SW Engineering CSC648-848-05 Spring 2024

Project Title: ToDo-Today

Team: 02

Names of Group Members:

Name	Email	Role
Alex Nikols	anikols@mail.sfsu.edu	Team-Lead
Jason Tran		Frontend-Lead
Sulav Jung Hamal		Backend-Lead
Jadon Hoang		Github Master
Randale Reyes		Database Admin
Christian Gopez		Docs-editor

Milestone 2

04/04/2024

History Table

Date	Revision
4/4/2024	Milestone 2 Revision 1
3/25/2024	Final Milestone 1 Revision 2
3/1/2024	Submitted Milestone 1 Revision 1

I. Data Definitions

- **ToDoToday:** The application name.
- **Dashboard:**
  - The main page of the application where the tasks are displayed.
- **Calendar:**
  - The calendar allows for tasks to be displayed on the dates they are assigned.
- **Journal:**
  - The journal is where the user would be able to write down their journal entries and view past entries.
- **Prompt:**
  - A prompt will be provided to the user for the journal entry.
- **Journal Entries:**
  - Journal entries would hold the input the user enters for the journal.
- **ToDo List:**
  - The ToDo List is what will be holding all the tasks that the user creates.
- **TodoItem:**
  - the TodoItem class represents individual tasks with various attributes such as priority, status, and due date.
- **Main Task:**
  - A Main Task would be a task that can hold subtasks or also be independent itself.
- **Sub Task:**
  - A Sub Task would be tasked under the main task usually splitting the main task into separate parts.
- **Priorities:**
  - Priorities are used to differentiate the importance of the different tasks
    - High: Very Important
    - Medium: Important
    - Low: Not as Important as of now
- **Progression Percentage:**
  - This would be used to name the progression bar that we plan to implement. The progression percentage will show the progress that the user has on certain tasks visually through a progression bar.
- **Due Date**
- **Log In**
- **Log Out**
- **Register**

- **Username**
- **Email**
- **Password**
- **Account**
- **Settings**

II. Prioritized Functional Requirements

Priority 1:

User:

1. A user shall be able to register for a new account using a username.
2. A user shall be able to register for a new account using a valid email.
3. A user shall be able to register for a new account using a password.
4. A user shall be able to accept the terms and conditions.
5. A user shall be able to log into their account using their email.
6. A user shall be able to log into their account using their username.
7. A user shall be able to log into their account using their password.
8. A user shall be able to change their password.

Task:

9. A task shall be able to be deleted.
10. A task shall be able to be added.
11. A task shall display the percentage completed.
12. A task shall display the overall percentage completion of its subtasks.
13. A task shall be able to nest subtasks inside of other subtasks.
14. A task shall display the progress of a task using a progress bar.
15. A task shall update depending on the subtask's progress.

Journal:

16. A Journal shall generate a prompt for the user based on their higher-priority tasks.
17. A Journal shall have an input section for user reflections.
18. A journal shall be accessible by an account.
19. A Journal shall contain a prompt generated from the user's top 5 priority tasks.

Calendar:

20. A calendar shall have a monthly view.
21. A calendar shall present the title of the task on the task's due date.
22. A calendar shall have the due date of the task on the day the task is due.

Account:

23. An account shall be created by one user.
24. An account shall only be created by a user able to properly consent to our terms and conditions without the assistance of a parent or guardian.
25. An account shall be created with a username.
26. An account shall be created with an email.
27. An account shall be created with a password.
28. An account shall be created with a confirmed password.
29. An account shall be able to access the “app” only when they have an internet connection.
30. An account shall be able to mark tasks as done.
31. An account shall be able to delete tasks.
32. An account shall be able to name new tasks.
33. An account shall be able to rename existing tasks.
34. An account shall be able to add a new task, regardless if a previous task is incomplete.
35. An account shall have multiple journal entries.
36. An account shall display the account username.

ToDo List:

37. A ToDo list shall be searchable by the user.
38. A ToDo list shall delete all subtasks beneath it upon deletion of a task.
39. A ToDo list shall upon deletion of a to-do list delete all tasks inside that list.

Settings:

40. A setting shall allow the user to change their username.
41. A setting shall allow the user to change their email.
42. A setting shall allow the user to change their password.

Dashboard:

43. A dashboard shall allow the user to see an overview of their tasks.
44. A dashboard shall allow the user to see an overview of their upcoming tasks.

Priority 2:

User:

1. A user shall be able to edit their profile information.
2. A user shall be able to update their email address.

Task:

3. A task shall be able to hold dates and times.
4. A task shall be able to hold certain priority flags depending on the importance as high, medium, or low.
5. A task shall be raised up when given a high priority.
6. A task's deadline shall be able to be shown in the calendar.

Journal:

7. A Journal shall ask the user how they can improve.
8. A Journal shall remind them to reflect on their work.
9. A Journal shall be deleted when the user chooses to delete it.
10. A journal shall have a collection of previous journal entries from the user.
11. A journal shall have a prompt of less than 150 words.
12. A journal shall have an open-ended prompt.
13. A journal shall have a date created.

Calendar:

14. A calendar shall show the user's upcoming deadlines.
15. A calendar shall keep track of passed deadlines as well as upcoming deadlines.
16. A calendar shall present the time the task is due on the day of the task's due date.
17. A calendar shall present the priority of the task on the day of the task's due date.
18. A calendar shall have color-coded tasks.

Account:

19. An account shall be deleted upon the decision of the user.
20. An account shall be able to review their previous journal entries.
21. An account shall be able to specify completion via percentage.
22. An account shall be able to sort through all their tasks by priority.
23. An account shall be able to search through all their subtasks and main tasks.
24. An account shall have a profile icon.

25. An account shall allow the user to edit profile information.
26. An account shall display the account icon.
27. An account shall have a delete profile option.

ToDo List:

28. A ToDo list shall be able to delete a task when the user chooses to delete it.
29. A ToDo list shall be able to be renamed by the user.
30. A ToDo list shall be named by the user upon creation.
31. A ToDo list shall have a name to differentiate it from other To Do lists.

Dashboard:

32. A dashboard shall allow the user to see an overview of their overdue tasks.
33. A dashboard shall allow the user to mark completed quickly and easily.

Priority 3:

Task:

1. A task shall be pushed down when given a low priority.
2. A task shall warn the user if they reach the limit for subtasks.
3. A task shall have a category listed.
4. A task shall be able to be color-coded.

Journal:

5. A Journal shall be searchable by date.
6. A Journal shall be able to be undeleted within 12 hours of deletion.
7. A Journal shall prompt a user to delete old journals if there is not enough space.
8. A Journal shall mark days that the user has not written in.

Calendar:

9. A calendar shall be able to differentiate between todo lists.
10. A calendar shall have a weekly view.
11. A calendar shall be able to transition between different views.
12. A calendar shall have a QR code that will allow the user to view a mobile view of their calendar.

13. A calendar shall have an add task button allowing the user to add tasks from the calendar view.
14. A calendar shall show the main or sub-task completion percentage on their due date.
15. A calendar shall be able to mark tasks as complete.

Account:

16. An account shall be able to undelete a task.
17. An account shall be able to add new subtasks even if the parent task is complete.
18. An account shall be able to sort through all their tasks by percentage.
19. An account shall be able to sort through all their tasks by date.
20. An account shall be able to sort through all their tasks by subtask.
21. An account shall be able to sort through all their tasks by main task.
22. An account shall be able to sort through all their tasks by amount of subtasks.
23. An account shall be able to add an icon next to their task.
24. An account shall be able to color code their tasks.
25. An account shall display how many in-progress tasks the user has.
26. An account shall display how many high, medium, and low tasks the user has completed.

ToDo List:

27. A ToDo list shall be shareable.
28. A ToDo list shall be importable.
29. A ToDo list shall have a QR code that will allow the user to view a mobile view of their ToDo list.
30. A ToDo list shall be exportable as a static PDF.
31. A ToDo list shall be printable.

Settings:

32. A setting shall allow the user to change the app from light mode to dark mode.
33. A setting page shall allow the user to set the default dashboard page on login.
34. A setting page shall allow the user to set the default view mode for all task views.

Dashboard:

35. A dashboard shall allow the user to see an overview of their completed tasks.
36. A dashboard shall allow the user to sort tasks based on due date.
37. A dashboard shall allow the user to sort tasks based on priority.
38. A dashboard shall allow the user to view stubs for their tasks on a mini calendar.
39. A dashboard mini calendar view shall allow the user to click and mark it complete from within

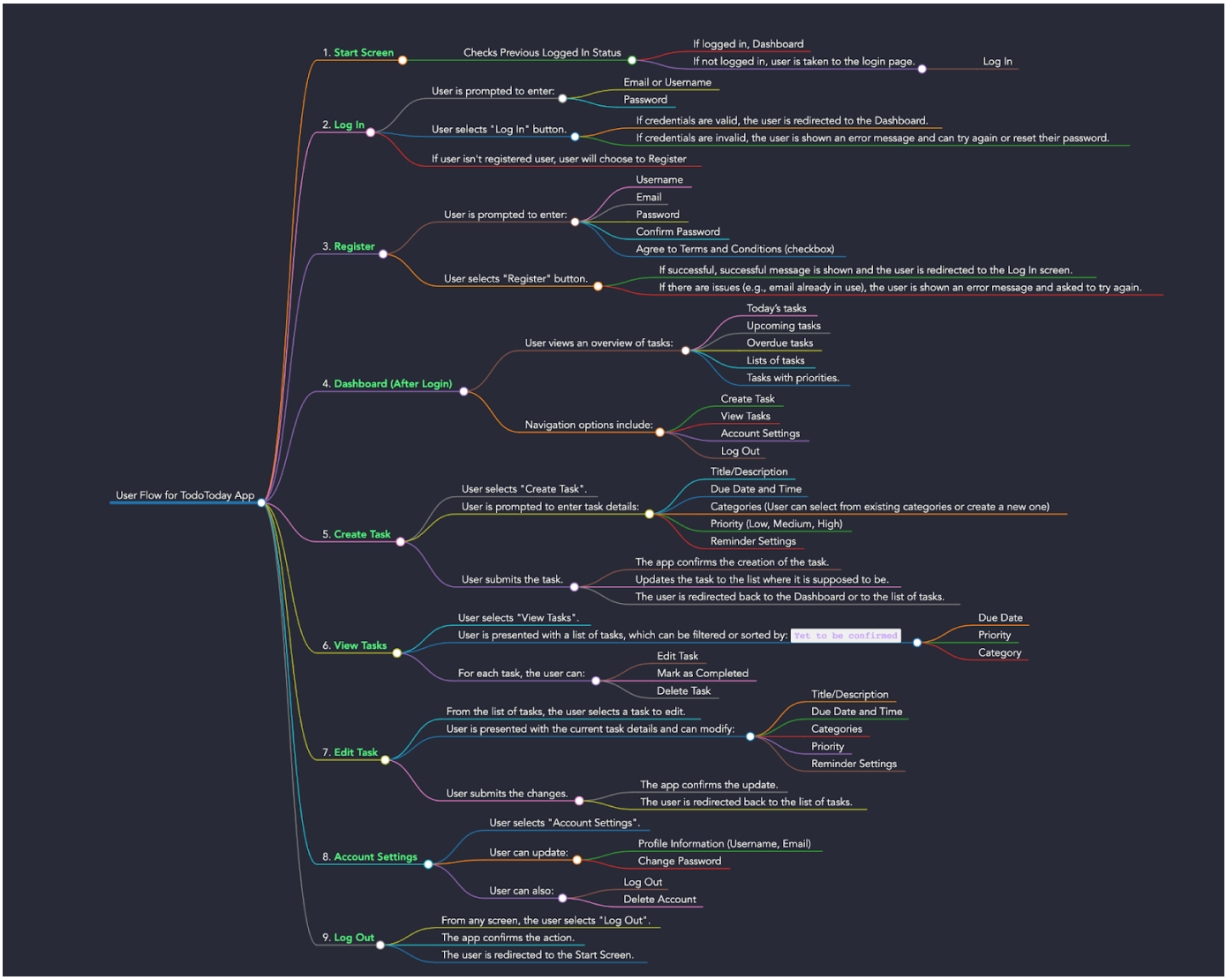
the view.

40. A dashboard mini calendar view shall allow the user to move the task at different times and dates.
41. A dashboard mini calendar view shall allow the user to see the priority of the task.
42. A dashboard shall allow the user to toggle between different categories of tasks.
43. A dashboard shall reflect the newly added task in the appropriate category.

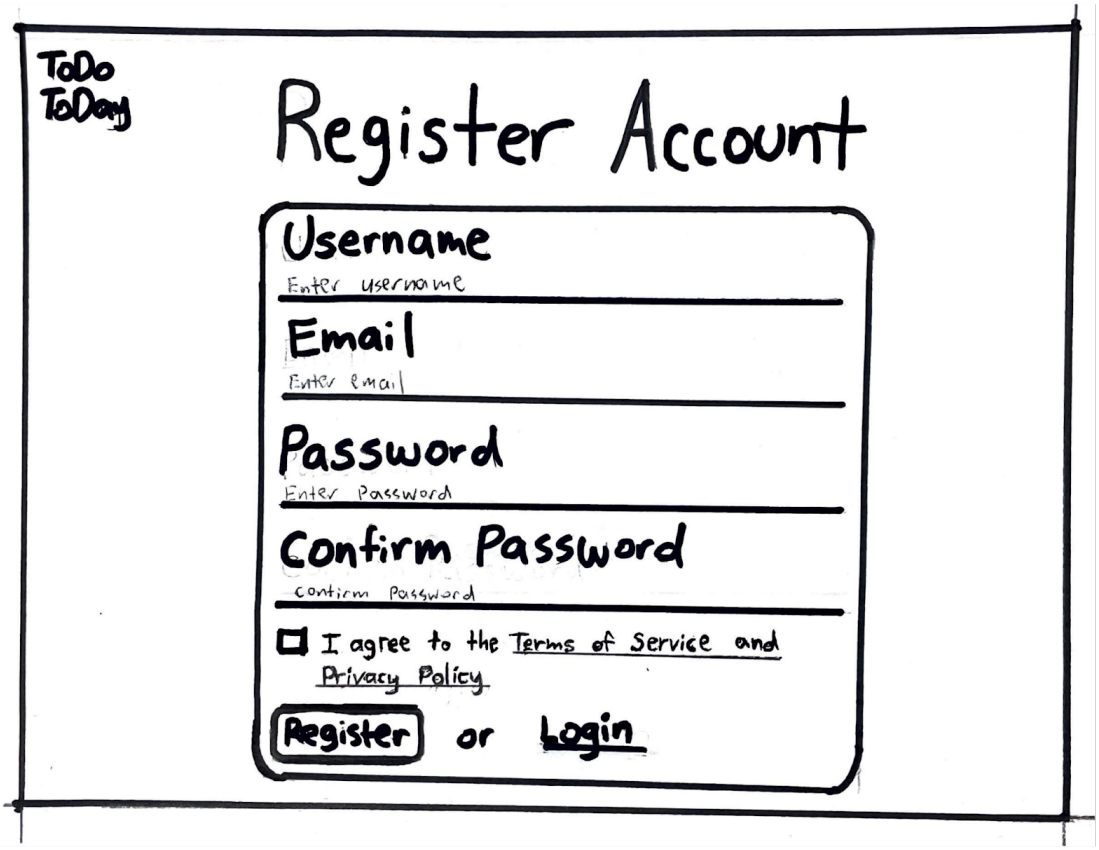
III. UI Mockups and Storyboards (high level only)

UI Mockups:

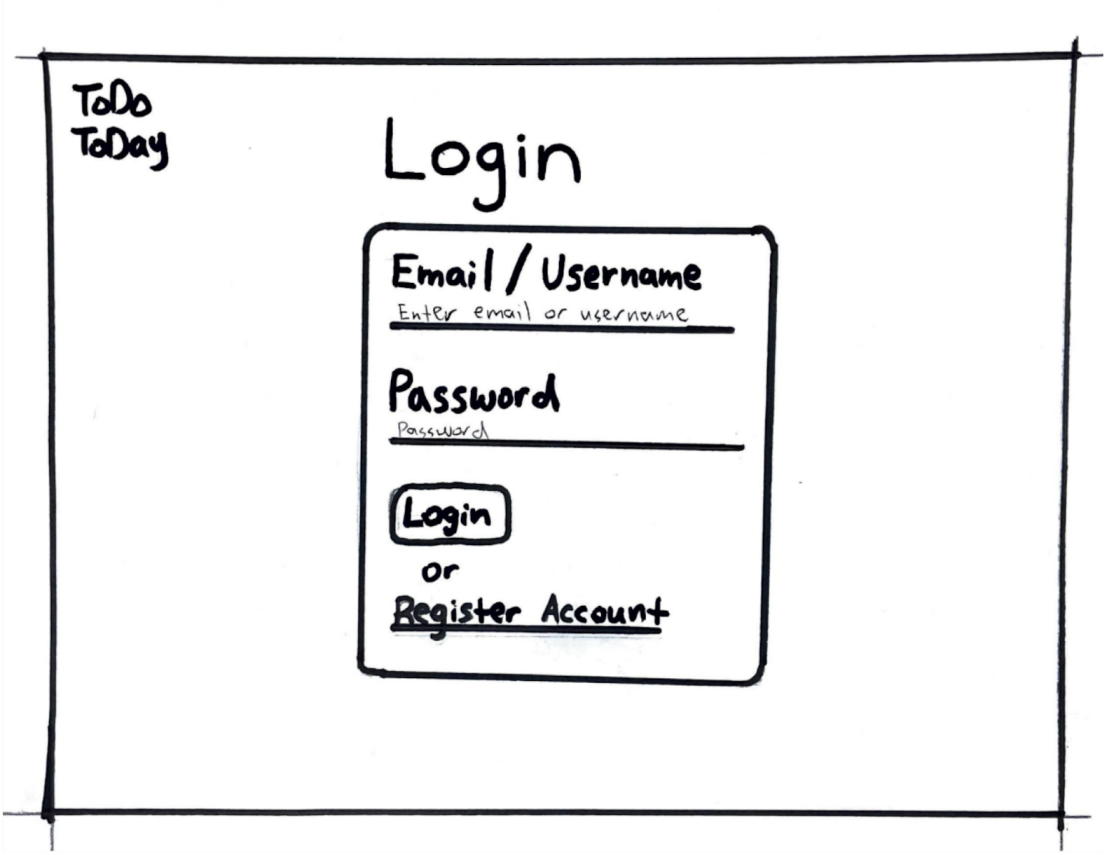
User Flow Diagram



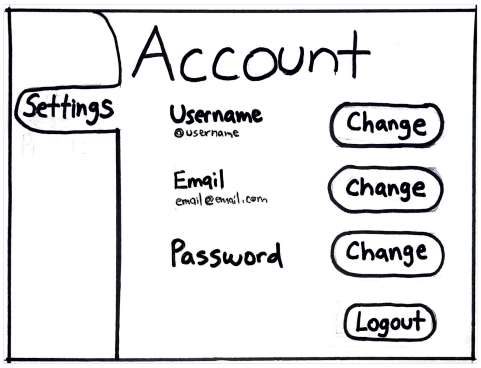
Register Account Page:

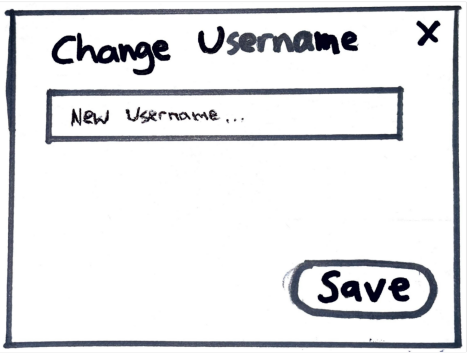


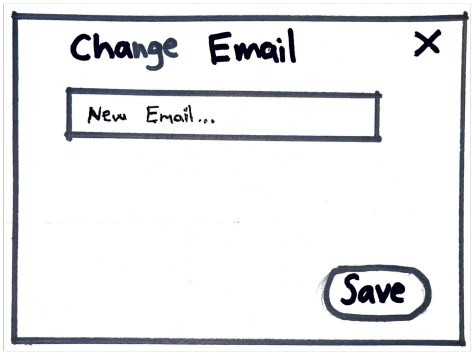
Login Page:

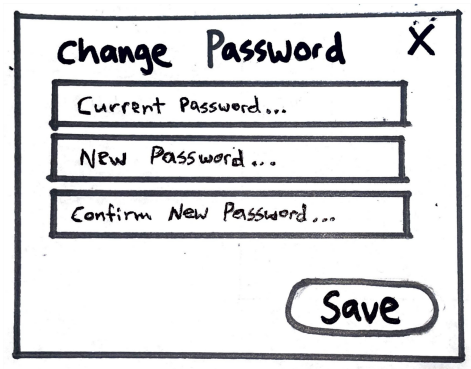


Account Page:

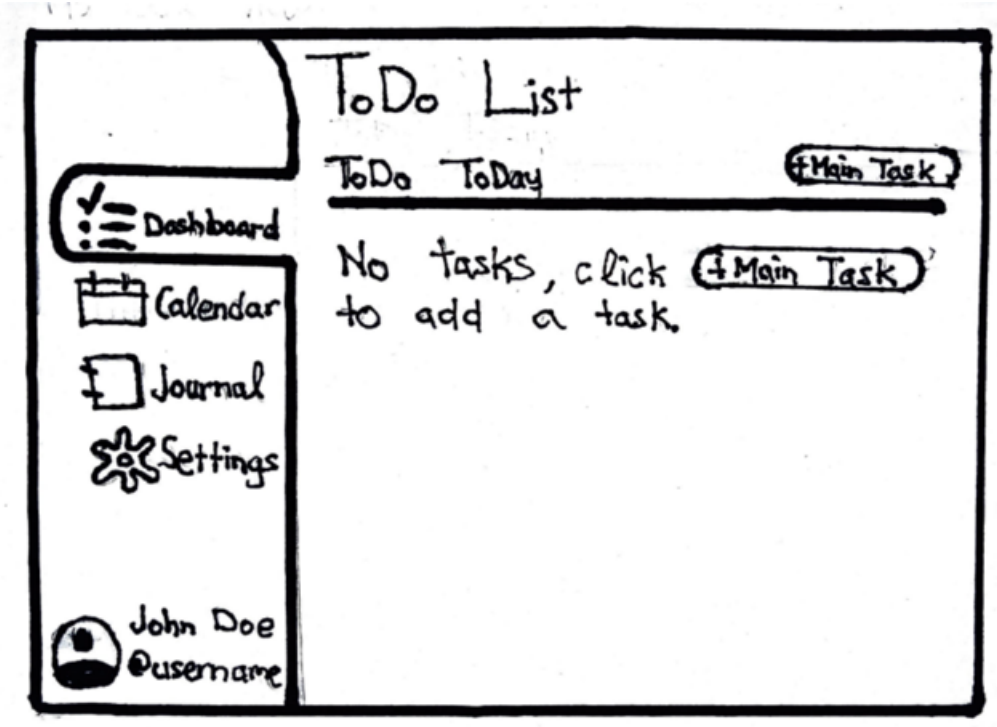




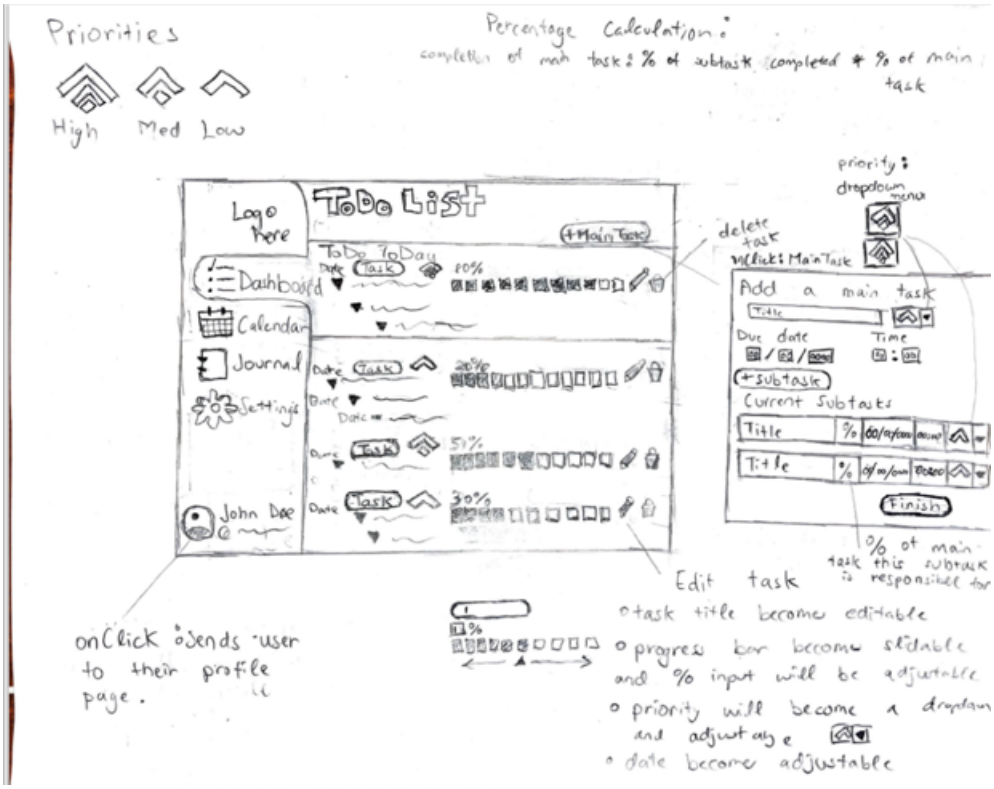




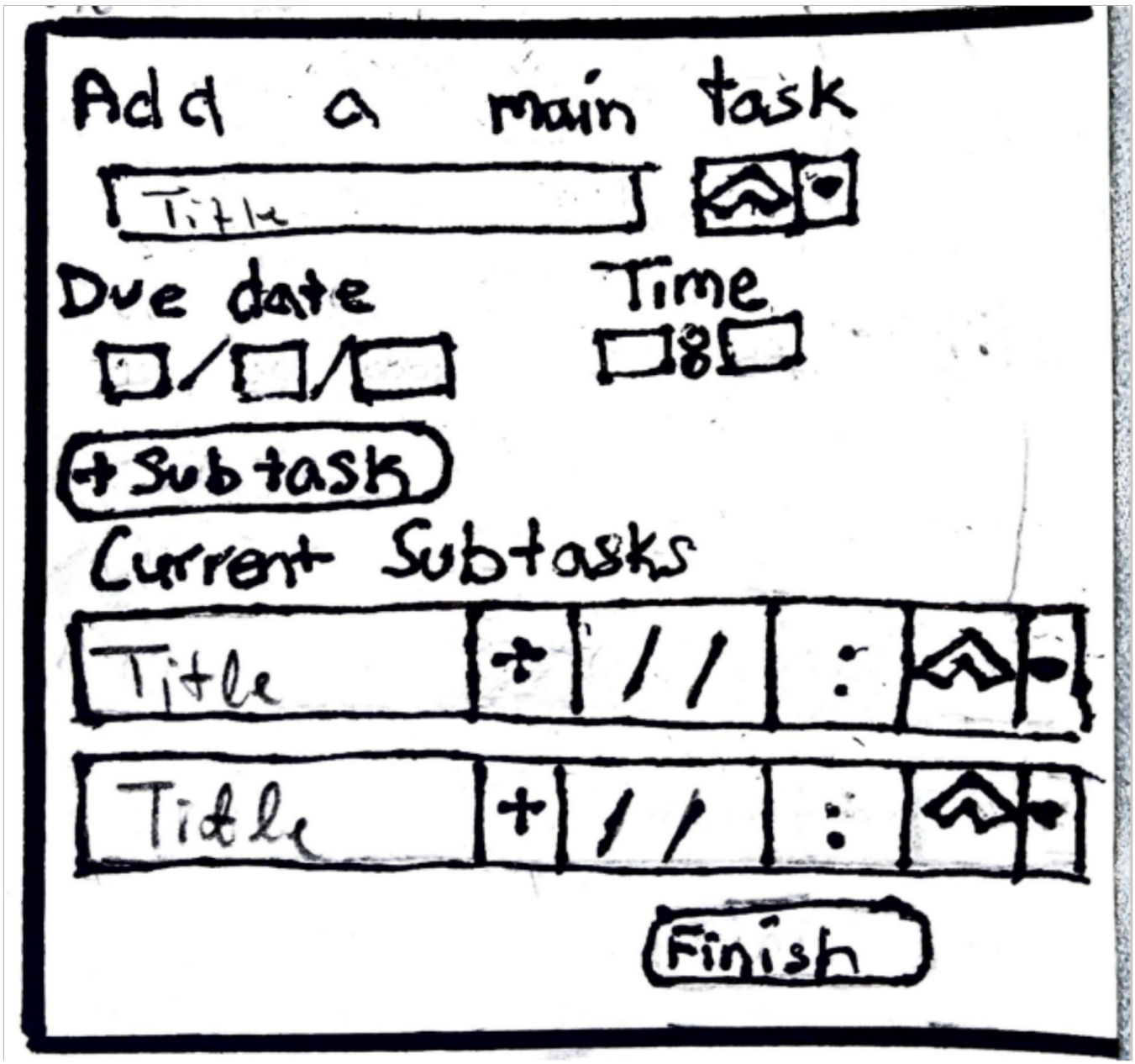
Empty Dashboard Mockup:



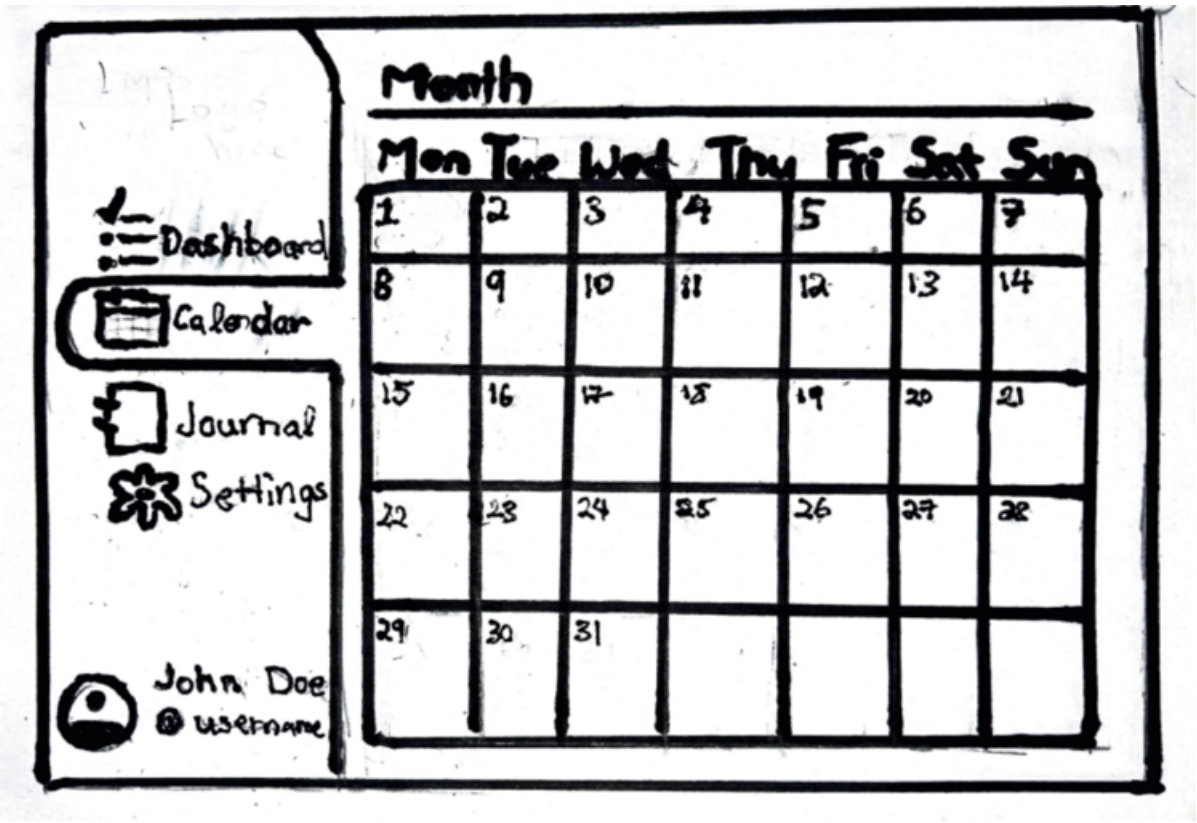
Dashboard Mockup:



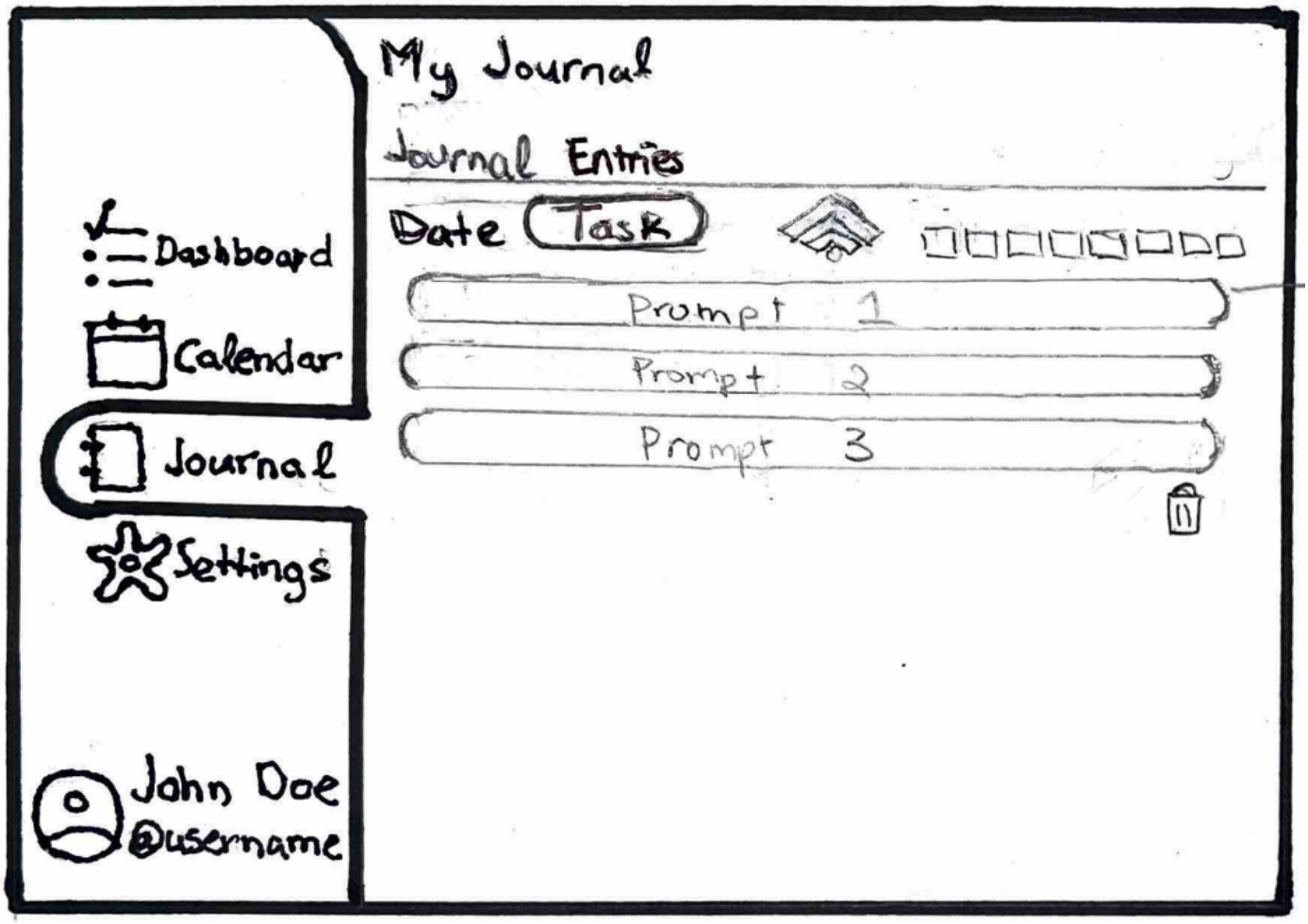
OnClick + Main Task Mockup:

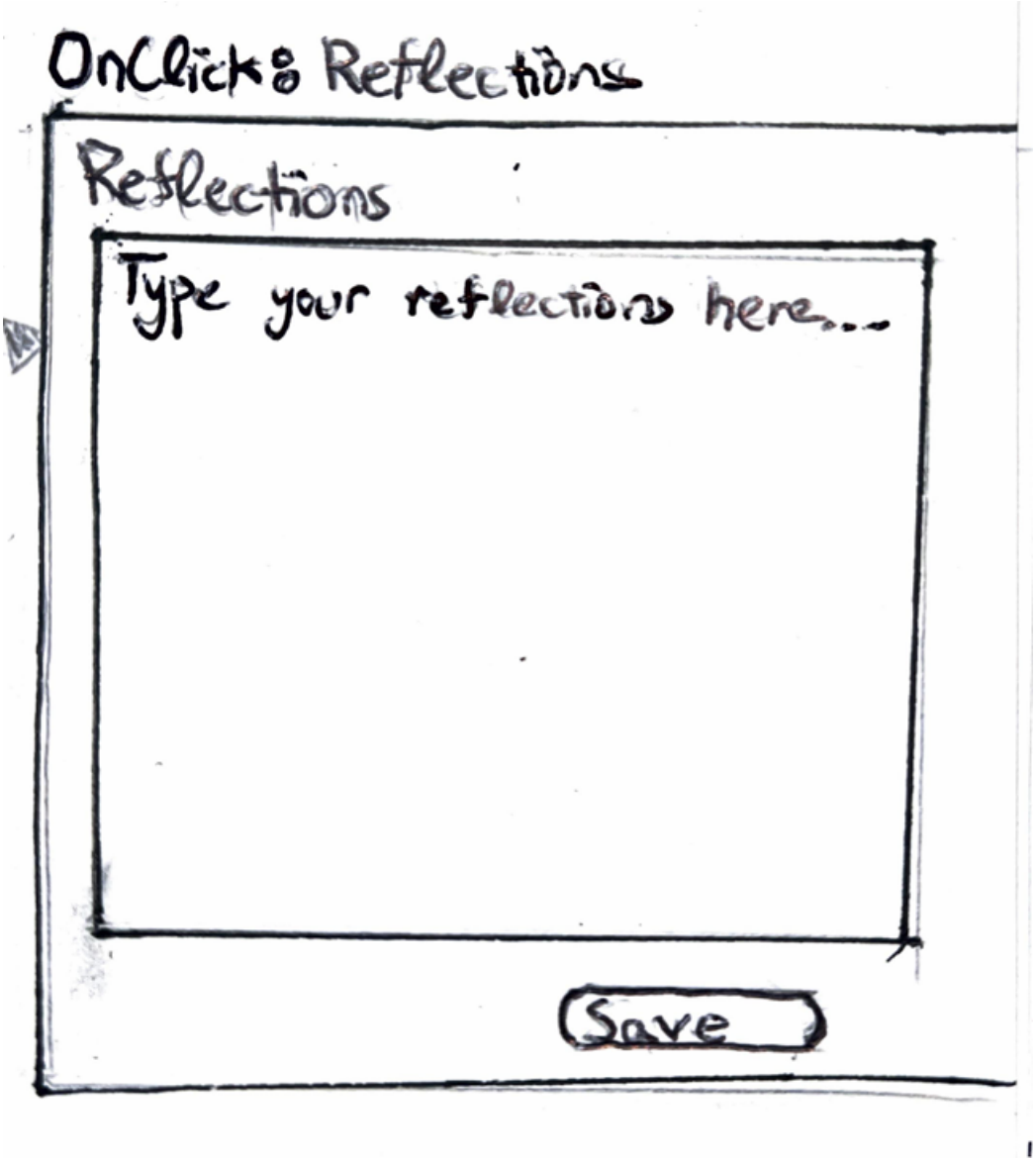


Calendar View Mockup:



Journal View Mockup:

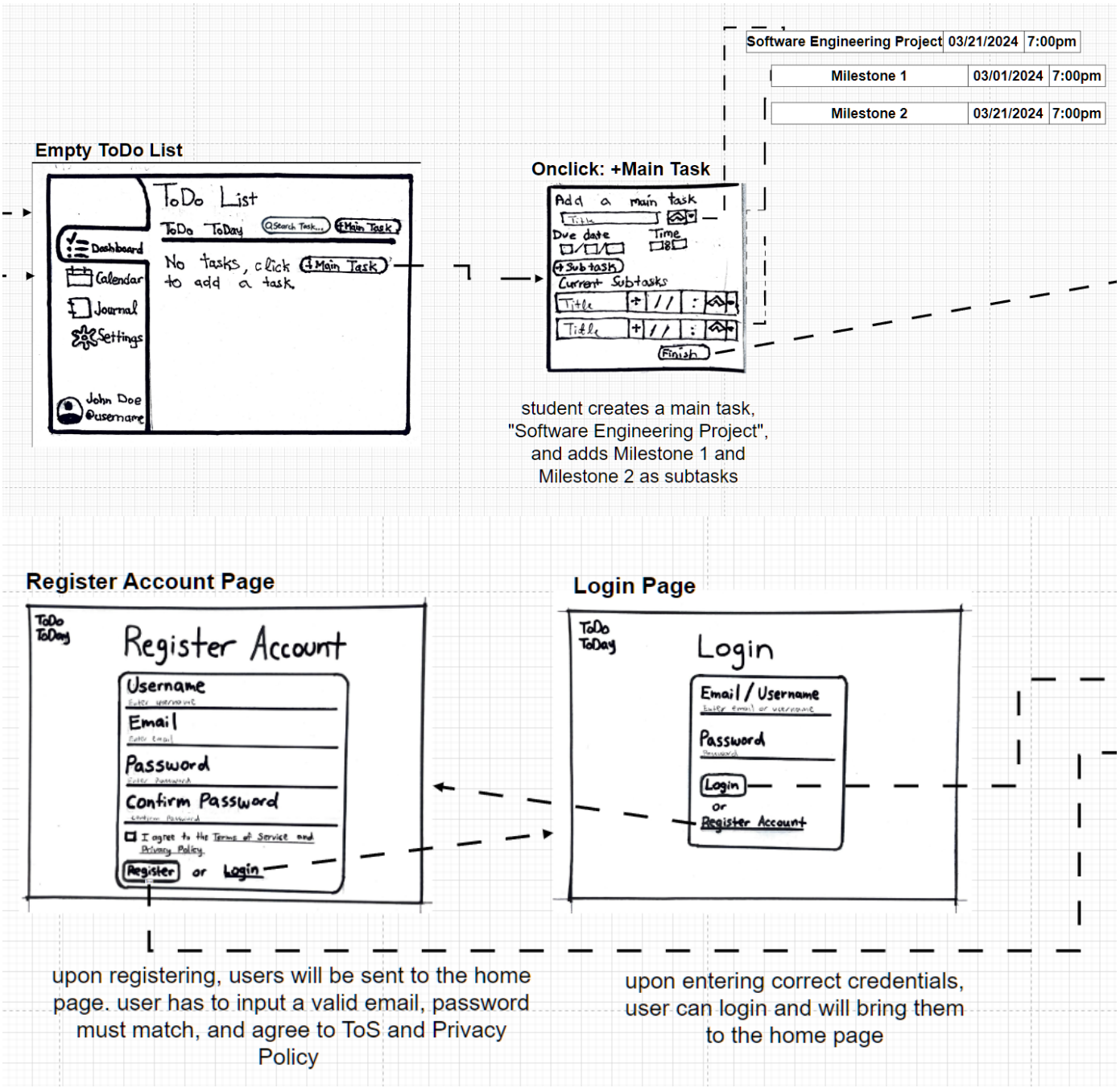



OnClick of Reflections
Reflections
Type your reflections here...
Save

Use Case Storyboards:

The storyboards progress from left to right while moving downwards.

