Project Design Group 3

UMGC CMSC 495 Section 6381

Team Members:
Cook, Sean
Coutros, Peter
Daley, Tasciana
Geisler, Nicholas
Holman, Trevor
Jefferson, Michael
Malixi, Matthew Arvhie
Mussa, Nezifa
Suarez, Karolyn

Table of Contents

1.0 Project Scope

The design phase of the web-based task management application will focus on creating an intuitive, user-friendly interface that facilitates easy task and schedule management. The scope includes the design of the core user interface, integration of essential functionalities, and the development of visual prototypes. The design will emphasize simplicity, usability, and compatibility across various web browsers and devices.

1.1 Inclusions

- **UI/UX Design: Develop wireframes, mockups, and high-fidelity prototypes for the core features, including scheduling, task management, reminder notifications, recurring events, and task categorization.**
- **Feedback Integration: Conduct user feedback sessions and incorporate insights into the design to improve usability.**
- **Design Documentation: Provide comprehensive documentation outlining the design process, including wireframes, UI components, interaction flows, and usability guidelines.**

1.2 Exclusions

- **Advanced Features: The design phase will not include AI-driven functionalities or multi-language support.**
- **Third-Party Integration: The design will not incorporate features related to integration with third-party applications like social media, email platforms, or cloud storage.**

2.0 Project Requirements

This section outlines the functional and non-functional requirements of the application. It is intended to be used as a guide for designers, developers, and testers who are working on the "Calendar and Task Management Application" project. It provides the essential information needed to design, develop, and test the software.

2.1 Functional Requirements

The functional requirements describe the behavior of the software. These requirements do not contain any design elements allowing for creativity and innovation in their implementations.

2.1.1 Create Event

Name	FR-1 Create Event
Summary	The create event feature must create new calendar events.
Rationale	A user needs to create new calendar events to utilize the calendar portion of this application.
Requirements	When the user invokes a “Create Event” function, the software must allow the user to enter in details surrounding the event. These details must include the following: event name, event date, event time, reminder date, reminder time, event description, event type (work, personal, or school), and event recurrence (daily, weekly, or monthly).
References	User Story 1 – Schedule Creation, User Story 4 – Recurring Events

2.1.2 Delete Event

Name	FR-2 Delete Event
Summary	The delete event feature must delete an existing calendar event selected by the user.
Rationale	A user will need to delete existing calendar events that have been canceled or are no longer a requirement for the user.
Requirements	When the user invokes a “Delete Event” function, the software must allow the user to select an existing event in their calendar. When the existing event is deleted, it must be removed from the user’s calendar.
References	User Story 1 – Schedule Creation

2.1.3 Edit Event

Name	FR-3 Edit Event
Summary	The edit event feature must allow editing of an existing calendar event selected by the user.
Rationale	A user will need to edit existing calendar events when dates, times, descriptions, etc of the event change. Also, the user should be able to correct any errors made during the original event's creation.
Requirements	When the user invokes an “Edit Event” function, the software must allow the user to select an existing event in their calendar to edit. The user must have options to edit all fields that are present in the “FR-1 Create Event” function. When the existing event is edited, the changes should be reflected in the user’s calendar.
References	User Story 1 – Schedule Creation

2.1.4 Create Task

Name	FR-4 Create Task
Summary	The create task feature must create new tasks.
Rationale	A user will need to create new tasks to utilize the task management portion of the application.
Requirements	When the user invokes a “Create Task” function, the software must allow the user to enter in details surrounding the task. These details must include the following: task name, task due date, reminder date, reminder time, task description, task completion status (not started, in progress, or complete), priority level (high, medium, or low), and task type (work, personal, or school).
References	User Story 2 – Task Management

2.1.5 Delete Task

Name	FR-5 Delete Task
Summary	The delete task feature must delete an existing task selected by the user.
Rationale	A user will need to delete existing tasks that have been canceled or are no longer a requirement for the user.
Requirements	When the user invokes a “Delete Task” function, the software must allow the user to select an existing task in their task manager. When the existing task is deleted, it must be removed from the user’s task manager.
References	User Story 2 – Task Management

