Software Design Document

for

Want2Remember

Version 3.1 approved

Prepared by

Antonio Campos	Project Co-Lead
Alec Kaczmarek	Project Co-Lead
Amy Guttman	Documentation Co-Lead
Alexandra Strong	Documentation Co-Lead
Vincent Li	Engineer
Saiyang Liu	Engineer
Ricardo Marroquin	Engineer
Miguel Nonoal-Garcia	Engineer
Jonathan Sum	Engineer
Edwin Zapata Minero	Engineer
Zilong Ye	Faculty Advisor
Kevin Benavente	We2Link Liaison
Michael Malone	We2Link CEO

13 May 2022

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Amy Guttman, Alexandra Strong	08/24/2021	Initial Access	1.0
Amy Guttman, Alexandra Strong	10/26/2021	Cross-Reference with Past SDD	1.1
Amy Guttman, Alexandra Strong	12/10/2021	First Draft	2.0
Amy Guttman, Alexandra Strong	05/09/2022	Second Draft Completed	3.0
All Project Members	05/13/2022	Final Draft Approved	3.1

1. Introduction

1.1 Purpose

Want2Remember is a mobile application that shall assist those with cognitive impairments remember day-to-day tasks and memories. This system shall keep track of user-generated data, such as memories, to-do lists, and appointments, as well as contacts and payment information. This document shall explain the functionality, design, architecture, and requirements of the Want2Remember application.

1.2 Scope

The purpose of this document is to provide an overview of the design and development of this project, as well as the recommended procedures and available resources for developers and maintainers of Want2Remember. This document is not an expression of formal policy; it contains documentation for the Want2Remember system and generally agreed-upon best practices.

1.3 Intended Audience and Reading Suggestions

This document is for the use in the California State University of Los Angeles (CSULA) Computer Science department. This document will cover the details of the Senior Design Project of the We2Link sponsored group. The intended audience for this project is the professors and students of CSULA, as well as the employees of We2Link. Please see the Want2Remember Project Report and the Software Requirements Document for Want2Remember for additional information on this project.

1.4 System Overview

Want2Remember uses a component-based architecture that allows for the reusability of components so that it reduces the size and complexity of our codebase.

2. Design Considerations

2.1 Assumptions and Dependencies

Software used to support the Want2Remember application include:

- Redux
- NodeJS
- JavaScript
- React Native
- GitHub
- JIRA & Agile Development Technology
- Firebase
- Test Fairy, Test flight

Additionally, the user is expected to have a mobile device running on the minimum required operating systems (iOS 7 or above; Android Oreo (8) or above).

2.2 General Constraints

The major hurdles associated with this project were learning the JavaScript mobile application framework and React Native and working as a group remotely through the COVID-19 pandemic. We were given online resources to solve both challenges. We were given access to the Udemy course “React Native - The Practical Guide” to guide us through the framework and language. To collaborate, we formed sub-teams to break down the project objectives into manageable pieces. We met virtually twice weekly for status updates and sprint retrospectives and communicated via Slack with our sub-team members.

We added our updates on top of what had been built by previous teams. We had to implement bug fixes and additional user requirements based off the existing design. Our time with this project was limited – approximately 9 months of development with this team.

2.3 Goals and Guidelines

Want2Remember is a mobile phone application to help those with memory issues - whether they are from brain injuries, Alzheimer’s, or other cognitive impairments. The Want2Remember app shall provide the user templates to log memories, passwords, to-do lists, medications, appointments, and other important things the user may want to remember. The user shall be able to record events and reminders from the past, present, and future.

The software features shall help facilitate the user’s ability to live independently, return to work, maintain social interaction, increase work efficiency, maintain personal safety, as well as any

other needs that may come up in development. The app features shall also help facilitate caregiver needs, as well as improve medical support.

Due to the need for simplicity and reliability, we chose the respective technologies because of the support and modularization they offer. We take the feedback from our beta testers to refactor the code and make improvements. The hope is for Want2Remember to be a viable product and to go to market by Summer 2022.

2.4 Development Methods

Our dedicated team spent months developing Want2Remember, building off the efforts of last year's Senior Design project. We followed their same methodology; we used the Agile Development Process. We broke down tasks and assign them to sub-teams in Sprints of one-to-two-week durations. Agile architecture allowed us to pivot and adapt quickly to new changes, which allowed us to respond swiftly to any bug fixes or feedback from beta testers. We also visualized the task board using Atlassian's Jira software to improve workflow.

3. Architectural Strategies

3.1 Use of Particular Products

3.1.1 JavaScript and React Native: Developing with the React Native framework gives us the advantage of developing simultaneously for Android and iOS projects. The code is written in JavaScript but is then rendered with native code. React Native also puts an emphasis on creating and reusing components within many screens of the application. This helps save time during development and encourages us to take full advantage of the utilities offered by React Native such as a multi-platform codebase, native app development, and fast refresh as soon as new code is saved.

3.1.2 Redux: We use Redux for data operations.

3.1.3 GitHub: We use GitHub for version control.

3.2 Reuse of Existing Software Components

3.2.1 This software is built off last year's Senior Design project for We2Link. We are refactoring their codebase and extending functionalities.

3.3 Future Plans for Software Enhancement

Several features have been marked for future development:

- Medication tracker template
- Additional customization
- Caregiver support, multiple caregiver access
- Medication Tracker template
- Mobile advertisement
- Premium features
- Proximity support
- Invites

3.4 User Interface Paradigms

3.4.1 The user interface shall present the Home Screen upon initial access.

3.4.2 The system shall provide a uniform look and feel between all pages. Headings, banners, fonts, and buttons shall follow the same style guide.

3.4.3 All screens shall include dynamic header component and bottom navigation bar.

3.4.3.1 Bottom navigation bar shall include Home, Create, Search, and Contacts.

3.4.3.2 Back button shall be in the top left to return the user back to the Home Screen.

3.4.4 Home Screen (default screen) shall show brief overview of all existing memories.
- 3.4.4.1 Help, settings, and filters shall be in the top left corner for navigation.
- 3.4.4.2 The Memory navigation bar shall show “All Memories” by default, with other subcategory options listed in the scroll bar to the right.
- 3.4.4.3 Below this navigation bar, the preview of the memory shall show with date, title, category, and other relevant summary information.
- 3.4.4.4 A button at the bottom right shall allow the user quick access to “Create,” leading them to the Create Memory Screen.

3.4.5 Create Memory Category Screen shall show all memory templates.
- 3.4.5.1 Memory tiles shall be labeled and color-coded in an accessibility-friendly way.
- 3.4.5.2 Clicking on a memory tile shall lead the user into a custom template.
- 3.4.5.3 Clicking on Change Color Palette shall allow the user to change or edit the color palette of the memory tiles

3.4.6 Search Screen shall let users filter through their memory tiles
- 3.4.6.1 Search bar.
- 3.4.6.2 When a user searches, the scrollable results shall display below the search bar.

3.4.7 Contacts Screen shall show all contacts in scrollable view.
- 3.4.7.1 Add Contact shall be in the top right corner
- 3.4.7.2 Search bar.
- 3.4.7.3 Quick view shall include name, quick links to contact, and associated memories.
- 3.4.7.4 Clicking on View More shall allow the user to see full details of that contact.

3.4.8 Settings Screen
- 3.4.8.1 The settings options shall allow the user the ability to import and export JSON format memories.
- 3.4.8.2 The user shall be able to create and reset their secured PIN.
- 3.4.8.3 The user shall be able to send feedback.

3.5 Hardware and/or Software Interface Paradigms

3.5.1 Hardware interface paradigm
Device must have a display (may or may not be touch display) for user input /interaction.

3.6 Error Detection and Recovery

3.6.1 Implement exception handling in the codes

3.6.2 Debugging the software interfaces/programs

3.6.3 To recover the issue, discussion shall be made within the group.

3.7 Memory Management Policy

3.7.1 Memory management of the software shall be managed by the framework

3.8 External Databases

Cloud Firestore is a cloud-hosted, NoSQL database. Following Cloud Firestore's NoSQL data model, you store data in documents that contain fields mapping to values. These documents are stored in collections, which are containers for your documents that you can use to organize your data and build queries.

3.9 Distributed Data/ Control Over a Network

N/A

3.10 Generalized Approaches to Control

N/A

3.11 Concurrences and Synchronization

N/A

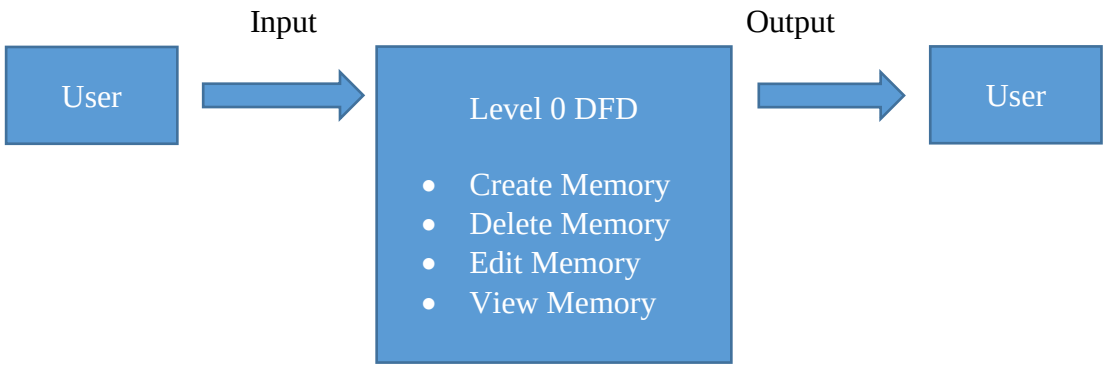
3.12 Communication Mechanisms

3.12.1 The software shall allow the user to contact the administrators through Test Fairy for Android and Test flight for iOS.

