PROJECT PLAN

PulseCheck

Trinity Dhillon, Daniel Sirias, Michael Campos, Brandon Budhan, Peter Georgaklis, & Zaira Garcia

2025-02-24 v1.0.0

Table of Contents

1. Introduction

PulseCheck is an interactive quiz and polling web application designed to enhance engagement and real-time feedback in large lecture hall environments. The platform enables instructors to create polls and quizzes that students can respond to instantly using their devices, fostering dynamic interaction and immediate knowledge assessment.

This project plan outlines the activities, schedules, resource allocations, and quality assurance measures to guide the development of the system. By utilizing Firebase for authentication, data storage, and real-time capabilities, the system will provide a scalable and user-friendly solution for both instructors and students.

The objective of this plan is to ensure that all project activities align with the requirements, mitigate potential risks, and deliver a polished, fully operational system by the presentation week starting on Tuesday, April 29, 2025. By following the principles of the CMM(i) methodology, this plan also includes traceability to the requirements specification, risk management strategies, and a robust testing and configuration management approach.

2. Project Schedule and Resourcing

The development PulseCheck follows a structured timeline, ensuring efficient progress from planning to deployment. Given the complexity of managing 76 tasks, we adopted a black-box abstraction approach for task visualization, grouping tasks into broader categories to simplify tracking and management.

Every activity of the project is outlined below. The duration of each activity is in days, their dependencies on other activities (e.g setting up the development environment, firebase, and

learning React), and resources (the number of team members working on each activity) are also listed.

Due to the complexity of the project and the tasks the development team enlisted in the software requirements specification. Constructing an activity chart of 76 tasks (see Appendix A) would be difficult to visualize in a 2D activity chart. Instead, the chart will display black-box tasks (tasks that may be grouped together) while maintaining necessary details to track project dependencies and timelines as seen in Section 3. Activity Chart.

The table includes the following:

- Task - the name of activity, followed by sub-tasks that are referenced from the software requirement specification.
- Duration (in Days) - The estimated time required for completion.
- Dependencies - Tasks that must be completed before this task can begin.
- Resources - Assigned team members to work on the task.

For clarification, *Brando* is an alias for Brandon in our development team.

Task	Duration	Dependencies	Resources
DEV-1 ***Setup Development Environment***	2	NONE	Michael
DEV-2 ***Setup Github/SSH***	1	DEV-1	Brando Daniel Michael Peter Trinity Zaira
DOC-1 ***Upload SRS to Github***	1	DEV-2	Trinity
FB-1 ***Setup Firebase***	2	DEV-1	Michael Trinity

RCT-1 ***Learn React***	9	DEV-1	Brando Daniel Trinity Peter Zaira
SR-1 ***Style Requirements***	1	DEV-1	Michael
LU-A ***Splash Page*** *LU-[1:6]*	5	DEV-1	Brando Daniel Michael Peter Trinity Zaira
NM-1 ***App Bar***	1	DEV-1	Brando Daniel Michael Peter Trinity Zaira
NM-A1 ***User Auth Menu Items*** *NM-[8-9]* *Login* *Register*	1	NM-1	Brando Trinity
NM-B **Home Menu Items** *NM-[2:7]* *Home* *Features* *About* *FAQs* *Privacy Policy* *Terms of Service*	1	NM-1	Brando Daniel Peter
NM-C ***Polish System*** *NM-10: Responsive Design* *NM-11: Consistent Styling* *NM-12: Access Control*	7	DEV-1	Brando Michael Peter
NM-13 ***Error Handling*** ***(Unauthorized Content)***	7	DEV-1	Brando Daniel Michael Peter Trinity Zaira
NM-14 ***Footer Links***	1	UD-A	Brando