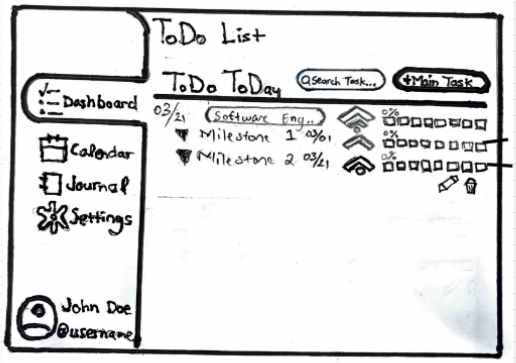
Use Case 1:



Milestone 1

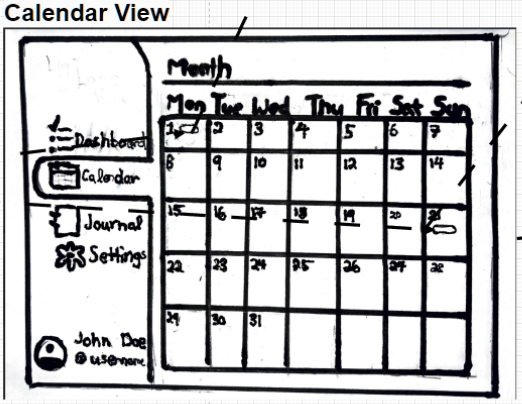
Milestone 2

Added main task and subtask to Dashboard

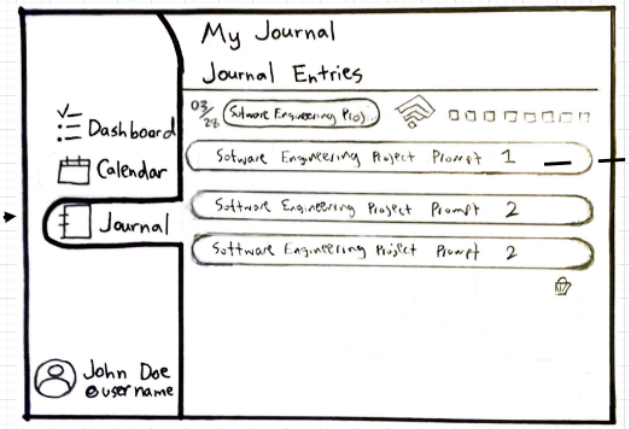


student clicks finish and the main task along with its subtasks now appears in the dashboard showing the due dates, priorities, and progress bar of each task

Calendar View

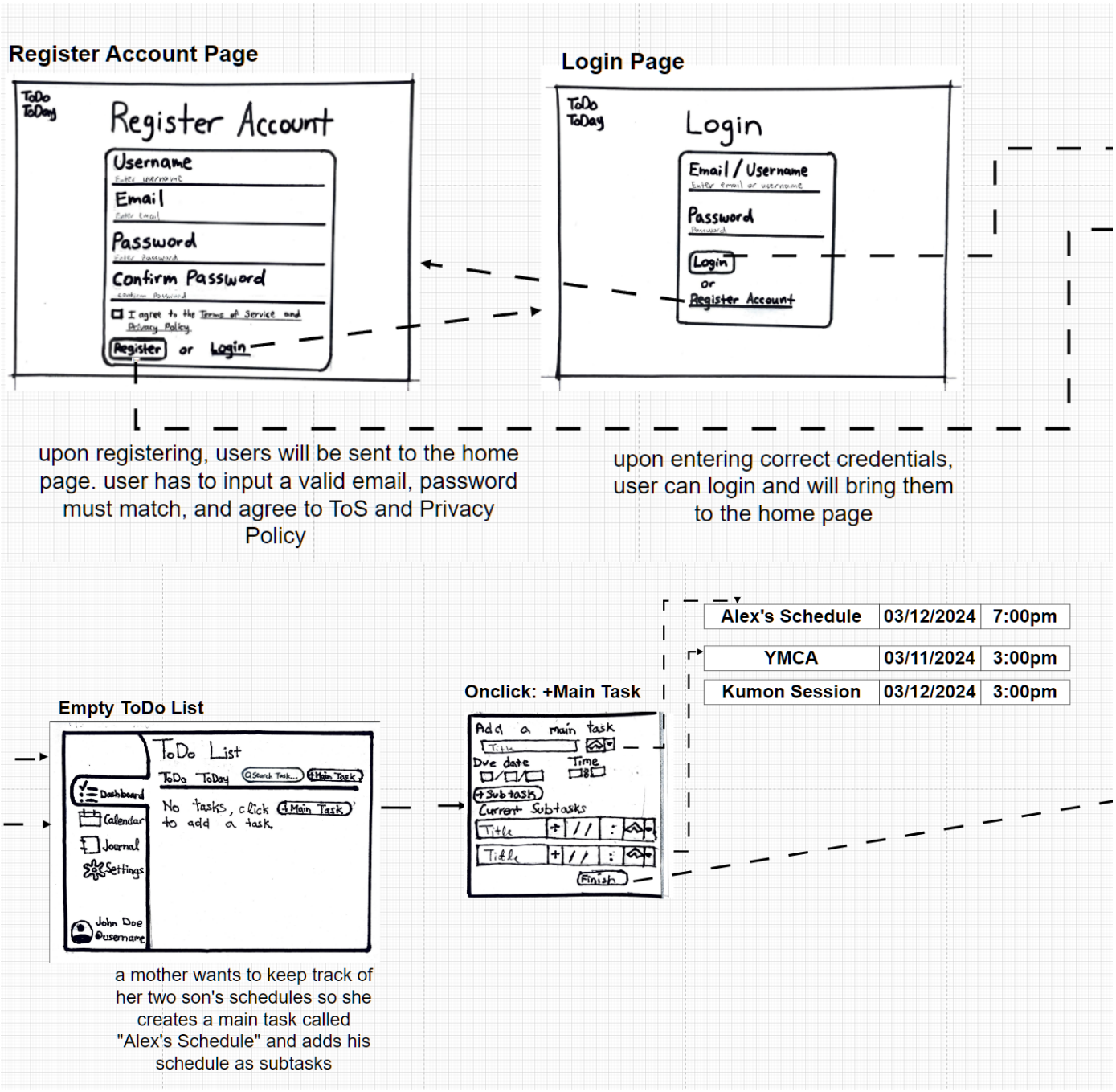


student checks the calendar and each subtask appears on its due date



the student can use the journal to make reflections and talk about what he worked on and how he can improve

Use Case 2:

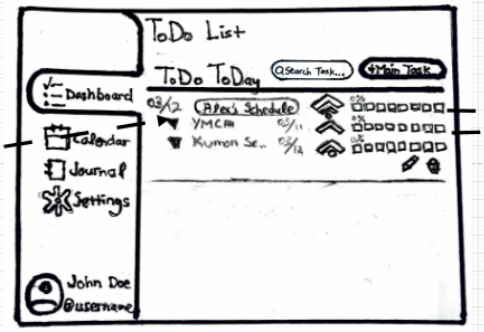

Register Account Page
ToDo ToDay
Register Account
Username
Email
Password
Confirm Password
I agree to the Terms of Service and Privacy Policy
Register
or
Login
Login Page
ToDo ToDay
Login
Email / Username
Password
Login
or
Register Account
upon registering, users will be sent to the home page. user has to input a valid email, password must match, and agree to ToS and Privacy Policy
upon entering correct credentials, user can login and will bring them to the home page
Empty ToDo List
ToDo List
ToDo Today
Search Task...
+Main Task
Dashboard
Calendar
Journal
Settings
John Doe
@username
No tasks, click +Main Task to add a task.
Onclick: +Main Task
Add a main task
Title
Due date
Time
+Subtask
Current Subtasks
Title
Title
Finish
Alex's Schedule 03/12/2024 7:00pm
YMCA 03/11/2024 3:00pm
Kumon Session 03/12/2024 3:00pm
a mother wants to keep track of her two son's schedules so she creates a main task called "Alex's Schedule" and adds his schedule as subtasks

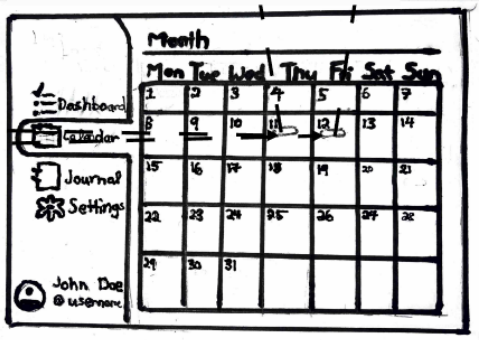
YMCA

Kumon Session

Added main task and subtask to Dashboard

Added subtasks to Month view

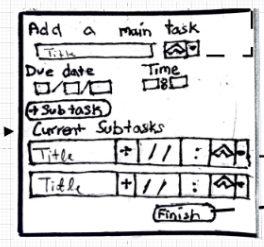


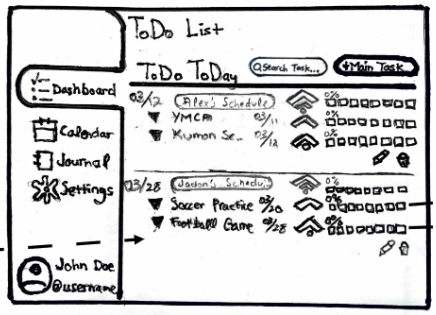


"Alex's Schedule" is now in the home page along with its subtasks and can be viewed in the calendar

Jadon's Schedule	**03/28/2024**	**7:00pm**
Soccer Practice	**03/20/2024**	**3:00pm**
Football Game	**03/28/2024**	**9:00pm**

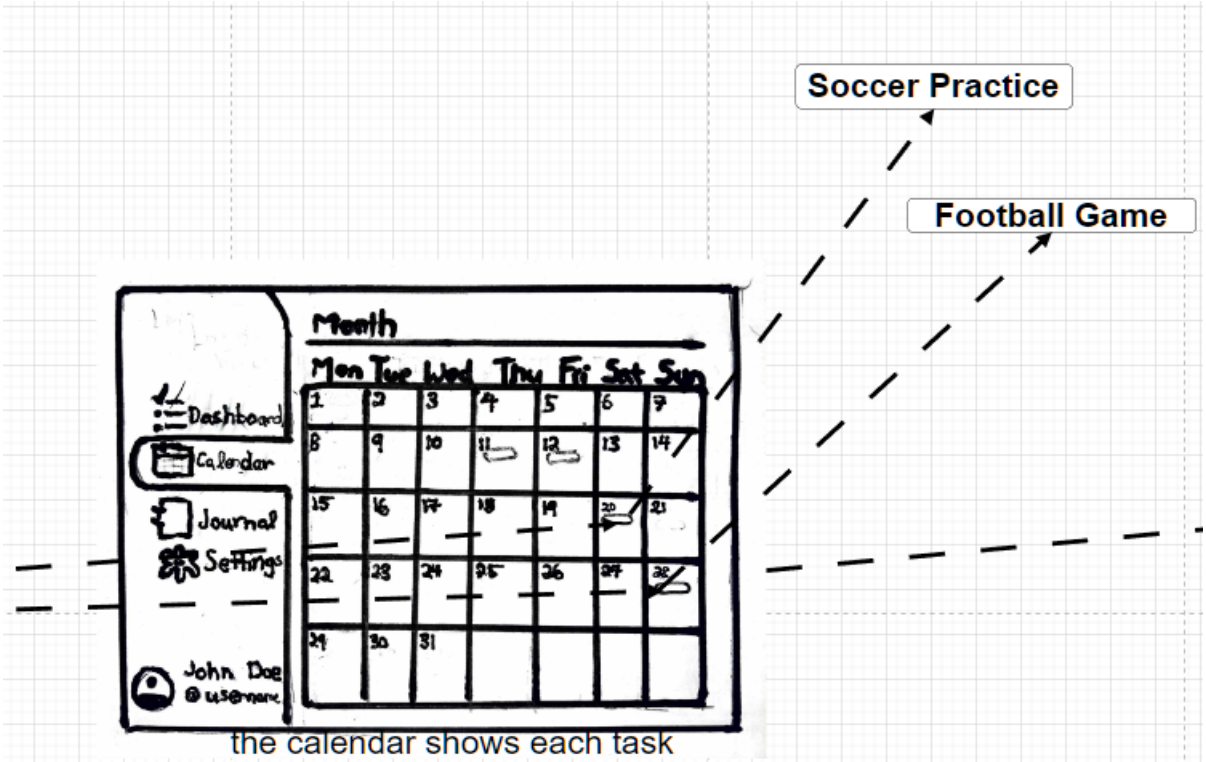
Onclick: +Main Task



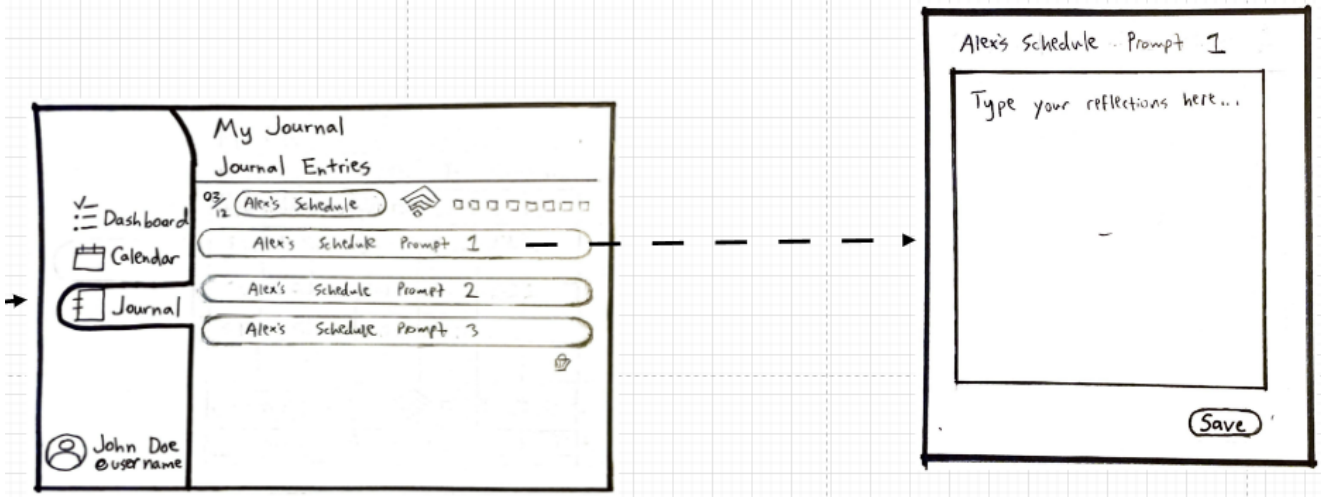


the mother now creates a second main task for her other son titling it "Jadon's Schedule" and creates subtasks for this schedule

both tasks now appear in the dashboard with their corresponding due dates, progress bars, and priorities

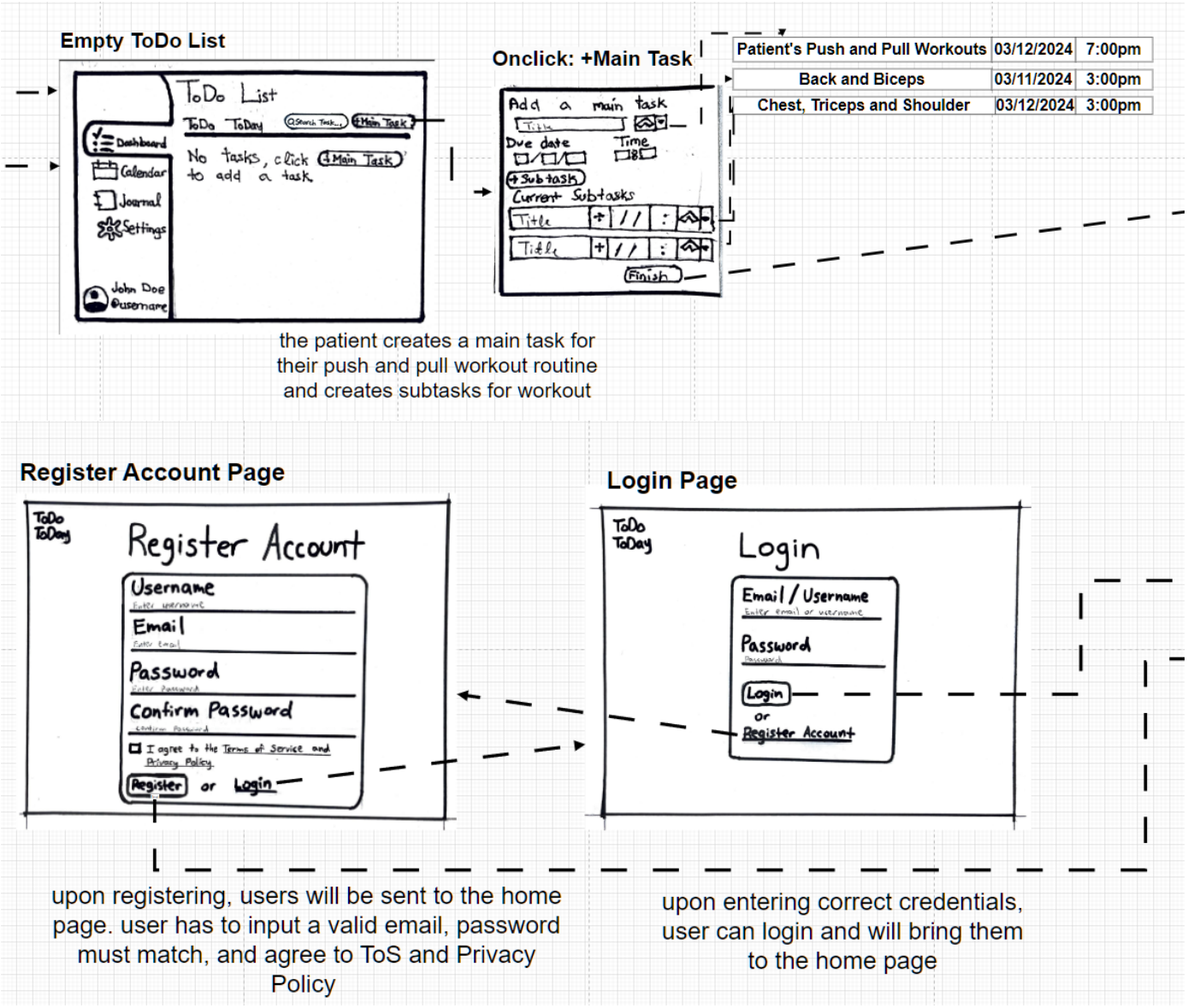


the calendar shows each task on its respective due date, making it easier for the mother to keep track of her sons's schedules



the mother uses the journal to reflect on how she manages her son's schedule

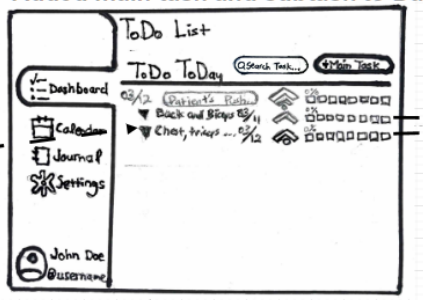
Use Case 3:


Empty ToDo List
ToDo List
ToDo
ToDay
+Main Task
No tasks, click +Main Task to add a task.
Dashboard
Calendar
Journal
Settings
John Doe
@username
Onclick: +Main Task
Add a main task
Title
Due date
Time
+Subtask
Current Subtasks
Title
Title
Finish
Patient's Push and Pull Workouts 03/12/2024 7:00pm
Back and Biceps 03/11/2024 3:00pm
Chest, Triceps and Shoulder 03/12/2024 3:00pm
the patient creates a main task for their push and pull workout routine and creates subtasks for workout
Register Account Page
ToDo ToDay
Register Account
Username
Email
Password
Confirm Password
I agree to the Terms of Service and Privacy Policy
Register
or
Login
Login Page
ToDo ToDay
Login
Email / Username
Password
Login
or
Register Account
upon registering, users will be sent to the home page. user has to input a valid email, password must match, and agree to ToS and Privacy Policy
upon entering correct credentials, user can login and will bring them to the home page

