Written By: Eugene Min, Adalia Truong, Claire Holmes, Anthony Donberger, and Harris Naseh

Team Hygiene

Requirements Workshop

1. Provide an example of five hypothetical non-functional requirements for this system. Be sure to include the specific type of requirement discussed in class, with each requirement coming from a unique category.
    a. **Usability**: Easy to download/use. Should not require any other software to work properly.
    b. **Implementation/Constraints**: Have cross platform capabilities. This should be accessible on multiple platforms.
    c. **Performance**: Low load times. We should be as efficient as possible. Be able to load all tasks in under 2 seconds.
    d. **Usability:** Provide a “Getting started” document to walk a user through how to use the software.
    e. **Supportability**: Must be able to receive regular updates without affecting user data.
    f. **Reliability**: Must keep to-do list items backed up so that items are not lost.
2. Provide an example of five hypothetical functional requirements for this system.
    a. **Prioritization:** The user should be able to categorize tasks has low, medium, or high priority
    b. **Completion:** The user should be able to assign a task as “finished” or complete
    c. **Add/Remove:** The user should be able to add/remove tasks from their to-do list
    d. **Tagging:** The user should be able to add tags to each task on their to-do list
    e. **Calendar:** The user should be able to set reminders for specific days using calendar integration
3. Think of a specific task required to complete each of the functional requirements and non-functional requirements mentioned above. Estimate the amount of effort needed to complete this task using function points (i.e., using the values here). Briefly explain your answer.
    a. Non-Functional
    Task: Create a standalone installer for PCs and an app for mobile devices. This is a challenging task but manageable but we need to make sure all dependencies are included with the installer.
        - 80 pts

- Our scale ranges from 0 to 100, so since this is a challenging task, it should have a high number of points. It won't have 100 points because it is somewhat manageable.

Task: Ensure software compatibility on Windows, MacOS, Linux, iOS, and Android platforms. This is a complex challenge that will require significant effort not only to develop but maintain the app for all these platforms.

- 90 pts
- This task is one of the most challenging, so it should have a very high number of points.

Task: Optimize the app to load all tasks under 2 seconds. This task is not the most challenging as long as we make sure to write efficient queries to the database.

- 15 points
- The reason this task is 15 points is because in order to ensure the code is well optimized good coding conventions must be followed. This is a somewhat easy task to follow.

Task: Write a comprehensive "Getting started" guide
A simple task but will require a good amount of effort for the docs to be complete.

- 10 points
- This task doesn't require any coding, so it should be easy, albeit time consuming, to complete.

Task: Implement an update mechanism that doesn't affect user data.
A challenging and time consuming task that will probably be the hardest task of the app.

- 100 points
- This is the most difficult and complex task, so it should have the most points.

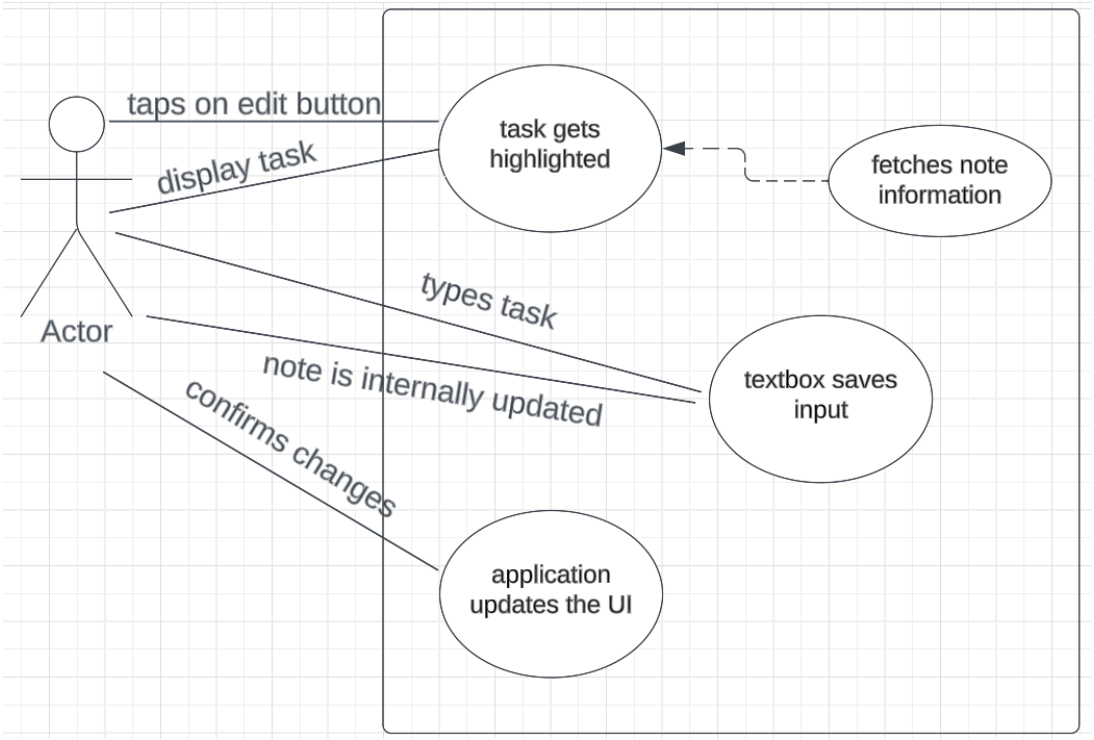
b. Functional:
   i. Prioritization: A user interface must be made in order to allow a user to see the tasks with their different priority levels. In addition, this UI must allow for priority levels to be changed.
   ii. Completion: Give each task object a field that indicates whether it is finished or not
   iii. Add/Remove: Code needs to be made within the application to allow for an add and remove button. Furthermore, this add and remove button must make the task appear or disappear from the UI depending on which button is used.
   iv. Tagging:
   v. Calendar: Implement google calendar integration