NM-A2 ***User Menu Items*** *NM-[15:20]*	1	NM-1	Brando Daniel Michael Trinity Zaira
GJP-A ***Guest Join Poll*** *GJP-1: Guest Join Poll Form* *GJP-1.1: Join Poll Button*	14	LU-A	Brando Daniel Michael Trinity Zaira
GJP-B *GJP-1.2* *Create an Account Button*	1	NM-A1	Brando
UR-A *UR-1: User Registration* *UR-2: Password Validation* *UR-3: Error Handling*	5	GJP-B	Michael Peter Trinity
UR-B *UR-4: Register with Google*	2	UR-A	Zaira
UR-C *UR-5: Register with Apple*	2	UR-A	Zaira
UL-A *UL-1: User Login Form* *UL-2: Error Handling*	5	UR-A	Michael Peter Trinity
UL-B *UL-1.1: Login with Google*	2	UR-B	Zaira
UL-C *UL-1.2: Login with Apple*	2	UR-C	Zaira
PPTS *PP-1* *TOS-1*	1	LU-A	Peter
UD-A ***User Dashboard (Controls)*** *UD-2: Create Poll Button* *UD-3: Join Poll Button*	3	UR-A	Brando Daniel Michael Peter
UD-B ***User Dashboard (Visuals)*** *UD-1: Most Recent Poll Chart* *UD-4: Most Recent Polls*	5	UD-A	Daniel Michael Zaira
JP-A *JP-1: Join Poll Form* *JP-2: Join Poll Button* *JP-3: Error Handling*	7	UR-A	Daniel Michael Trinity

PRM-A *PRM-1: Results Tabs* *PRM-3: Poll List*	7	UR-A	Daniel Michael Zaira
PRM-B *PRM-2: Filter Poll Field*	3	PRM-A	Trinity Zaira
UPM-A ***Profile Page*** *UPM-[1:5]*	3	UR-A	Michael Trinity
UQS-A ***User Post Quiz Statistics/Storage*** *UQS-1: Post Quiz Statistic Display*	20	UQCH-A	Brando Daniel Michael Zaira
UQS-B ***User Post Quiz Statistics/Storage*** *UQS-[2-5]* *Save Button* *Export button* *Viewing Past Quizzes* *Saved Quizzes Drop Menu Items*	20	UQCH-A	Brando Daniel Michael
UQC-A ***User Quiz Creation*** *UQC-1: Question Settings* *UQC-1.1: Question Input*	10	UD-A	Brando Daniell Michael Peter
UQC-B ***User Quiz Creation*** *UQC-1.2: Upload Images*	10	UQC-A	Brando Michael
UCNH **User Quiz Controls** UCNH-1: Waiting Room UCNH-2: Viewing and Answering Questions	14	GJP-A	Brando Daniel Michael Peter
UQCH-A **User Quiz Controls Hosts** UQCH-2: Initiating Quiz and Waiting Room UQCH-4: In-Quiz View (Anonymous) UQCH-5: Quiz Termination	14	GJP-A	Brando Daniel Michael Peter Trinity Zaira
UQCH-B **User Quiz Controls Hosts** UQCH-1: Accessing Previously Created Quiz UQCH-3: In-Quiz View	20	UQCH-A	Brando Daniel Michael Peter Trinity

3. Activity Chart

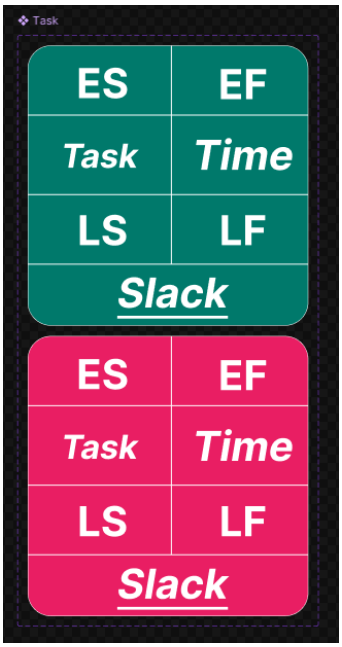


Figure 1. Activity Boxes

Boxes on the activity chart are colorized teal and pink according to the category they represent. Teal boxes on the chart are a representation of project tasks that have subsequent tasks dependent on them. In comparison, pink boxes represent tasks that do not have relying dependencies to them; once completed they approach the “end” of their path.

