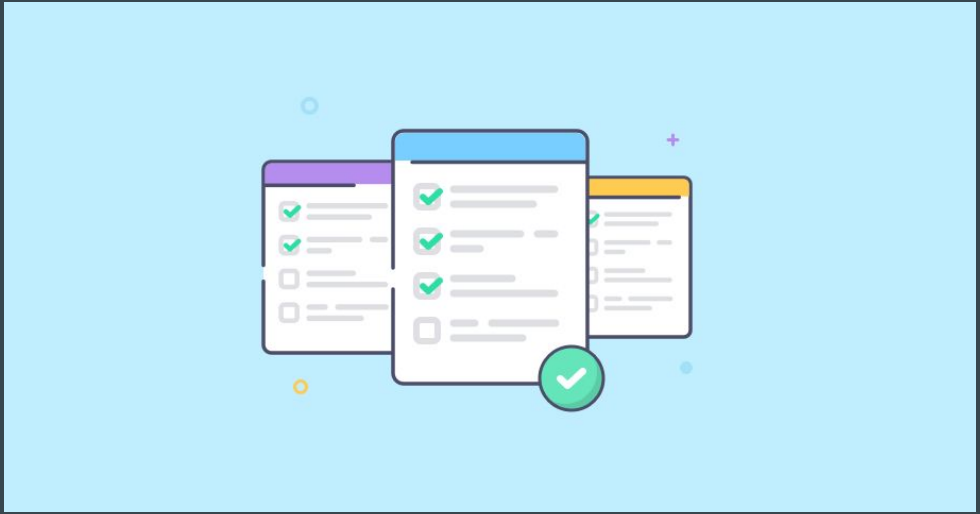
TASKMASTER

Group Name: Tech Titans

By: Alexander Georgiev, Daniel Lanigan, Akshath Majumder, Prahaara Malli Sh



Problem Statement:

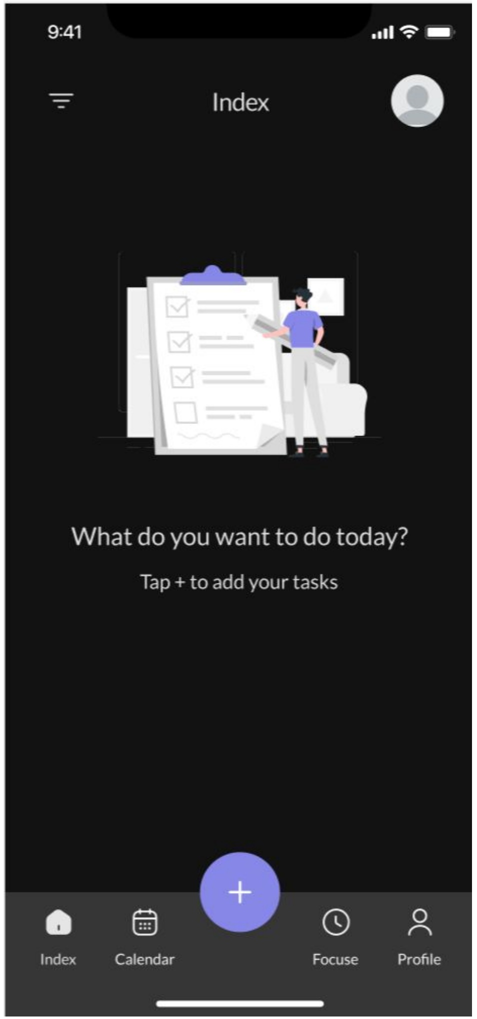
The problem: Task Management

- Volume: An abundance of tasks to manage.
- Organization: Difficulty in keeping tasks organized.
- Overwhelm: Feeling overwhelmed by the workload.

Proposed Solution:

Our solution: TaskMaster - All-in-One Task Management Solution

- Streamlined Prioritization:
  - Easily identify and focus on high-impact tasks.
- Seamless Organization:
  - Keep your tasks and projects neatly organized.
- Stress-Free Workload:
  - Manage your tasks without feeling overwhelmed.



Related Work

Related Tools:

Asana: Known for its project tracking and team collaboration features

Trello: Popular for its Kanban-style boards and ease of use.