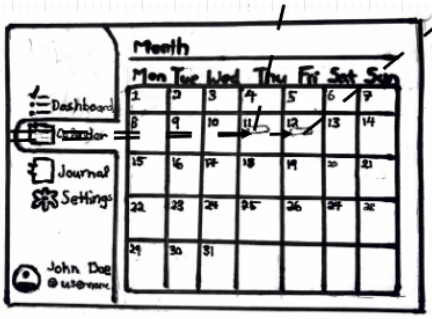
Back and Biceps

Chest, Triceps and Shoulder

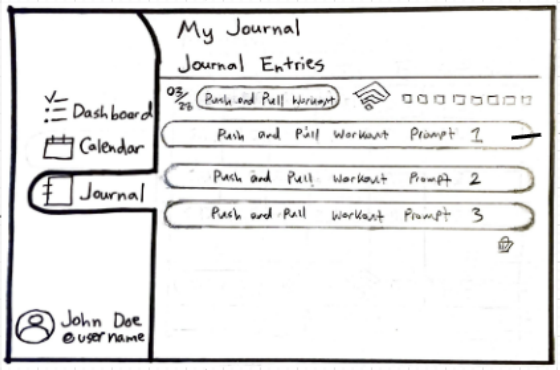
Added main task and subtask to Dashboard



the workout routine is displayed on the dashboard

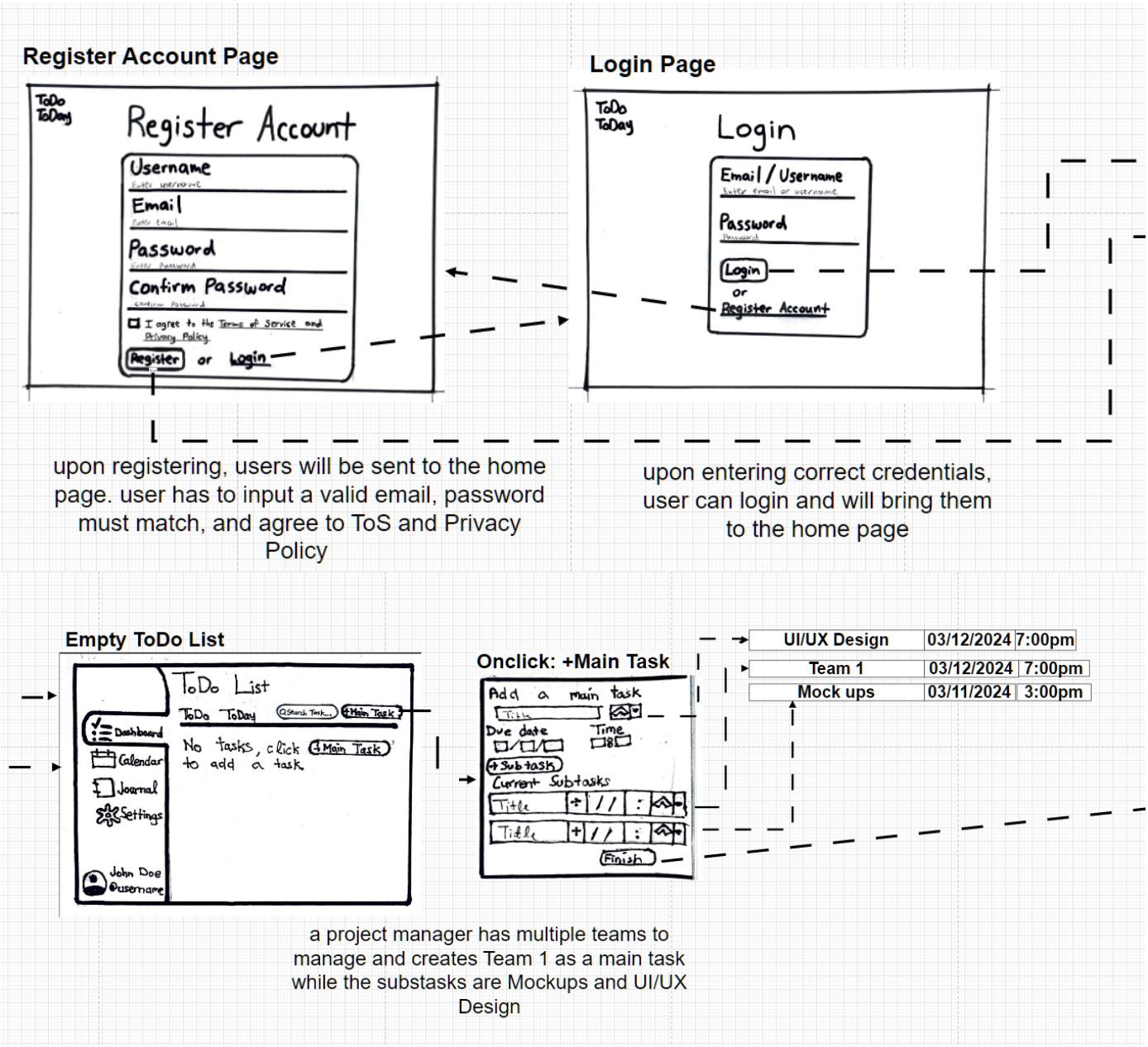


the patient utilizes the calendar to have a better visualization of their workouts throughout the week



the patient uses the journal feature to talk about what they like and didn't like about their workout

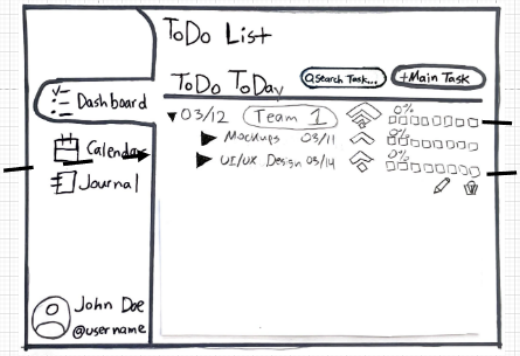
Use Case 4:

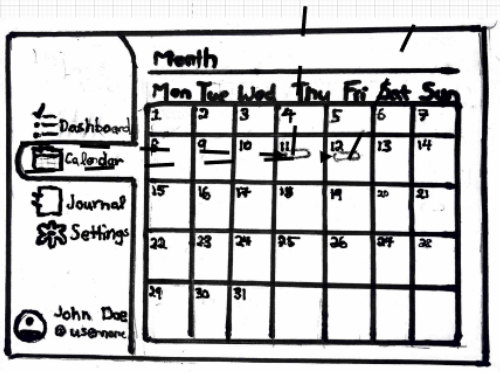

Register Account Page
ToDo ToDay
Register Account
Username
Email
Password
Confirm Password
I agree to the Terms of Service and Privacy Policy
Register or Login
Login Page
ToDo ToDay
Login
Email / Username
Password
Login
or
Register Account
upon registering, users will be sent to the home page. user has to input a valid email, password must match, and agree to ToS and Privacy Policy
upon entering correct credentials, user can login and will bring them to the home page
Empty ToDo List
ToDo List
ToDo ToDay
Search Task...
+Main Task
Dashboard
Calendar
Journal
Settings
John Doe
@username
No tasks, click +Main Task to add a task
Onclick: +Main Task
Add a main task
Title
Due date
Time
+Subtask
Current Subtasks
Title
Title
Finish
UI/UX Design 03/12/2024 7:00pm
Team 1 03/12/2024 7:00pm
Mock ups 03/11/2024 3:00pm
a project manager has multiple teams to manage and creates Team 1 as a main task while the substasks are Mockups and UI/UX Design

Mockups

UI/UX Design

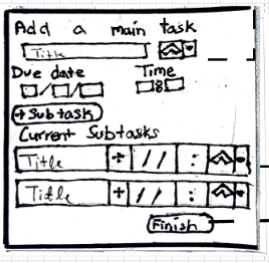
Added main task and subtask to Dashboard



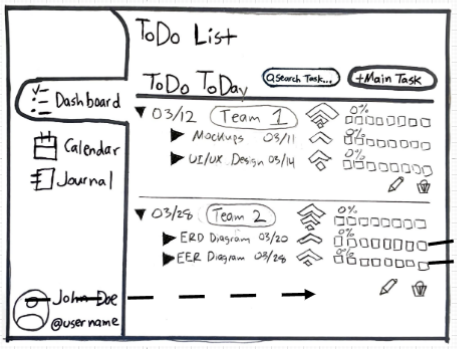


the due dates of the Mockups and UI/UX Design can now be seen in the calendar view

Onclick: +Main Task



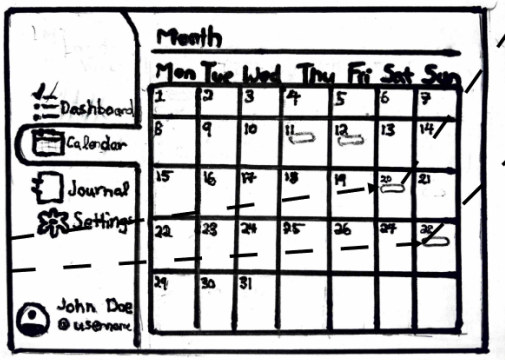
Team 2	**03/28/2024**	**7:00pm**
ERD Diagram	**03/20/2024**	**3:00pm**
EER Diagram	**03/28/2024**	**9:00pm**



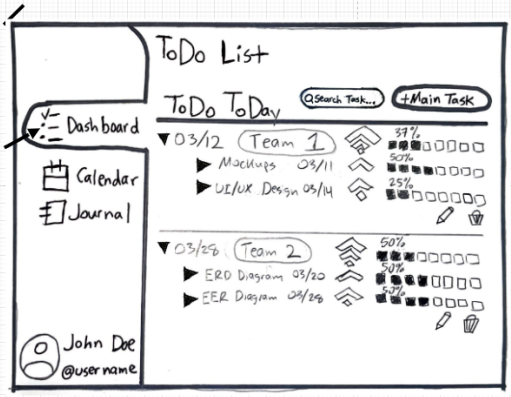
the project manager adds Team 2 with ERD Diagram and EER Diagram as the subtasks

ERD Diagram

EER Diagram

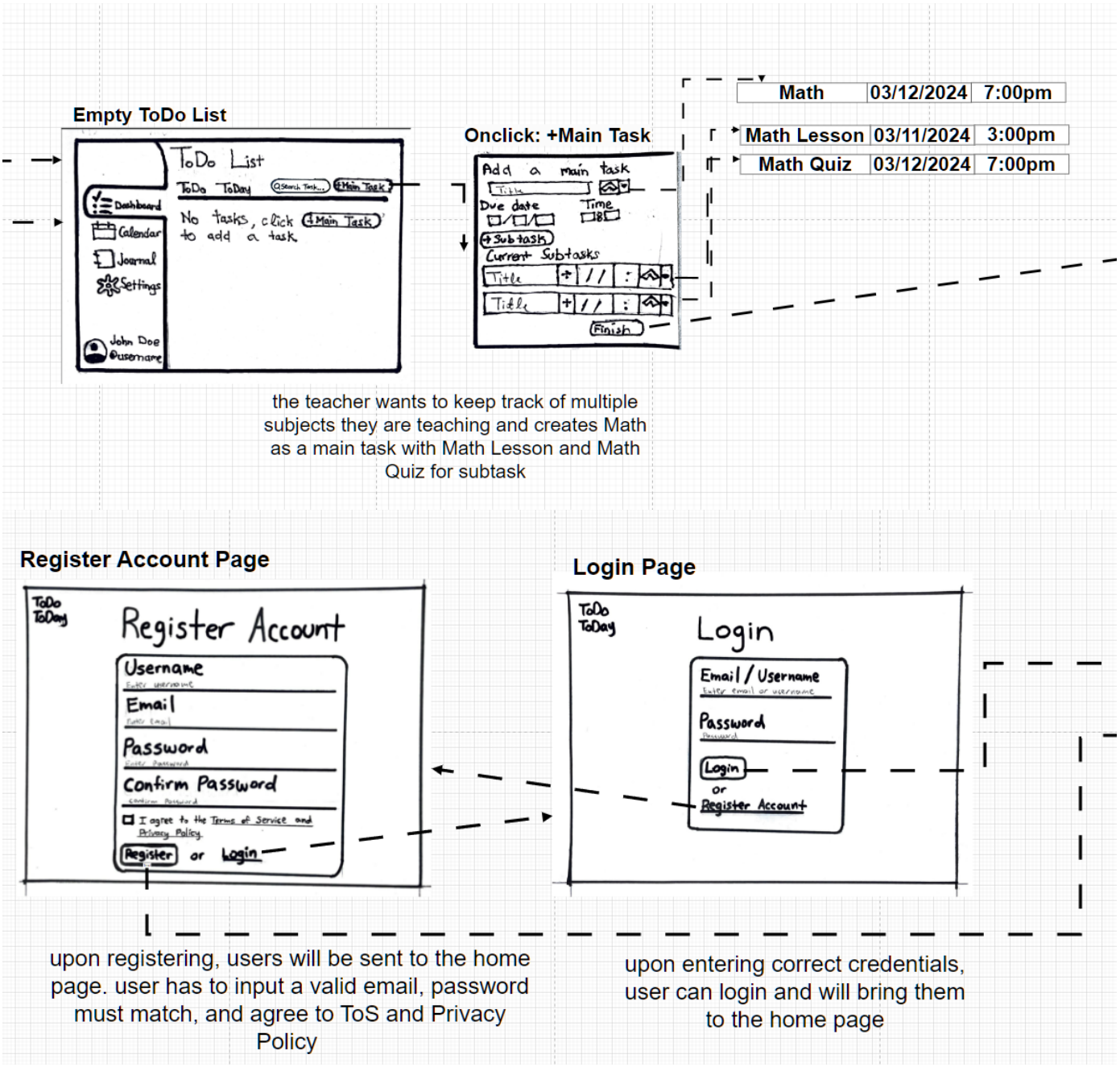


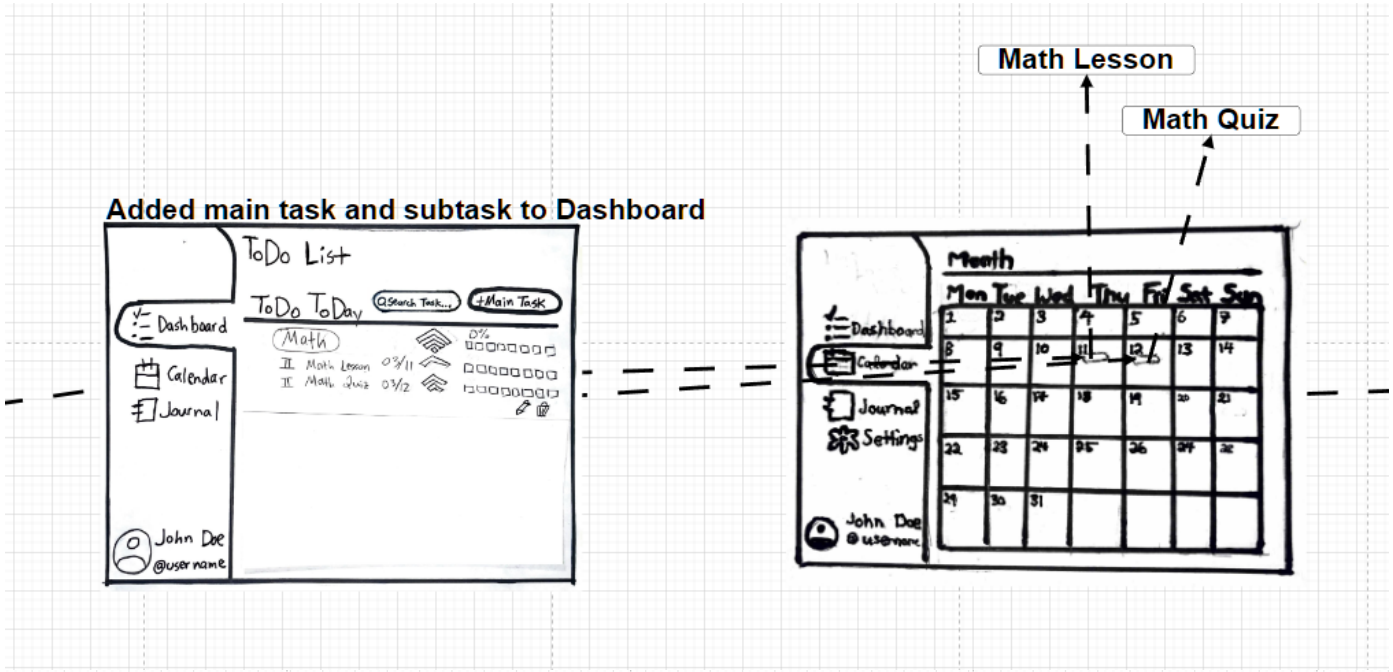
all the subtasks for Team 1 and Team 2 are now seen in the same calendar view

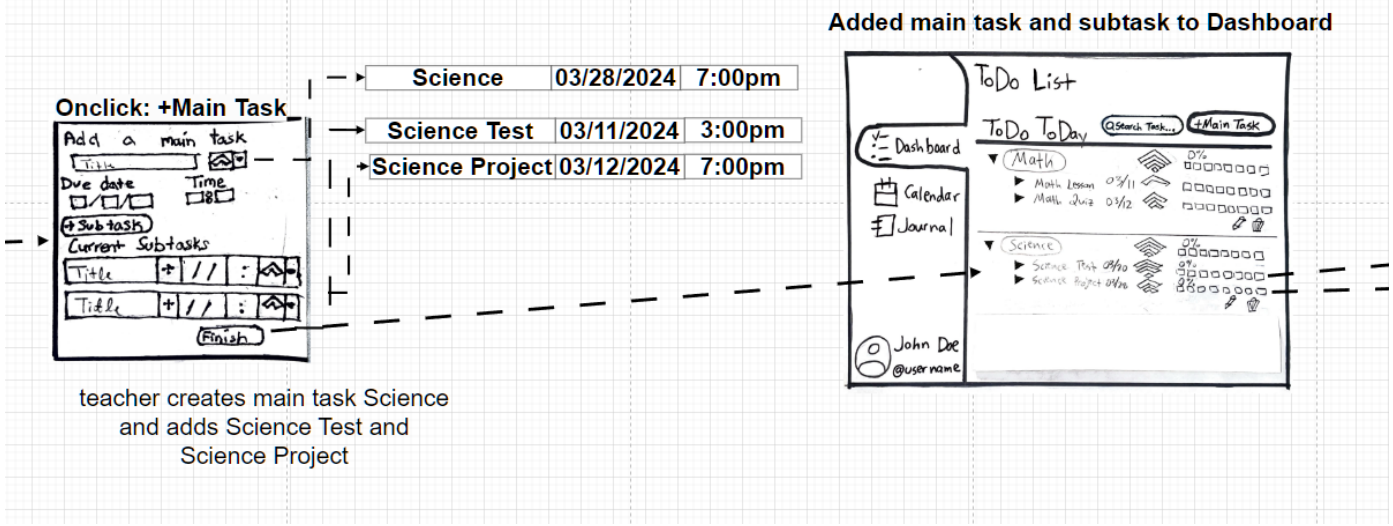


as each team progresses through their subtasks, the project manager marks them partially complete with the progress bar, increasing the progress bar for each team

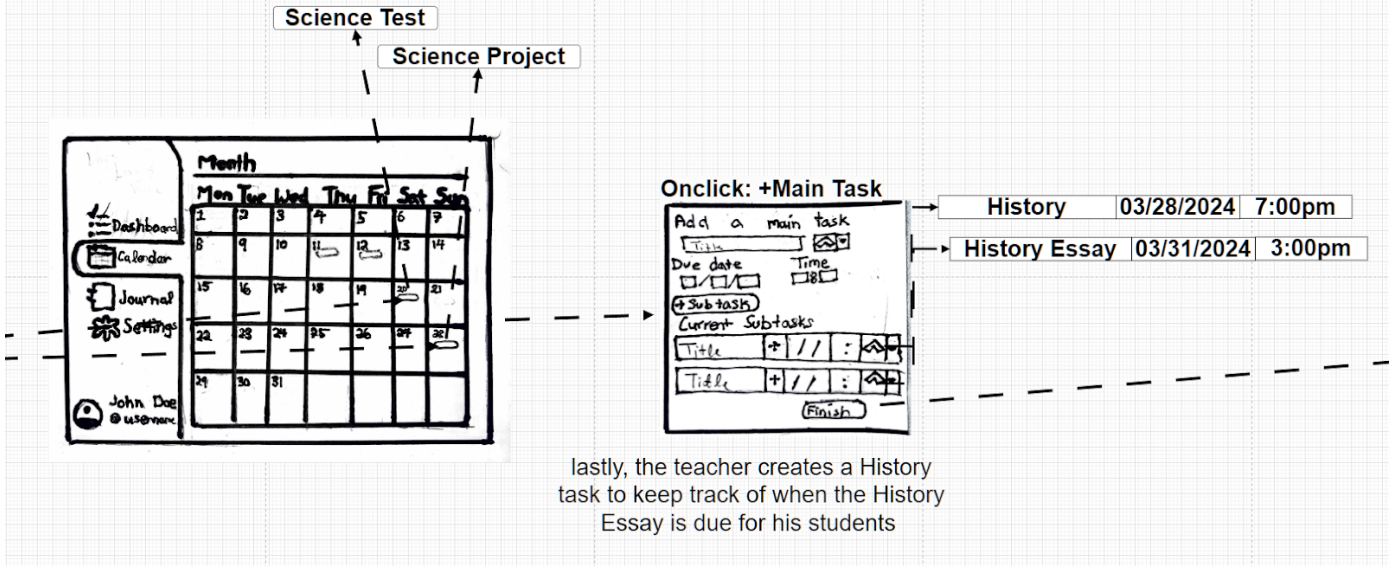
Use Case 5:


Empty ToDo List
ToDo List
ToDo ToDay
Search Task...
+Main Task
No tasks, click +Main Task to add a task
Dashboard
Calendar
Journal
Settings
John Doe
@username
Onclick: +Main Task
Add a main task
Title
Due date
Time
+Subtask
Current Subtasks
Title
Title
Finish
Math 03/12/2024 7:00pm
Math Lesson 03/11/2024 3:00pm
Math Quiz 03/12/2024 7:00pm
the teacher wants to keep track of multiple subjects they are teaching and creates Math as a main task with Math Lesson and Math Quiz for subtask
Register Account Page
ToDo ToDay
Register Account
Username
Email
Password
Confirm Password
I agree to the Terms of Service and Privacy Policy
Register
or
Login
upon registering, users will be sent to the home page. user has to input a valid email, password must match, and agree to ToS and Privacy Policy
Login Page
ToDo ToDay
Login
Email / Username
Password
Login
or
Register Account
upon entering correct credentials, user can login and will bring them to the home page



