Senior Design Final Report

Want2Remember

Version 1.0 - 05/10/2023

David Pablo	Project Co-Lead
Jesse Gonzalez	Project Co-Lead
Harshil Kotamreddy	Documentation Co-Lead
Roger Ramirez	Documentation Co-Lead
Salvador Cornejo	Engineer
Angelo Esmeralda	Engineer
Victor Flores	Engineer
Marco Gonzalez	Engineer
Jonah Kim	Engineer
Ryan Mckean	Engineer
Zilong Ye	Faculty Advisor
Ricardo Marroquin	We2Link Liaison
Michael Malone	We2Link CEO

Table of Contents

1. Introduction

1.1. Background

Michael C. Malone, a veteran of the United States Army for over three decades, sustained a traumatic brain injury (TBI) during an Airborne Jump in 2012. Despite the injury, he continued his education and pursued a master's degree in 2014, only to realize he was experiencing difficulties with learning, memory retention, and recognition. To address these challenges, Michael founded We2Link, but his plans were put on hold when he was redeployed to the Middle East and suffered another brain injury while receiving medical treatment.

During his recovery at Walter Reed National Military Medical Center, Michael recognized that many other service members were also struggling with TBIs and needed additional support. This realization inspired him to develop Want2Remember, a mobile and web application designed to assist individuals with cognitive impairments in remembering their daily tasks and memories. The Centers for Disease Control and Prevention (CDC) estimates that approximately 1.5 million Americans sustain a TBI annually, and 5.3 million Americans live with a permanent TBI-related disability. Memory loss, whether from TBIs or progressive memory conditions such as Alzheimer's or dementia, can affect an individual's social and familial relationships and lead to caregiver burnout and social isolation.

Want2Remember seeks to support individuals with memory impairments and their caregivers by providing an intuitive user interface for logging important information and memories. The app aims to promote greater personal autonomy, social interaction, independent living, and return to work, while also enhancing medical support and personal safety. Ultimately, Want2Remember hopes to relieve caregiver burnout, decrease social isolation, and combat the stigma surrounding memory impairment.

1.2. Design Principles

To ensure that the technologies used in developing Want2Remember complemented each other and provided added functionality for the user, we carefully selected a technology stack. Our stack includes ReactJS, Google Firebase, Tailwind CSS.

ReactJS provides basic templates for Want2Remember web app, enabling users to easily log their memories, to-do lists, appointments, medications, passwords, interactions, and other essential notes. These templates eliminate the need for users to figure out the setup and organization of their entries on their mobile devices. All app screens and templates are composed of ReactJS components.

For backend development and database support, we opted for Google Firebase services and Google Cloud User Authentication. These technologies facilitate multi-platform support beyond local user data on the device.

1.3. Design Benefits

To ensure simplicity and reliability, we carefully selected technologies that provide support and modularization, allowing us to quickly incorporate feedback from beta testers and make code improvements.

Developing with the ReactJS framework gives us the advantage of developing seamlessly for all web compatible devices. The code is written in JavaScript but is then rendered with reactjs code. ReactJS also puts 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 ReactJS such as a multi-platform codebase, web app development, and fast refresh as soon as new code is saved.

Firebase is a highly effective and secure platform that allows us to develop Want2Remember while maintaining the security required for sensitive medical data. Google Cloud User Authentication provides secure user verification, ensuring a safe and reliable authentication process.

1.4. Achievements

Our team devoted two semesters to the development of Want2Remember, which was based on the work done by the previous year's Senior Design project. With only about nine months to complete the project, we focused on fixing bugs and incorporating additional user requirements into the existing design. To manage our workflow, we adopted the Agile Development Process and broke down tasks into Sprints that lasted one to two weeks. This allowed us to pivot quickly and respond promptly to feedback from beta testers. We used Atlassian's Jira software to visualize our task board and improve our workflow.

Our biggest challenges were learning the ReactJS framework and working remotely during the COVID-19 pandemic. To overcome these obstacles, we were provided with online resources such as the Udemy course “React - The Complete Guide (incl Hooks, React Router, Redux)”. We formed sub-teams to collaborate on manageable pieces of the project and communicated regularly through Zoom, Slack, and email.