4. System Architecture

4.1 Overview

Figure 1: The Application



- The User: This section represents the individual using the application, i.e., the user. Only the user can provide input for the mobile application.
- The Application: React Native encapsulates all Screens and Components which the user interacts with.

4.2 Data Flow

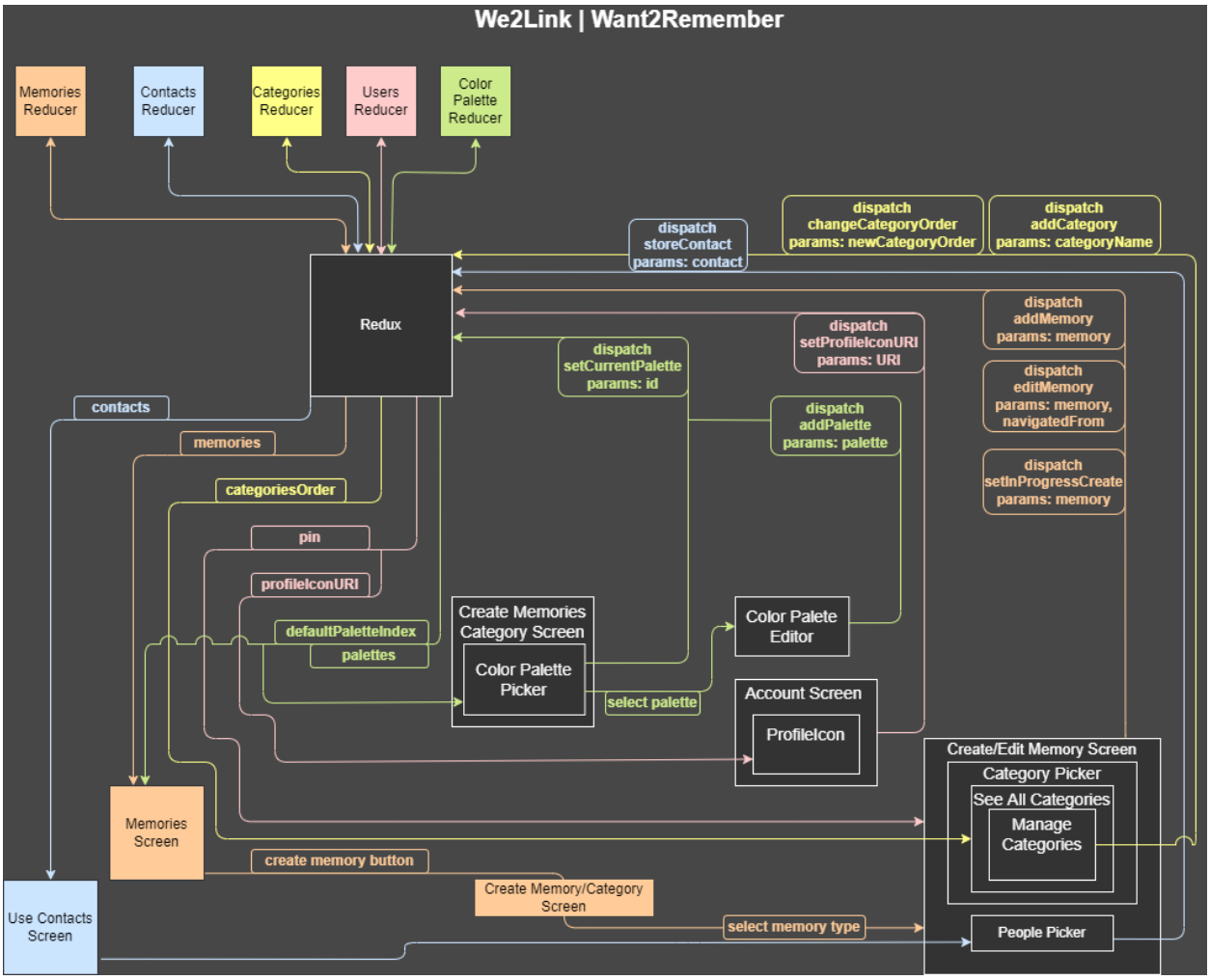


Figure 2: Redux Flow Diagram

Want2Remember mobile application consists of eight major screens:

4.2.1 HomeScreen

The HomeScreen provides the initial landing site for users. It provides a brief overview of all existing memories. It connects to the other screens below, allowing the user to navigate to other sub-screens.

4.2.2 CreateMemoriesCategoryScreen

The CreateMemoriesCategoryScreen allows the user to choose from different entry templates. Each memory template layout is unique based off the needs of that memory type.

4.2.3 MoreDetailsScreen

The MoreDetailsScreen displays all available metadata associated with a selected memory.

4.2.4 ContactsScreen

The ContactsScreen synchronizes Want2Remember's contact address book with the user's native address book. All contacts are displayed on this screen in the form of ContactTiles. ContactTiles show the contact's first name, last name, and memories the contact is associated with. The user can directly call, message, and/or email the chosen contact from this tile.

4.2.5 SearchScreen

The SearchScreen allows users to search through entries and display results.

4.2.6 SettingsScreen

The SettingsScreen allows user to change application settings, import or export app data. The user may also create and edit app reminders, customize the app, edit their secure pin, send feedback to developers (help screen), or clear all user data.

4.2.7 HelpScreen

The HelpScreen allows user to send feedback and report a bug.

4.2.8 FiltersScreen

The FiltersScreen has two tabs labeled "Memories" and "Date". The "Memories" tab can filter memories by memory type, while the "Date" tab can filter user memories by a specific date range. This date range can be a preset value, or a value selected by user input through the in-app calendar.

4.3 Software Development and Implementation

Because this application is built off the last Senior Design project, our first task was to understand their existing model and UI design. We started by refactoring the code and fixing bugs, followed by updates and new features based off user feedback. Some of these include but are not limited to Calendar support, cloud infrastructure design, new memory templates, and user customization.

5. Policies and Tactics

5.1 Specific Products Used

- Integrated Development Environment (IDE): Visual Studio Code
- Project Management Software: Atlassian JIRA board
- Communication: Slack
- Version Control: GitHub
- Mobile Application Framework: React Native
- Application State Management: Redux
- User Authentication: Google Cloud User Authentication
- Cloud Support: Google Firebase
- Android Emulator: Android Studio
- IOS Simulator: XCode

5.2 Requirements Traceability

GitHub allows us to have version control. It enables our developers to branch off and collaborate, which is especially useful in a virtual environment.

5.3 Testing the Software

We are currently beta testing this application. We use a combination of Test Fairy for Android and Test flight for iOS to gather feedback from the beta users. The issues generated from user feedback is pushed onto our JIRA board for further examination and response.

5.3 Maintaining the Software

This software shall be available on mobile and tablet devices for iOS and Android OS. A separate team is working on the Want2Remember website; a future team may continue to work on this application. Current We2Link developers shall keep this application maintained regularly, responding to help tickets and user feedback as well as fixing bugs. All software shall be optimized for older devices, as well as offline first functionality.

6. Detailed System Design

6.1 MemoriesScreen.js

6.1.1 Responsibilities

Memory landing page. Users shall see an overview of all the memories they have created on this application. The home screen is responsible for the most navigation and the most detailed user interface. This is the user's first impression of the application and must be simple, have good flow, and be intuitive to use.

6.1.2 Constraints

Memories must be created before they can be displayed, filtered, searched, or edited.

6.1.3 Composition

This component allows us to view and access every memory created by the user. This component also allows us to navigate to the Create Screen, Feedback Screen, Filters Screen, Settings Screen and Contacts Screen

6.1.4 Uses/Interactions

This component allows us to access each memory. This component also allows us to navigate to the Create Screen, Feedback Screen, Filters Screen, Settings Screen and Contacts Screen

6.1.5 Resources

Dependencies such as React, React Native Dimensions, React Native.

6.1.6 Interface/Exports

Views - Help organize the order and flow of the content within this specific screen.

Buttons - Help users navigate to other screens within the App.

6.2 CreateScreen.js

6.2.1 Responsibilities

This screen shall enable the user to enter all metadata associated to the memory type. After the user enters all the necessary data needed for a particular memory type, the user shall then be able to press a "Create" button to create an instance of that specific memory type.

6.2.2 Constraints

The title field shall be filled in before the user can create the memory. The user shall see different input fields depending on the type of memory.