2.1.6 Edit Task

Name	FR-6 Edit Task
Summary	The edit task feature must allow for the editing of an existing task selected by the user.
Rationale	A user will need to edit existing tasks when due dates, descriptions, and completion status of the event change. Also, the user should be able to correct any errors made during the original task’s creation.
Requirements	When the user invokes an “Edit Task” function, the software must allow the user to select an existing task in their task manager to edit. The user must have options to edit all fields that are present in the “FR-4 Create Task” function. When the existing event is edited, the changes should be reflected in the user’s task manager.
References	User Story 2 – Task Management

2.1.7 Track Tasks

Name	FR-7 Track Tasks
Summary	The track tasks feature must let the user visualize their tasks (and all their details) in priority order.
Rationale	The user wants to manage their tasks and will therefore wish to see them in priority order. In addition, they will see the task's status to determine which tasks should be completed first.
Requirements	When the user invokes an “Track Tasks” function, the software must display all the user’s tasks in priority order. The software must also display all the information/details associated with the task.
References	User Story 2 – Task Management

2.1.8 Reminder Notifications

Name	FR-8 Reminder Notifications
Summary	The software must send out a notification to the user when the reminder date/time of an existing event or task has been reached.
Rationale	The user wants to manage their events/tasks and will want to be notified on the reminder date/time they set so that they do not miss any important meetings/tasks.
Requirements	When the reminder date/time of the event/task is reached, the software must automatically send the user a notification reminding them of the event/task and include its details.
References	User Story 3 – Reminder Notifications

2.1.9 Filter by Category

Name	FR-9 Filter by Category
Summary	The software must allow the user to view all events/tasks based on the category of the events/tasks.
Rationale	The user will have different categories (work, personal, or school) for events/tasks and will wish to filter their events/tasks based on category.
Requirements	When the user invokes an “Filter by Category” function, the software must allow the user to select an event/task type. Following a selection, the software must display all events/tasks of the selected type to the user.
References	User Story 5 – Task Categorization

2.2 Non-Functional Requirements

Unlike functional requirements describing the application's behavior, the non-functional requirements describe the quality attributes, performance standards, and design constraints. These requirements focus on how the system operates rather than specific behaviors of the application.

2.2.1 Performance Constraints

Name	NFR-1 Performance Constraints
Summary	The application must load and respond to user input quickly.
Rationale	The user will need a quick responding application so they can manage their events/tasks efficiently. If the application is too slow to start or respond, the user may not see value in it and will go back to “pen and paper” management.
Requirements	When the user starts up the application or provides any input to the application, the software must load/respond within 3 seconds.
References	User Stories 1, 2, and 5

2.2.2 Usability Constraints

Name	NFR-2 Usability Constraints
Summary	The application interface must be intuitive for the user to navigate.
Rationale	The user needs to immediately begin managing events/tasks and therefore the interface should be intuitive, otherwise users may avoid this software.
Requirements	The user interface (UI) must be intuitive and user-friendly. The application must be easy to navigate with all components clearly labeled. The UI must provide Day, Week, and Month views of the user’s events/tasks. There should be little to no learning curve to use this application.
References	User Stories 1, 2, 4 and 5

2.2.3 Compatibility Constraints

Name	NFR-3 Compatibility Constraints
Summary	The application must be compatible with common web browsers and operating systems.
Rationale	The application should not be limited to specific browsers/operating systems to not exclude user populations. Users have preferred browsers/operating systems and will likely not change their preferences for one application.
Requirements	The software must function on Google Chrome and Mozilla Firefox browsers and on Windows, Linux, and Mac Operating Systems.
References	User Stories 1 - 5

2.2.4 Accessibility Constraints

Name	NFR-4 Accessibility Constraints
Summary	The application must be accessible 24/7 with any planned maintenance communicated to the users 24 hours in advance.
Rationale	The users will need access to their events and tasks at all times during the day to ensure proper management. If users are blocked from accessing their calendars, they may not find value in the application.
Requirements	The application must remain accessible for users at all times. Any planned maintenance of the system must be communicated to the users 24 hours in advance. Any reminders that are set to go out during planned maintenance should be sent out prior to the system coming down.
References	User Stories 1 - 5

2.2.5 Security Constraints

Name	NFR-5 Security Constraints
Summary	The application must validate a user and provide access to only their events/tasks.
Rationale	The users are only concerned with their events/tasks and will want to make sure no other user can access their events/tasks for privacy reasons and to avoid unwanted CRUD actions.
Requirements	The application must require that a user login with valid credentials and only give access to data that is associated to those credentials.
References	User Stories 1 - 5

3.0 Project Methodology

The chosen project management approach for this project is the Agile methodology. This is an iterative, incremental approach that emphasizes individuals and interactions, working software, customer collaboration, and responding to change (Cunningham, 2001).

