TODO Desktop Application

Team TODO:

Ryan Szilvasi

Tyler Streator

Junxiang Feng

Xiaolong Xuan

Kechen Yu

Problem Statement

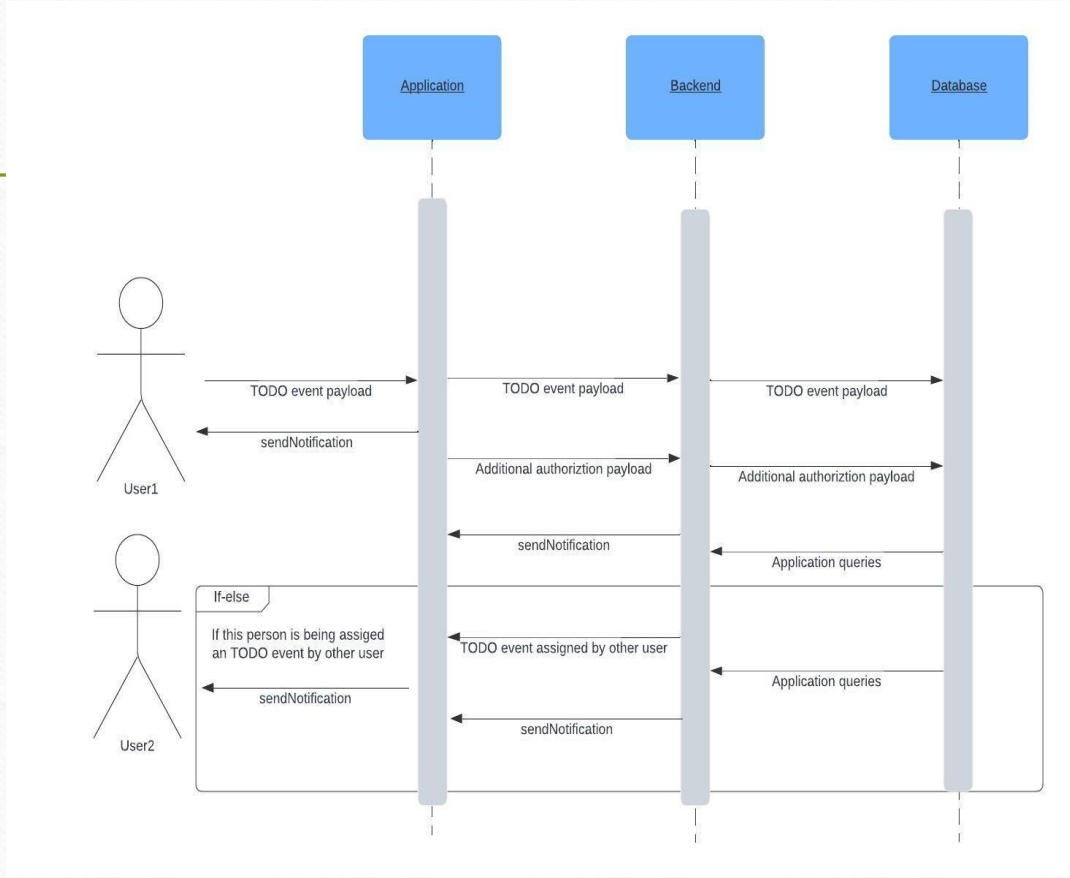
- Many programmers struggle to meet important deadlines for projects. As CS majors, we know it's important not to get behind on your work and this is only emphasized in professional settings.