6.2.3 Composition

This screen contains the input fields necessary to create a memory, which rely on the following subcomponents: CategoriesPicker, TagsPicker, TimePicker, EmojiPicker, FilePicker, DatePicker, StatusPicker, PeoplePicker, LocationPicker, and TitleInput.

6.2.4 Uses/Interactions

This component allows the user to select which type of memory they would like to create. From there, they can create the memory of their choice. Once the memory is created, a Quick Look of it shall be displayed on the Memories Screen.

6.2.5 Resources

Dependencies such as React, React Native Dimensions, React Native.

6.2.6 Interface/Exports

Views - Help organize the order and flow of the content within this specific screen.

Buttons - Help users navigate to other screens within the App.

6.3 MoreDetailScreen.js

6.3.1 Responsibilities

After a memory is created, the data inserted in that memory such as the title, dates, and tags shall be displayed. This screen shall display all the metadata associated with the selected existing memory. The MoreDetailsScreen shall also allow the user to edit or delete the selected memory.

6.3.2 Constraints

The memory shall, at minimum, have a Title associated with it for the MoreDetailsScreen to render.

6.3.3 Composition

This screen displays all metadata associated with a memory via the following subcomponents: TextDetail, WebsiteDetail, ContactsDetail, PeopleDetail, TagsDetail, CustomDetail, StatusDetail, LinkDetail, EmojiDetail.

6.3.4 Uses/Interactions

This component allows the user to edit and delete a selected memory.

6.3.5 Resources

Dependencies such as React, React Native Dimensions, React Native.

6.3.6 Interface/Exports

Views - Help organize the order and flow of the content within this specific screen.

Buttons - Help users navigate to other screens within the App.

6.4 ContactsScreen.js

6.4.1 Responsibilities

This screen displays all the contacts stored inside the local device. This screen also allows the user to input Contacts to be used in the App. These custom contacts shall also be imported to the native Contacts on the device.

6.4.2 Constraints

The ContactsScreen is reliant on being synchronized with the phone’s local contacts data. If permissions for accessing the phone's local contacts, Redux shall leave empty placeholders instead.

6.4.3 Composition

N/A

6.4.4 Uses/Interactions

ContactsScreen interacts with components that have a ‘People’ input field. Users select people from the list of contacts.

6.4.5 Resources

Dependencies such as React, React Native Dimensions, React Native.

6.4.6 Interface/Exports

Views - Help organize the order and flow of the content within this specific screen.

Buttons - Help users navigate to other screens within the App.

6.5 ContactDetailsScreen.js

6.5.1 Responsibilities

This screen displays all details of the selected contact.

6.5.2 Constraints

The ContactsDetailsScreen is reliant on synchronization with the device’s local contacts data. If permissions for accessing the phone’s local contacts are denied, Redux shall leave empty placeholders instead.

6.5.3 Composition

This component shows detailed information about contacts, such as phone number, address, full name, etc. It shall also display memory QuickLooks that the contact has been tagged in.

6.5.4 Uses/Interactions

ContactsDetailsScreen interacts directly with memory QuickLooks.

6.5.5 Resources

Dependencies such as React, React Native Dimensions, React Native.

6.5.6 Interface/Exports

Views - Help organize the order and flow of the content within this specific screen.

Buttons - Help users navigate to other screens within the App.

6.6 AccountScreen.js

6.6.1 Responsibilities

This screen shall provide users access to import/export capabilities, app reminder settings, customization features, secure pin creation, and feedback. Users can also log out from the AccountScreen.

6.6.2 Constraints

N/A

6.6.3 Composition

Import/Export are both separate components that go to their respective screens where the user can import or export their data. App reminder lets the user create their own reminders to use the app. Customization shall let the user customize multiple aspects of the App display. The pin component sends the user to a screen where they can change or create pins for the “Secured” tag. Feedback leads to a form where the user can submit feedback through text, images, or video and can include their emails so we can contact them regarding their feedback. The Logout button shall log the user out of the app.

6.6.4 Uses/Interactions

The pin creating/editing is the most directly interactive component between this component and the rest of the app. The pin is used for anything that has the “Secured” tag. The Import component is also very interactive with the rest of the app because imported data shall overwrite whatever local data is currently stored.

6.6.5 Resources

Dependencies such as React, React Native Dimensions, React Native.

6.6.6 Interface/Exports

Views - Help organize the order and flow of the content within this specific screen.

Buttons - Help users navigate to other screens within the App.

7. Detailed Lower-Level Component Design

7.1 QuickLookComponent

7.1.1 Classification

A small window displaying details of most recent memories.

7.1.2 Processing Narrative (PSPEC)

The order of the memories is a stack-based order. It can be filtered by typing, tags, and by containing select(ed) string(s). If the user selects on the QuickLook it transfers them to the moreDetails page of that memory.

7.1.3 Interface Description

A small window design that has select data from the memory, can be filtered by select fields.

7.1.4 Processing Detail

Grabs data from the selected memory, then sorts memories in stack order. If selected, it transfers the user to moreDetails page of that memory.

7.1.4.1 Design Class Hierarchy

Sorts memories by most new to most old.

7.1.4.2 Restrictions/Limitations

It loads a fixed select of memories. If the user scrolls down, more memories shall load.

7.1.4.3 Performance Issues

Constant when load on start, since it loads a fixed number of memories. If the user scrolls all the way to the final memory, it is linear loading.

7.1.4.4 Design Constraints

N/A.

7.1.4.5 Processing Detail For Each Operation

Refer to 7.1.4.3

7.2 Detail.js

7.2.1 Classification

The component displays different types of details in the details screen like TextDetail, ContactsDetail, PeopleDetail, LinkDetail, TagsDetail, EmojiDetail, CustomDetail, StatusDetail, and TaskDetail.

7.2.2 Processing Narrative (PSPEC)

The component receives information from props and displays it.

7.2.3 Interface Description

Displays items, ranging from plain test to image file.

7.3 ContactQuicklook.js

7.3.1 Classification

Display contact information such as name, tags, and associated memories.

7.3.2 Processing Narrative (PSPEC)

It can be filtered through search contact by name. Upon tapping, it shall transfer users to contact more details screen. It also generates call, text, and email buttons.

7.3.3 Interface Description

Shows information to the users and performs corresponding actions upon tapping buttons.

7.3.4 Processing Detail

7.3.4.1 Design Class Hierarchy

N/A

7.3.4.2 Restrictions/Limitations

Contact data must exist, otherwise nothing to show.

7.3.4.3 Performance Issues

N/A

7.3.4.4 Design Constraints

N/A

7.3.4.5 Processing Detail For Each Operation

N/A

7.4 GenericDescriptionFormat.js

7.4.1 Classification

The component contains text inputs for title and description

7.4.2 Processing Narrative (PSPEC)

The component feeds information into props

7.4.3 Interface Description

two text input fields

7.4.4 Processing Detail

7.4.4.1 Design Class Hierarchy

N/A

7.4.4.2 Restrictions/Limitations

N/A

7.4.4.3 Performance Issues

N/A

7.4.4.4 Design Constraints

N/A

7.4.4.5 Processing Detail For Each Operation

N/A

7.5 FormTextInput.js

7.5.1 Classification

Selected text data to fit a template of a design form.

7.5.2 Processing Narrative (PSPEC)

It is a design component; it is used by calling the component. Saves the text input data for future reference.

7.5.3 Interface Description

Have a box with a select amount of text data.

7.5.4 Processing Detail

7.5.4.1 Design Class Hierarchy

N/A

7.5.4.2 Restrictions/Limitations

N/A

7.5.4.3 Performance Issues

N/A

7.5.4.4 Design Constraints

N/A

7.5.4.5 Processing Detail For Each Operation

N/A

8. Database Design

Cloud Firestore is a cloud-hosted, NoSQL database. Following Cloud Firestore's NoSQL data model, you store data in documents that contain fields mapping to values. These documents are stored in collections, which are containers for your documents that you can use to organize your data and build queries.

Each user has a document in the "users" collection. Inside each user document exists respective sub-collections; one is "memories" which holds all the memories for that user. Full cloud functionality and database organization is to be implemented and integrated by a separate set of developers in the near future.