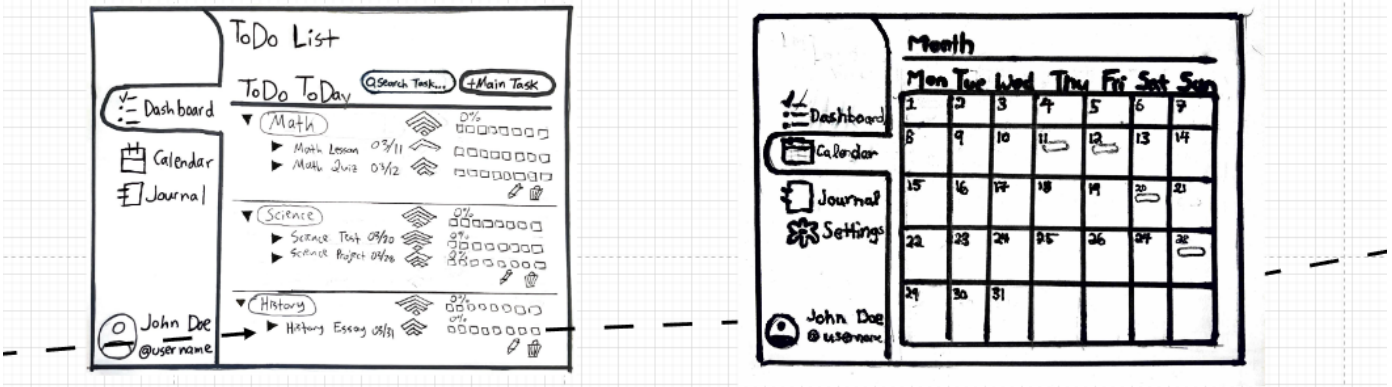


teacher creates main task Science and adds Science Test and Science Project



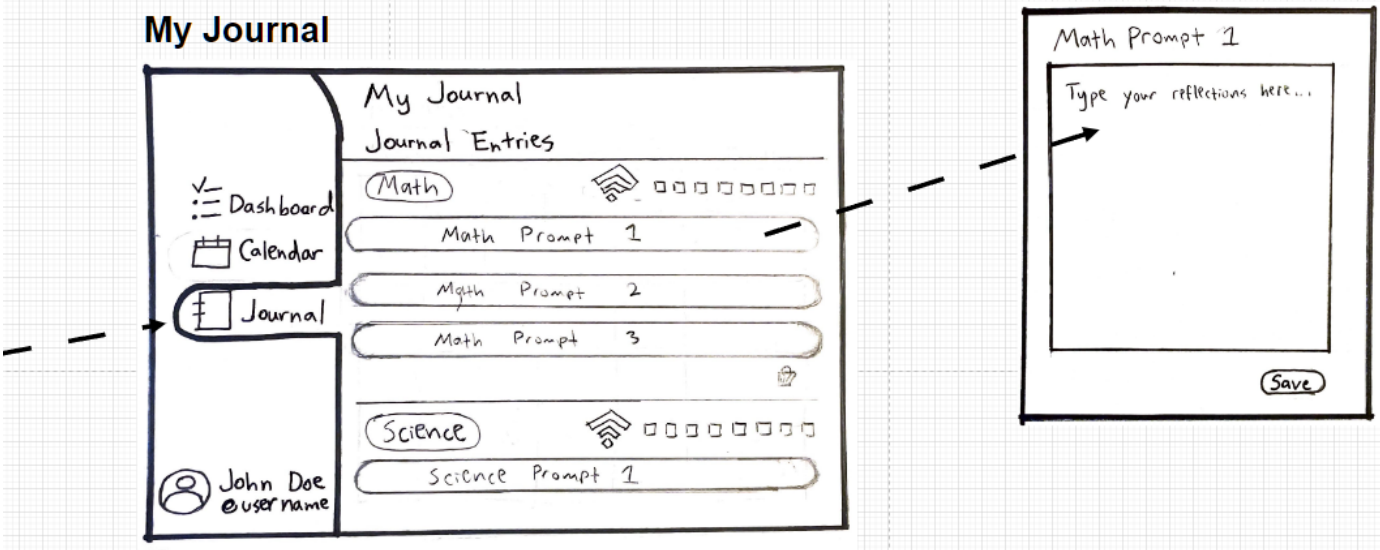
lastly, the teacher creates a History task to keep track of when the History Essay is due for his students

Added main task and subtask to Dashboard



all the tasks and subtasks can be seen in the dashboard with each priority next to each task as well as in the calendar for better tracking and organization for the teacher

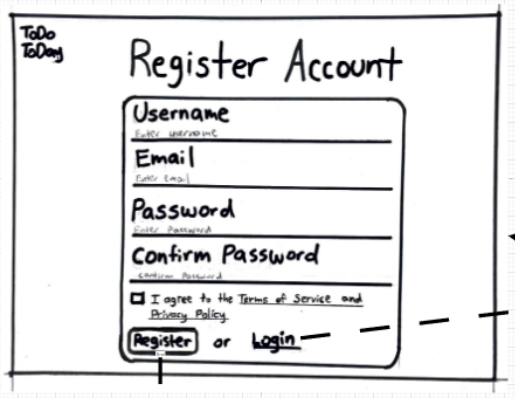
My Journal



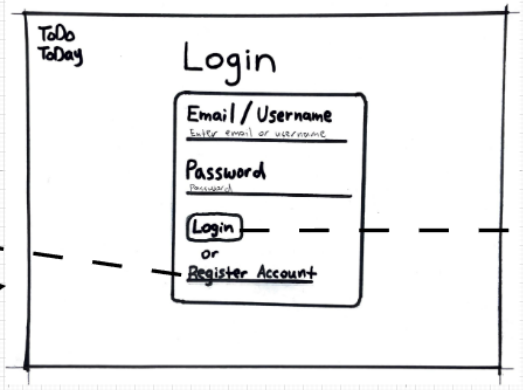
the teacher uses the journal feature to talk about things she liked about her lessons or things that she could improve on

Use Case 6:

Register Account Page



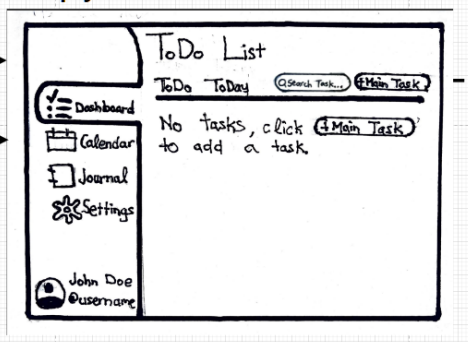
Login Page



upon registering, users will be sent to the home page. user has to input a valid email, password must match, and agree to ToS and Privacy Policy

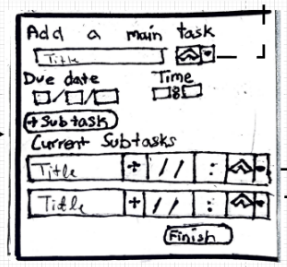
upon entering correct credentials, user can login and will bring them to the home page

Empty ToDo List



Sarah is planning a Europe trip. She wants to be more organized and wants a better way to keep track of her trip. She decides to use ToDoToday

Onclick: +Main Task



Europe Trip	03/28/2024	7:00pm
Item Checklist	03/11/2024	3:00pm
Paris	03/20/2024	7:00pm
London	03/28/2024	7:00pm

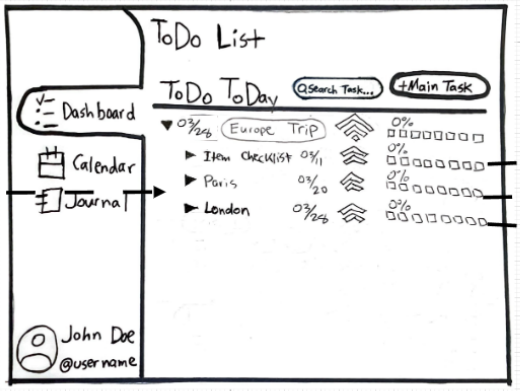
Sarah creates Europe Trip as the main task. She Item Checklist, Paris, and London as the subtasks

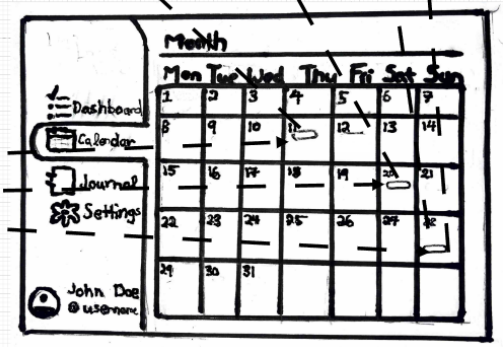
London

Paris

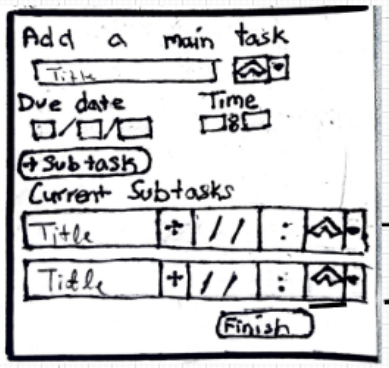
Item Checklist

Added main task and subtask to Dashboard





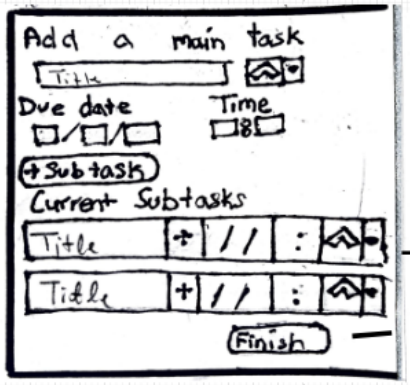
Onclick: +Subtask Item Checklist



Clothes	03/11/2024
Toothbrush	03/11/2024
Shoes	03/11/2024
Vlog Camera	03/11/2024
Passport	03/11/2024
Phone	03/11/2024

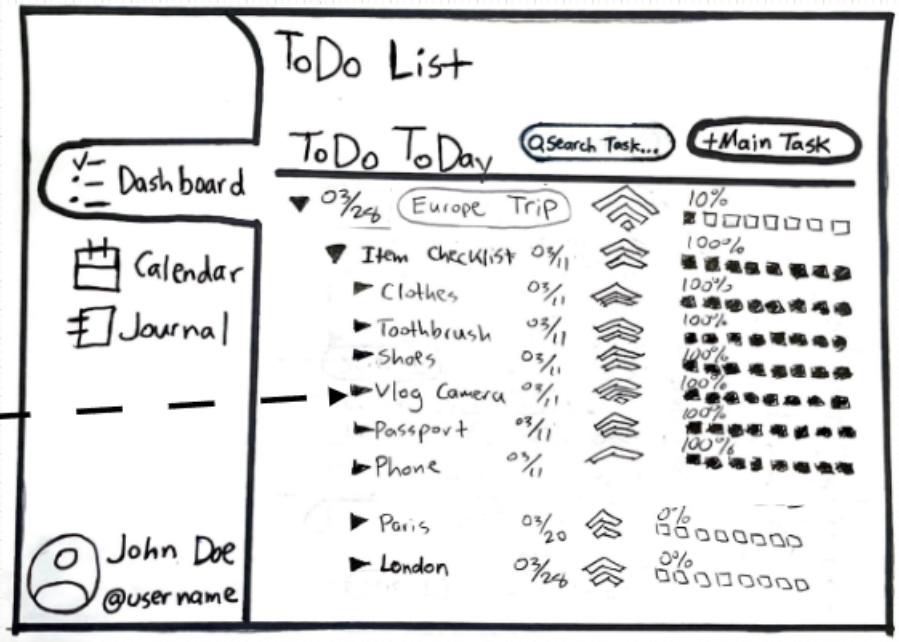
Sarah wants to make sure she has everything she needs for her trip. She adds items to the Item Checklist as subtasks

Onclick: +Subtask Paris

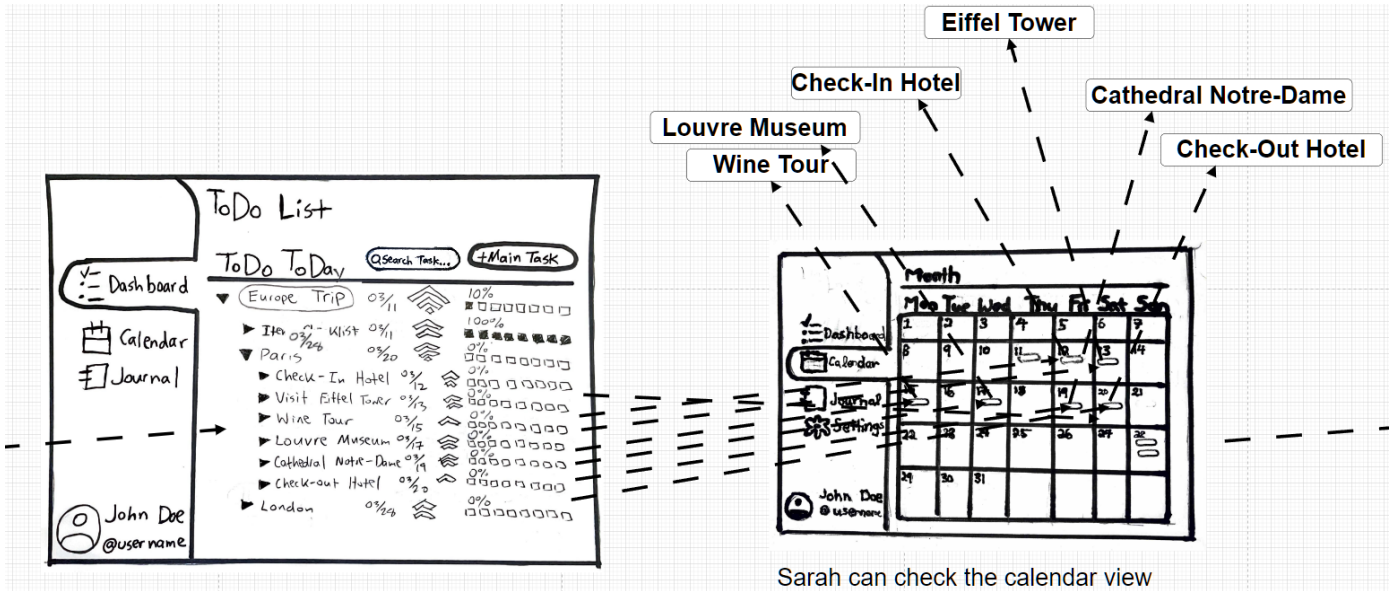


Check-In Hotel	03/12
Eiffel Tower	03/12
Wine Tour	03/15
Louvre Mueseum	03/17
Cathedral Notre-Dame	03/19
Check-Out Hotel	03/20

In the Paris subtask, she adds her main itinerary for her Paris trip and has a better idea of what she is going to do in Paris



After adding everything, she checks them off as she finishes packing them

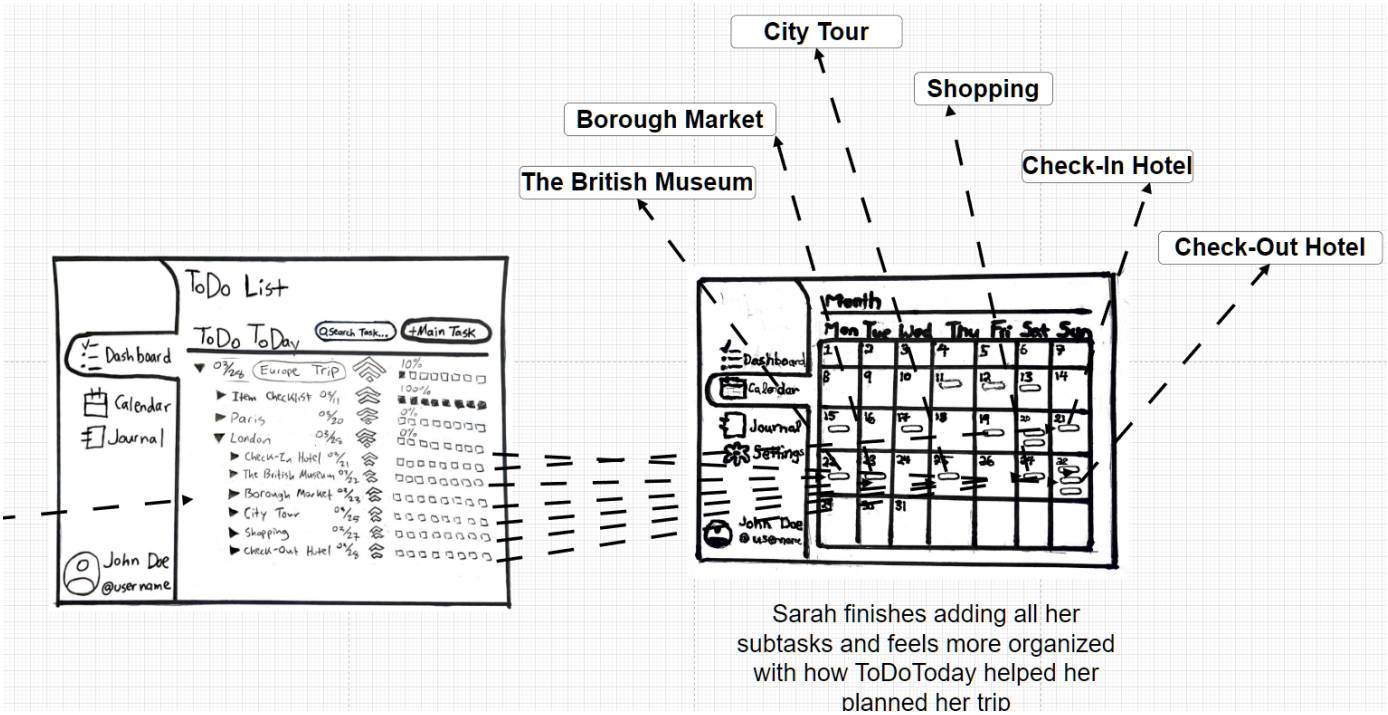


Sarah can check the calendar view for what she has planned during her trip.

Onclick: +Subtask London

Check-In Hotel	03/21
The British Museum	03/22
Borough Market	03/23
City Tour	03/25
Shopping	03/27
Check-Out Hotel	03/28

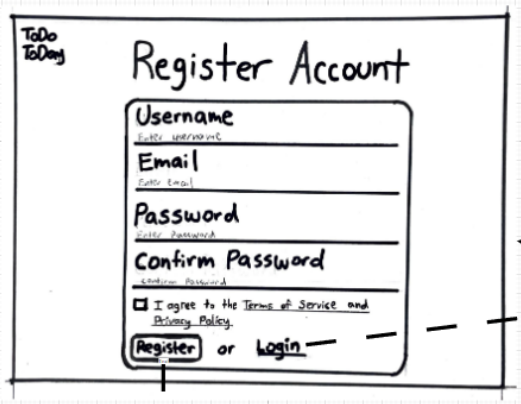
In the London subtask, she adds her main itinerary for her London trip and has a better idea of what she is going to do in London



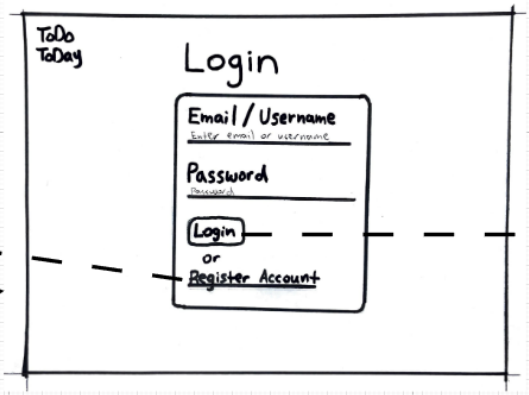
Sarah finishes adding all her subtasks and feels more organized with how ToDoToday helped her planned her trip

Use Case 7:

Register Account Page



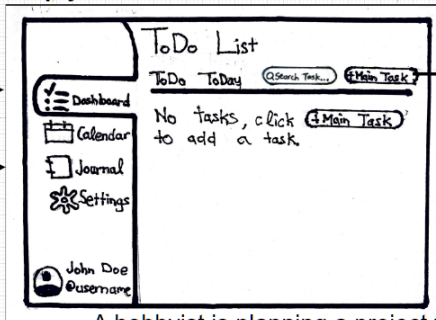
Login Page



upon registering, users will be sent to the home page. user has to input a valid email, password must match, and agree to ToS and Privacy Policy

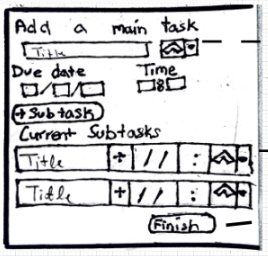
upon entering correct credentials, user can login and will bring them to the home page

Empty ToDo List



A hobbyist is planning a project to build a bookshelf. He heard about ToDoToday from a friend and decides to use it for his bookshelf project.