Each box contains the following fields:

- ES - Early Start
- EF - Early Finish
- Task - ID of the Task
- Time - The amount of time units it takes to complete a task. In this case, time is measured in days.
- LS - Late Start
- LF - Late Finish
- Slack - Duration that a specific task can be delayed without impacting the overall time of project

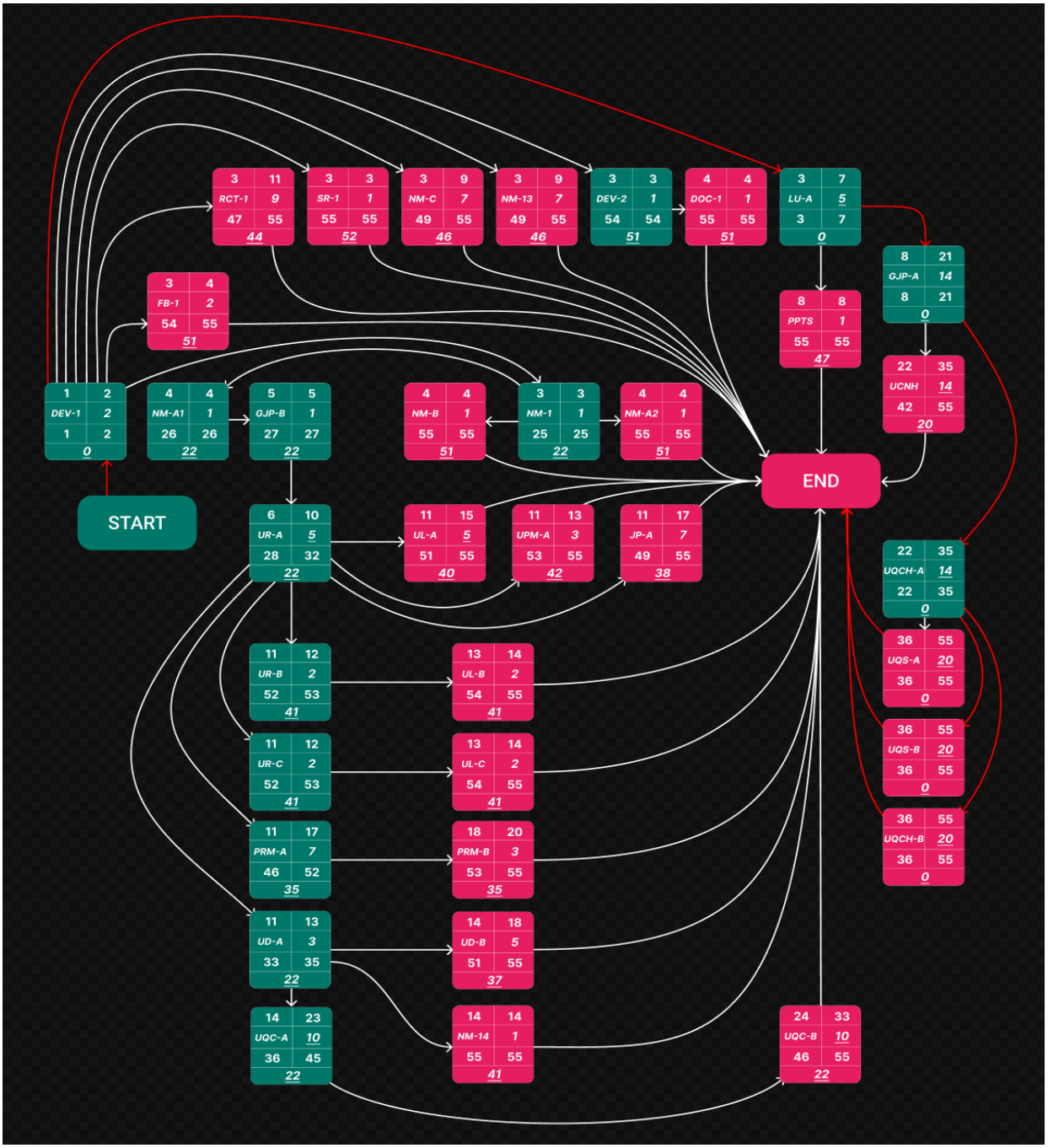
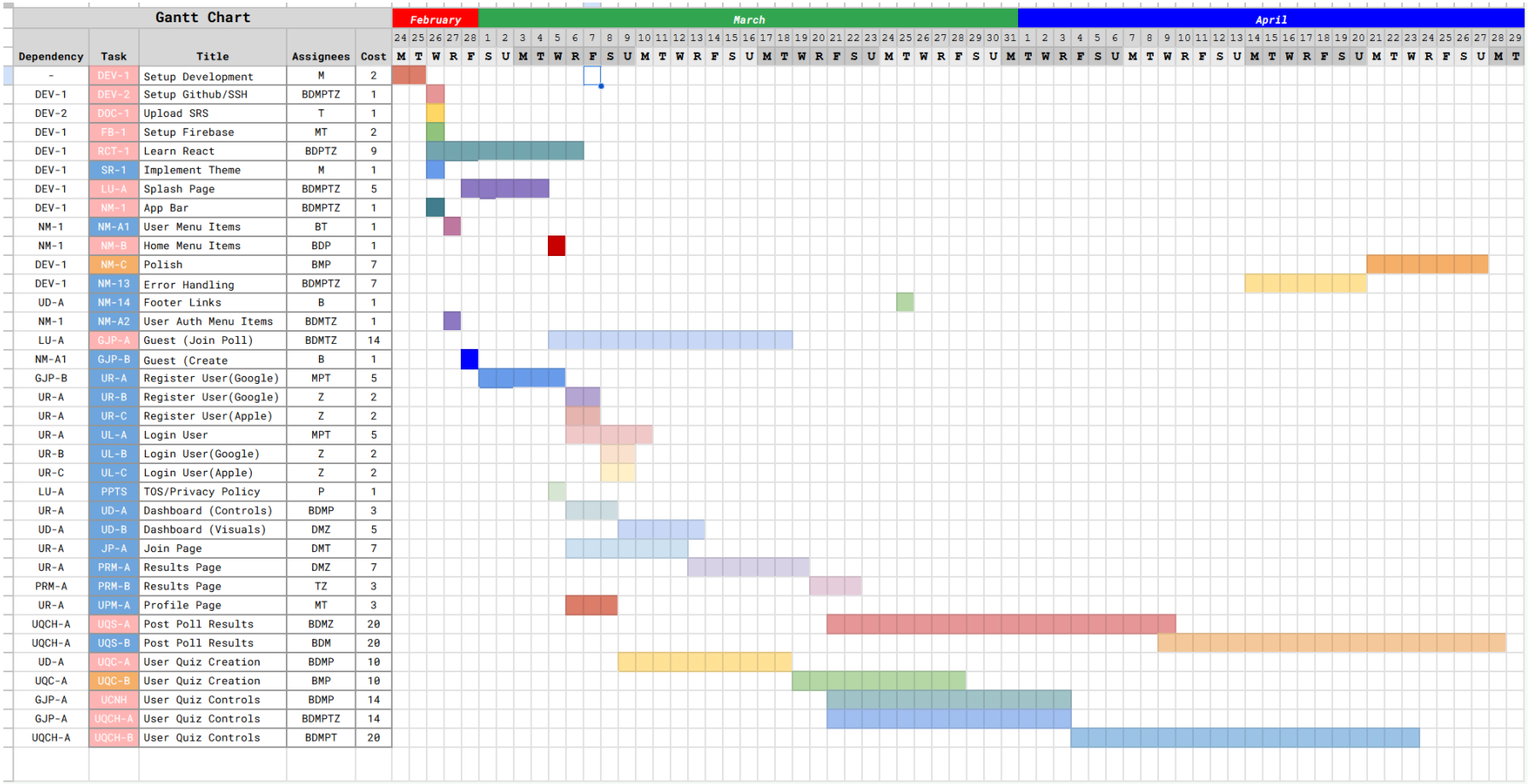