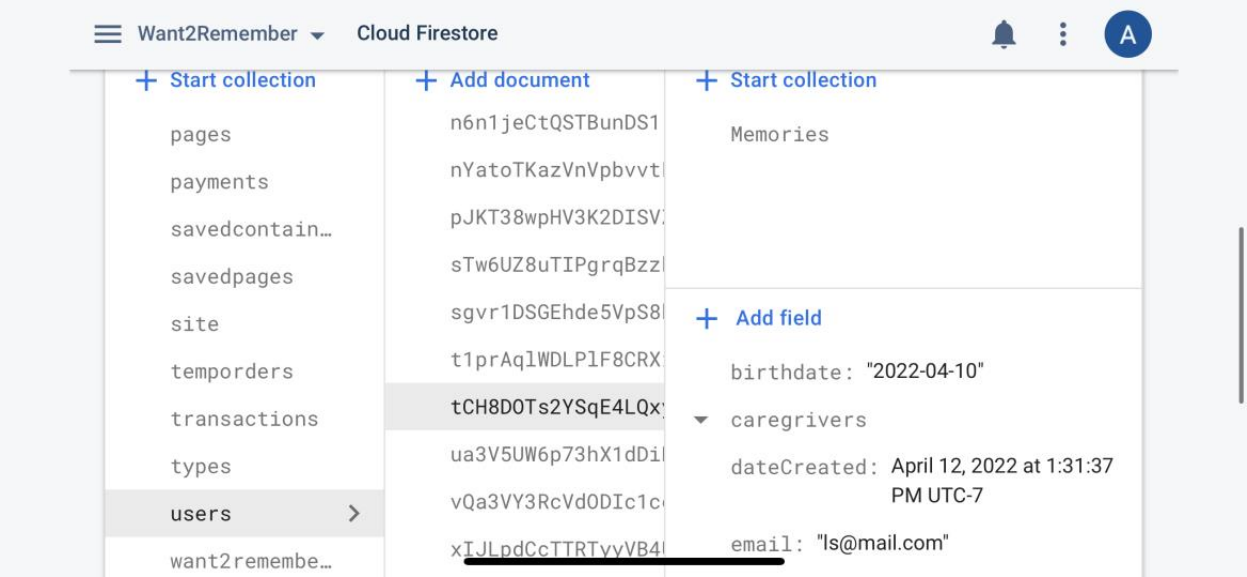


Figure 3: Firestore Test Collection

9. User Interface

9.1 Overview of User Interface

The system shall provide users access to the features of Want2Remember. There are four user interfaces that shall accompany this app: a free version user interface, a premium user interface (TBD), a free version caregiver interface (TBD), and a premium caregiver interface (TBD). The free versions shall have access to most features, while the other features shall be accessed once payment is processed.

9.1.1 The user interface shall present the Home Screen upon initial access.

9.1.2 The system shall provide a uniform look and feel between all pages. Headings, banners, fonts, and buttons shall follow the same style guide.

9.1.3 All screens shall include dynamic header component and bottom navigation bar.

- 9.1.3.1 Bottom navigation bar shall include Home, Create, Search, and Contacts.
- 9.1.3.2 Back button shall be in the top left corner to return the user back to the Home Screen.

9.1.4 Home Screen (default screen) shall show brief overview of all existing memories.

- 9.1.4.1 Help, settings, and filters shall be in the top right corner for navigation.
- 9.1.4.2 The Memory navigation bar shall show “All Memories” by default, with other subcategory options listed in the scroll bar to the right.
- 9.1.4.3 Below this navigation bar, the preview of the memory shall show with date, title, category, and other relevant summary information.
- 9.1.4.4 A button at the bottom right shall allow the user quick access to “Create,” leading them to the Create Memory Screen.

9.1.5 Create Memories Category Screen shall show all memory templates.

- 9.1.5.1 Memory tiles shall be labeled and color-coded in an accessibility-friendly way. Color customization shall generate options for those with color vision deficiency.
- 9.1.5.2 Clicking on a memory tile shall lead the user into the chosen template.

9.1.6 Search Screen shall let users filter through their memory tiles.

- 9.1.6.1 Search bar shall provide enough space for query.
- 9.1.6.2 Search results shall display below the search bar and be scrollable.

9.1.7 Contacts Screen shall show all contacts in scrollable view.

- 9.1.7.1 Add Contact shall be in the top right corner.
- 9.1.7.2 Search bar shall provide enough space for query.
- 9.1.7.3 Quick view shall include name, quick links to contact, and associated memories.
- 9.1.7.4 Clicking on View More shall allow the user to see full details of that contact.

9.1.8 Settings Screen

- 9.1.8.1 The settings options shall allow the user the ability to import and export JSON format memories.
- 9.1.8.2 The user shall be able to create and reset their secured PIN.
- 9.1.8.3 The user shall be able to send feedback.