4. Write three user stories from the perspective of at least two different actors. Provide the acceptance criteria for these stories.
    a. Joe has been constantly forgetting his deodorant. He needs something to remind him. He downloads our impressive to-do list app. He now has a reminder every morning to put on deodorant.
    b. Gertrude is a mother of three and needs to keep track of what to buy for her children. They all need new clothes and school supplies this fall. The app helps her make separate lists for each child.
    c. Sam is a student in college trying to manage his time properly. He comes home to start working on his homework but is unsure of what to do first. He then uses our to-do list app to order his homework assignments in terms of lowest to highest priority.
5. Provide two examples of risk that could potentially impact this project. Explain how you would mitigate these risks if you were implementing your project as a software system.
    a. Having too many items on the to-do list causing performance issues. To mitigate these risks we can take into consideration the performance requirements of this application.
    b. Users may put personal or sensitive information in their to-do lists, which could lead to data leaks. We will avoid this by not storing any user data.
6. Describe which process your team would use for requirements elicitation from clients or customers, and explain why.
    a. Our team would use the interview process to develop requirements for our system. We would gather software developers plus some non-software developers and hold individual interviews with each stakeholder. Holding interviews would help us gain a greater understanding of how participants structure their personal to-do lists and daily work schedule, as well as learn their expectations of our system.

Requirements Analysis

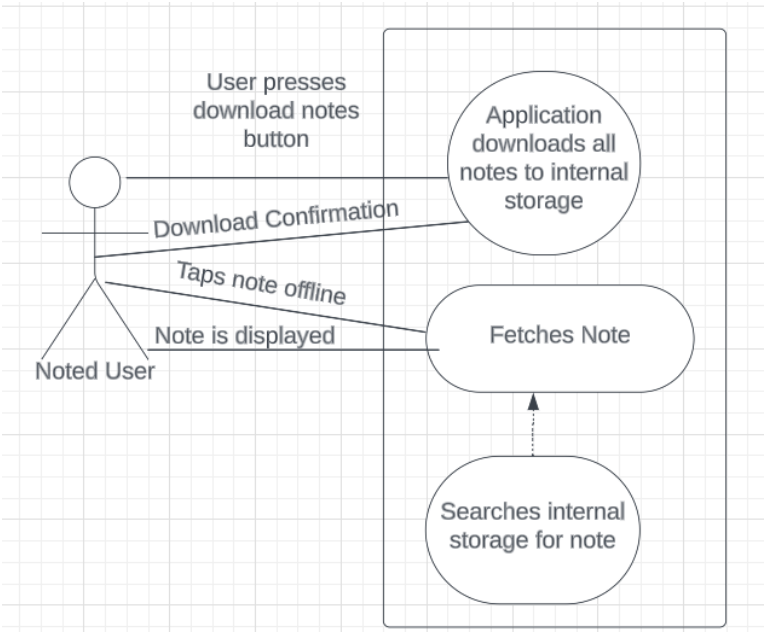
1. **Editing a note on the go**
    a. Precondition: The user's device must be connected to the internet
    b. Main Flow:
        i. The user taps on task edit button
        ii. The user provides text input to change task description
        iii. The user confirms their changes
    c. Subflows:
        i. The application will highlight task to be edited
        ii. The application will take input
        iii. The application will update the UI
    d. Alternative Flows: No tasks were added to the task list

e. Model:

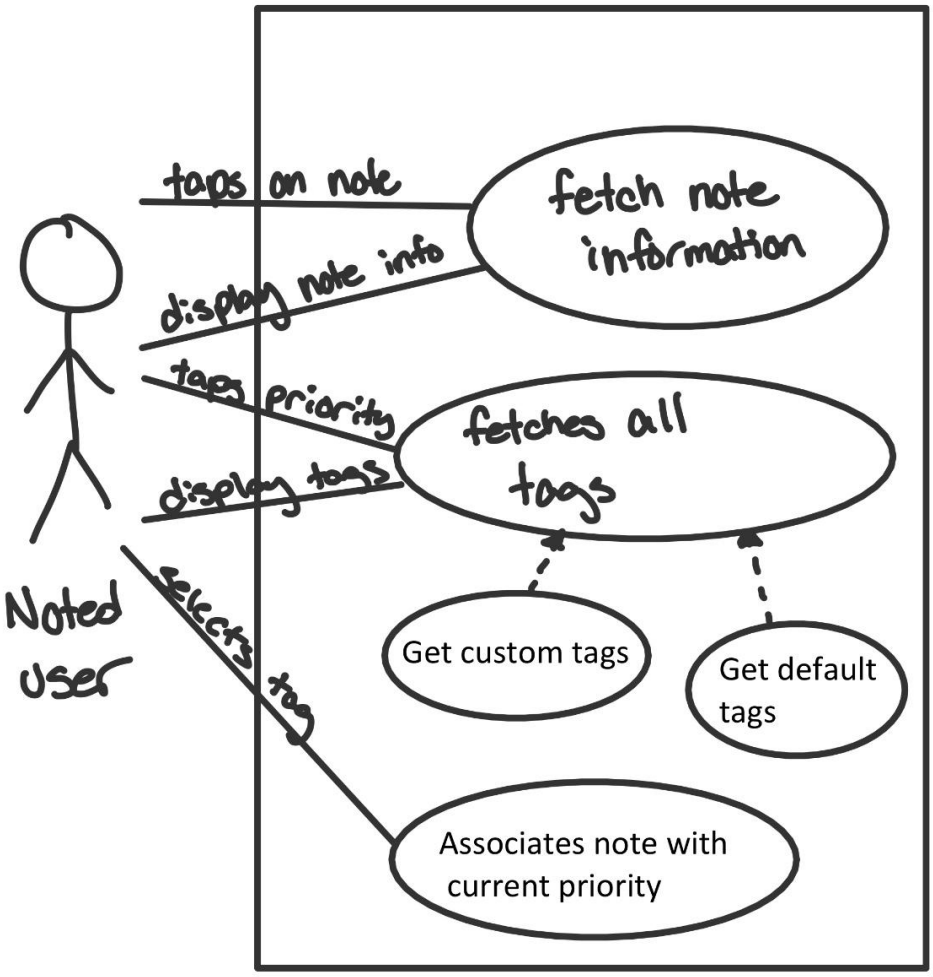


2. Accessing a note offline

a. Precondition: The user must have created a note already
b. Main Flow:
   i. The user must enable automatic note downloading in the app settings
   ii. Once the user is offline, they will open the app and find the downloaded note
   iii. The user can now view and edit the note
c. Subflows:
   i. The application is toggled to store all notes on the device
   ii. The application will look for the note in the device's storage
   iii. The application will display note to user in the UI
d. Alternative Flows:
   i. No tasks were added to the task list.
   ii. The user could not download the note before going offline (unable to access cell service or wifi)
e. Model:



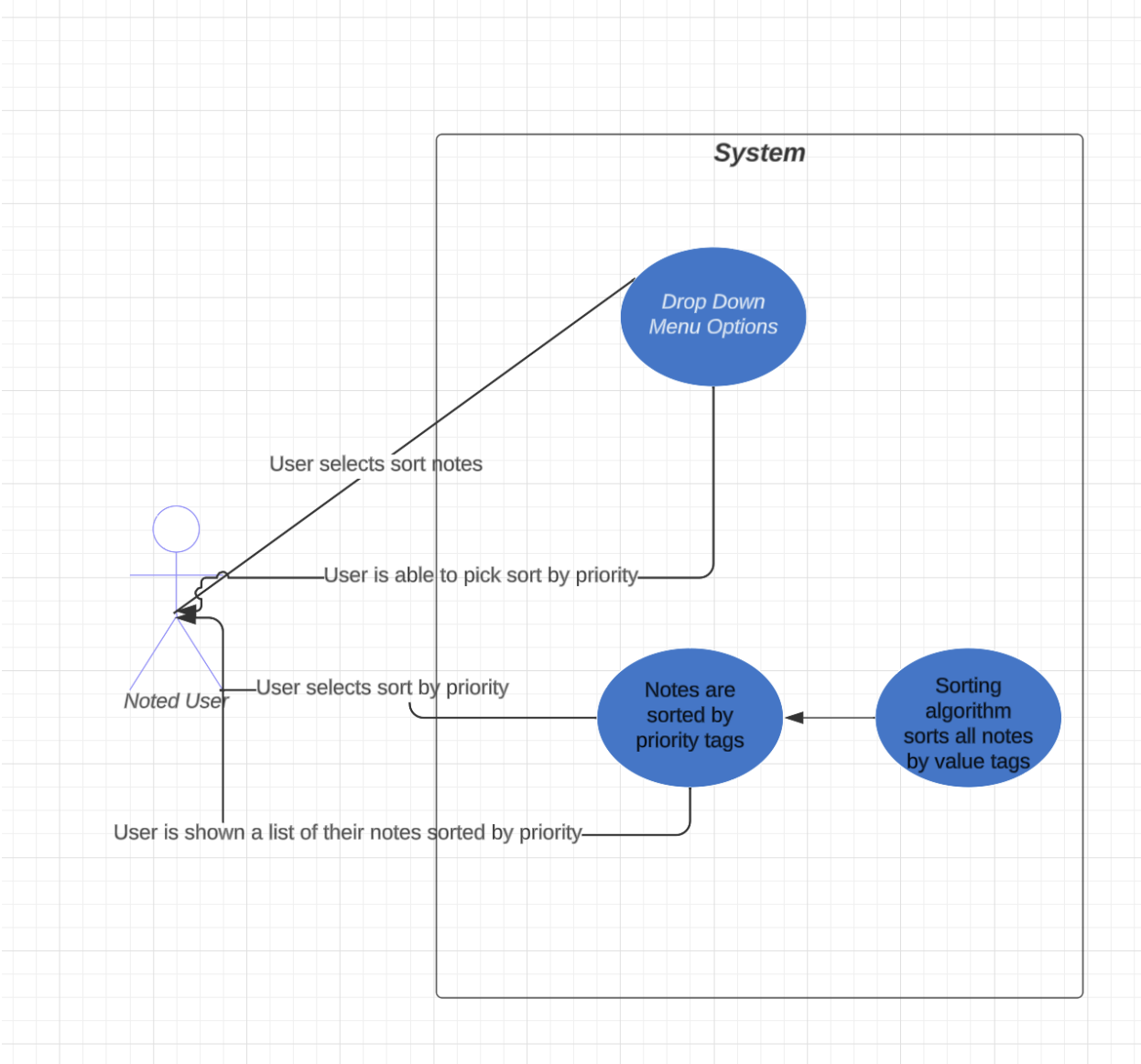
3. **Adding a priority tag to a preexisting note**
    a. Precondition: The user must have created a note already
    b. Main Flow:
        i. The user taps on a note to open up a detailed note view
        ii. The user taps the “Priority” button to open a dropdown containing default priority tags and custom priority tags
        iii. The user selects the desired tag from the list
    c. Subflows:
        i. The application fetches information about the note including priority tag, category, due date, creation date, sub-notes, and attachments
        ii. The application fetches default tags and any custom tags that the user has created
        iii. The application associates the current note with the selected priority. This will affect how the note is displayed in the app
    d. Alternative Flows:
        i. Adding a new tag:
            1. When the user taps on the “Priority” button, they select “Add new tag”
            2. The user types in the new tag name
            3. The user selects a tag color
            4. The user selects “Add” and the tag is added to the list of tags to be used on any note
    e. Postcondition: The note has a priority tag associated with it
    f. Model:



4. **Sorting notes by priority**
    a. Precondition: The user must have at least one note with a tag and the user must be in the list view on the app
    b. Main Flow:
        i. The user taps on the “Sort” button in the top right corner of the app
        ii. A dropdown menu appears with sorting options.
        iii. The user selects "By Priority" from the dropdown menu.
        iv. The application sorts and displays the notes based on their priority tags, from highest to lowest.
    c. Subflows:
        i. The application fetches all notes available in the list view.
        ii. The application identifies notes with associated priority tags.
        iii. The application organizes notes based on these tags, typically placing high-priority items at the top.
    d. Alternative Flows:

i. The user has no notes with tags, in which case a message is displayed: "No notes with priority tags found."
ii. User chooses to revert back to the original sort order or select a different sorting method.

e. Postcondition: Notes are displayed in a sorted manner based on priority.
f. Model:

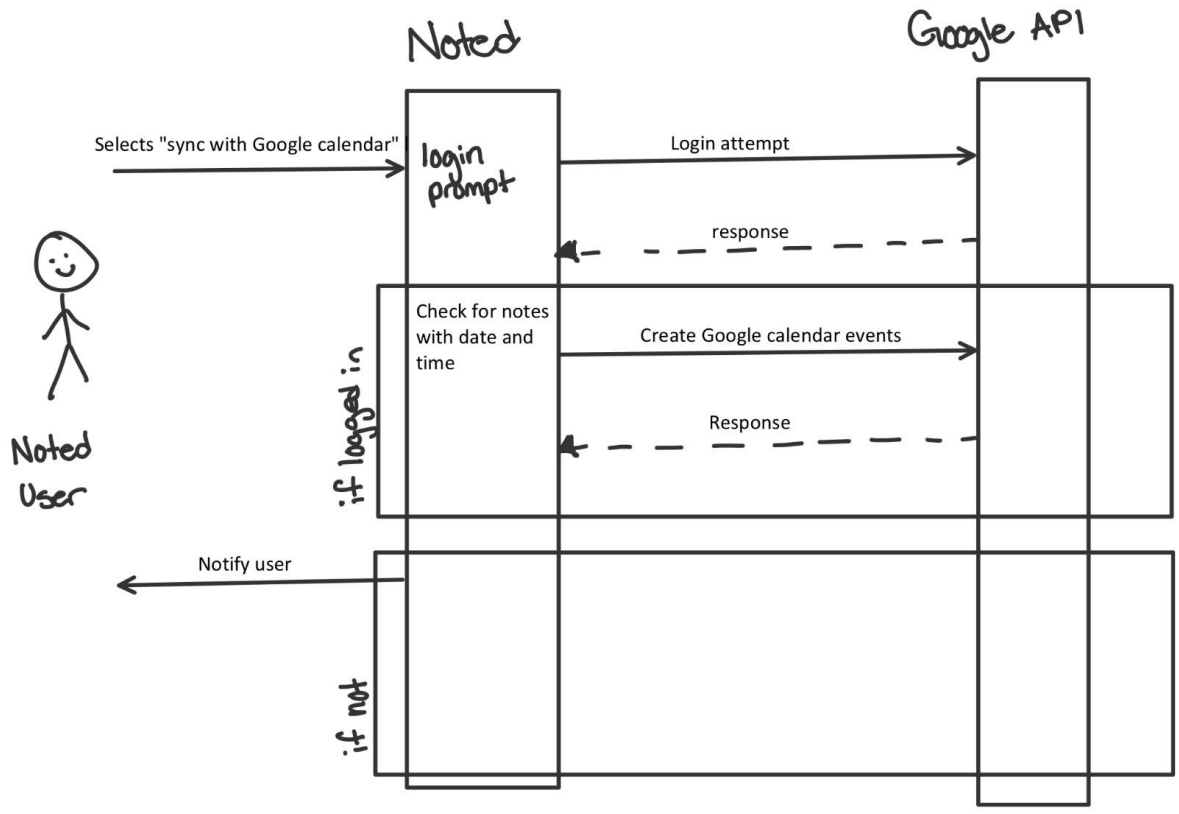


5. Display notes on the user's Google Calendar

a. Precondition: User must have a Google account and has granted the app permission to access their Google Calendar.
b. Main Flow:
i. The user goes to the app settings and selects the "Sync with Google Calendar" option.
ii. The user is prompted to log in to their Google account.

iii. The application starts syncing notes that have specific dates associated with them to the user's Google Calendar.