Figure 2. The Activity Chart

- Arrows colored in red represent the critical path.
- All boxes represent tasks that will be completed throughout the duration of the project.



Gantt Chart

Dependency	Task	Title	Assignees	Cost
-	DEV-1	Setup Development	M	2
DEV-1	DEV-2	Setup Github/SSH	BDMPTZ	1
DEV-2	DOC-1	Upload SRS	T	1
DEV-1	FB-1	Setup Firebase	MT	2
DEV-1	RCT-1	Learn React	BDPTZ	9
DEV-1	SR-1	Implement Theme	M	1
DEV-1	LU-A	Splash Page	BDMPTZ	5
DEV-1	NM-1	App Bar	BDMPTZ	1
NM-1	NM-A1	User Menu Items	BT	1
NM-1	NM-B	Home Menu Items	BDP	1
DEV-1	NM-C	Polish	BMP	7
DEV-1	NM-13	Error Handling	BDMPTZ	7
UD-A	NM-14	Footer Links	B	1
NM-1	NM-A2	User Auth Menu Items	BDMTZ	1
LU-A	GJP-A	Guest (Join Poll)	BDMTZ	14
NM-A1	GJP-B	Guest (Create	B	1
GJP-B	UR-A	Register User(Google)	MPT	5
UR-A	UR-B	Register User(Google)	Z	2
UR-A	UR-C	Register User(Apple)	Z	2
UR-A	UL-A	Login User	MPT	5
UR-B	UL-B	Login User(Google)	Z	2
UR-C	UL-C	Login User(Apple)	Z	2
LU-A	PPTS	TOS/Privacy Policy	P	1
UR-A	UD-A	Dashboard (Controls)	BDMP	3
UD-A	UD-B	Dashboard (Visuals)	DMZ	5
UR-A	JP-A	Join Page	DMT	7
UR-A	PRM-A	Results Page	DMZ	7
PRM-A	PRM-B	Results Page	TZ	3
UR-A	UPM-A	Profile Page	MT	3
UQCH-A	UQS-A	Post Poll Results	BDMZ	20
UQCH-A	UQS-B	Post Poll Results	BDM	20
UD-A	UQC-A	User Quiz Creation	BDMP	10
UQC-A	UQC-B	User Quiz Creation	BMP	10
GJP-A	UCNH	User Quiz Controls	BDMP	14
GJP-A	UQCH-A	User Quiz Controls	BDMPTZ	14
UQCH-A	UQCH-B	User Quiz Controls	BDMPT	20

Figure 3. The Gantt Chart

The Gantt Chart depicted in Figure 3 visually represents the project's tasks on a projected timeline. The bars corresponding to each task are colored solely for visual distinction. The number of boxes assigned to each task corresponds to the scheduled duration of that task in days.

Tasks on the leftmost chart are labeled with one of three colors, indicating their priority. Pink tasks are designated as high-priority tasks, orange as medium priority, and blue as low priority.

4. Risks & Mitigation

4.1 Risks

Throughout the development of PulseCheck, various risks may occur that may impact the project's progress and/or functionality. These risks range from technical challenges to team-related obstacles. The following table outlines key risks along with their potential effects on the project. Identifying these risks early allows the team to anticipate challenges and develop strategies to mitigate their impact.

Risk	Effect	Description
Lack of JavaScript/TypeScript Knowledge Among Team Members	Catastrophic	Coding languages used throughout the project, if team members are not comfortable using them, it majorly impedes progress as nothing can get done.
Lack of React Knowledge Among Team Members	Catastrophic	React is the primary library for building the User Interface, enabling efficient real-time rendering and state management. A lack of knowledge in React among team members may result in difficulties creating UI components, inefficient coding practices, and delays in overall development.
Lack of Material-UI Knowledge Among Team Members	Serious	Material-UI (MUI) is the primary framework for React in this project, providing pre-built components to streamline development. Without a solid understanding of MUI, team members may struggle with styling, component usage, and theme customization, leading to delays and inconsistent UI implementation.
Firebase Service Disruption or Limitations	Tolerable	Firebase is used as the back-end service for hosting the system, handling authentication, real-time database, and storage. If Firebase experiences downtime, API limitations, or unexpected changes in service, it may temporarily impact application functionality. Firebase is the back-end service for hosting the system, user authentication, real-time database operations, and storage. If Firebase experiences downtime, API limitations, or unexpected changes in service, it may temporarily impact application functionality. Additionally, the team has limited experience with Firebase, which could slow down troubleshooting and implementation.