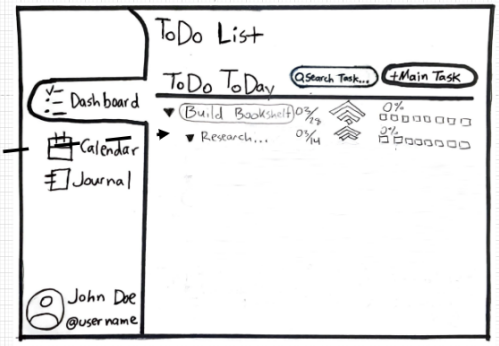
Onclick: +Main Task



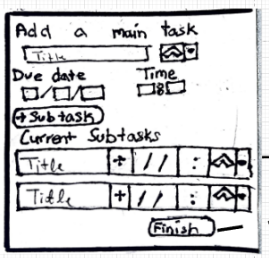
Build Bookshelf	**03/28/24**
Research...	**03/11/24**

The hobbyist creates the main task of building the bookshelf and creates a subtask for research

Added main task and subtask to Dashboard

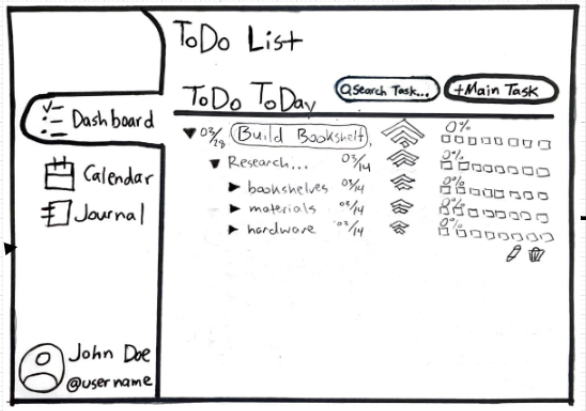


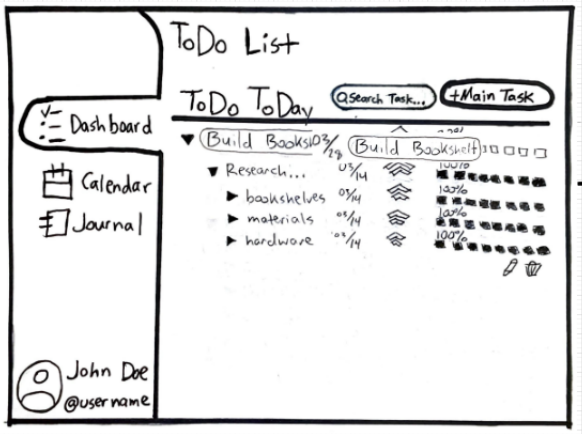
Onclick: +Subtask Research...



bookshelves	03/12/2024
materials	03/12/2024
hardware	03/12/2024

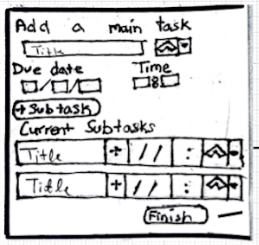
He adds subtasks: bookshelves, materials, and hardware as his research subtasks



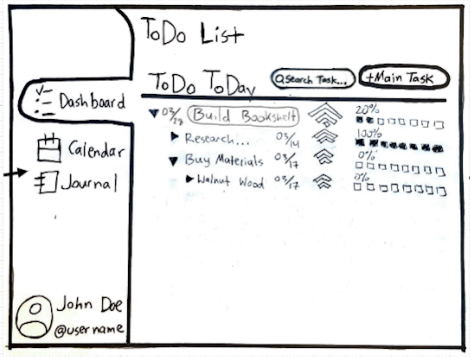


While the hobbyist is researching for his bookshelf, he starts marking them off in ToDoToday

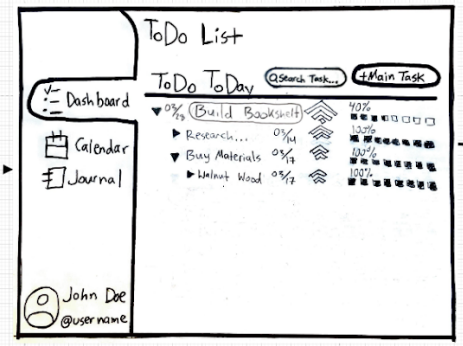
Onclick: +Subtask Build Bookshelf



Buy Materials	03/17/2024
Walnut Wood	03/17



After researching, the hobbyist knows what kind of wood to use. He creates a subtask to buy materials. Under the buy materials subtask, he adds walnut wood

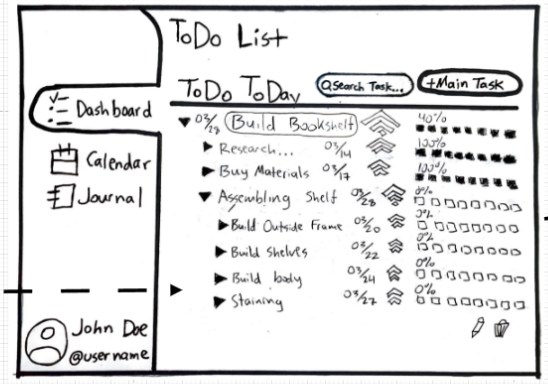


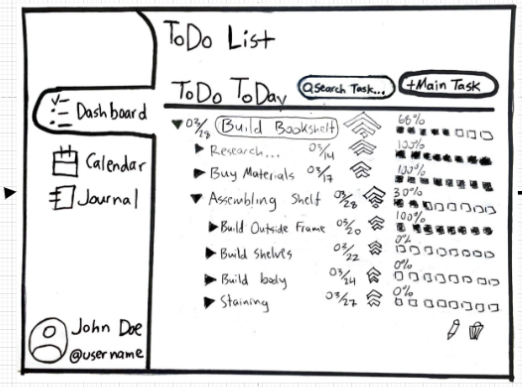
As he finished buying the walnut wood, he checks it off, completing the buy materials task

Onclick: +Subtask Build Bookshelf

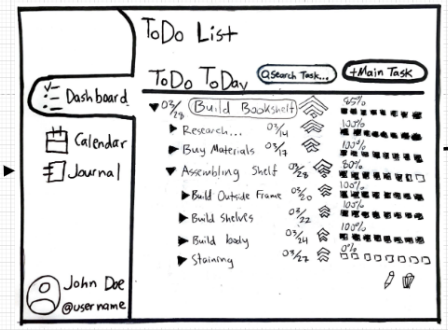
Assembling Shelf	03/12/2024
Build Outside Frame	03/12/2024
Build Shelves	03/12/2024
Build Body	03/12/2024
Staining	03/12/2024

With all the materials, the hobbyist creates the final subtask of assembling the shelf. The subtasks he creates for this are: Build Outside Frame, Build Shelves, Build Body, and Staining

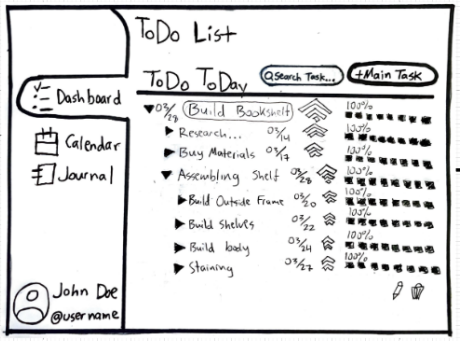




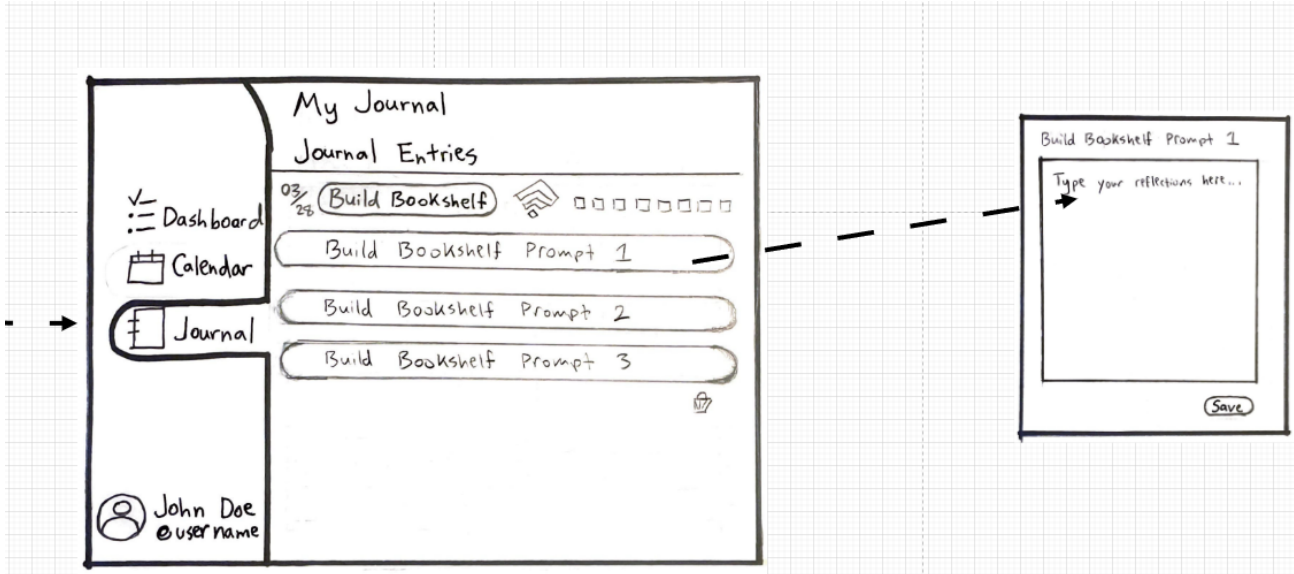
The hobbyist finishes building the outside frame, marking the whole subtask complete. The assembling shelf subtask is now 30% complete and the whole main task is now 68% complete.



The hobbyist goes through his subtasks as he is now done with building the shelves and the body of the bookshelf. Assembling Shelf is now 80% complete and the main task is 85% complete

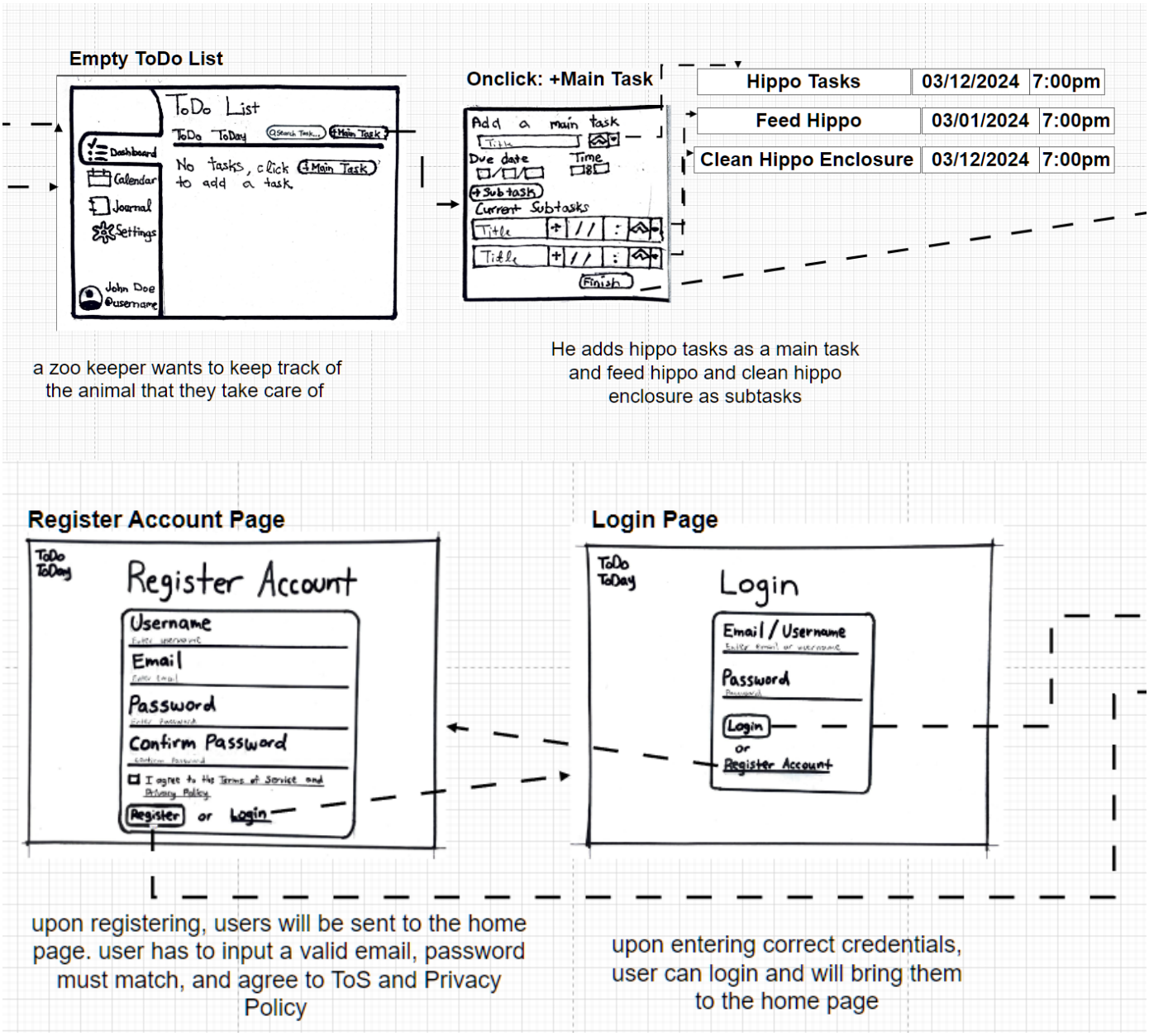


Now, the hobbyist finishes his bookshelf staining. He marks it complete, making assembling the shelf 100% complete as well as the whole task.



The hobbyist has finally finished his bookshelf project and decides to use the journal to reflect on what he struggled with and how he can improve for future projects

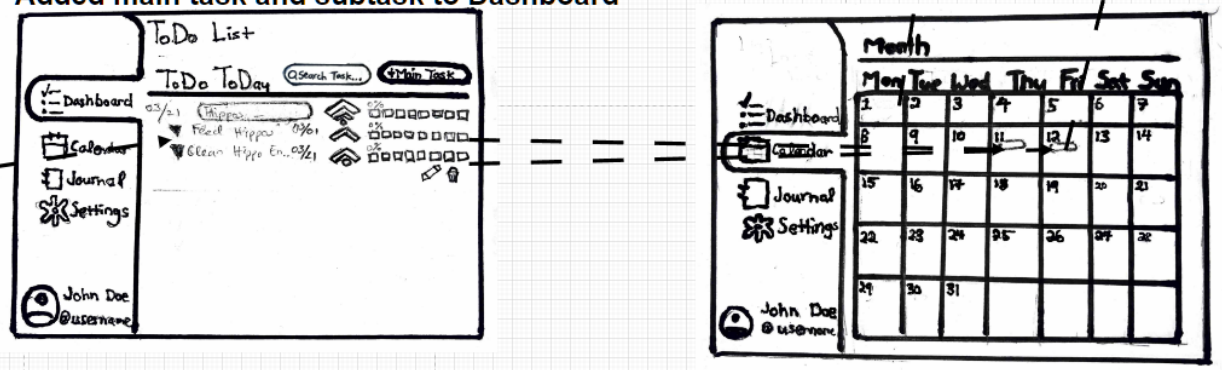
Use Case 8:


Empty ToDo List
ToDo List
ToDo Today
+Main Task
No tasks, click +Main Task to add a task.
Dashboard
Calendar
Journal
Settings
John Doe
@username
Onclick: +Main task
Add a main task
Title
Due date
Time
+Subtask
Current Subtasks
Title
Finish
Hippo Tasks
03/12/2024
7:00pm
Feed Hippo
03/01/2024
7:00pm
Clean Hippo Enclosure
03/12/2024
7:00pm
a zoo keeper wants to keep track of the animal that they take care of
He adds hippo tasks as a main task and feed hippo and clean hippo enclosure as subtasks
Register Account Page
ToDo Today
Register Account
Username
Email
Password
Confirm Password
I agree to the Terms of Service and Privacy Policy
Register
or
Login
Login Page
ToDo Today
Login
Email / Username
Password
Login
or
Register Account
upon registering, users will be sent to the home page. user has to input a valid email, password must match, and agree to ToS and Privacy Policy
upon entering correct credentials, user can login and will bring them to the home page

Feed hippos

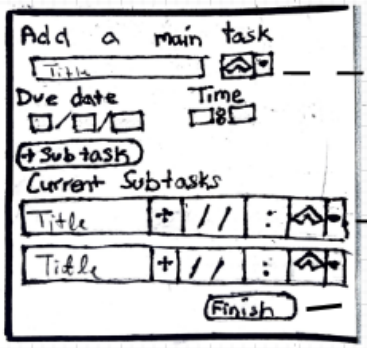
clean hippo enclosure

Added main task and subtask to Dashboard



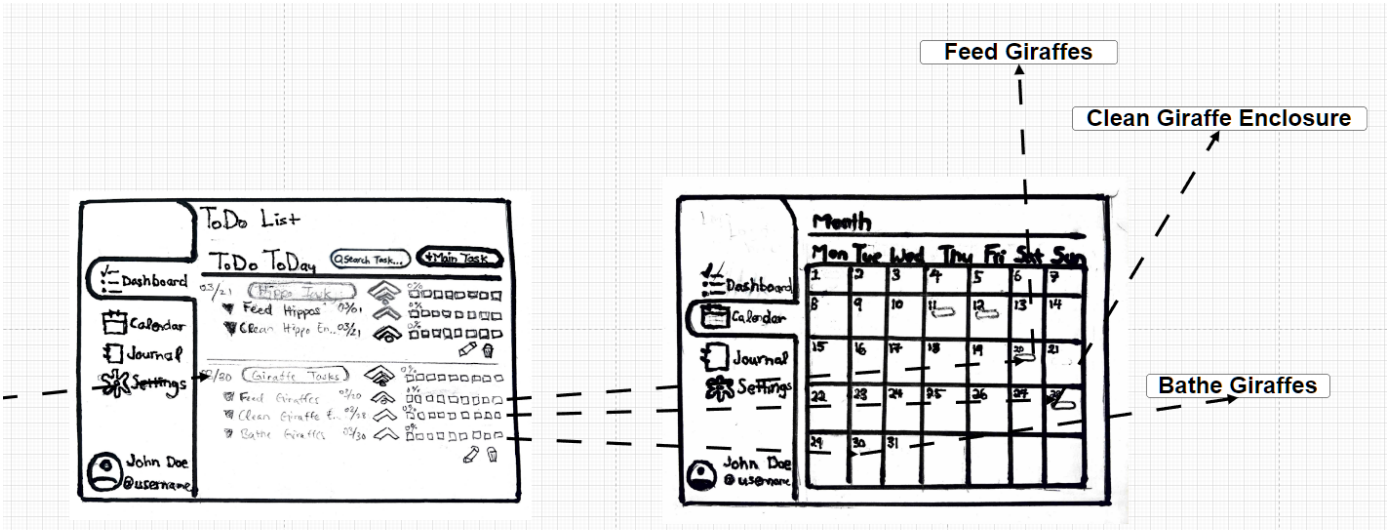
The zookeeper can now look the the calendar to look at future tasks

Onclick: +Main Task



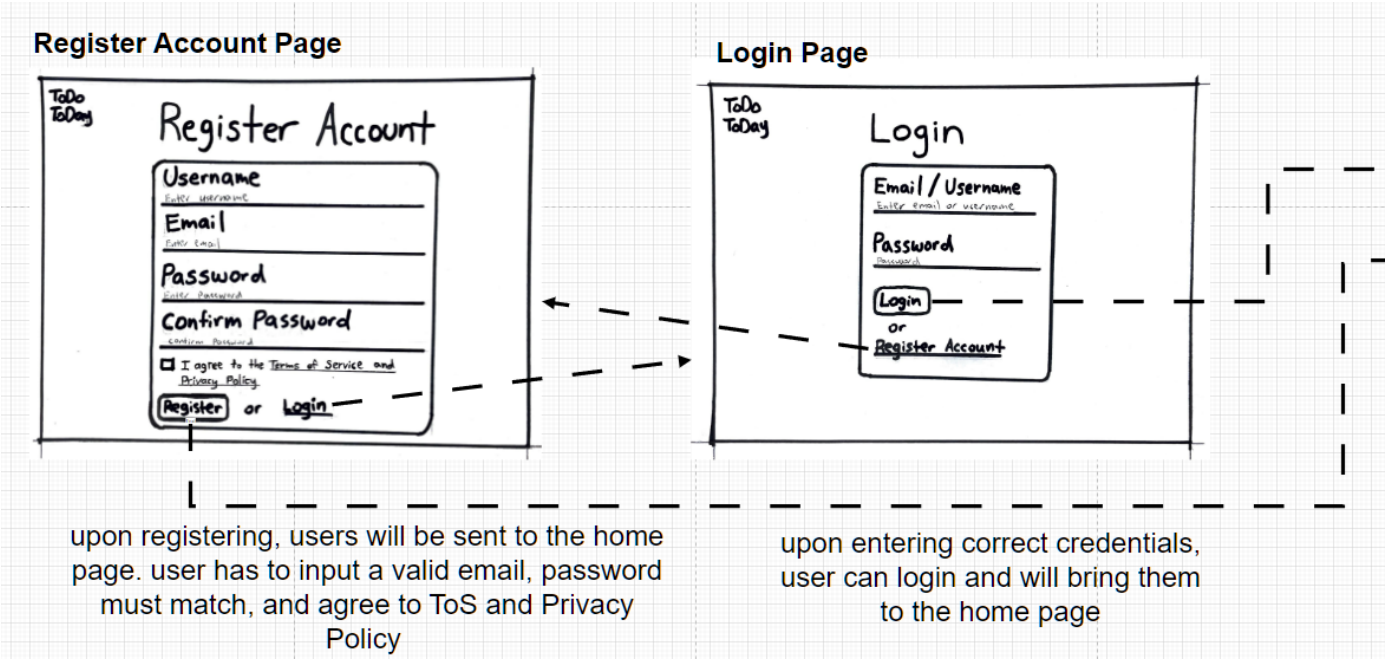
Giraffe Tasks	03/30/2024	7:00pm
Feed Giraffe	03/20/2024	7:00pm
Clean Giraffe Enclosure	03/28/2024	7:00pm
Bathe Giraffe	03/30/2024	7:00pm