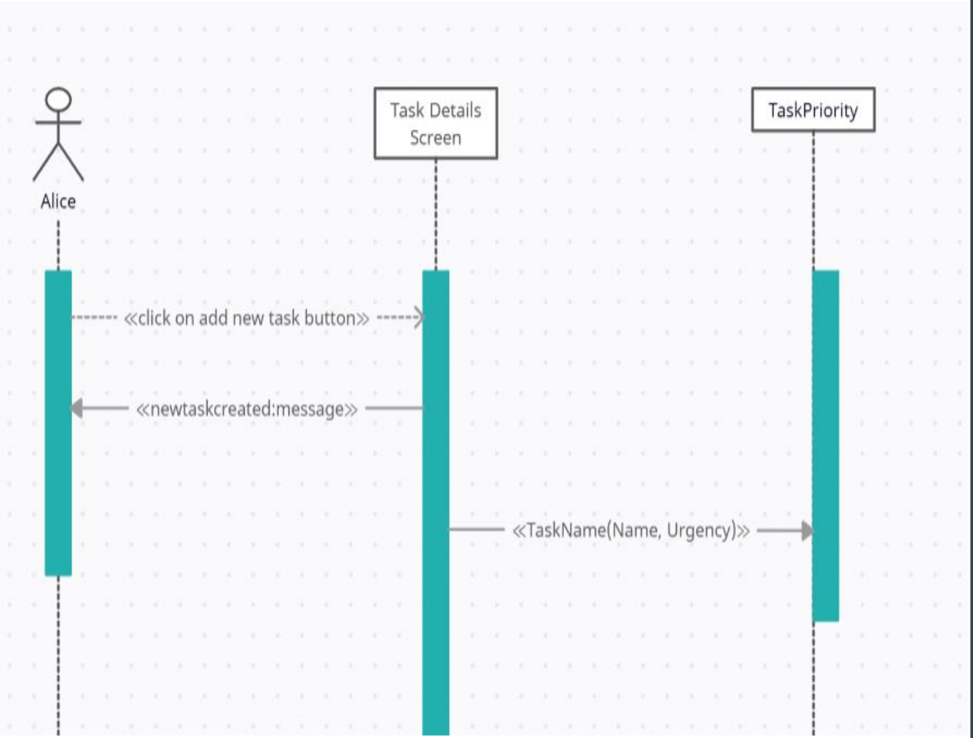
Monday.com: Offers a wide range of customization and automation features.

Related Articles: The article "The Future of Team Collaboration: Trends and Best Practices for 2023" from Almanac.io is related work for TaskMaster in several key ways:

It outlines emerging trends in team collaboration and remote work, areas that are directly relevant to TaskMaster's goal of enhancing task management and team productivity.

The focus on the integration of AI in work processes and the importance of fostering a positive team culture aligns with TaskMaster's objectives to streamline task management using technology.

The article's insights into the evolution of collaboration tools offer valuable context for TaskMaster's development, particularly in enhancing team communication and collaboration features.



Use Case 1

Use Case 1: Streamlining Task Prioritization

Scenario: Alice, a project manager, struggles with managing a large volume of tasks for multiple projects.

Use Case: Alice uses TaskMaster to streamline task prioritization, focusing on high-impact tasks first.

Stakeholders: Project managers, team members.

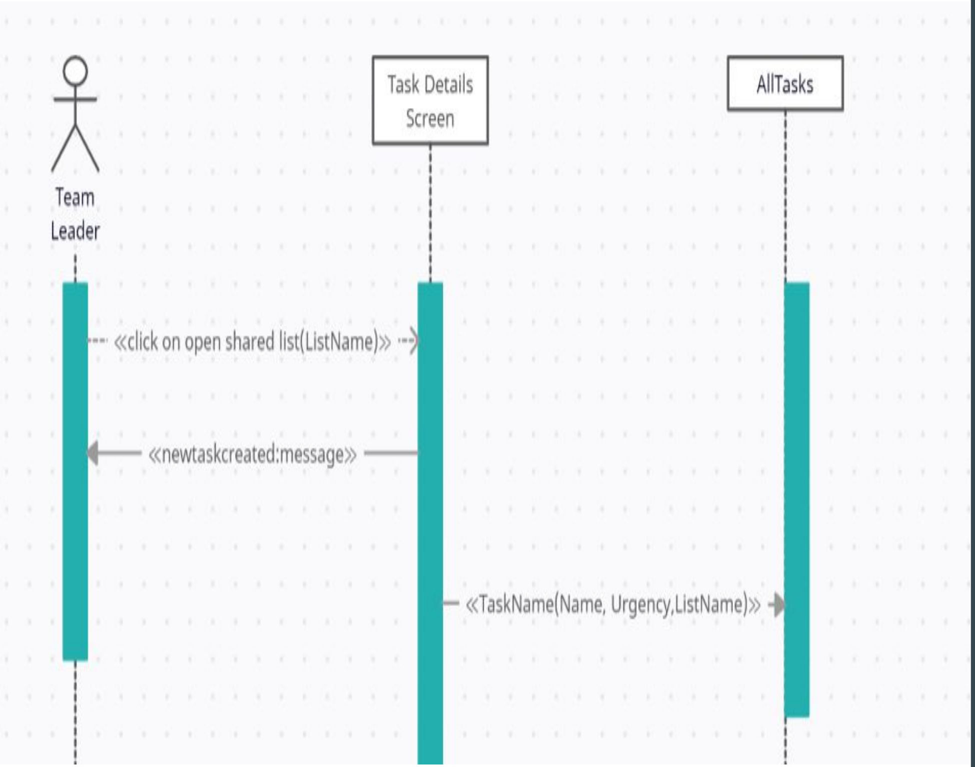
Preconditions: TaskMaster is installed; Alice has an account.