3.1 Agile Outline

The Agile methodology, characterized by its iterative and incremental nature, allows developers to adapt to project requirements and stakeholder needs throughout the life cycle of the project. This methodology breaks the project timeline down into sprints and since this project spans 8 weeks, each sprint will be 1 week. Each sprint starts with a planning session where the team defines the goal of the sprint, identifies some outstanding requirements that will be worked on during the sprint, and estimates the effort required to complete the requirements. Normally daily standups are held to discuss the work done the prior day, the work to be done today, and any challenges that need to be addressed. Due to obligations outside of this class, the daily standup model will not be feasible for our team. Instead, we will be meeting weekly to discuss progress. During the week, development and testing of the requirements identified during the planning session will occur concurrently. At the end of each sprint a review is conducted to demonstrate the completed work and gather feedback. The sprint then concludes with a retrospective where success, challenges, and areas for improvement are discussed. This iterative process then repeats, allowing for continuous refinement of the product and its features based off feedback (PathToAgility, 2024).

3.2 Agile Justification

Agile is a good fit for this project due to its emphasis on collaboration and responding to change to produce a high-quality software product. The iterative nature of the methodology allows the team to adjust project priorities based on feedback and requirements. The collaborative nature allows the team to keep the development process on track and better align to the user’s needs and expectations. Since development and testing occur concurrently, the product maintains its quality and any issues can be identified and addressed early on. The retrospective aspect of Agile allows the team to reflect upon the prior week’s work and identify where improvements can be made to enhance productivity and improve product quality. While this project revolves around having a complete application that satisfies the stakeholders’ requirements, it also teaches the teams how to work collaboratively, which the Agile methodology inherently fosters.

4.0 Work Breakdown/Project Tasks

4.1 Planning and Initial Meetings

- Task: Finalize project scope, objectives, and design goals.
- Description: Discuss and agree on the overall vision for the application, including core functionalities and design priorities.
- Deliverables: Finalized project scope and design objectives document.
- Collaboration: Team discussion and agreement on design goals.

4.2 Project Design

Task 1: Establish Design Timeline and Allocate Tasks

- Description: Develop a timeline for the design phase, including key milestones and the allocation of tasks among team members.
- Deliverables: Design timeline and task allocation chart.
- Collaboration: Regular check-ins to ensure alignment and progress.

Task 2: Select Technology Stack and Application Architecture

- Description: Choose the appropriate technologies, database model, and architecture for the application, ensuring compatibility with design goals.
- Deliverables: Document outlining the selected technology stack and architecture.
- Collaboration: Input from developers and designers to ensure feasibility.

Task 3: Develop Wireframes and UI Prototypes

- Description: Create low-fidelity wireframes and high-fidelity prototypes for the main screens and features of the application.
- Deliverables: Wireframes, UI mockups, and prototypes.
- Collaboration: Review sessions with the team to refine designs based on feedback.

Task 4: Identify Potential Risks and Challenges

- Description: Assess potential design-related risks (e.g., usability issues, technical constraints) and develop strategies to mitigate them.
- Deliverables: Risk assessment report and mitigation strategy.
- Collaboration: Team discussion and brainstorming.

Task 5: Specify Requirements and Design Prototypes

- Description: Define detailed design requirements and create prototypes for user testing and feedback.
- Deliverables: Requirements document and design prototypes.
- Collaboration: Iterative design process with ongoing team feedback.