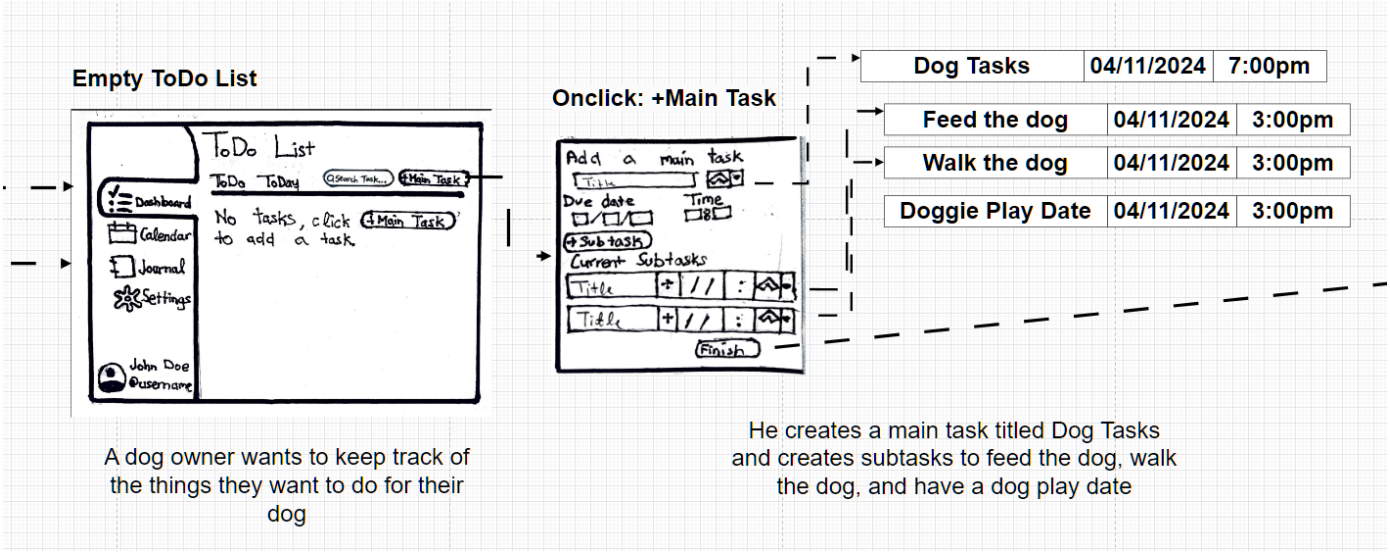
He adds another main task, Giraffe Tasks, and adds subtasks: feed giraffe, clean giraffe enclosure, and bathe giraffe

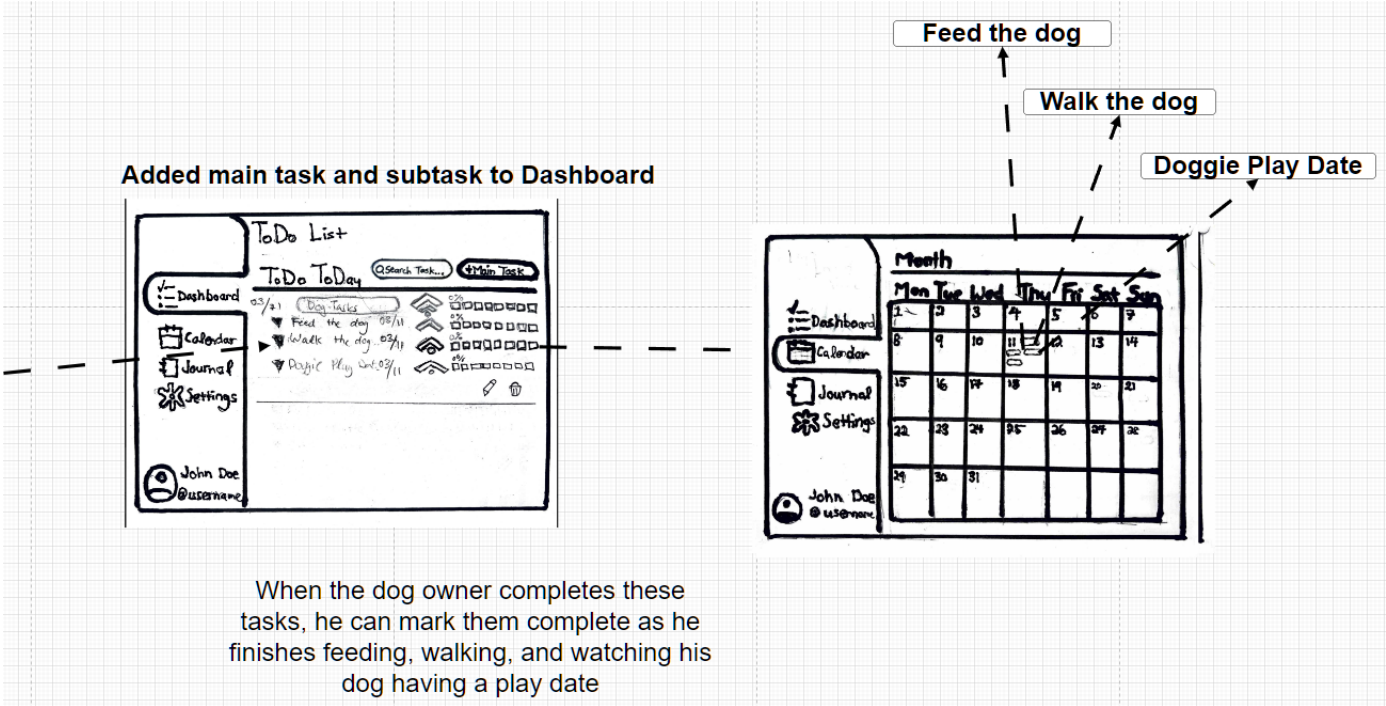


All main tasks can be seen in the dashboard and the zookeeper now has a better way to keep track of all his tasks for the animals

Use Case 9:


Register Account Page
ToDo ToDay
Register Account
Username
Email
Password
Confirm Password
I agree to the Terms of Service and Privacy Policy
Register or Login
Login Page
ToDo ToDay
Login
Email / Username
Password
Login
or
Register Account
upon registering, users will be sent to the home page. user has to input a valid email, password must match, and agree to ToS and Privacy Policy
upon entering correct credentials, user can login and will bring them to the home page

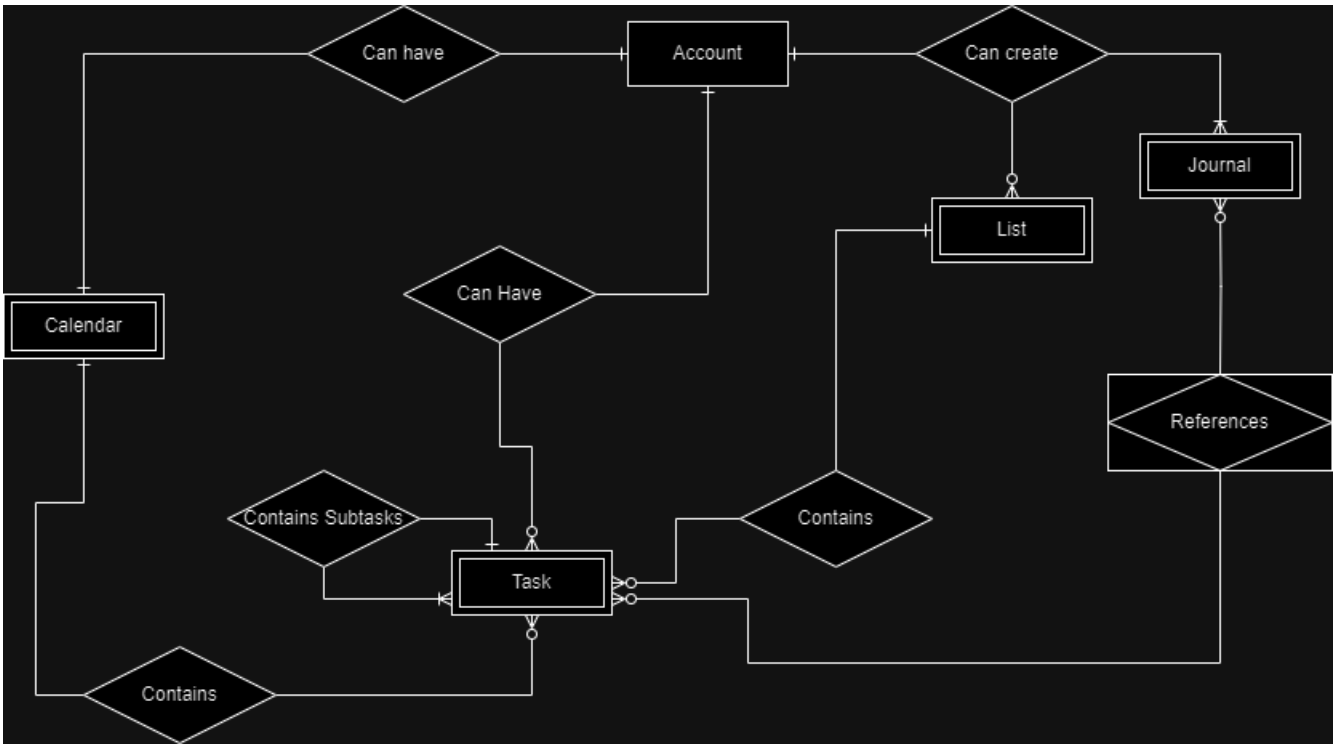

Empty ToDo List
ToDo List
ToDo Today
+Main Task
Dashboard
Calendar
Journal
Settings
John Doe
@username
No tasks, click +Main Task to add a task
Onclick: +Main Task
Add a main task
Title
Due date
Time
+Subtask
Current Subtasks
Title
Title
Finish
Dog Tasks 04/11/2024 7:00pm
Feed the dog 04/11/2024 3:00pm
Walk the dog 04/11/2024 3:00pm
Doggie Play Date 04/11/2024 3:00pm
A dog owner wants to keep track of the things they want to do for their dog
He creates a main task titled Dog Tasks and creates subtasks to feed the dog, walk the dog, and have a dog play date



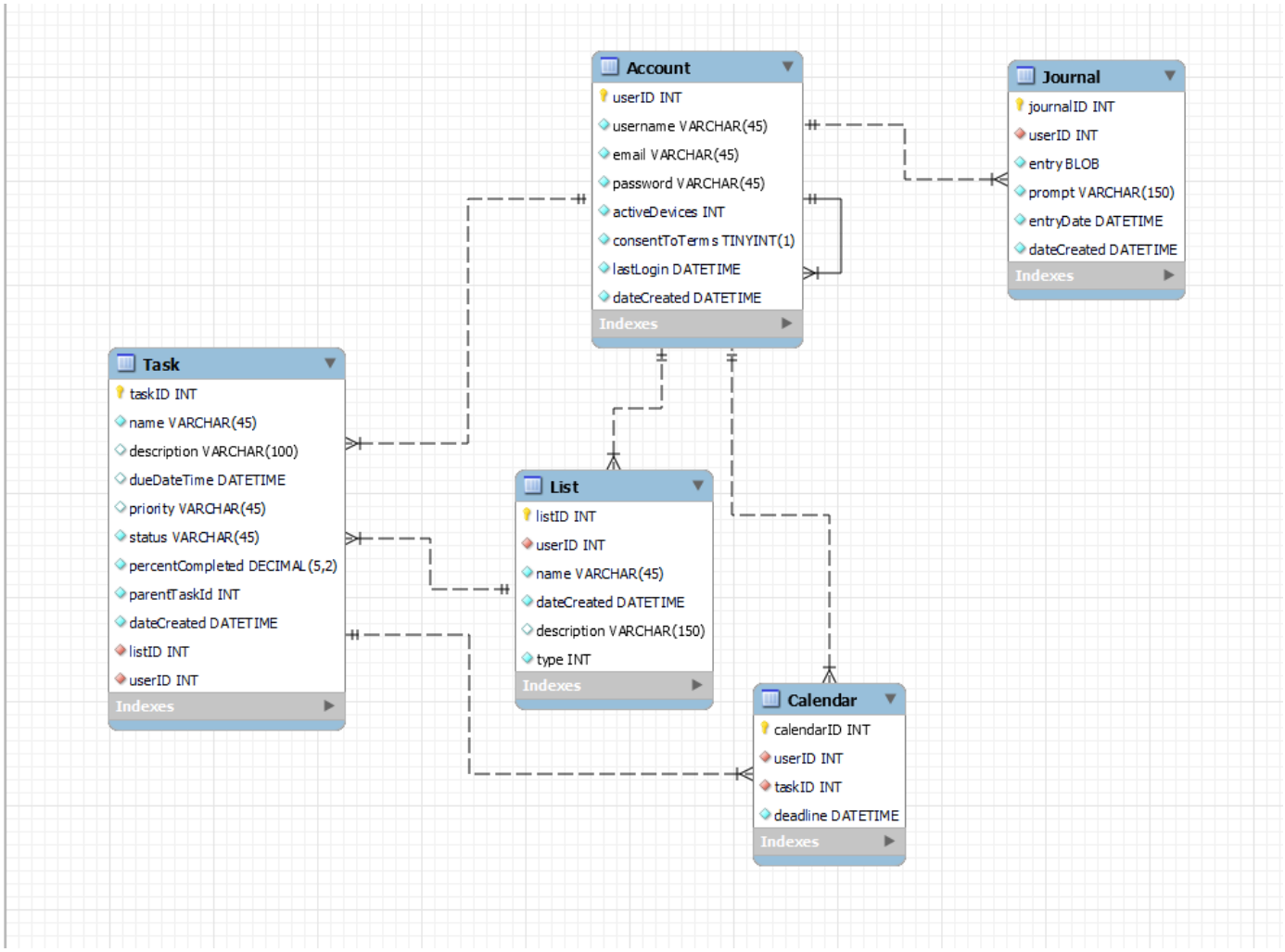
IV. High-level database architecture and organization

MySQL is a great tool to use due to its ability to efficiently handle the relational structure needed for organizing entities, such as tasks, users, lists, and other entities. It's also easy to use, as well as having the ability to convert ERDs to EERs easily and can export the actual SQL database

ERD:


Can have
Account
Can create
Journal
List
Can Have
Calendar
References
Contains Subtasks
Contains
Task
Contains

EER:


Account
userID INT
username VARCHAR(45)
email VARCHAR(45)
password VARCHAR(45)
activeDevices INT
consentToTerms TINYINT(1)
lastLogin DATETIME
dateCreated DATETIME
Indexes
Journal
journalID INT
userID INT
entry BLOB
prompt VARCHAR(150)
entryDate DATETIME
dateCreated DATETIME
Indexes
Task
taskID INT
name VARCHAR(45)
description VARCHAR(100)
dueDateTime DATETIME
priority VARCHAR(45)
status VARCHAR(45)
percentCompleted DECIMAL(5,2)
parentTaskId INT
dateCreated DATETIME
listID INT
userID INT
Indexes
List
listID INT
userID INT
name VARCHAR(45)
dateCreated DATETIME
description VARCHAR(150)
type INT
Indexes
Calendar
calendarID INT
userID INT
taskID INT
deadline DATETIME
Indexes