Postconditions: Tasks are organized and prioritized efficiently.

Main Flow:
Alice logs into TaskMaster.
She reviews her task list, sorting by impact and urgency.
High-impact tasks are tackled first, enhancing productivity.

Alternative Flow:
If a task becomes less critical, Alice re-prioritizes it.

Requirements: TaskMaster installed with prioritization features enabled.


Team Leader
Task Details Screen
AllTasks
«click on open shared list(ListName)»
«newtaskcreated:message»
«TaskName(Name, Urgency,ListName)»

Use Case 2

Use Case 2: Collaborative To-Do List

Scenario: A team of developers needs to collaborate on tasks in real-time.

Use Case: The team uses TaskMaster's shared to-do list feature for real-time collaboration and updates.

Stakeholders: Developers, project managers.

Preconditions: All team members have TaskMaster installed.

Postconditions: Tasks are collaboratively managed and updated.

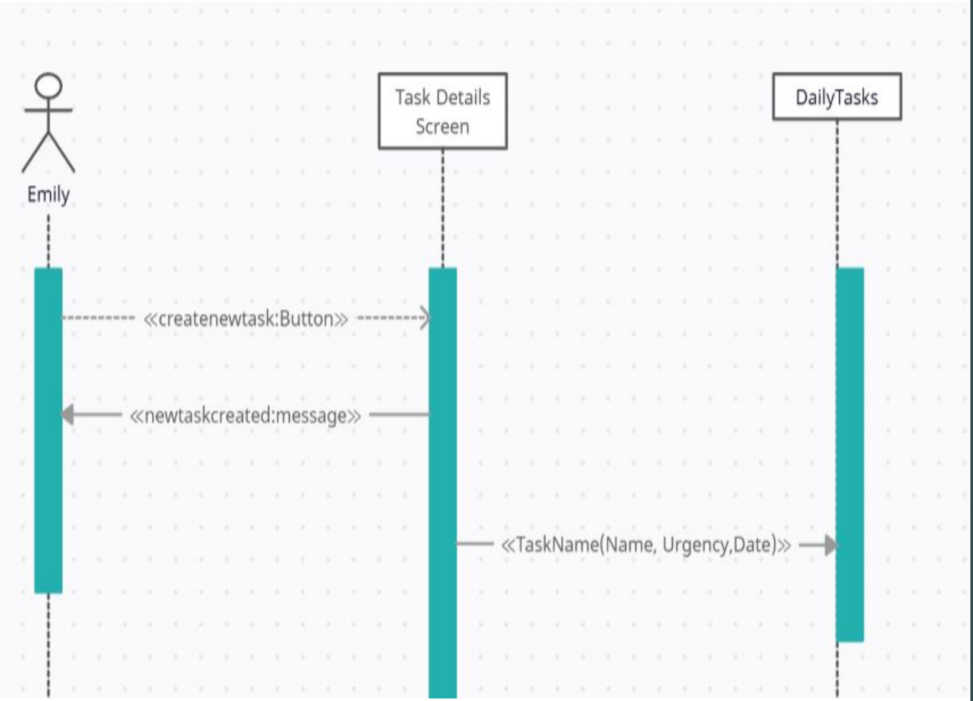
Main Flow: Team leader sets up a shared to-do list in TaskMaster.

Team members add and update tasks in real-time.

