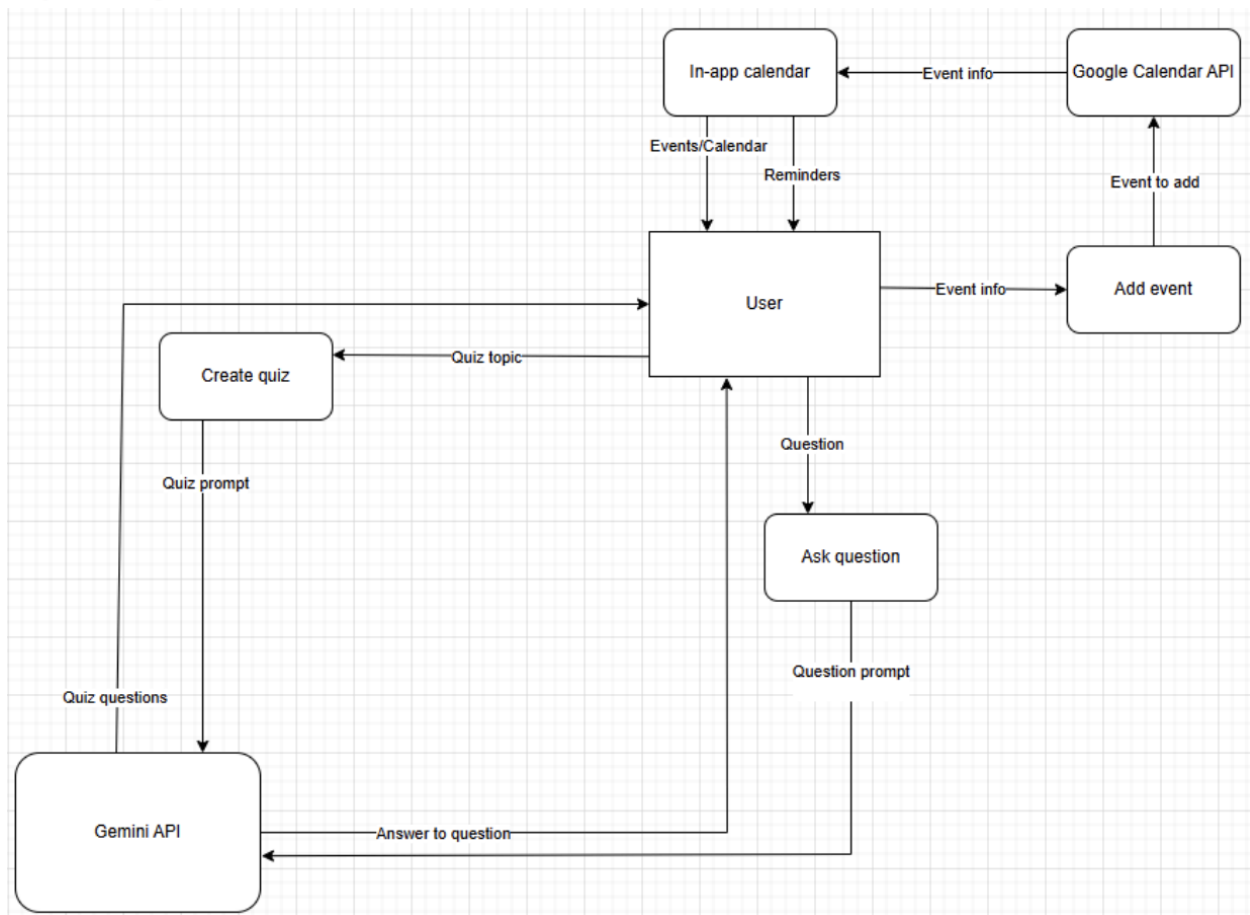
In our project, our testing consisted of both manual and unit testing. We implemented unit tests for each function we created which checks if correct values are being stored and if what's being displayed to the screen in the UI is correct. For testing the correctness of values, we compare those values with the expected values to see if they match using the Jest testing framework for unit testing.

We also test to see if what's displayed is correct, like the text being shown. For this, we check if what gets displayed to the screen matches what we expect to be displayed. This ensures that the information that's being displayed to the user is correct. This is also done using the Jest framework.

For the manual user testing, we had users complete tasks without assistance to gauge how easy-to-understand our project is to use. The feedback collected gave us insight on what to modify to better improve the user experience and user interface.