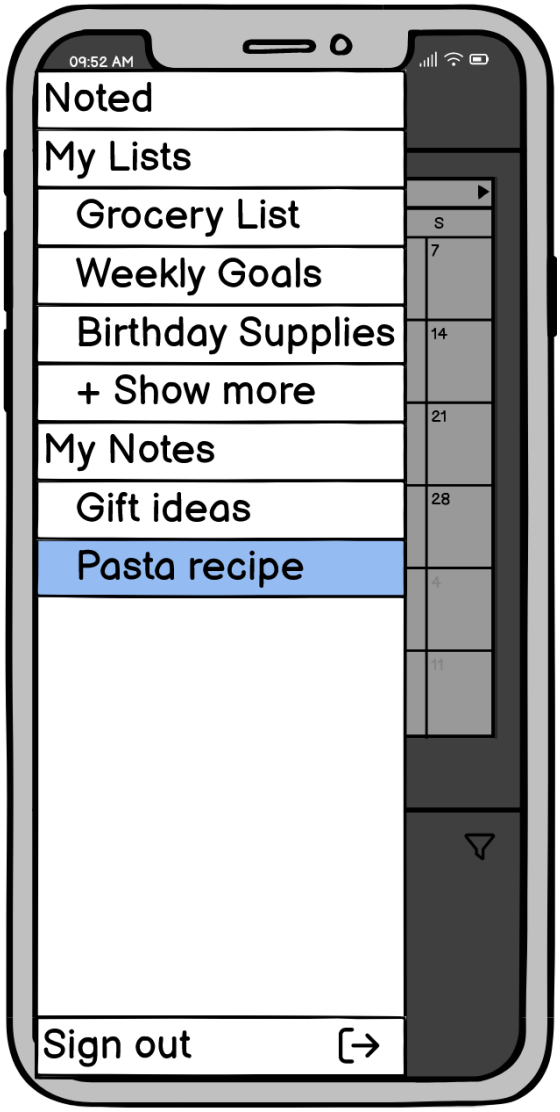
c. Subflows:
   i. The application checks for notes that have dates and times associated with them.
   ii. The application creates events in the user's Google Calendar based on these notes.
   iii. The application places the note title as the event name, and the note description in the event details.
d. Alternative Flows:
   i. The user denies permission to access Google Calendar, in which case the process ends.
   ii. Sync fails due to a connection error. The user is notified and can attempt the sync again.
   iii. User chooses to unsync or stop sharing notes with Google Calendar from the app settings.
e. Postcondition: Notes with associated dates are visible as events in the user's Google Calendar.
f. Model:

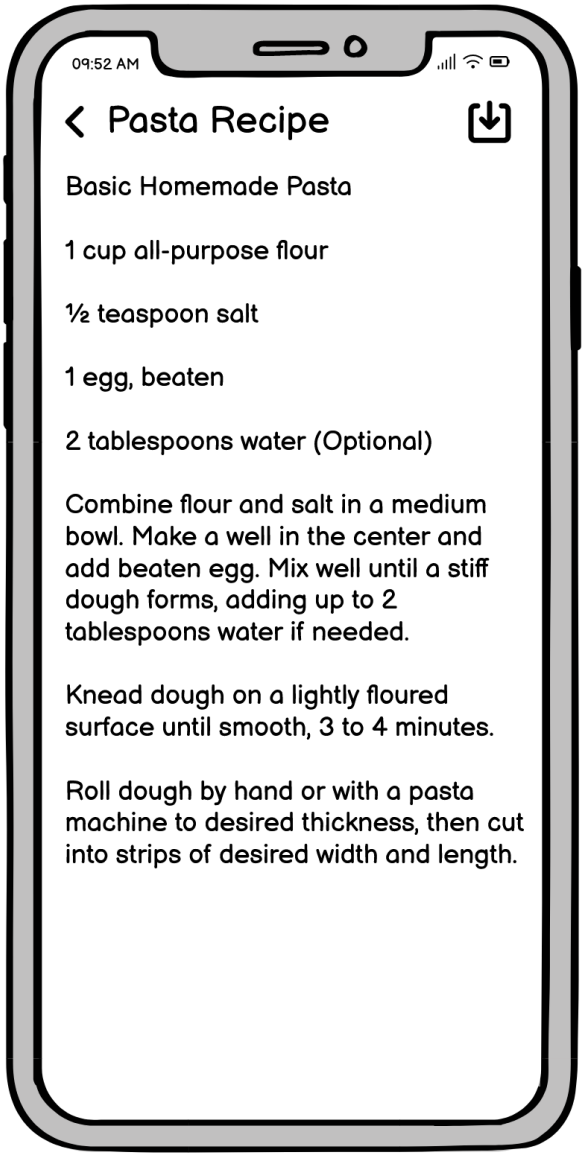


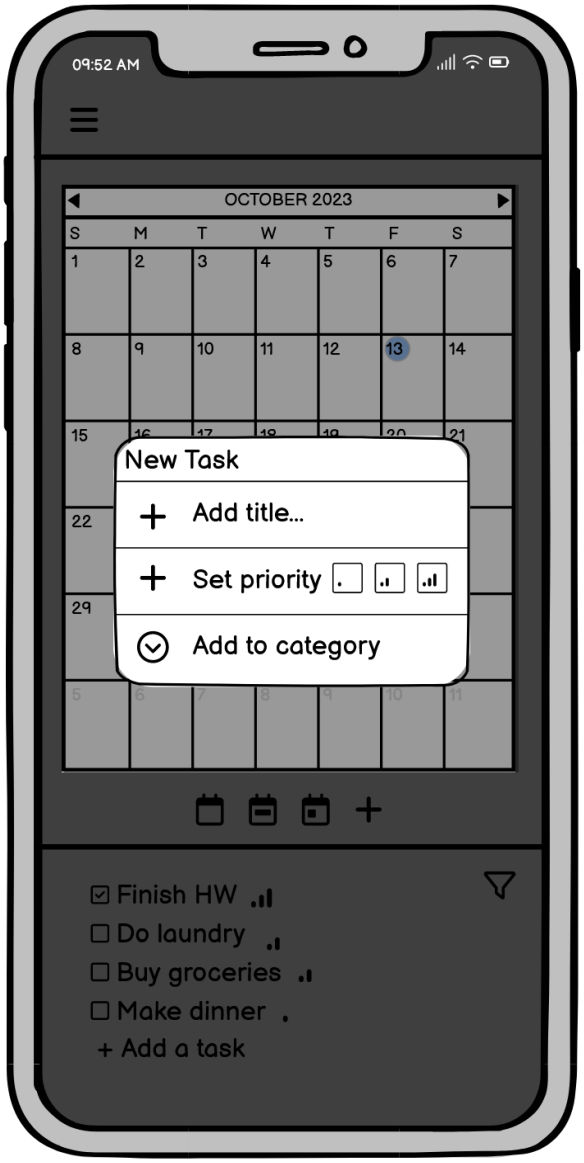
Process Deliverable (Prototyping)

https://balsamiq.cloud/s9dm1u8/pc1xa47/r2278
Wireframe screenshots:


09:52 AM
OCTOBER 2023
S M T W T F S
1 2 3 4 5 6 7
8 9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
29 30 31 1 2 3 4
5 6 7 8 9 10 11
Sort by
Highest priority
Lowest priority
Alphabetical


09:52 AM
Noted
My Lists
Grocery List
Weekly Goals
Birthday Supplies
+ Show more
My Notes
Gift ideas
Pasta recipe
Sign out


09:52 AM
Pasta Recipe
Basic Homemade Pasta
1 cup all-purpose flour
½ teaspoon salt
1 egg, beaten
2 tablespoons water (Optional)
Combine flour and salt in a medium bowl. Make a well in the center and add beaten egg. Mix well until a stiff dough forms, adding up to 2 tablespoons water if needed.
Knead dough on a lightly floured surface until smooth, 3 to 4 minutes.
Roll dough by hand or with a pasta machine to desired thickness, then cut into strips of desired width and length.


09:52 AM
OCTOBER 2023
S M T W T F S
New Task
Add title...
Set priority
Add to category
Finish HW
Do laundry
Buy groceries
Make dinner
+ Add a task