Github/SSH Keys	Tolerable	Allows for version control, and sharing code between team members. Without the ability to share code it could lead to major confusion on who is doing what and how much progress has been made, making it difficult to put the project together without Github.
Yarn	Tolerable	Yarn is the package manager used for managing JavaScript dependencies. Provides efficient dependency resolution and faster package installations, any issues (such as version conflicts or dependency errors) could slow down the development process.
Communication Risks	Catastrophic	If communication between team members is unclear or inconsistent, it may lead to a misunderstanding of requirements, delays in decision-making, or incomplete task execution.
Task Overload	Serious	If the development team takes on too many tasks, productivity may drop, leading to delays and difficulty in maintaining code quality. Task overload can also impact morale and overall project efficiency.

4.2 Mitigation

To mitigate the knowledge gaps and technical risks, the team will utilize official documentation and supplementary online resources for each framework, tool and programming language. By utilizing the following resources, the team will gain the knowledge required to address any technical challenges that may arise during the project's development.

Risk	**Strategy (Mitigation Plan)**

Lack of JavaScript/TypeScript Knowledge Among Team Members	• Learning the languages referring to (but not limited to) the documentation for both languages.
	
	• JavaScript:
	  ○ MDN Web Docs (https://developer.mozilla.org/en-US/docs/Web/JavaScript) - Detailed documentations for JavaScript features and syntax.
	  ○ JavaScript.info (https://javascript.info/) - In-Depth JavaScript tutorials
	  ○ ECMAScript Specification (https://tc39.es/ecma262) - Official language specification
	• TypeScript:
	  ○ TypeScript Documentation (https://www.typescriptlang.org/docs/) - Official TypeScript Documentation
	  ○ TypeScript Handbook (https://www.typescriptlang.org/docs/handbook/intro.html) - Introductory guide to TypeScript for developers
	  ○ DefinitelyTyped Repository (https://github.com/DefinitelyTyped/DefintelyTyped) - A community-maintained library of TypeScript type definitions for JavaScript.

Lack of React Knowledge Among Team Members	• Require team members to refer to React Documentation (official guide and API documentation available at https://react.dev/) and the React TypeScript Cheat Sheet (best practices at https://react-typescript-cheatsheet.netlify.app/). • Encourage the use of additional resources such as YouTube tutorials, articles, and online courses to reinforce learning. • Organize internal training sessions and code reviews to share best practices and ensure consistency across the team. • Establish mentorship or pair programming arrangements where experienced developers guide those new to React.
Lack of Material-UI Knowledge	• Require team members to review Material-UI documentation

Among Team Members	(https://mui.com/) and explore example implementations. ● Encourage learning through YouTube tutorials, blog articles, and community forums. ● Promote hands-on practice by building small UI components before integrating them into the main project. ● Conduct code reviews and UI consistency checks to ensure proper use of Material-UI components.
Firebase Service Disruption or Limitations	● Monitor Firebase Status Dashboard for service outages. ● Upgrade Firebase plan if needed, utilizing the $300 credit deal for 90 days avoiding hitting limits. ● Improve team knowledge by referring to the official documentation of Firebase (https://firebase.google.com/docs and use tutorials or community resources for learning. ● Use error handling and retry mechanisms for API calls. ● Upgrade Firebase plan if necessary utilizing the $300 credit deal for 90 days to handle increased demand.
Github/SSH Keys	● Create a Github repository for the project and require each member of the team to join. ● Require everyone to set up their SSH Keys for the project in order to be able to share and download changes made to the code with ease. ● Require team members to be familiar with git commands (https://git-scm.com/doc) ● Require team members to be familiar with Github's features (https://docs.github.com/en)
Yarn	● Use the latest stable version of Yarn (v4). ● Use lockfiles (yarn.lock) to maintain consistent dependencies across environments for all team members. ● Document all necessary scripts and commands for development, production, and linting in README.md to ensure smooth execution. ● Use community resources and official Yarn documentation (https://yarnpkg.com/) for troubleshooting.
Communication Risks	● Use Discord or other collaboration tools for regular updates. ● Maintain proper documentation in an accessible place (Google Docs, Github Issues, Github Projects). ● Conduct regular check-ins and stand-ups to ensure alignment. ● Use when2meet app to know all members from the development team availability to schedule meetings.
Task Overload	● Prioritize tasks based on project goals and deadlines, focusing on essential features first. ● Use Github Issues and Projects to track task status and workload distribution. ● Drop non-essential tasks if the workload becomes unmanageable. ● Hold regular check-ins to assess progress and reallocate work if necessary.