Proposed Solution

- Our solution is to provide a desktop application to give programmers notifications of deadlines, providing them with customizable views and ordering deadlines based on importance. The notifications will provide programmers with specific tasks they need to complete by the end of the day. Individuals can customize their views to ensure that important information is being highlighted. Lastly, individuals can order their tasks based on importance to ensure they are contributing to their team in a proper manner.
- As explained in, "Maximizing Efficiency A Comprehensive Review of Daily Productivity Growth with Todo Manager," a task manager leads to an increase in productivity and a decrease in distractions

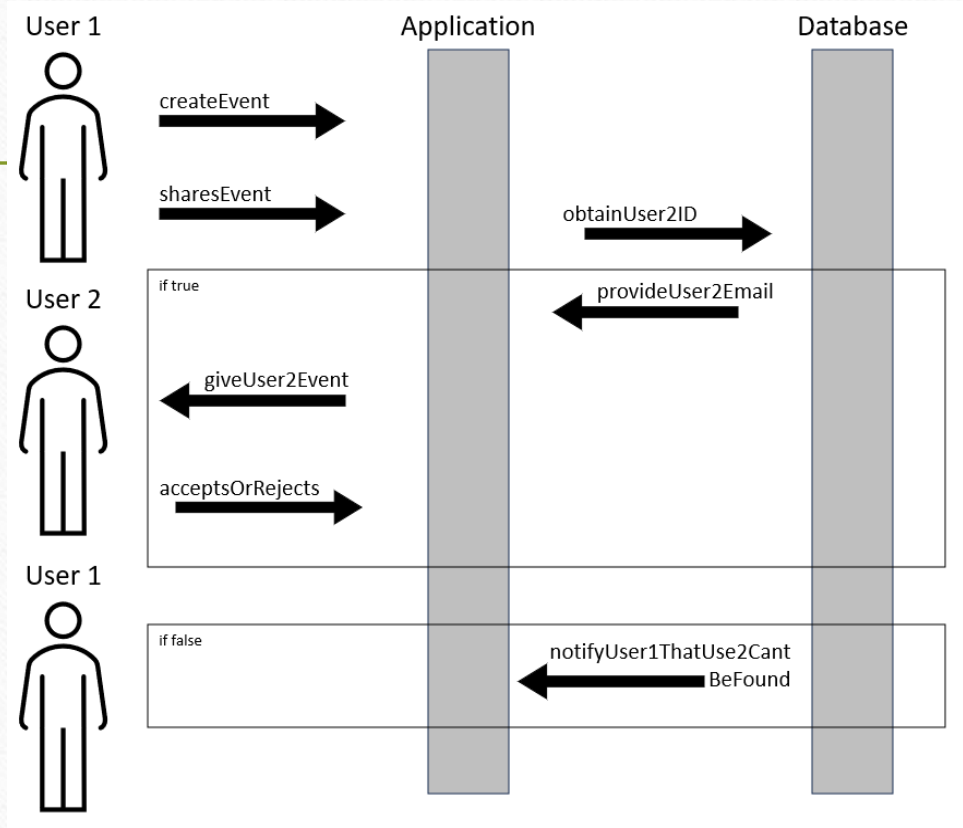
Use case: Creating a TODO event

- Preconditions
  - The user must provide information about the event to the application.
- Main Flow
  - The user can set the title and details of TODO events[S1]
  - The application will store and display them.[S2]
- Subflows
  - [S1] The application provides an option to add an event for the user to fill in the information
  - [S2] The application arranges and displays the event titles
- Alternative Flows
  - [E1] There are no events to show