Throughout our time on the project, we accomplished a great deal, including refactoring the codebase, integrating calendars for the web, expanding color palettes for individuals with color vision deficiencies, using dyslexia-friendly fonts, adding new

memory types, implementing a customization tool, improving sorting and filtering functionalities, revising the site map, and adding full cloud support for user data.

2. Related Technologies

2.1. Existing Solutions

There are currently no software solutions publicly available that are well supported, functional, and tailored toward helping those with cognitive impairment complete daily tasks. The most similar solution available is Qcard, a mobile app that was created to help users complete daily tasks through reminders, guided tasks, and appointments. However, the app is not cross-platform and it does not provide any support to caregivers. Additionally, the app has not been updated for nearly three years, meaning the app suffers from several bugs and issues leading to user frustration.

Other solutions exist, however there are no solutions geared toward helping cognitively impaired individuals with daily tasks and reminders. Reminder and to-do apps are vast and are very popular for the average user. However, they are often difficult to use for those with cognitive impairment due to ambiguous user interfaces. Note-taking apps are also immensely popular. An excellent example of a note-taking app is Notion. Notion is incredibly versatile, providing a vast amount of features and customizability. However, like the aforementioned reminder and to-do apps, Notion's user interface is ambiguous. The functionality of interactable components in the app are not always readily apparent. Additionally, Notion and other note-taking apps have a relatively high learning curve even for the average user. Those with cognitive impairment would find it incredibly difficult to use such apps.

Thus, in order to address these gaps in existing solutions, our solution involves a non-ambiguous user interface where all buttons and other interactive components are labeled through multiple cues (text, icons, colors). Additionally, our solution involves implementing features that allow caregivers to assist their patients through the app's interface.

2.2 Reused Products

The web application is developed with the following languages: HTML, CSS, JavaScript, TypeScript, and Tailwind CSS. The web application uses the ReactJS framework, an open-source JavaScript library, allowing the creation and re-use of components throughout the website. All components made in reference to the

components made for the mobile application. The languages mentioned before are heavily used within the ReactJS library. Firebase storage is used as a database storage for holding the user's data for their account and memory information. Google Cloud works in parallel with Firebase Storage for hosting the web application locally for testing and deploying.

3. System Architecture

3.1. Overview

The architecture of the application can be divided into three elements: The user, the components, and the database. The user creates and logs into their account and they are able to interact with the various screens in the application. The screens are built through the amalgamation of components that were made using React JS, a JavaScript library. When the user enters information into the application, the details are saved by Firebase, a set of cloud computing services.

I. The User:

The application ultimately fits the needs of its users and it is essential that the specifications of the app are made when considering their wants too. This is where the user will help give an idea of what they want using the feedback feature, which will result in modifications to the existing base functionalities and even additional features.