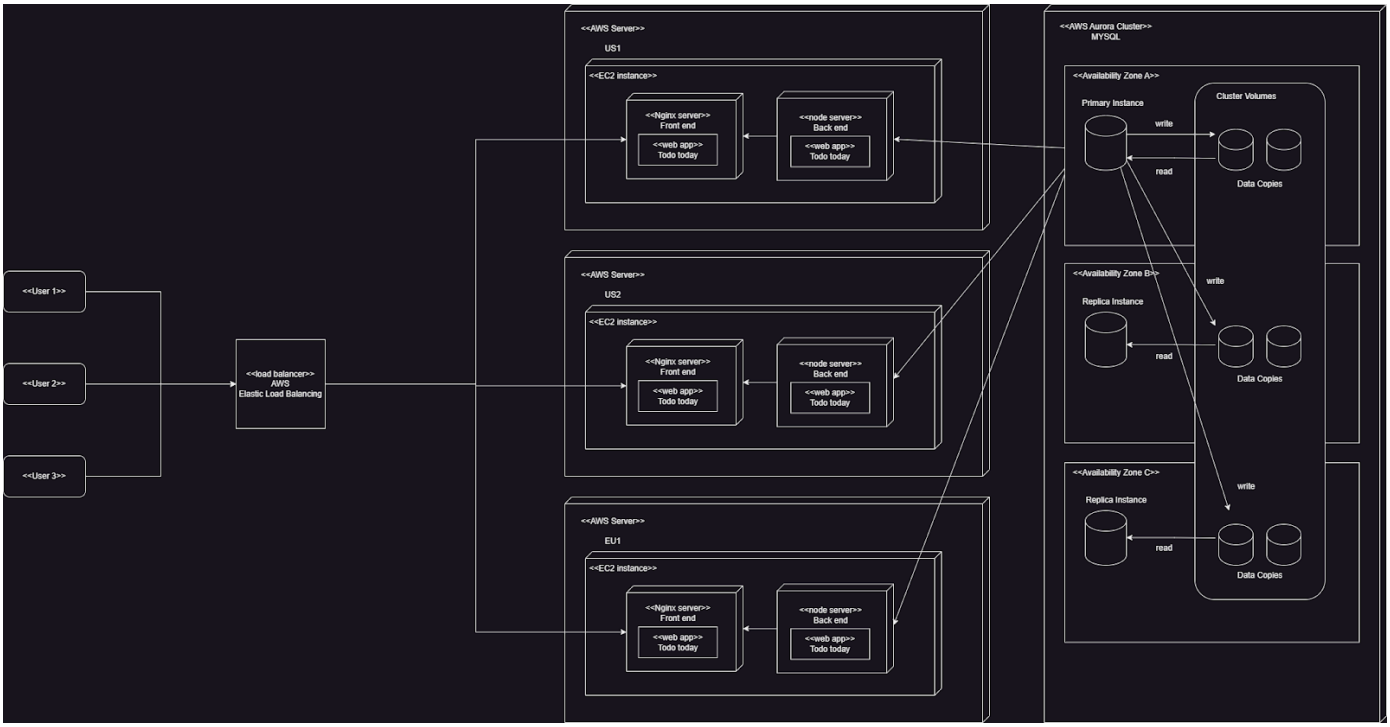
V. High-Level APIs and Main Algorithms

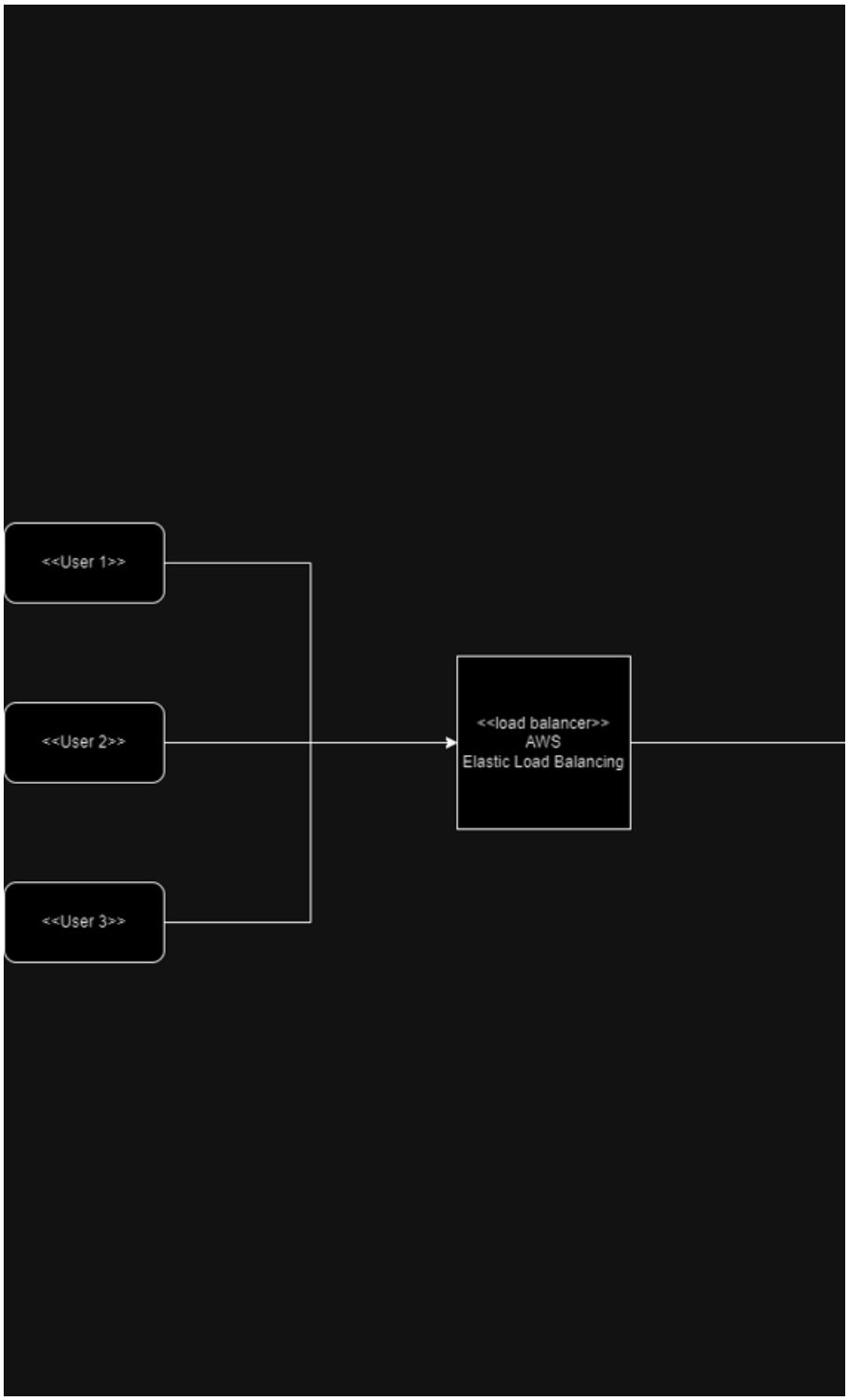
5. High-Level APIs and Main Algorithms

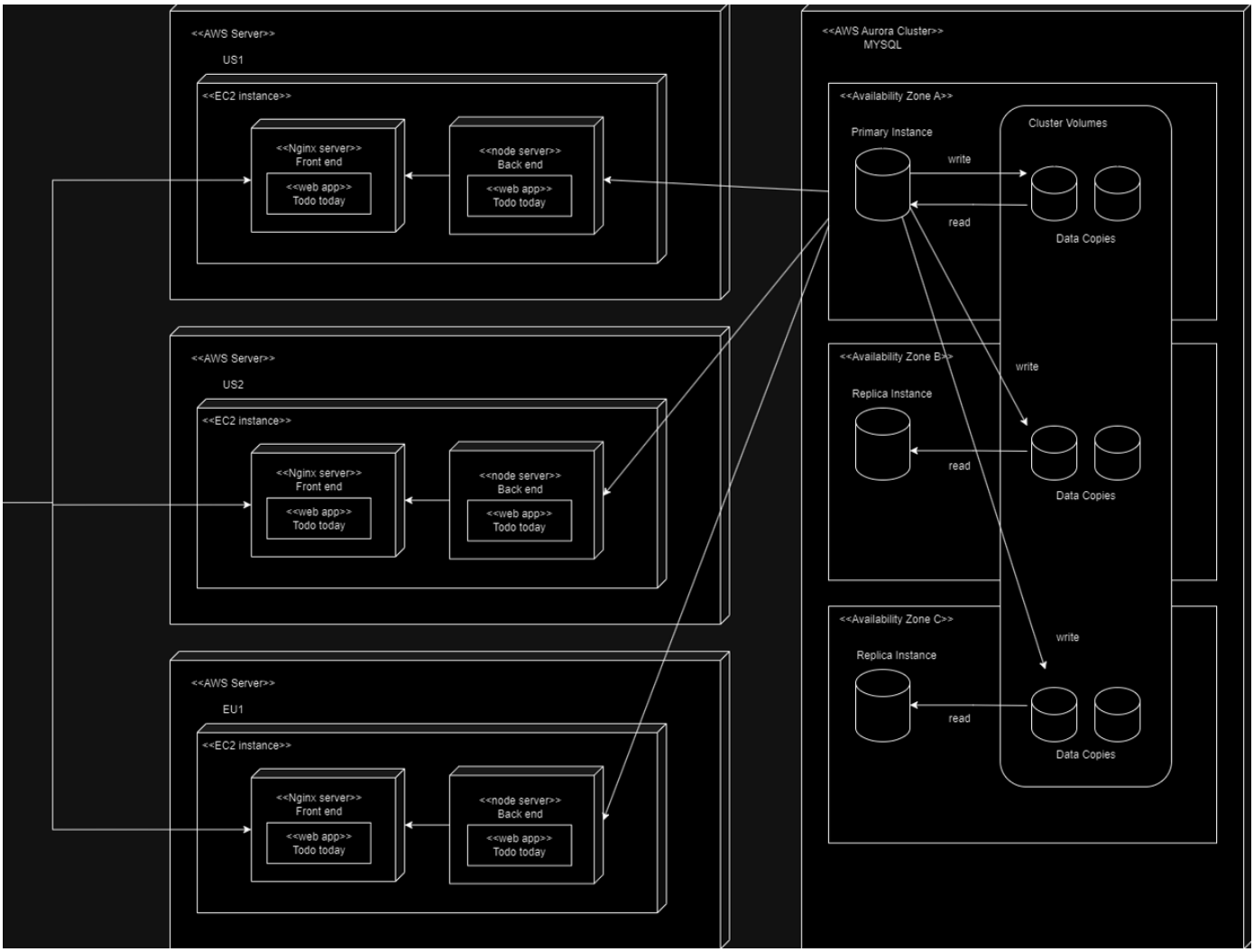
- Priority sorting algorithm
- Sorts subtasks based on priority but only for layer. Each set of subtasks is sorted independent of subtasks under a different task.
- Tree search algorithm
- Searches the tasks and subtasks in the tree structure of our todo lists
- GPT2 API(HuggingFace)
- Calls to the GPT2 API to generate text for our prompts
- List Search
- Basic search of both the name and description

VI. System Design

Scalability Diagram (Zoomed in sections below):




<<User 1>>
<<User 2>>
<<User 3>>
<<load balancer>>
AWS
Elastic Load Balancing



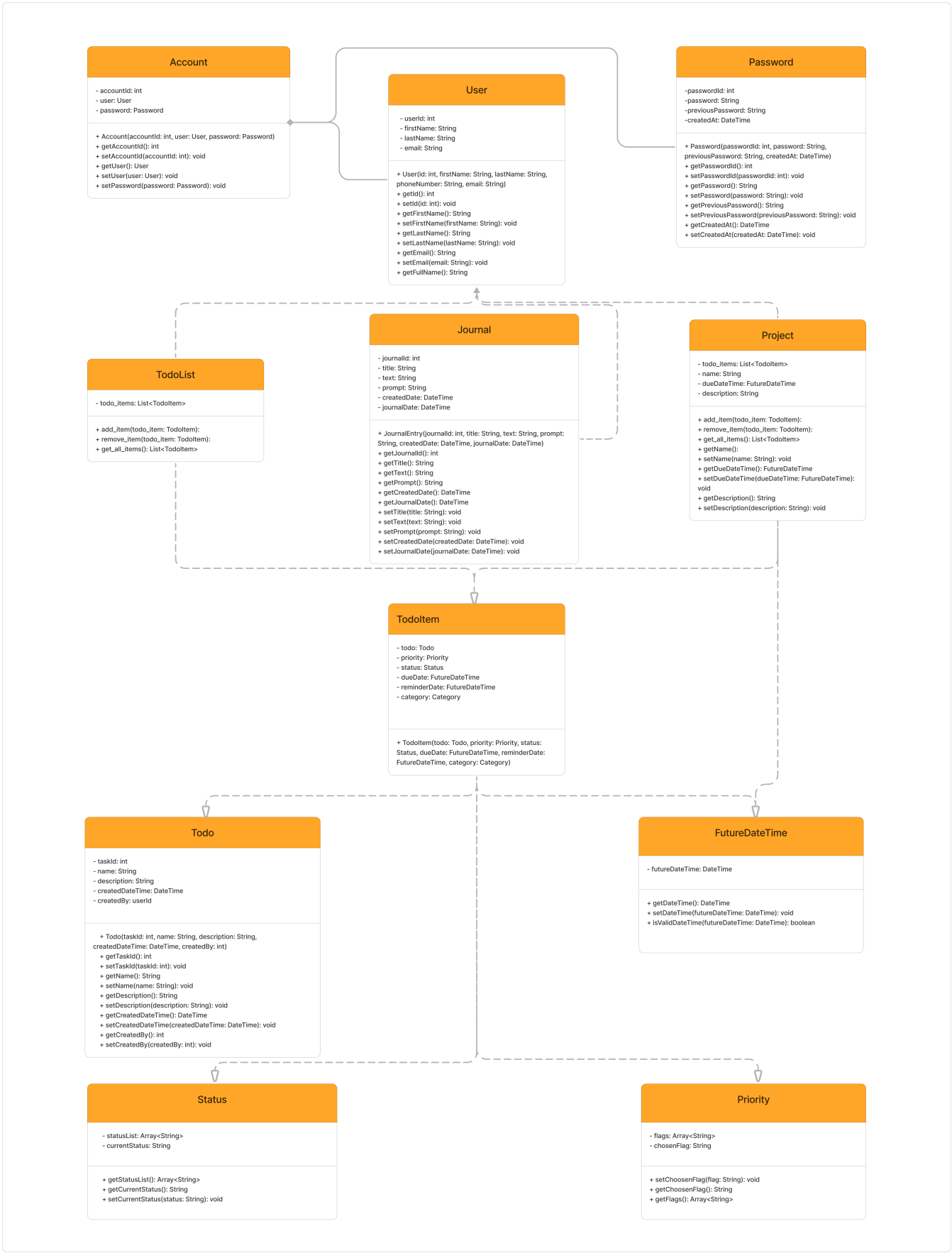
Scalability Diagram Summary:

Starting at the user end on the left side of the diagram the first microservice we're using is Amazon Web Services elastic load balancing to manage incoming traffic for the instances of our system. Beyond the load balancing we have 3 instances of our application one on the East Coast, one on the West Coast, and one in Europe, the reasoning being that no critical failure could have happened in all three locations simultaneously, which makes the entire system much more fault tolerant allowing for an entire critical failure in one of those regions without interruption to service. The next microservice we would use would be Amazon Web Services Aurora data cluster system specifically with the MySQL compatibility. This allows us to keep our same database structure but replicate it over multiple instances with a clustered volume backup. In the version that is in the diagram, we have one primary instance with two replica instances ready and immediately waiting to take over if there is a critical failure, along with 6 supplemental copies of the data. This gives us an incredible level of fault tolerance and error correction as if something goes wrong or something gets overwritten Aurora is designed to be robust and have safeties in place that will help us recover in the case of accidental or intentional data corruption or deletion. Another advantage of using Amazon Web Services systems for our upgrade path is that it keeps everything in one ecosystem. This unification of the ecosystem means that all of our systems are in one place

which is convenient for maintenance and upkeep and also is a potential security vulnerability depending on the security for our login for admin. But that is the only major drawback is that since we are using only one system if Amazon Web Services goes down so does our entire application we would have to recover what we can and then try to transfer to the Azure cloud or some other system but we would lose significant time as everything is integrated into amazons web ecosystem. The main advantage to keeping it all within the AWS framework is that this basic structure is scalable so if we need more space and if or if we need to increase our throughput we can add new instances and increase the redundancy and distribution for our data cluster very easily with just a couple clicks. If we were using a disparate set of components from various companies it would be much harder for us to scale up and down depending on usage. If the application becomes extremely popular we might need to migrate out of Amazon Web Services into a more proprietary system and that will take a lot of research and development time but for the current scale and the current scope that we are looking at this should be viable even as a phase two for development of our product.

UML Class diagram:

UML Diagram ToDoToday



Summary

System design patterns are categorized into three types: Creational, structural, and Behavioral patterns. We explored different design patterns, including the Factory Method and Singleton patterns.

While suitable for creating objects of related types without specifying their concrete classes, the Factory Method pattern did not provide the level of flexibility and customization we needed for configuring `Account` and `TodoItem` instances. Additionally, the Singleton pattern, which ensures a class has only one instance and provides a global point of access to it, did not apply to our scenario. The reason is, that it would limit our ability to create multiple `TodoItem` instances with different configurations.

Other patterns within Structural Patterns and Behavioral Patterns seemed the next step for the scalability. So, after evaluating these patterns and their suitability for our application's architecture, we determined that the Builder pattern offered the best fit. Its ability to encapsulate the construction process and provide a flexible way for creating complex objects made it an ideal choice for constructing `TodoItem` instances with varying configurations. By adopting the Builder pattern, we ensure that our application remains scalable, maintainable, and adaptable to future changes and requirements.

How are we using this pattern in our System?

Here is how it is being used in the above UML Diagram:

1. **Building `Account` with `Password` and `User` Class**:
   In our system architecture, the `Account` entity comprises user credentials and personal information, which are represented by the `Password` and `User` classes, respectively. To streamline the creation of `Account` instances with varying configurations and to make each class responsible for small tasks, we employed the Builder pattern.

   The `Account` class serves as the primary builder for `Account` objects. It uses the construction process by utilizing methods to set the properties of the `Password` and `User` components directly within the `Account` class.

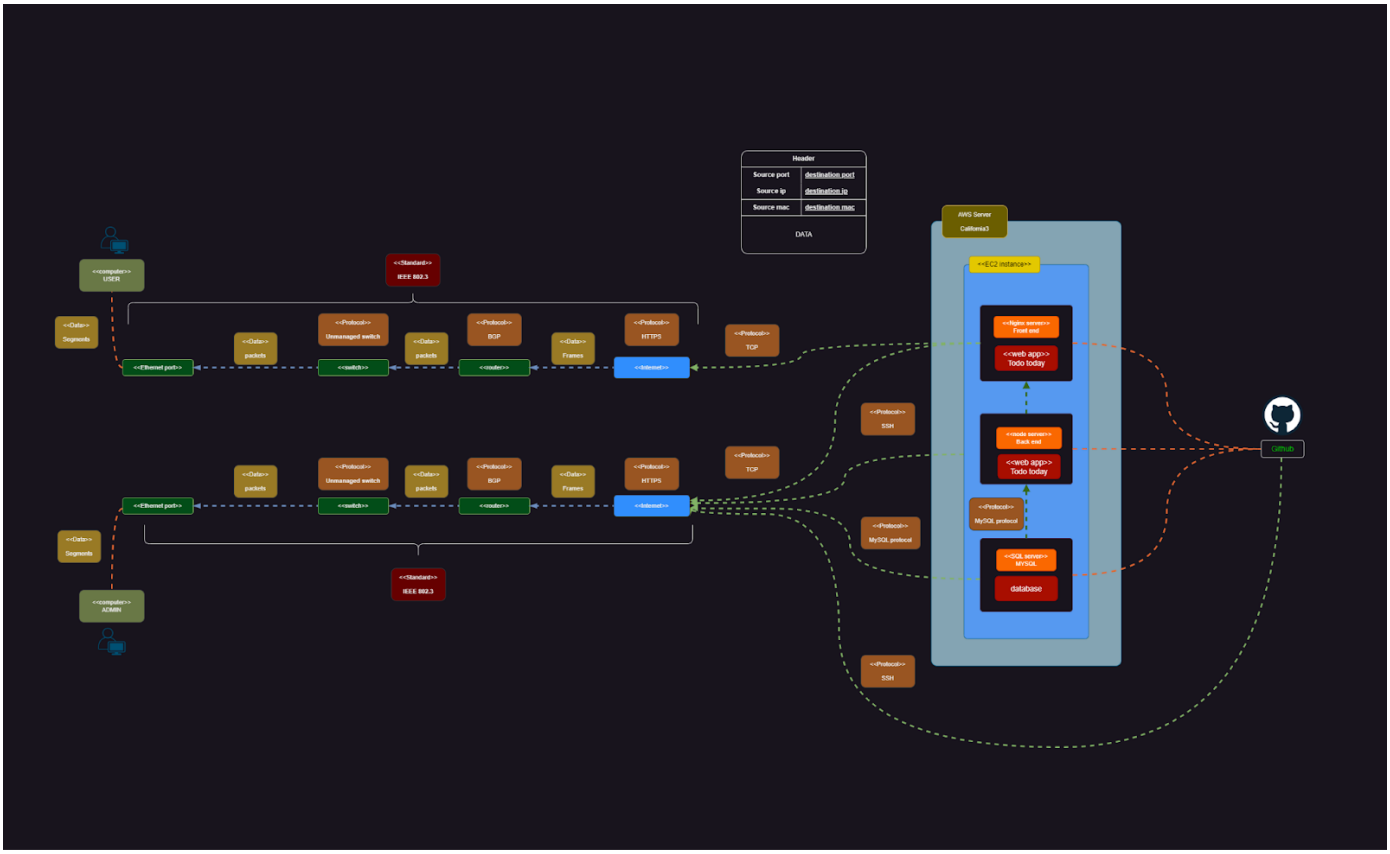
   By utilizing the Builder pattern in this context, we ensure that the creation of `Account` objects remains flexible and maintainable. The encapsulation of construction logic within the `Account` class and dividing the responsibilities across multiple other classes simplifies the instantiation process and promotes code readability.
2. **Building `TodoItem` with Different Configurations**:
   Within our task management module, the `TodoItem` class represents individual tasks with various attributes such as priority, status, and due date. To facilitate the creation of `TodoItem` instances with customizable configurations, we adopt the Builder pattern.

The `TodoItem` class itself acts as the builder for `TodoItem` objects. It provides methods for setting the attributes of the `TodoItem` with `Todo`, `Priority`, `Status`, and `Due Date`, allowing clients to specify the desired configuration through method calls.

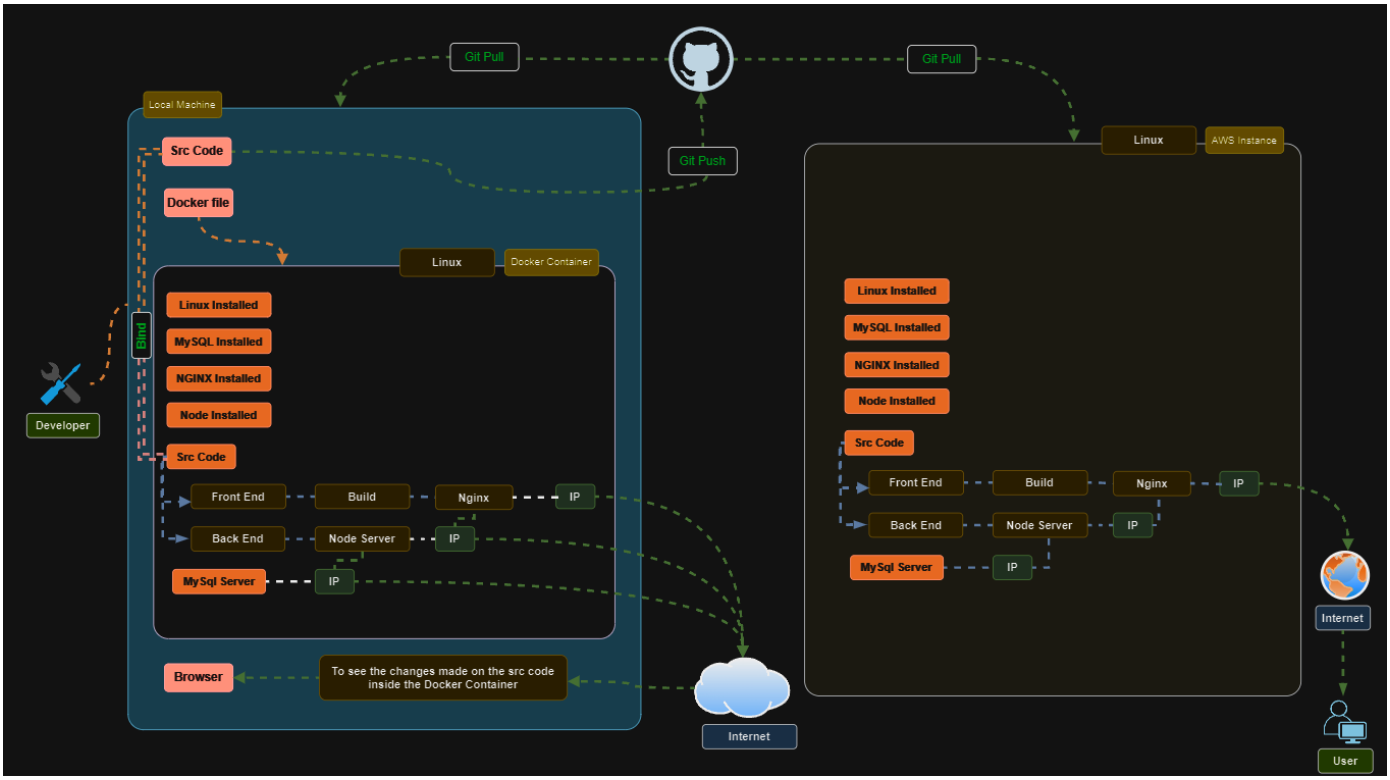
By leveraging the Builder pattern for `TodoItem` creation, we ensure that the construction process remains clear and adaptable. The `TodoItem` class encapsulates the logic for building `TodoItem` objects. It enables clients to construct instances with ease while maintaining code's flexibility and maintainability.

VII. High-Level Application Network and Deployment Design

Network Diagram:



Deployment Design:



VII. Identify actual key risks for your project at this time

Identify only actual and specific risks in your current work such as (list those that apply:

• skills risks (do you have the right skills),

• schedule risks (can you make it given what you committed and the resources),

• technical risks (any technical unknowns to solve),

• teamwork risks (any issues related to teamwork);

• legal/content risks (can you obtain content/SW you need legally with proper

licensing, copyright)

Known Risks:

1. Application Network Diagram Knowledge (skill)
2. Initial lack of AWS knowledge (technical / skill)
3. Chat GPT (technical unknown)
4. Data Leakages (security risk)

5. Lack of communication
6. Lack of knowledge regarding legal terms of condition content
7. Difficulty in splitting up tasks

IX. Project management (Team Lead)

During milestone one as the team lead I used an excel spreadsheet to manage our time and completion of tasks. It was a very manual system that I constantly was having to update as I was the only one with access to that data. Moving over to notion that allowed me to share the tasks between all of the teammates made it much easier for us to understand exactly where we were at and who had finished what and what still needed to be done. It also allowed us to define subtasks and build out a solid structure for the step by step sub steps that we need to do to complete the larger pieces that are needed for completion of our milestone 2. It also allowed us to divide and conquer in a much more effective way than me just sort of handing out assignments directly I was able to hand out high level assignments saying OK you're in charge of this task you're in charge of that task and then you tell us what we need to do to help you. That sort of subdivision of Labor has significantly improved both the quality and speed of our work.

X. Detailed list of contributions (Team Lead)

	Alex Nikols	Jason Tran	Sulav Jung Hamal	Jadon Hoang	Randale Reyes	Christian Gopez
overall Code	1	4	5	2	1	1
overall Documentation	5	4	4	4	3	5
Data Definitions	2	1	4	1	2	4
Prioritized Functional Requirements	3	3	3	3	3	3
UI Mockups and Storyboards (high level only)	2	5	2	5	1	1
High level database architecture and organization	3	1	3	1	5	1

	Alex Nikols	Jason Tran	Sulav Jung Hamal	Jadon Hoang	Randale Reyes	Christian Gopez
High Level APIs and Main Algorithms	4	1	4	1	1	4
System Design	4	1	5	1	1	2
High Level Application Network and Deployment Design	5	1	4	1	1	2
Identify actual key risks for your project at this time	3	3	3	3	3	3
Project management	5	2	3	2	1	2
Vertical SW Prototype	2	4	5	2	2	2