9.2 Screen Frameworks or Images

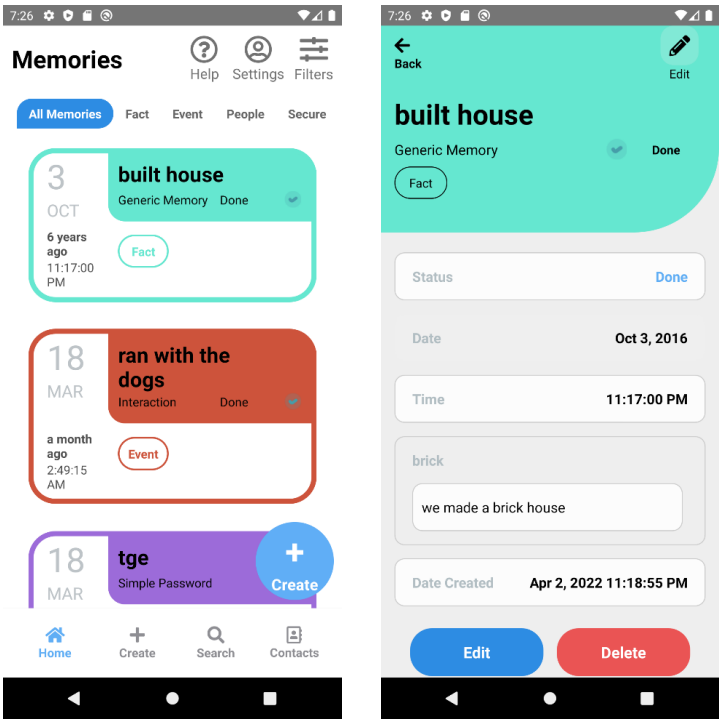


Figure 4: Homepage and Memory Details Screens

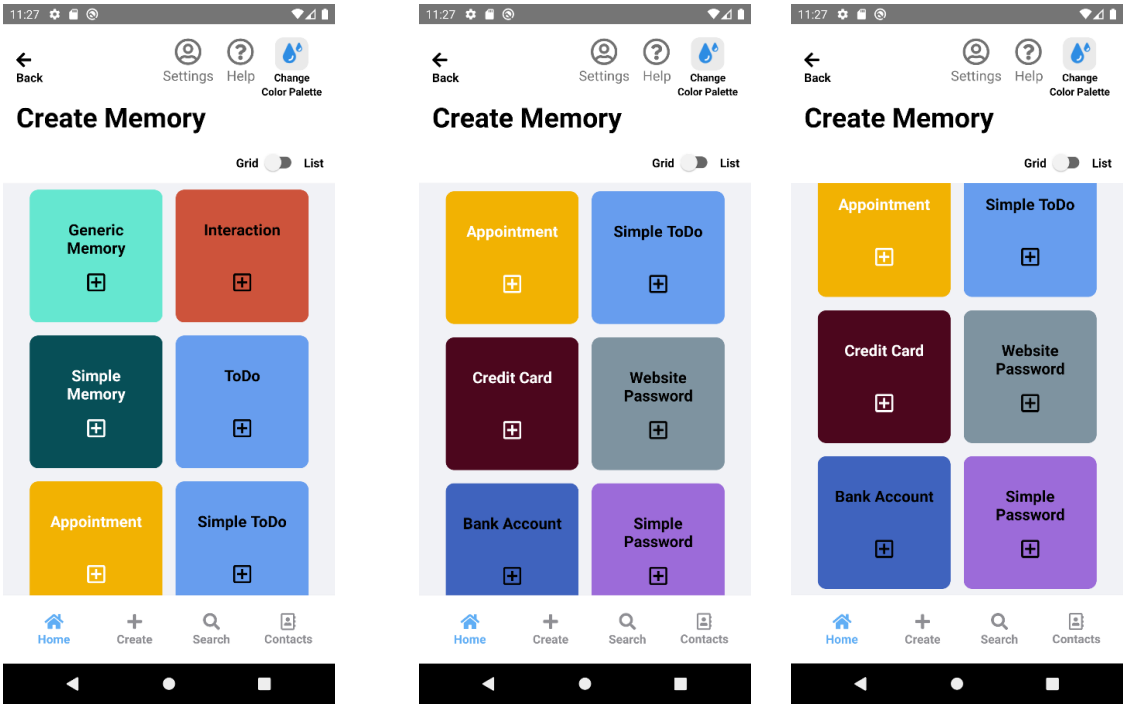


Figure 5: Create Memories Category Screen

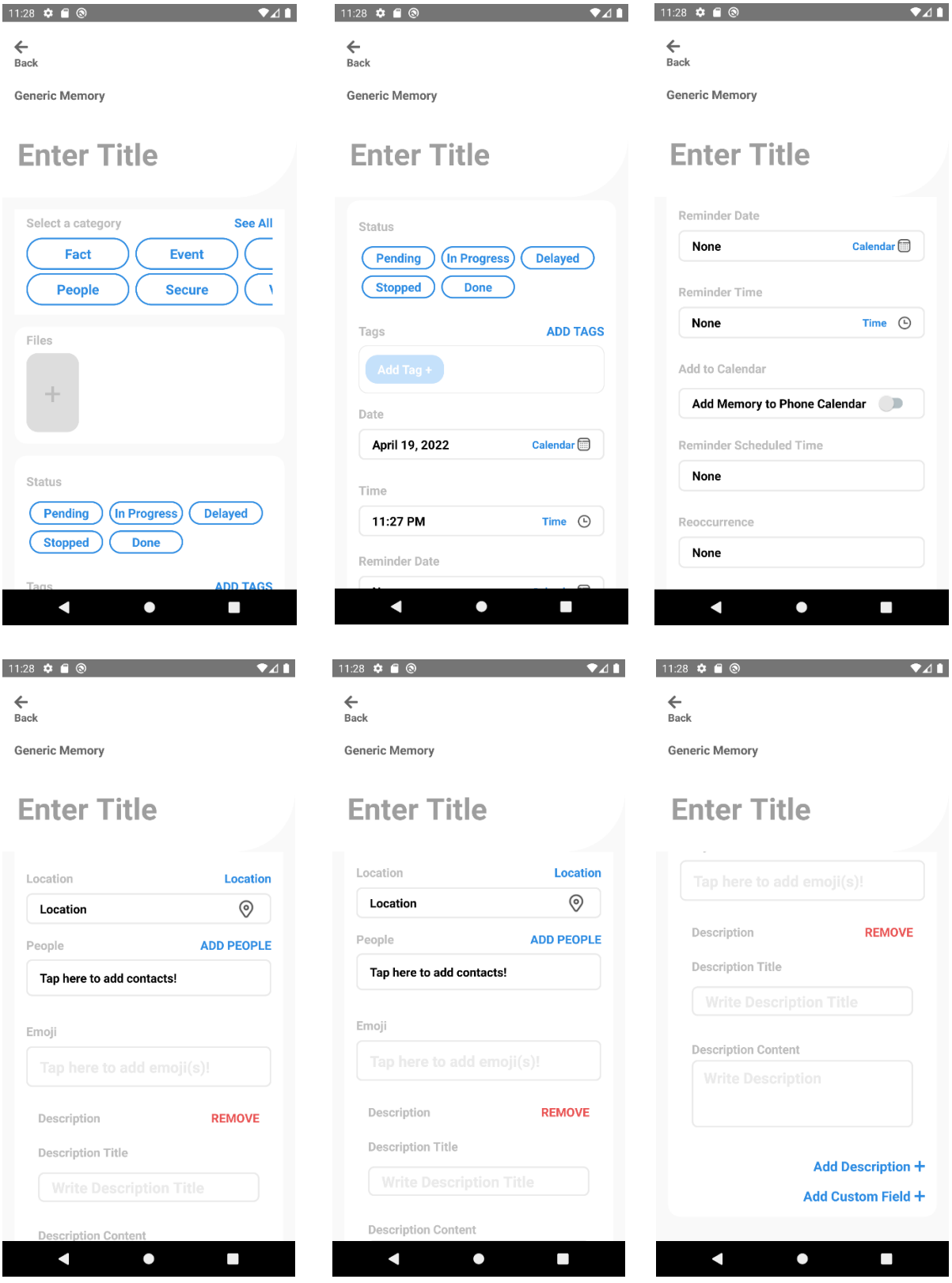


Figure 6: Create Memory Screen

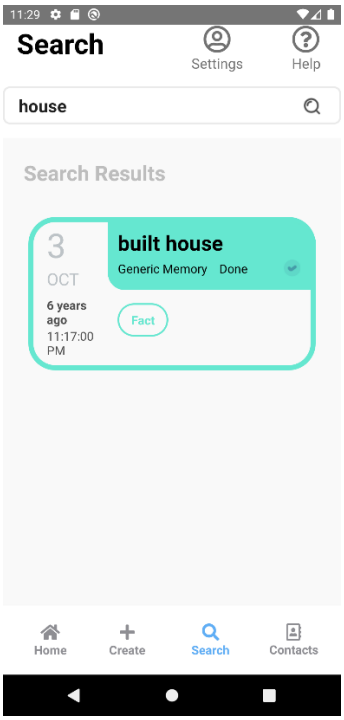


Figure 7: Search Screen

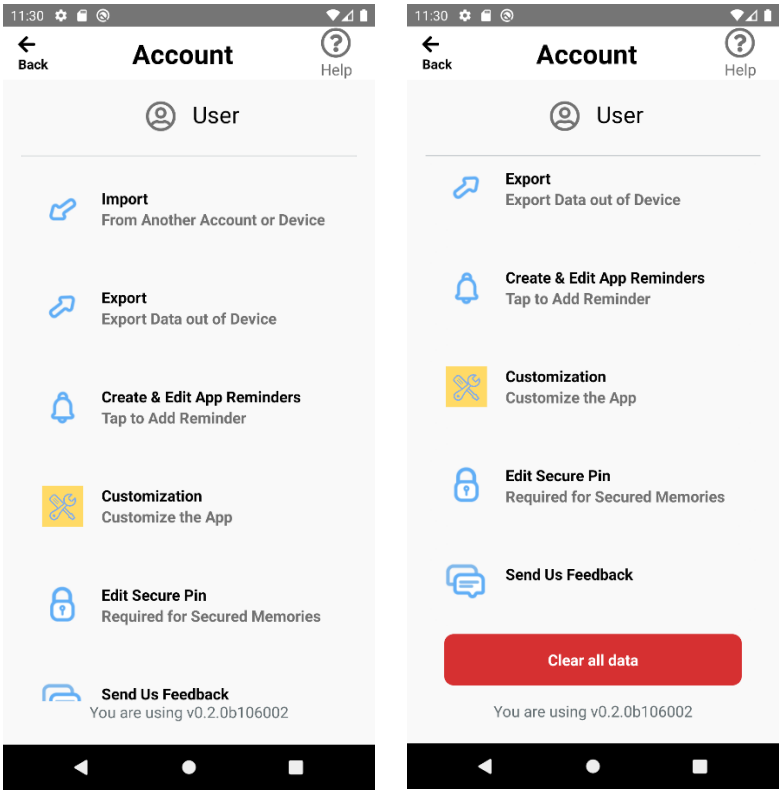


Figure 8: Settings Screen (Account)