Notifications keep everyone informed of changes.

Alternative Flow: If a team member cannot complete a task, it is reassigned.

Requirements: Real-time update and notification features.



Use Case 3

Use Case 3: User-Friendly UI for Task Management

Scenario: Emily, a new user, needs an intuitive task management tool.

Use Case: Emily uses TaskMaster's user-friendly and responsive UI for her daily task management.

Stakeholders: New and existing users of TaskMaster.

Preconditions: TaskMaster installed with an updated UI.

Postconditions: Emily efficiently manages her tasks using the intuitive interface.

Main Flow: Emily logs into TaskMaster.

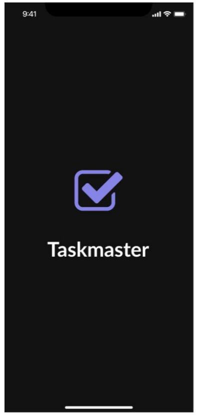
She easily navigates through the app, adding and managing tasks.

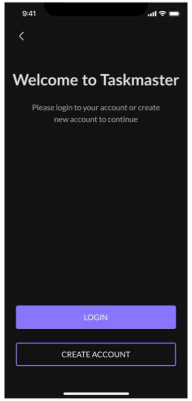
The responsive design adapts to her device, ensuring accessibility.

Alternative Flow: Emily customizes the UI settings according to her preferences.

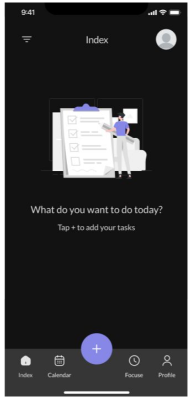
Requirements: TaskMaster with an updated, responsive UI.

Mock User Interface



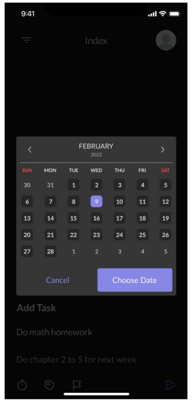


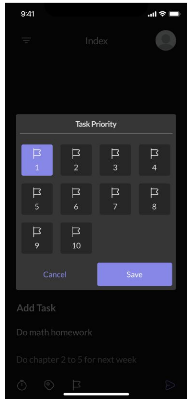
Please login to your account or create new account to continue

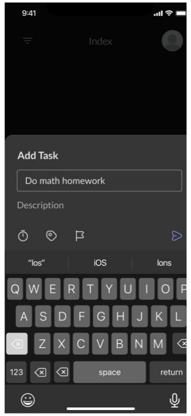


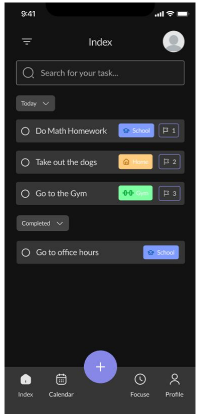
What do you want to do today?