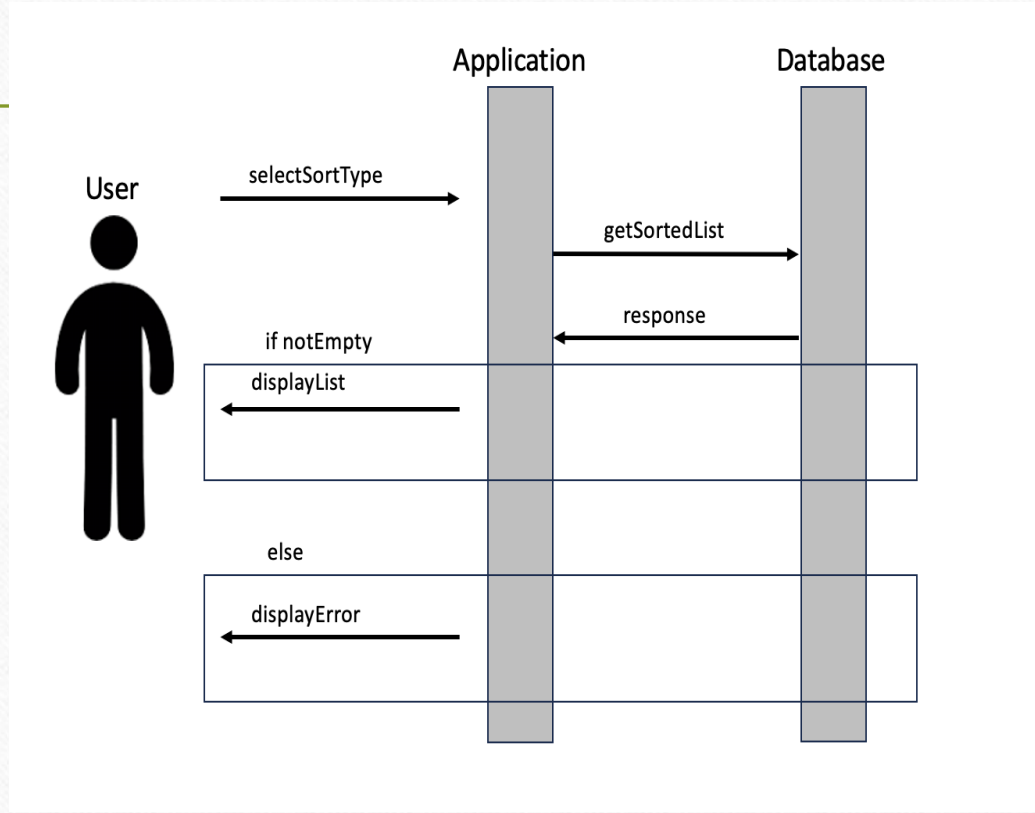
Use case: Sharing Event Between Users

- Preconditions
  - The user must have an event/task to share with another user Both users must be a networked user
- Main Flow
  - User 1 creates an event and sends it to User 2 [S1].
  - User 2 must accept or reject the event [S2].
- Subflows
  - [S1] User 1 provides the event and User ID of User 2 that needs the event
  - [S2] User 2 must review the event to determine if it should be rejected or not
- Alternative Flows
  - [E1] There is no event that needs to be shared