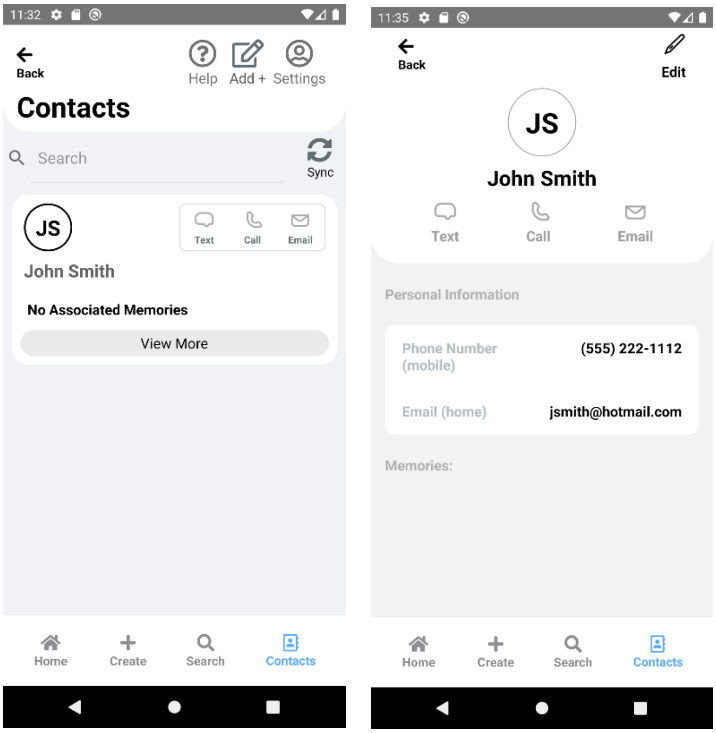


Figure 9: Contacts and Contact Detail Screen

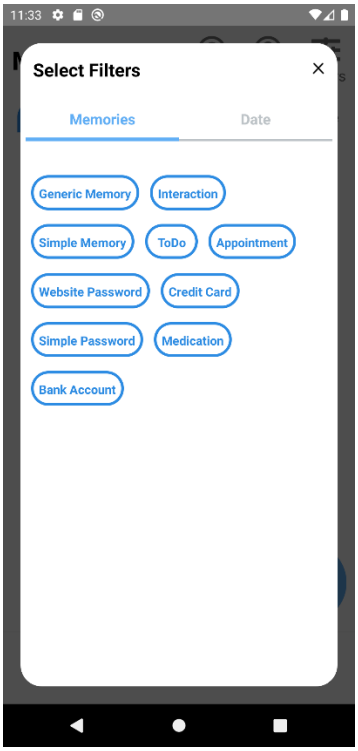


Figure 10: Filters Screen

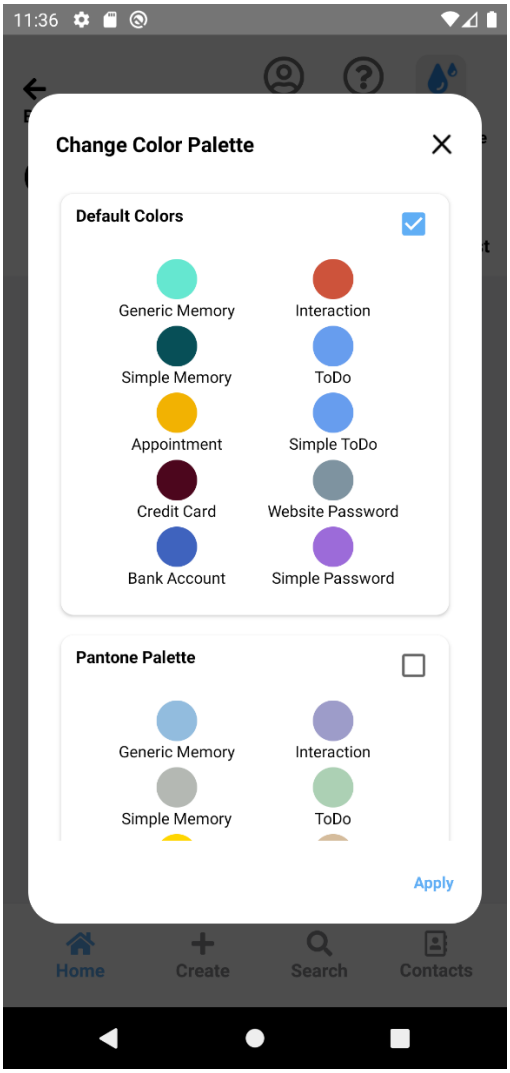


Figure 11: Color Palette Screen

9.3 User Interface Flow Model

9.3.1 Home Screen

Provides initial landing site for users. Provides brief overview of all existing memories. Allows user to filter memory categories and navigate to other sub-screens.

9.3.2 Register User

Allows the user to create an account.

9.3.3 Login/Logout

Allows users to login and logout.

9.3.4 Create Memories Category Screen

Allows user to choose from different entry templates. Layout is unique for each premade memory template. User may reorder the memory templates with long press and drag.

9.3.4.1 Generic Memory

Provides interface for user to edit and create generic memory from template. User may enter title, category, file, status, tag, date, time, reminder date, reminder time, reminder scheduled time, recurrence, location, people, emoji, description, and/or custom field.

9.3.4.2 Simple Memory

Provides interface for user to edit and create simple memory from template.

9.3.4.3 ToDo

Provides interface for user to edit and create to-do list from template.

9.3.4.4 Simple ToDo

Provides interface for user to edit and create simple to-do list from template.

9.3.4.5 Interaction

Provides interface for user to edit and create interaction from template.

9.3.4.6 Appointment

Provides interface for user to edit and create appointments from template.

9.3.4.7 Credit Card

Provides interface for user to edit and store credit card information from template.

9.3.4.8 Bank Account

Provides interface for user to edit and store bank account information from template.

9.3.4.9 Website Password

Provides interface for user to edit and store website password from template.

9.3.4.10 Simple Password

Provides interface for user to edit and store simple password from template.

9.3.4.11 Transaction

User may track past and/or current purchases. User may enter title, category, vendor name, purchase amount, purchase date, purchase time, transaction type, purchase method, account used, billing address, due date, next due date, and/or notes.

9.3.4.12 Customization Tool

User may create their own custom memory types or edit ones they have already created. User may enter a name for the memory type as well as the individual components they want to include for this custom memory type.

9.3.4.13 Change Color Palette

User may select the color palette they would prefer for the display of memory tiles. Provides default palettes (default, pantone, rainbow, deuteranopia, protanopia, and tritanopia) or the user may create a custom palette.

9.3.5 Search Screen

Allows users to search through entries and display results.

9.3.6 Contacts Screen

Create contact or add existing contact to application.

9.3.7 Help Screen

Allows user to send feedback and report a bug.

9.3.8 Settings Screen

Allows user to change application settings, import or export app data. User may also create and edit app reminders, customize the app, edit their secure pin, send feedback to developers (help screen), or clear all user data.

9.3.9 Filters Screen

Allows user to filter through entries using keyword tags or dates.

See following Application Flow Maps for navigation connections between screens.