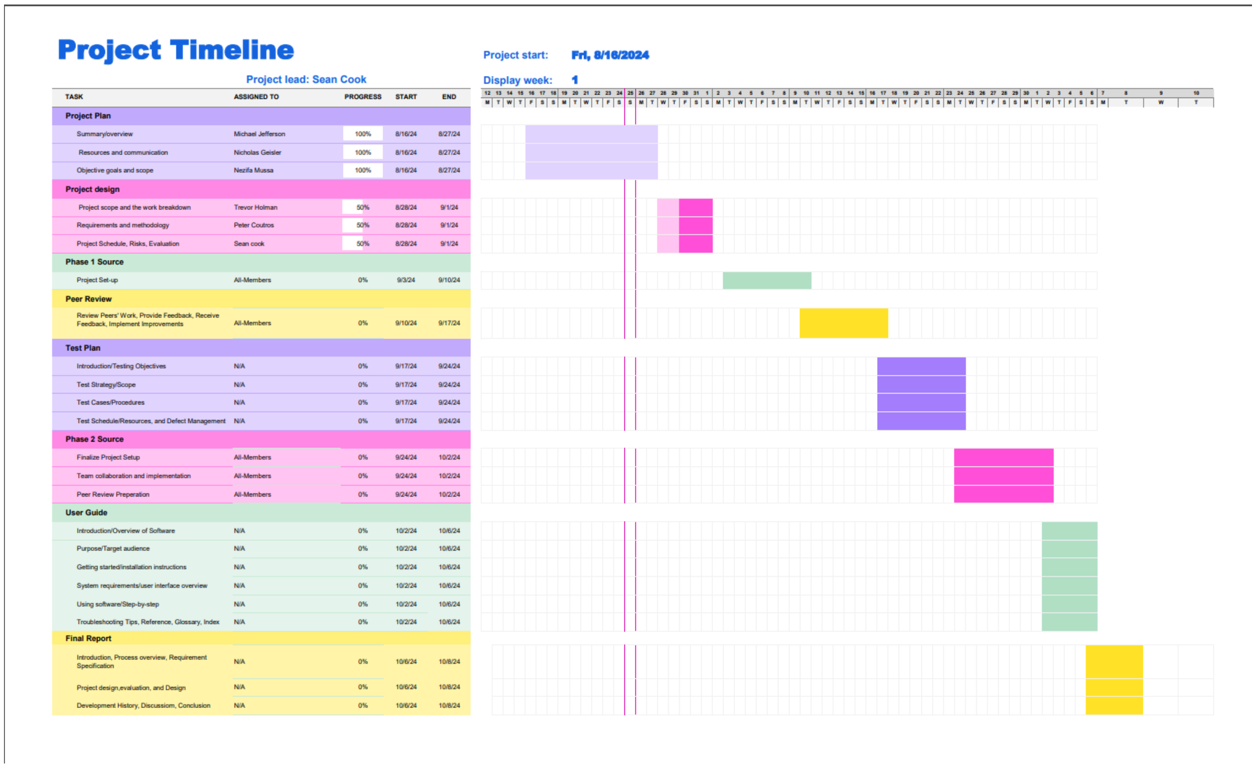
Task 6: Conduct User Feedback Sessions

- Description: Test the design prototypes with potential users to gather feedback on usability and functionality.
- Deliverables: User feedback report and revised design prototypes.
- Collaboration: Collaboration between designers, developers, and users.

Task 7: Finalize Design Documentation

- Description: Compile all design documents, including wireframes, UI components, interaction flows, and usability guidelines, into a final design document.
- Deliverables: Complete design documentation.
- Collaboration: Final review and approval by the team.