Use case: Ordering Events

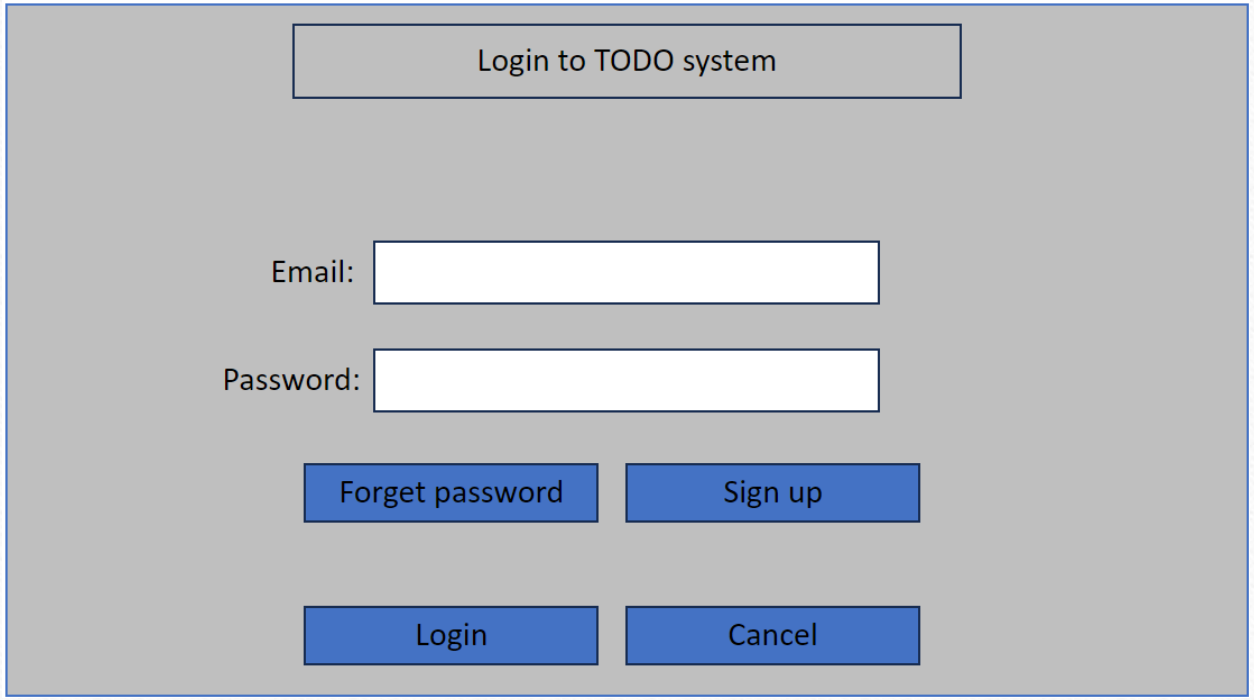
- Preconditions
  - The user must have entered tasks in the TODO app
- Main Flow
  - The user selects an option for prioritizing events [S1]
  - The application provides a list of events sorted using the given ordering [S2]
- Subflows
  - [S1] The application retrieves and sorts the list of events from the database
  - [S2] The application displays the sorted list to the user.
- Alternative Flows
  - [E1] There are no events entered.