Project's Architecture

DFD



MVC

Calendar Page

Model	View	Controller
<ul style="list-style-type: none">-Remove event-Add event-Get events	<ul style="list-style-type: none">-Calendar-Events-Sign in/out	<ul style="list-style-type: none">-Change home-Sign in-Add events-Delete events

Quiz Generation Page

Model	View	Controller
<ul style="list-style-type: none">-Get quiz topic-Make quiz	<ul style="list-style-type: none">-Quiz questions-Quiz prompt-Past quiz results	<ul style="list-style-type: none">-Change home-Generate quiz-Get past quiz results

Home Page

Model	View	Controller
<ul style="list-style-type: none">-Send AI messages-Get AI Response-Switch to Calendar page-Switch to Quiz Generation page	<ul style="list-style-type: none">-To-Do list-AI Messaging	<ul style="list-style-type: none">-Toggle To Do-Toggle AI-Change to calendar view-Change to quiz view-Message Gemini AI

List of Bugs

GitHub Issue	Recreate	Problem	Severity
<u>Modal location</u>	[Confirmation modal to delete an event appears in the center of the screen] vs [Confirmation modal when an event is deleted appears in the top left corner]	Modal is inconsistent in location	Medium – While a UI/UX issue and is functional as is, it takes away from the experience and flow
<u>Returning to Website</u>	[Remaining signed in but navigating away from the page for many hours]	When the user navigates away from the website and returns, the page will think they are signed in, but not have any save data, resulting in errors	Medium – Unless the user is aware of it, it can be difficult for them to troubleshoot and could lead to frustrations
<u>Gemini API returning HTML causes error</u>	[Ask code help with HTML]	Code replies from ChatBot come in as formatted with HTML, rather than showing the actual code	Medium – Depending on the usage by the user. It can be high if they are a CS student learning HTML though.
<u>No user feedback if ChatBot is processing a request</u>	[Ask a question to ChatBot that requires a longer processing time – IE: How do you determine the integral of a graph]	User doesn't know if they input their question or not. They aren't aware the ChatBot is processing their request	High – It leaves the User questioning if they did the process right or if there's something wrong with the website
<u>User sign in and out button</u>	[Sign in on one page, navigate to another]	The sign in / out button defaults to offer user to sign in on page load until it initializes the user and switches to the sign out button	Low – this is more of a distraction than an issue
<u>To Do List doesn't store items</u>	[Add items to the To Do List on Home => Navigate away and return]	User must remain on the Home page to keep their To Do List	Critical – users will lose any to do lists when navigating from that page, resulting in lost information and important lists

Looking Forward

Potential Future Implementation

1. Future Google Calendar implementation could include:
 - Multiple calendars integration
 - Users can add, edit, view events from multiple calendars
 - Users can view all calendars as events on the calendar with the agenda tabs being filters to view upcoming activities for specific calendars
 - Colour coding the events on the calendar based on user's Google calendar settings
 - Including all events, not just primary, on the calendar grid

Potential Future Improvements

1. Google Calendar could be improved by allowing for AI generated study blocks
 - AI can view calendar and block time for studying based on preferences ("I like to study in the morning", "I need 15-minute blocks, 4 times a day instead of one hour solid")
2. To Do List
 - When creating tasks, they save to the User.js object so the user can navigate from the page and it will repopulate when the user returns
 - This could be implemented with long term storage, not just local storage which is currently implemented

Lessons Learned

Team Work

A lesson we learned is that when working on a group project, it is not just about yourself, it is about everyone involved. When one person makes a mistake, it can affect everyone. This emphasizes the fact that each member involved must uphold their work and take responsibility for their actions. We had to make sure deadlines were set so that the person in charge of combining everything together would have sufficient time to do so. If one person were to delay themselves, then the whole thing would be delayed.

Diligence and Responsibility

Another lesson we learned is that one should always remain diligent. It should not matter what you are doing, you should always remain diligent. Diligence is a vital aspect of any project and it will benefit everyone since if tasks are done earlier it allows others to work off of your work and collaborate easily. During development, some of us were not present during the peer review which easily could have been avoided if we were more diligent and responsible. This prevented us from being able to effectively improve our project.

Collaboration

Throughout the development process, we got to understand the value of group work. As we head towards the deadline of our project, we realized that our project could have been improved if we had been more collaborative. Rather than each going our own paths and then trying to reconnect, we work together on the same path and thus building a more solid product.

Planning is Critical

Another thing we learned in the development process is the importance of the planning phase. The planning phase is a crucial component to any project and should never be overlooked. It offers the chance for everyone to be on the same page and figure out the project. During our project, we had to redecide on a lot of ideas which overall took more time than it would have if we just put more effort during the design process.

Communication

One last lesson we learned, and think is important is the value of communication. Communication is the most important skill in any collaborative project as it helps the entire team make progress and understand the requirements. During the production of our project, we found that communicating with each other really benefited the creation process of our project and our desired product.