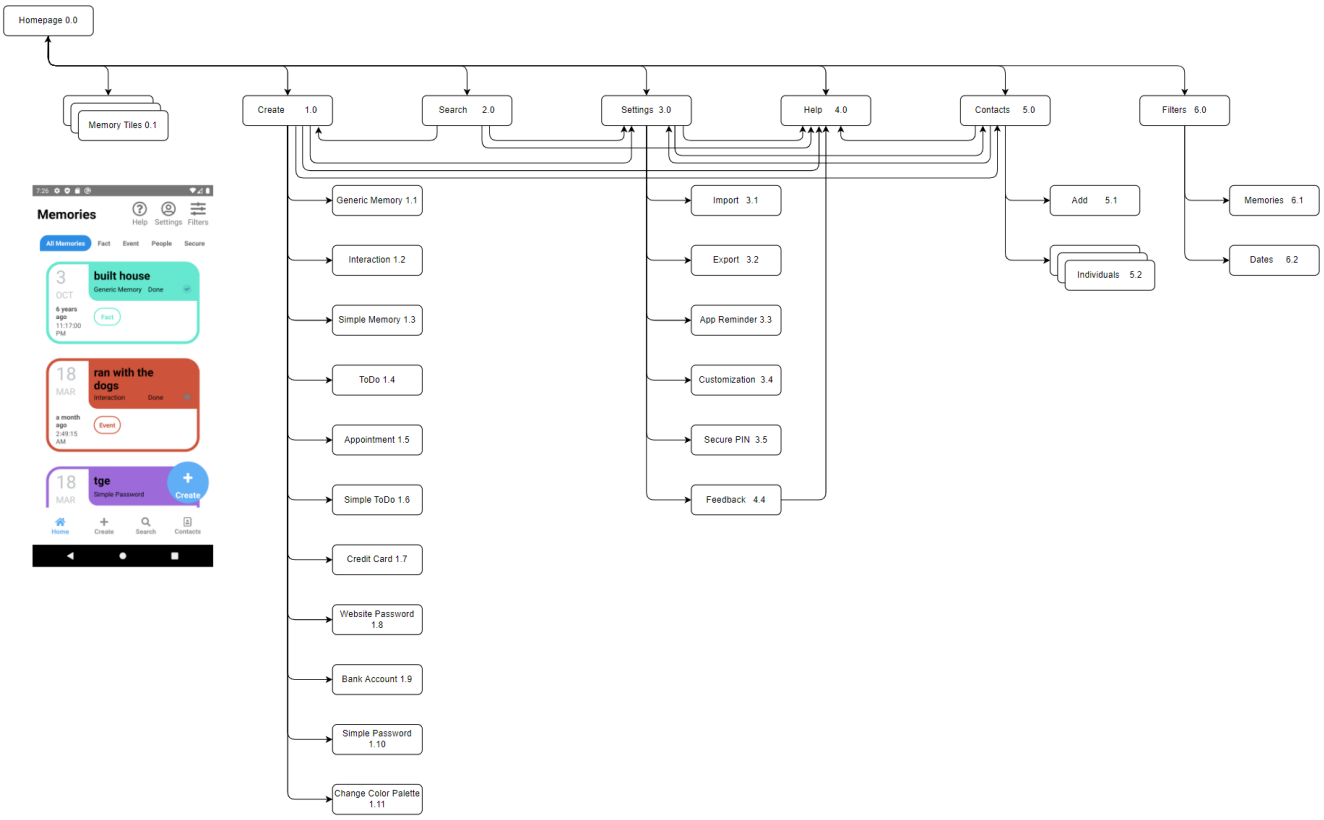


Figure 12: Application Map from Home Screen

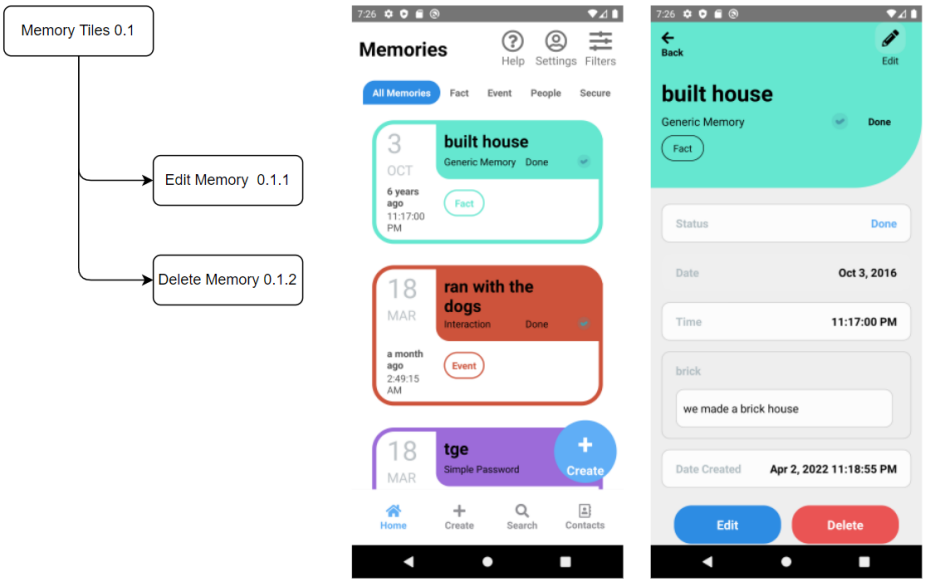


Figure 13: Memory Tile Map

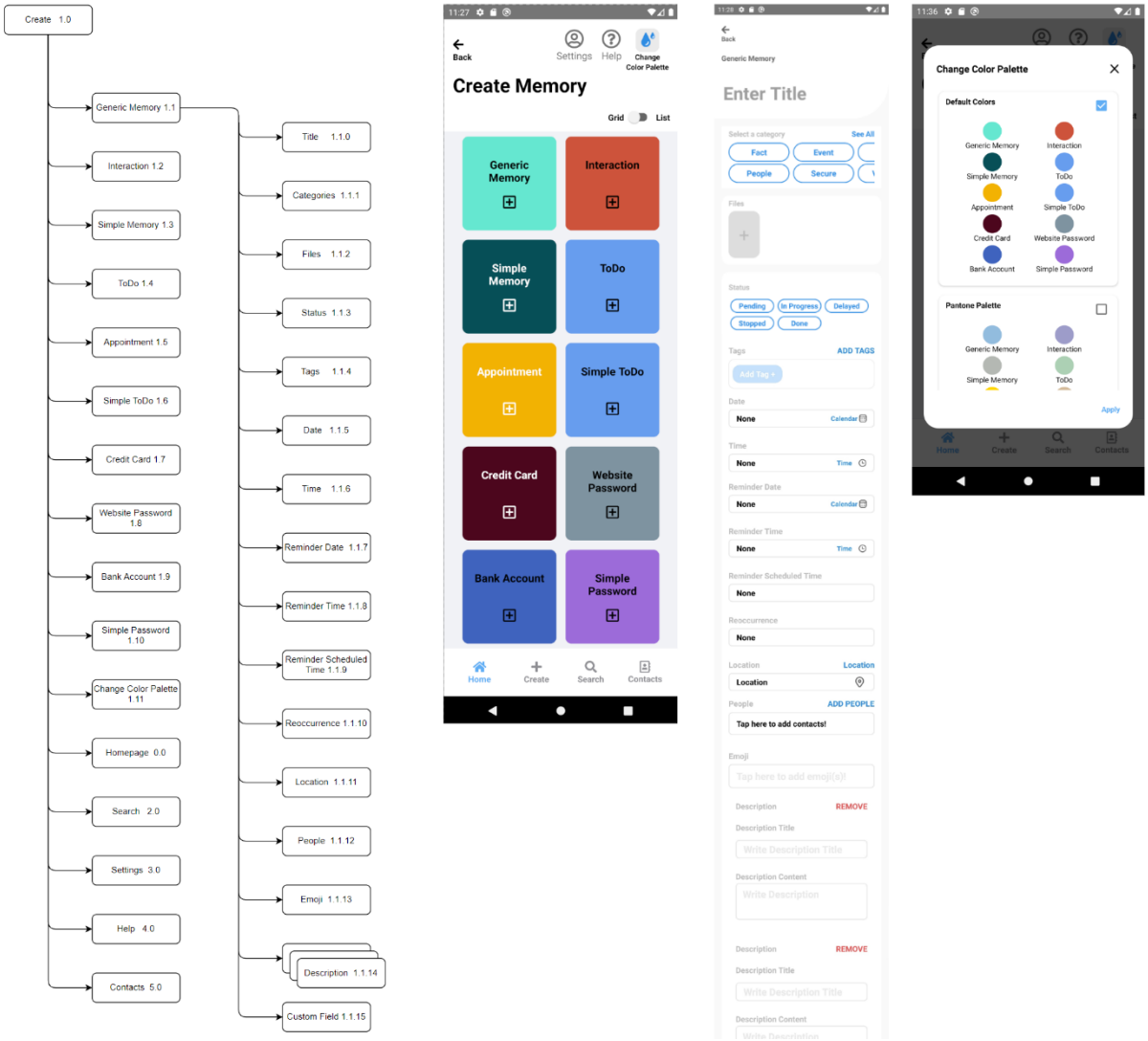


Figure 14: Create Memories Category Screen and Create Memory Screen Map

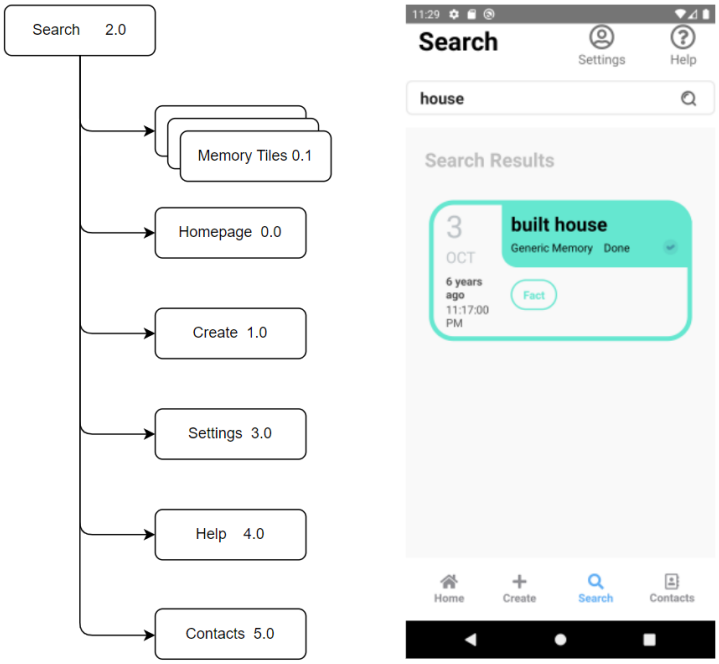


Figure 15: Search Screen Map

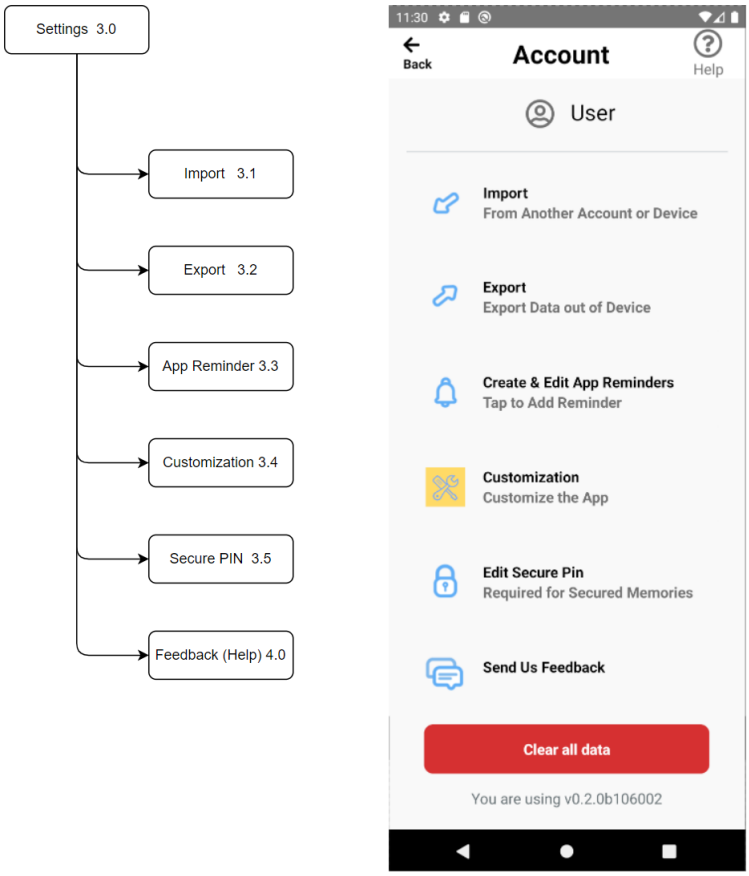


Figure 16: Settings Screen Map

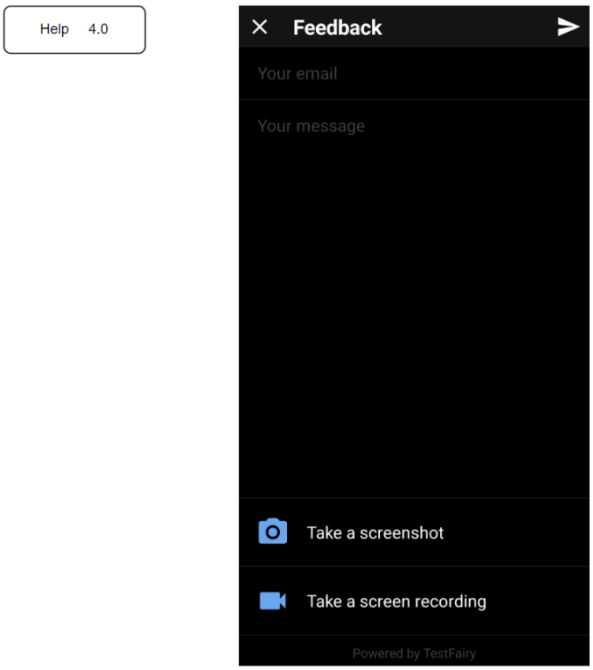


Figure 17: Help Screen

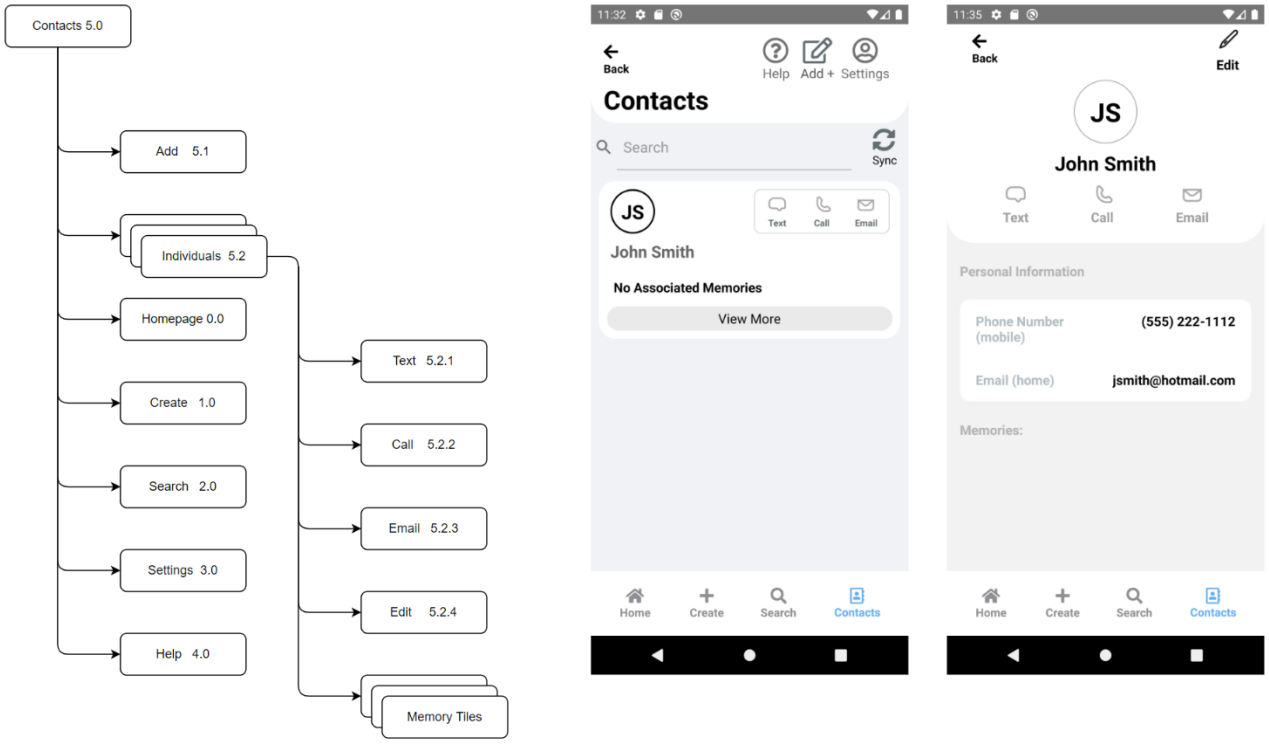


Figure 18: Contacts Screen Map

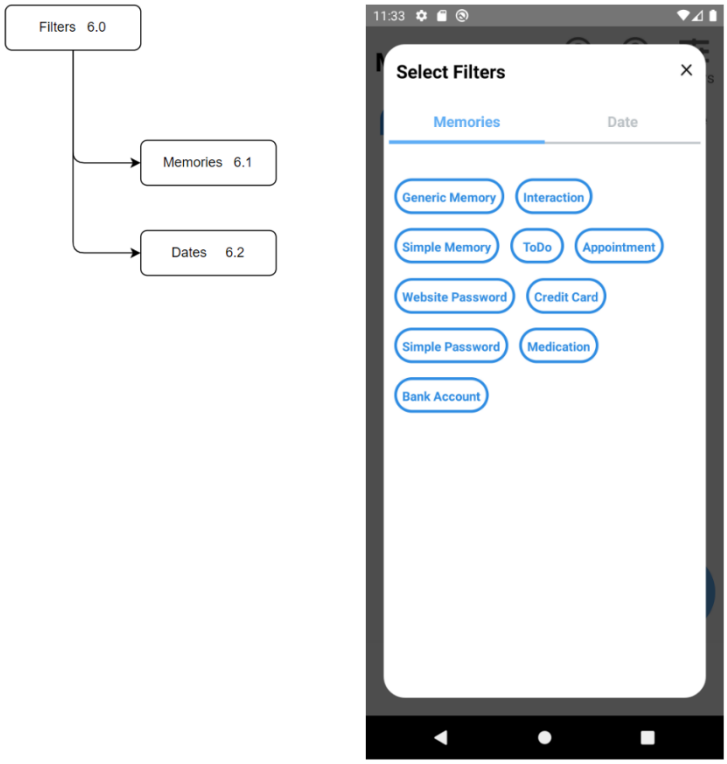


Figure 19: Filters Screen Map

10. Requirements Validation and Verification

10.1 Home Screen

10.1.1	The system shall display created memories on the home screen	
10.1.2	The system shall display memories within a category selected by the user (Filter by category)	
10.1.3	The system shall display memories within a type selected by the user (Filter by memory type)	
10.1.4	The system shall display memories within a time frame selected by the user (Filter by time/date)	
10.1.5	The system shall allow the user to create a new memory	
10.1.6	The system shall allow the user to select a memory to view	

10.2 Create Screen

10.2.1	The system shall allow the user to select from premade memory templates	
10.2.2	The system shall allow the user to give memories a title	
10.2.3	The system shall allow the user to select a category for a memory	
10.2.4	The system shall allow the user to create a PIN for secured memories if not already created	
10.2.5	The system shall allow the user to select a date for a memory	
10.2.6	The system shall allow the user to select a time for a memory	
10.2.7	The system shall allow the user to select a reminder time for a memory	
10.2.8	The system shall allow the user to select a status for a memory	
10.2.9	The system shall allow the user to enter tags for a memory	
10.2.10	The system shall allow the user to enter a location for a memory	
10.2.11	The system shall allow the user to select people for a memory	
10.2.12	The system shall allow the user to add a custom field to a memory	
10.2.13	The system shall allow the user to enter a title to a custom field	
10.2.14	The system shall allow the user to enter content to a custom field	

10.3 Edit Screen

10.3.1	The system shall allow the user to edit a memory	
10.3.2	The system shall allow the user to delete a memory	

10.4 More Details Screen

10.4.1	The system shall allow the user to view the details of a screen	