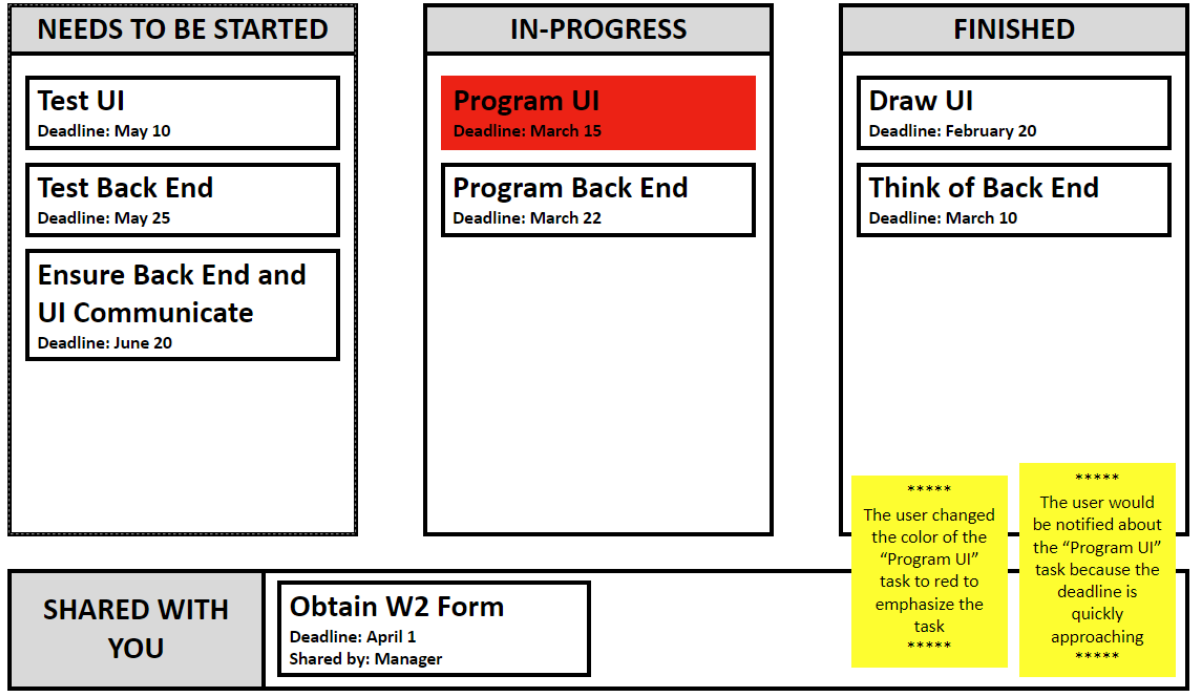
UI windows

- Login window
- Main UI window
- Add TODO event window
- Share TODO event window
- Delete TODO event window