5.0 Project Schedule



Project Timeline

Project lead: Sean Cook

Project start: **Fri, 8/16/2024**

Display week: **1**

TASK	ASSIGNED TO	PROGRESS	START	END
Project Plan				
Summary/overview	Michael Jefferson	100%	8/16/24	8/27/24
Resources and communication	Nicholas Geisler	100%	8/16/24	8/27/24
Objective goals and scope	Nezifa Mussa	100%	8/16/24	8/27/24
Project design				
Project scope and the work breakdown	Trevor Holman	50%	8/28/24	9/1/24
Requirements and methodology	Peter Coutros	50%	8/28/24	9/1/24
Project Schedule, Risks, Evaluation	Sean cook	50%	8/28/24	9/1/24
Phase 1 Source				
Project Set-up	All-Members	0%	9/3/24	9/10/24
Peer Review				
Review Peers' Work, Provide Feedback, Receive Feedback, Implement Improvements	All-Members	0%	9/10/24	9/17/24
Test Plan				
Introduction/Testing Objectives	N/A	0%	9/17/24	9/24/24
Test Strategy/Scope	N/A	0%	9/17/24	9/24/24
Test Cases/Procedures	N/A	0%	9/17/24	9/24/24
Test Schedule/Resources, and Defect Management	N/A	0%	9/17/24	9/24/24
Phase 2 Source				
Finalize Project Setup	All-Members	0%	9/24/24	10/2/24
Team collaboration and implementation	All-Members	0%	9/24/24	10/2/24
Peer Review Preperation	All-Members	0%	9/24/24	10/2/24
User Guide				
Introduction/Overview of Software	N/A	0%	10/2/24	10/6/24
Purpose/Target audience	N/A	0%	10/2/24	10/6/24
Getting started/installation instructions	N/A	0%	10/2/24	10/6/24
System requirements/user interface overview	N/A	0%	10/2/24	10/6/24
Using software/Step-by-step	N/A	0%	10/2/24	10/6/24
Troubleshooting Tips, Reference, Glossary, Index	N/A	0%	10/2/24	10/6/24
Final Report				
Introduction, Process overview, Requirement Specification	N/A	0%	10/6/24	10/8/24
Project design,evaluation, and Design	N/A	0%	10/6/24	10/8/24
Development History, Discussiom, Conclusion	N/A	0%	10/6/24	10/8/24

6.0 Project Risks

The team faces several risks that could impact the project's success. First, there is a risk related to team availability, as each member lives in a different time zone. This could lead to slow or difficult communication, potentially causing project delays. To mitigate this risk, weekly meetings will be established where questions can be answered, and duties for the week will be discussed. Additionally, all team members will be encouraged to check the group chat daily to stay updated on any new messages.

Second, there is a risk associated with teamwork. The team members have limited experience working on collaborative projects, which may lead to coordination challenges, particularly when writing the project plan, test plan, and project design. To address this, open communication will be encouraged, and roles and responsibilities for each team member will be clearly defined. Collaborative tools such as version control systems (e.g., Git) and project management tools (e.g., Trello, Asana) will be utilized to facilitate teamwork and ensure everyone is aligned with the project goals.

Finally, the team faces a risk of unfamiliarity with the programming languages and frameworks required for the project, such as Java and React. Not all team members are familiar with these technologies, which may cause delays as they learn. To mitigate this, the programming skills of each team member will be assessed, and tasks will be allocated according to their strengths. Study sessions or shared learning resources will be organized to quickly bring everyone up to speed. Pair programming or code reviews will also be considered to help less experienced members learn from those with more experience.

7.0 Project Evaluation Plan

The below sections cover the criteria for evaluating the project's progress and outcomes.

7.1 Functionality

The primary objective is to ensure that the app meets all the functional requirements as outlined in the project design document. The criteria for success include the full implementation and operation of all core features, such as task creation, calendar integration, and task reminders. Additionally, the app must handle user inputs correctly, providing the expected outputs. It should also support the specified number of concurrent users without experiencing any performance degradation.

7.2 Usability

The objective here is to assess the app's user experience (UX) to ensure it is intuitive and user-friendly. The criteria for achieving this include having an interface that is easy to navigate, with a clear layout and intuitive design. User feedback should indicate satisfaction with the ease of use. Moreover, accessibility features must be implemented to ensure the app is usable by individuals with disabilities.

7.3 Performance

The goal is to measure the app's performance under various conditions to ensure it meets the required standards. The criteria include the app loading within an acceptable time frame, ideally under 3 seconds. The app should perform efficiently under both normal and peak load conditions, with resource usage, such as CPU and memory, remaining within acceptable limits during operation.

References

Cunningham, Ward. *Manifesto for Agile Software Development*, 2001, agilemanifesto.org/.

PathToAgility. “A Comprehensive Guide to Agile Methodology.” *Path To Agility*, 10 July 2024, pathtoagility.com/blog/a-comprehensive-guide-to-agile-methodology/.