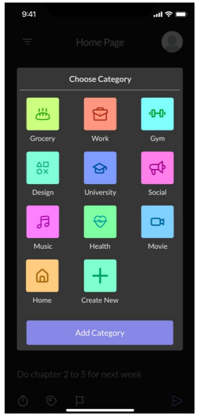
Tap + to add your tasks

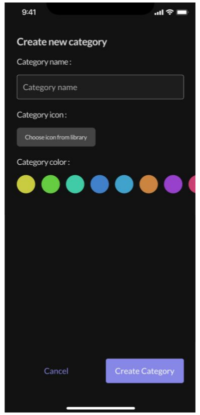


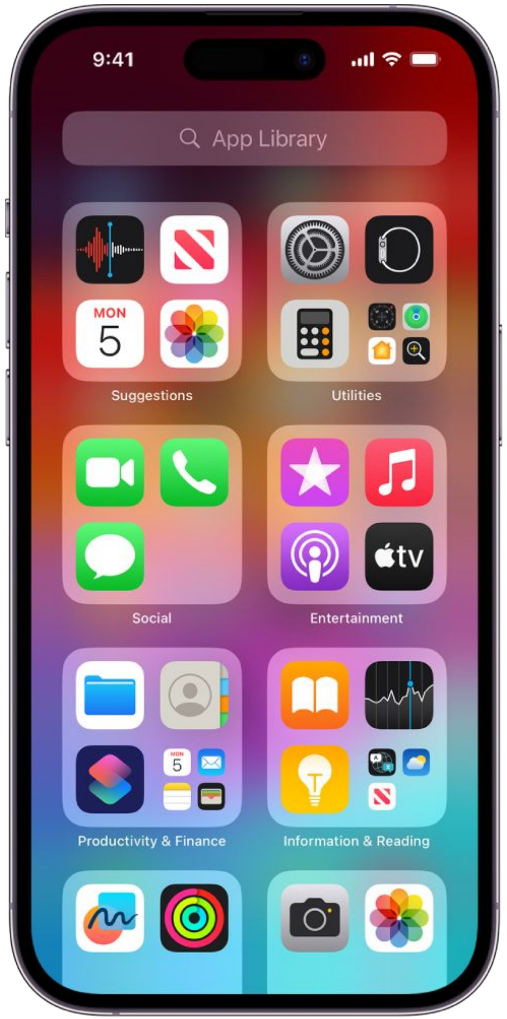












Limitations

Scalability Challenges: As the number of users grows, the app may face performance issues, especially with real-time features like shared to-do lists.

Integration with Other Tools: Limited or no integration with other productivity tools and platforms could restrict the app's utility for users who rely on multiple tools.

User Interface Complexity: While the app aims for a user-friendly design, the complexity of features might overwhelm new users, especially those not tech-savvy.

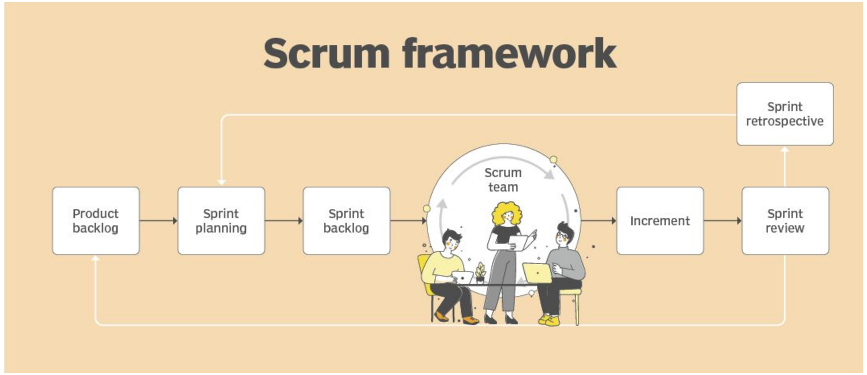
Data Security and Privacy: Handling sensitive user data, especially with features like file attachments and collaborative lists, could pose data security and privacy concerns.

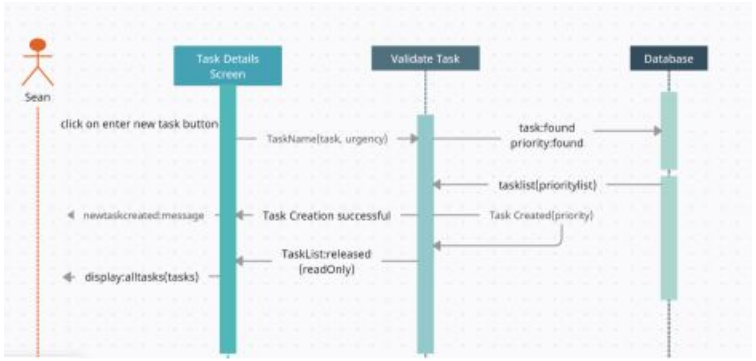
Processes and Tools Used

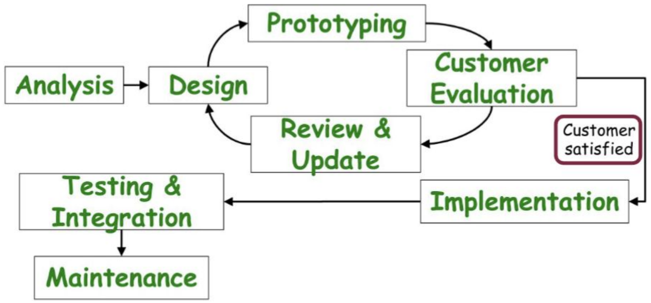
Agile Methodology (Scrum and Kanban): We adopted Scrum for focused sprints with specific goals, aided by daily stand-up meetings. Kanban helped visualize tasks, ensuring team alignment and adaptability to changes, a crucial aspect of Agile.

Figma Prototyping: Utilizing Figma, we designed and refined our app's user interface collaboratively, facilitating efficient experimentation with layouts, colors, and user flows. This saved time and ensured a user-friendly design.