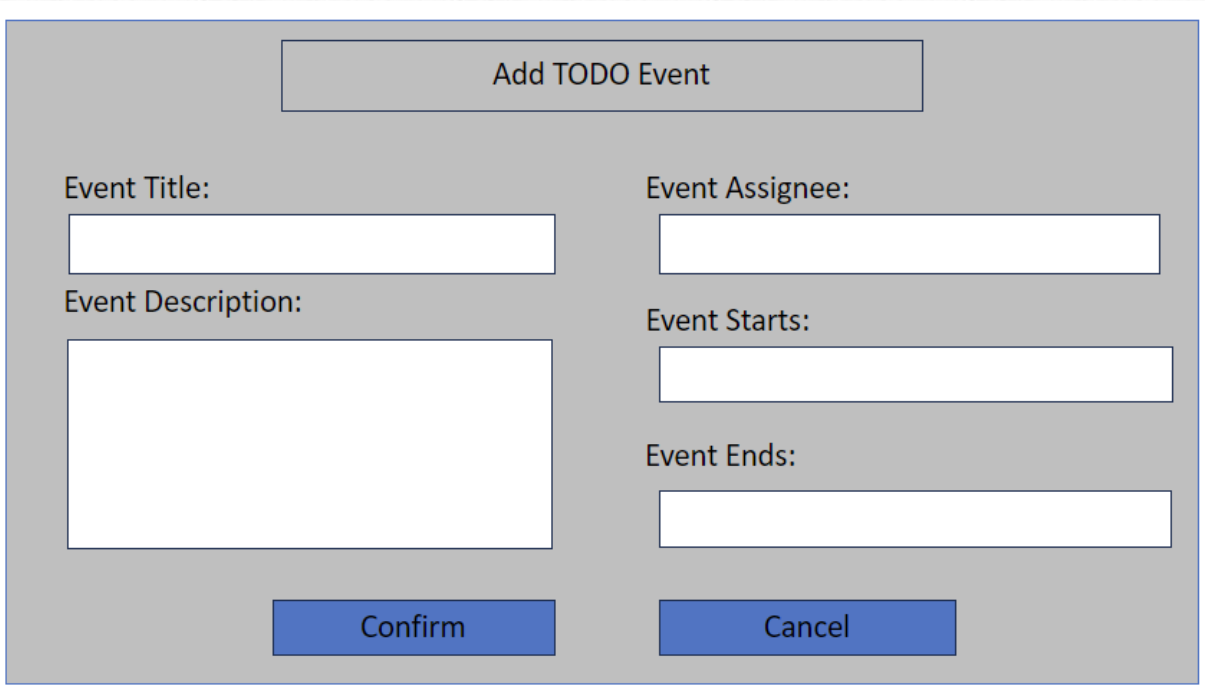
Login window



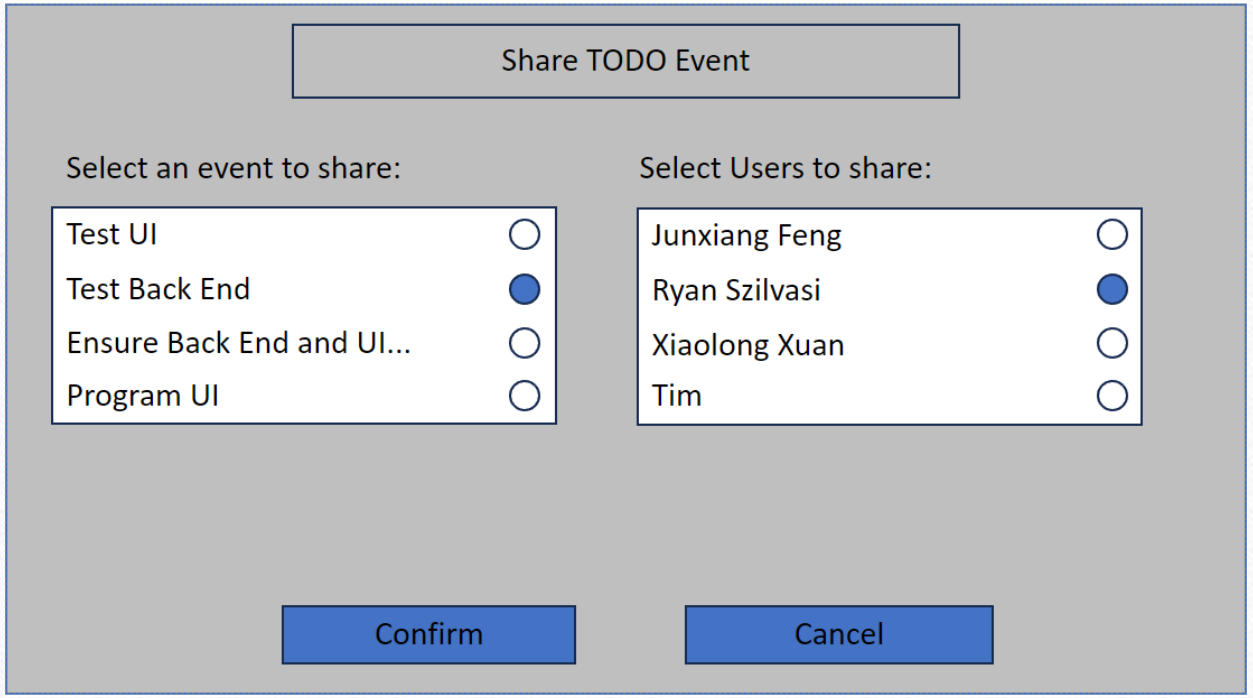
Main window


NEEDS TO BE STARTED
Test UI
Deadline: May 10
Test Back End
Deadline: May 25
Ensure Back End and UI Communicate
Deadline: June 20
IN-PROGRESS
Program UI
Deadline: March 15
Program Back End
Deadline: March 22
FINISHED
Draw UI
Deadline: February 20
Think of Back End
Deadline: March 10
*****
The user changed the color of the "Program UI" task to red to emphasize the task
*****
*****
The user would be notified about the "Program UI" task because the deadline is quickly approaching
*****
SHARED WITH YOU
Obtain W2 Form
Deadline: April 1
Shared by: Manager