Challenges

Some challenges we faced were during the development process and testing. We encountered heavy difficulty with the coding process as some of us were unfamiliar with web development and the learning process as well as the development took a lot of our time. Another problem we faced was during the testing phase. Due to the lack of testing for our application, we could not accurately pinpoint the errors in our design. Working with the data returned by Gemini to create the quiz because Gemini wouldn't return the data in a consistent format, even if the prompt was the exact same. Sometimes it will return it as a string, and other times as an object. This caused errors to appear when the API data was being handled because the code was set up to handle it as a string value, but string functions don't work with objects which led to the errors. This issue was resolved by ensuring that before moving onto the next part of the code, the API always returns the data in string format. Another challenge was figuring out how to design the user interface while adhering to the 10 heuristics. This was a challenge as it was hard to work on specific heuristics while maintaining the other heuristics. Sometimes modifying one part of the UI to improve a specific heuristic led to other heuristics being violated. This challenge was overcome by spending time testing the interface and each time focusing on a specific heuristic to ensure that its present in the UI.

Project Takeaway

The key takeaways from this project are time and effort that is required by any project, what it means to work effectively in a group, and the importance of each process throughout the

project's lifecycle. Since this is the first time the majority of our group has worked on any projects professionally, we struggled and the problems of our lack of experience became apparent. However, our group collectively put in a lot of effort and was able to push through. We were able to effectively communicate on a regular basis and keep up with the workload of the project. We looked back at our work and realized that there is a lot of room for improvement. There could have been many things that we could have done differently near the beginning of the lifecycle such as the planning phase, but due to inexperience, we did not know the value that the planning process offered. Overall, this project has significantly improved our way of approaching projects and group work.

Appendix

Group Member Contribution

Contribution List – Karl

- Setup the CI/CD pipeline with linting, testing, and deployment to Github pages jobs
- Setup the HTML, CSS, and JavaScript files for initial phase of project development
- Designed and developed the user interface of the quiz generation page
- Developed the AI quiz generation feature which includes setting up Gemini API, developing the logic behind the quiz functionality and how the UI will change as a result, created error messages, and implemented a loading screen
- Created a quiz history section where previous quiz topics entered along with their scores out of 10 are displayed and stored in the browsers local storage so that it doesn't disappear upon refreshing the page
- Documented code
- Resolved various bugs
- Wrote tests for the quiz feature that test both the data being stored and if what's being displayed to the UI is correct
- Made adjustments to the user interface of the home page
- Assisted other group members with various issues they faced during the development process
- Reviewed pull requests and gave suggestions along with helped fix errors
- Created help and documentation file for to-do list, AI chatbot, and quiz generation features
- Added help and documentation button to the nav bar that opens the documentation pdf when clicked

Contribution List – Tasha

- Created the calendar system UI prototype
- Created calendar system implementation

- Implemented user log in to sync with google calendar
- Separated calendar and user functionality to create site-wide User functionality with logging in and save data
- Designed website header and implemented it across all pages
- Implemented sitewide navigation in header code
- Created overall UI scheme for the website
- Resolved various bugs
- Logged various bugs
- Assisted with help documentation
- Peer reviewed and monitored peer-reviewing in class session
- Monitored user testing from external sources (family, friends)
- Compiling reports for all milestones
- Documented code

Contribution List – Wesley

- Created AI question bot
- Created to do list
- Created icons for navigation
- Created and implemented favicon
- Tested application
- Reviewed code for redundancy
- Made mid-fidelity prototype for the website

Peer Testing Feedback

During Peer Testing (November 26th) we had three classmates test the calendar part of our project.

Peer A:

Recommendations:

1. Colour coding and having all events on the calendar (with the agenda tabs acting as filters)
2. Making user more aware that they can click and drag to select a timeframe to auto-populate the event creation

Reception:

1. This idea was explored but was determined to be too time consuming for project completion. A branch with some basic logic and implementation is on GitHub, but the idea had to be discarded to avoid further time creep.

Peer B:

Recommendations:

1. Allowing user to log in within the same window, rather than the popup window, would provide a better experience

Reception:

1. We decided that was out of scope and not something we wanted to explore at the time

Peer C:

Recommendations:

1. Change the colour of the button to create events (was a light blue, resembled an inactive button at first glance)

Reception:

1. Changed to dark blue with a light blue for hover effect

Additional Information

Account Information

To test Study Buddy, a dummy account was created:

Project.20.Waves

PW: CMPT276Waves

Please feel free to use this account as needed for testing calendar functionality. Events are not sensitive and can be changed and deleted freely.

Group division of work spreadsheet

https://docs.google.com/spreadsheets/d/1e8y_XtjkwU4ZIZtovzaM5bRjwNMXcrSpcO2RqQhGG3c/edit?gid=662553510#gid=662553510