UML Diagrams for Use Cases and Personas: Through UML diagrams, we visualized use cases and created user personas, enabling a deeper understanding of user interactions and needs. This guided feature prioritization and development, aligning the app with user expectations.

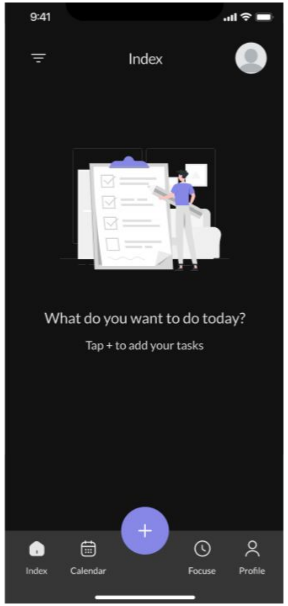






Useful tools:

- Figma
- Miro
- App Builder
- Adobe XD
- Balsamic
- Usability Hub
- ...



How it Relates Back to Class

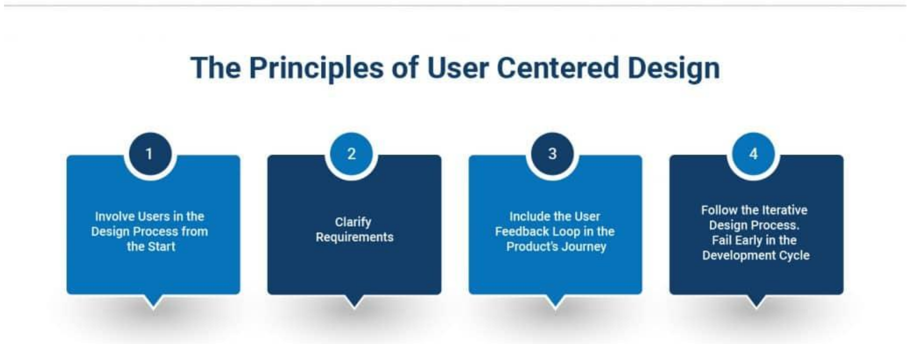
Requirement Analysis: We created user personas to define key features like task categorization and reminders, applying class concepts of understanding user needs.

Prototype Design Pattern: In our to-do app, we used the Prototype design pattern from class to create a basic task feature template, which we then replicated and modified for different task types, enhancing development efficiency and consistency.

Agile Methodology: Adopted Scrum with two-week sprints and frequent stand-ups, using a Kanban board for task management, mirroring Agile practices discussed in class.

Design and Prototyping Tools: We extensively used design and prototyping tools like Figma, aligning with our class's emphasis on visualizing app interfaces and workflows before development, essential for planning and conceptualizing our to-do app.

UI/UX Design Principles: Focused on a clean, responsive design for a user-friendly app experience, applying UI/UX design principles taught in class.


The Principles of User Centered Design
1
Involve Users in the Design Process from the Start
2
Clarify Requirements
3
Include the User Feedback Loop in the Product's Journey
4
Follow the Iterative Design Process. Fail Early in the Development Cycle

Things We Learned

Agile Software Engineering Process: Implementing the Agile process was invaluable, teaching us to work in short, productive cycles, quickly adapt to changes, and foster team collaboration, reflecting the Agile practices emphasized in our class.

Prototyping for Visualization: Using prototyping tools to visualize our app before coding was incredibly useful. It allowed us to experiment with different designs and functionalities, aligning with class lessons on the importance of prototyping in software development.

Feature Prioritization: We learned to identify and prioritize essential features, a key skill from our class, crucial for effective scope and resource management in our software project.

User-Centric Design Approach: Focusing on user-centric design was a major learning point. It helped us refine our app's usability and functionality, directly applying our class's teachings on the significance of a user-focused approach.

Future Work:

Complete Coding the Features:

- Ensure all planned features are implemented according to project requirements.
- Integrate user feedback and make necessary adjustments.

Make a Shared To-Do List:

- Implement collaborative to-do list feature for multiple users in real-time.
- Include real-time notifications to keep users informed about changes.

Adding More Information to Each Task:

- Integrate feature allowing users to attach files, images, or additional documents.
- Implement tagging system, due dates, and reminders for task organization.

User Interface (UI) Enhancements:

- Ensure user-friendly design for an intuitive experience.
- Implement responsive design for accessibility on various devices.