	○ Non-essential tasks include the implementation of user accounts. ● Use When2Meet to schedule working sessions based on availability of team members.

5. Quality Assurance Planning & Configuration Management Strategy

5.1 Quality Assurance Planning

Quality assurance planning for the system will include both preventive and reactive actions. These include following a universal set of coding conventions, conducting regular testing and using external tools. Standard coding conventions include consistent naming, and appropriate syntax in accordance with TypeScript and React standards. The development team will also keep descriptive documentation of the functionality of the code written. Following these conventions is a very important preventative measure for the product; increasing transferability, maintainability and reducing bugs in the code. An important aid to help enforce these coding standards is ESLint. ESLint is an external tool that identifies syntax errors in TypeScript code and fixes them. This tool also allows its users to customize their code style requirements, meaning that the development team is able to fine tune the tool to the system's requirements. An additional tool that will be used for quality assurance is Prettier. Prettier is a code formatter, which will allow code format to stay consistent and transferable through the product. These tools can be accessed by developers by running its respective command on the source code.

Frequent testing will also be conducted throughout the development process as a way to protect the system's integrity. An external tool that will be used for unit testing is Vitest. Vitest is a testing software that facilitates code coverage, mocking, writing, and running customizable tests among other things. Vitest will be configured to run on watch mode, which will enable affected tests to rerun each time changes are made to the code. Additional tests that will be done manually, include browser tests and merge tests. Browser tests will ensure that features implemented are responsive and can be viewed across different browsers. The browsers that must be covered are Edge, Firefox, Chrome, Safari. Merge tests will be conducted by

developers and peer reviewers to certify that tested code can merge without compromising other components.

A tool that will be used to handle existing complications is Github Issues. This feature allows the development team to organize issues, assign members to specific tasks, and plan resolutions. Github issues will be used by developers to handle instances in which changes in the code do not pass its tests or need adjustments. Additionally, developers will hold weekly meetings to provide quality assurance status updates on their current task. These meetings will also be used to plan for future tasks and plan for adjustments on the system.

5.2 Configuration Management Strategy

The development team will use Git with Github for version control throughout the duration of the development phase. To ensure the integrity of the final production code, all changes must go through pull requests and code reviews before merging into the main branch. Manual code reviews will be requested through Github's pull request feature; where new code is reviewed by another developer before being integrated into the production build.

For task management and collaboration, the development team will use Github Projects. This feature allows the team to plan tasks, organize issues, and facilitate project management. The development team will also be able to view everyone's past contributions, current tasks, and tasks that are currently in review.

Appendix A

Task	Duration	Dependencies	Resources
DEV-1: Setup Development Environment	2	-	Michael
DEV-2: Setup Github/SSH	1	DEV-1	Daniel Zaira Trinity Brando Michael Peter
DOC-1: Upload SRS to Github	1	DEV-2	Trinity
FB-1: Setup Firebase	2	-	Michael Trinity
RCT-1: React	9	-	Daniel Zaira Trinity Brando Peter
SR-1	1	DEV-1	Michael
LU-1: Splash Page Access	4	DEV-1	Daniel Zaira Trinity Brando Michael Peter
LU-2: Brief Explanation	1	LU-1	Brando
LU-3: Get Started Button	1	LU-1	Daniel
LU-4: About Section	1	LU-1	Trinity
LU-5: Features Section	1	LU-1	Peter
LU-6: FAQs Section	1	LU-1	Zaira
NM-1: App Bar	3	DEV-1	Daniel Zaira

			Trinity Brando Michael Peter
NM-2: Home Menu Item	1	NM-1	Peter
NM-3: Features Menu Item	2	NM-1	Brando
NM-4: About Menu Item	2	NM-1	Michael
NM-5: FAQs Menu Item	2	NM-1	Zaira
NM-6: Privacy Policy Menu Item	1	NM-1	Daniel
NM-7: Terms of Service Menu Item	1	NM-1	Daniel
NM-8: Register Menu Item	1	NM-1	Trinity
NM-9: Login Menu Item	1	NM-1	Trinity
NM-10: Response Design	*63*	-	Daniel Zaira Trinity Brando Michael Peter
NM-11: Consistent Styling	*63*	-	Daniel Zaira Trinity Brando Michael Peter
NM-12: Access Control	*63*	-	Daniel Zaira Trinity Brando Michael Peter
NM-13: Error Handling	*63*	-	Daniel Zaira Trinity Brando Michael