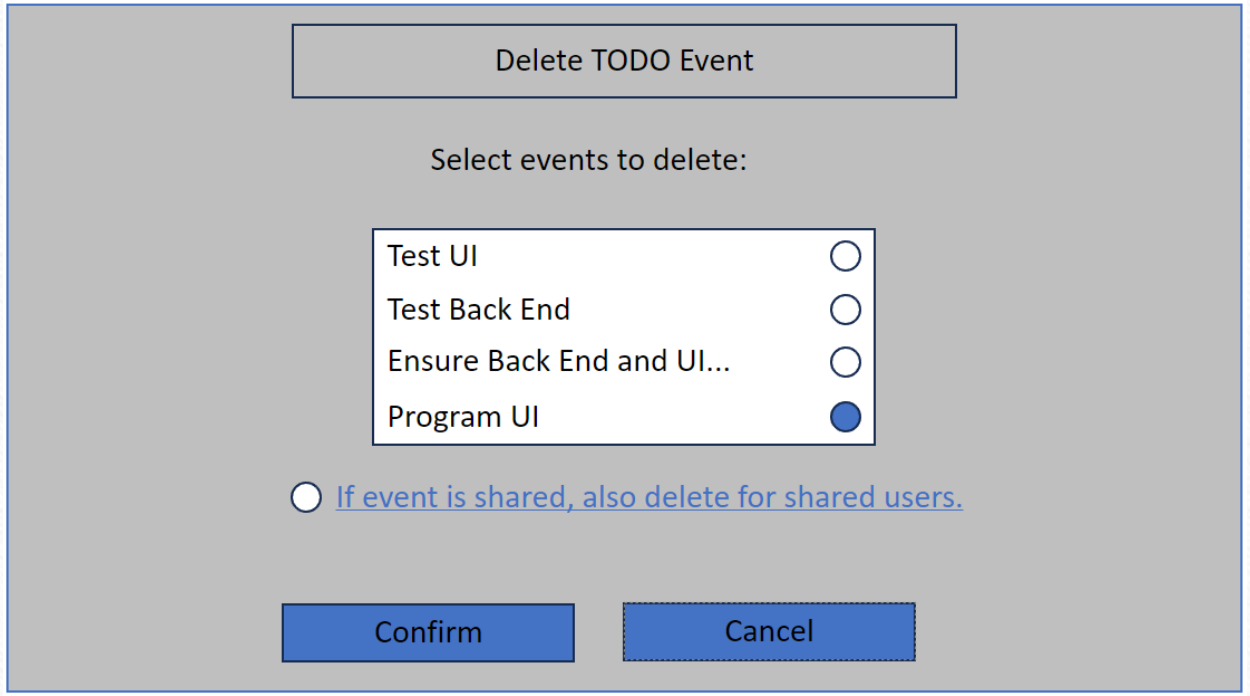
Add TODO event window



Share TODO event window



Delete TODO event window



Limitation

- Platform Dependency
  - Most TODO desktop applications are developed for specific operating systems, limiting their usability across different platforms without additional develop
- Lack of Real-Time Collaboration
  - Desktop applications often lack integrated real-time collaboration features, making it difficult for multiple users to work together synchronously on the same tasks.
- Limited Accessibility
  - Desktop applications require installation on individual computers, restricting access to the application and its data to only those devices where it is installed.

Future work

- Cross-Platform Compatibility
  - Developing applications using technologies that allow them to run seamlessly across various operating systems, such as containerization or platform frameworks.
- Cloud Integration and Synchronization
  - Implementing cloud-based synchronization and storage to facilitate real-time collaboration and access from any device, anywhere.
- Advanced Features Incorporation
  - Integrating artificial intelligence to automate task scheduling, priority setting, and predictive task creation based on user behavior and preferences.

Process and Tools Use

- Our project was developed iteratively through a prototyping process
- Figma
  - Used to design our app's initial screen layout
- Lucidcharts
  - Used to model dependencies
- Microsoft Office
  - We used PowerPoint for project milestones, as well as to produce quick mockups during the prototyping stages

Things We Learned

- Requirement Analysis and Use Cases helped us understand our project better
- Prototyping Process helped us continuously improve our project
- Storyboarding and wireframing helped us visually think of our project

References

Perez, G., & Sheikh, N. (2023, March). *Maximizing Efficiency A Comprehensive Review of Daily Productivity Growth with Todo Manager*. International Journal of Research Publication and Reviews. https://www.researchgate.net/profile/Ramesh-Byali/publication/369440033_Maximizing_Efficiency_A_Comprehensive_Review_of_Daily_Productivity_Growth_with_Todo_Manager/links/641b464592cfd54f84206e99/Maximizing-Efficiency-A-Comprehensive-Review-of-Daily-Productivity-Growth-with-Todo-Manager.pdf