10.5 Contacts Screen

10.5.1	The system shall display contact quick looks	
10.5.2	The system shall display associated memories	
10.5.3	The system shall allow the user to call a contact	
10.5.4	The system shall allow the user to text a contact	
10.5.5	The system shall allow the user to email a contact	
10.5.6	The system shall allow the user to search for a contact	
10.5.7	The system shall allow the user to update contacts	

10.6 Search Screen

10.6.1	The system shall allow the user to search through the database with the search bar.	
10.6.2	The system shall display scrollable results below the search bar.	

10.7 Settings Screen

10.7.1	The system shall allow the user the ability to import/export JSON format memories.	
10.7.2	The system shall allow the user to create and reset their secured PIN.	
10.7.3	The system shall allow the user to send feedback.	

10.8 Backend

10.8.1	The system shall allow synchronization between local and cloud user data.	
10.8.2	The system shall encrypt user data.	

11. Glossary

API: Application Programming Interface
CRUD: Create, Read, Update, and Delete
GUI: Graphical User Interface
HIPAA: Health Insurance Portability and Accountability Act
iOS: Apple's mobile operating system
JSON: JavaScript Object Notation
JS: JavaScript
MacOS: Apple's Macintosh operating system
OS: Operating System
PIN: Personal Identification Number
RSA Encryption: public key/private key encryption method

12. References

Title: Software Requirements Specification Template
Author: Professor Jiang Guo
Date: (Accessed) August 24, 2021

Title: React Native - The Practical Guide
Authors: Academind by Maximilian Schwarzmüller, Maximilian Schwarzmüller
Date: (First Accessed) August 31, 2021
https://www.udemy.com/course/react-native-the-practical-guide/

Title: Software Design Document for Want2Remeber (Ver 2)
Authors: Kevin Benavente, Thomas Weatherell, Alejandro Salazar, Jesus Roman, Leon/Liangbin Huang, Jesus B. Osuna, Edward Ramirez, Tanya Kitchaiskulrit
Date: May 13, 2021

Title: Senior Design Project Report for Want2Remeber (Ver 2.1)
Authors: Antonio Campos, Alec Kaczmarek, Amy Guttman, Alexandra Strong, Vincent Li, Saiyang Liu, Ricardo Marroquin, Miguel Nonoal-Garcia, Jonathan Sum, Edwin Zapata Minero
Date: May 13, 2022

Title: Software Requirements Specification for Want2Remeber (Ver 3.1)
Authors: Antonio Campos, Alec Kaczmarek, Amy Guttman, Alexandra Strong, Vincent Li, Saiyang Liu, Ricardo Marroquin, Miguel Nonoal-Garcia, Jonathan Sum, Edwin Zapata Minero
Date: May 13, 2022