			Peter
NM-14: Footer Links	1	LU-1 UD-1	Brando
NM-15: Dashboard Menu Item	1	NM-1	Daniel
NM-16: Join Poll Menu Item	1	NM-1	Zaira
NM-17: Results Menu Item	1	NM-1	Brando
NM-18: Profile Menu Item	1	NM-1	Trinity
NM-19: Logout Menu Item	1	NM-1	Daniel
NM-20: Avatar Icon	3	NM-1	Michael
GJP-1: Guest Join Poll Form (refer to figure 3P/D)	3	LU-1	Daniel Zaira
GJP-1.1: Join Poll Button	7	GJP-1	Daniel Trinity Brando Michael
GJP-1.2: Create an Account Button	1	GJP-1	Brando
UR-1: User Registration Form (refer to figure 5P/D)	3	-	Peter
UR-2: Password Validation	3	UR-1	Peter
UR-3: Error Handling	5	UR-1	Trinity Michael
UR-4: Register with Google Button	3	UR-1	Zaira
UR-5: Register with Apple Button	3	UR-1	Zaira
UL-1: User Login Form	1	-	Peter
UL-1.1 User Login with Google Button	3	UL-1	Zaira

UL-1.2 User Login with Apple Button	3	UL-1	Zaira
UL-2 Error Handling	2	UL-1	Michael Trinity
PP-1	2	NM-14	Peter
TOS-1	2	NM-14	Peter
UD-1: Most Recent Poll Chart	3	UR-1 UL-1	Michael Daniel
UD-2: Create Poll Button	9	UD-1	Daniel Michael
UD-3: Join Poll Button	9	UD-1	Daniel Michael Peter Brando
UD-4: Most Recent Polls	3	UD-1	Michael Zaira
JP-1: Join Poll Form	3	NM-16 UD-3	Daniel
JP-2 Join Poll Button	2	JP-1	Michael Daniel
JP-3 Error Handling	2	JP-1	Michael Trinity
PRM-1: Result Tabs	5	NM-17	Michael Daniel
PRM-2: Filter Polls Field	4	PRM-1	Zaira
PRM-3: Poll List	4	PRM–1	Zaira Michael
UPM-1: Profile Page	7	NM-18 NM-20	Michael
UPM-2: Profile Email Field	2	UPM-1	Michael
UPM-3: Display Name Field	2	UPM-1	Michael
UPM-4: Theme Option	3	UPM-1	Michael Trinity
UPM-5: Save Changes Button	2	UPM-1	Michael

UQS-1: Post Quiz Statistic Display	7	UQCH-5	Zaira Daniel Brando Michael
UQS-2: Save Button	4	UQS-1	Michael
UQS-3: Export Button	4	UQS-1	Michael Daniel
UQS-4: Viewing Past Quizzes	4	UQS-1	Michael Daniel
UQS-5: Saved Quizzes Drop Menu Items	7	UQS-1	Michael Brando
UQC-1: Question Settings	10	UD-2	Daniel Michael Brando
UQC-1.1: Question Input	9	UQC-1	Daniel Peter Michael
UQC-1.2: Upload Image Button	3	UQC-1	Michael Brando
UCNH-1: Waiting Room	14	JP-2	Daniel Michael Brando
UCNH-2: Viewing and Answering Questions	14	UCNH-1	Daniel Michael Peter Brando
UQCH-1: Accessing Previously Created Quiz	14	UQC-1	Michael Brando Trinity Peter
UQCH-2: Initiating Quiz and Waiting Room	14	UQCH-1	Michael Daniel Trinity Brando
UQCH-3: In-Quiz View	14	UQCH-2	Daniel Michael Brando Peter

UQCH-4: In-Quiz View (Anonymous)	14	UQCH-2	Daniel Michael Zaira
UQCH-5: Quiz Termination	14	UQCH-4	Michael Daniel Peter