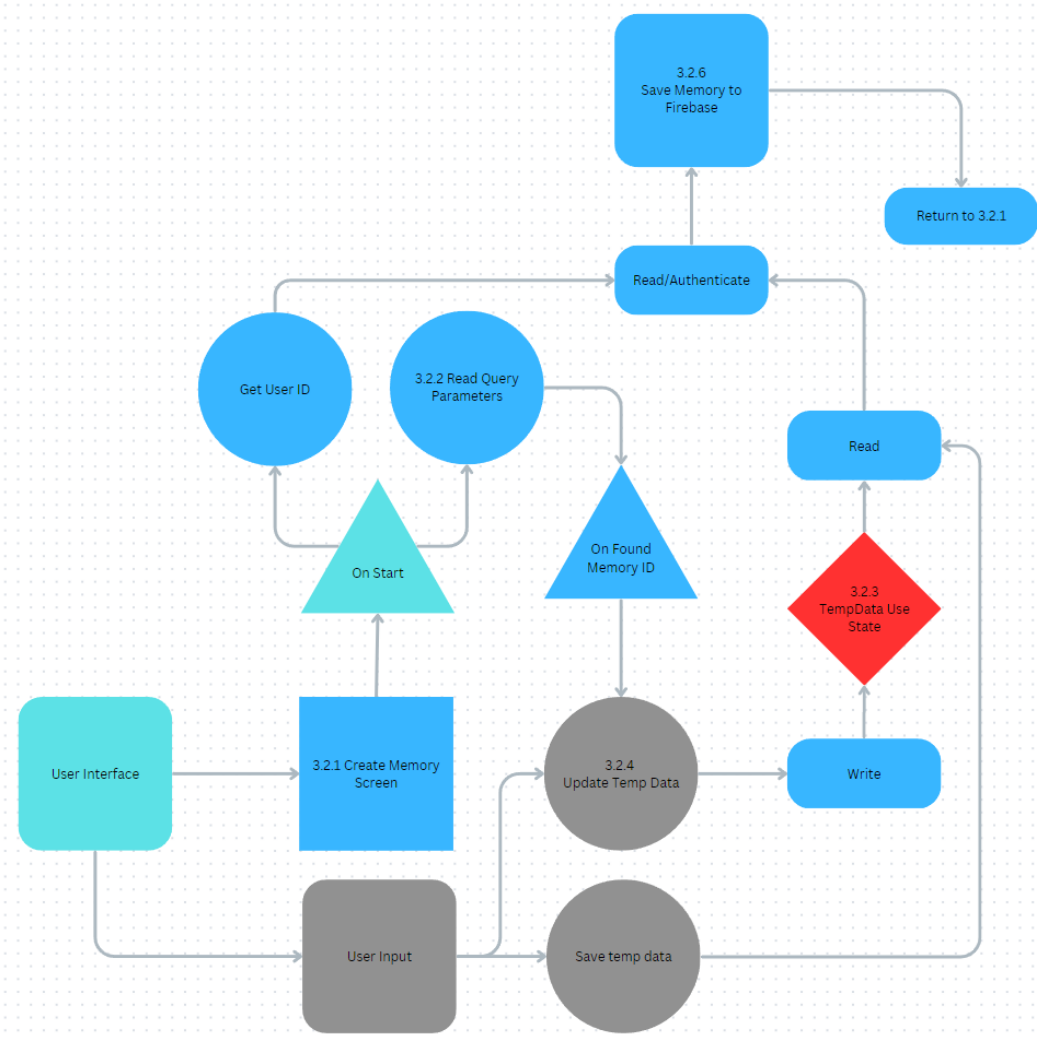
II. Components:

Components can be thought of as the essential building blocks that make up each screen and function on those screens. Each component is made keeping the idea of reusability in mind to keep the simplicity of the app to a maximum. Reusing components reduces the amount of code needed to produce each part of the app.

III. Database:

The data for each user that accesses the app needs to be stored somewhere and this is where Firebase cloud services come into play. Firebase keeps track of any changes that the user makes to the information they had previously entered. For example, when a user deletes a Memory, the data is then deleted and when a memory is edited, the new details are recorded in real time.

3.2 Data Flow



3.2.1 Create memory screen: The screen that the user sees upon entering the application is the initial interface that provides access to various features. To reach this screen, the user must click on the 'Create Memory' button, which takes them to a wide array of memory templates. After selecting a memory template, the user will be directed to this screen, which is where they can enter the necessary information to create a memory. This interface is user-friendly and intuitive, allowing users to navigate the application with ease.

3.2.2 Reading Query Parameters: After setting up the use states, the component sets up the 'useSearchParams' hook. The page is designed to expect either the type of memory that the user picked or the type of memoryID that is in the query parameters. The ideal function to use here is 'searchParams.get()'. If the memoryID is present but the memory type is not included in the URL query strings, the program will automatically determine

the memory type that the user is looking for from the memory data given the ID. Conversely, if the memoryID is not present, there should be a memory type in the parameters, which allows the program to determine which empty memory template the user should be looking at. Only one of these options can happen at a time; both cannot occur simultaneously.

3.2.3 React hook states: The 'temp data' object is implemented using the 'useState' hook because it's an object that the user can manipulate without changing any data in Firebase. Another hook used in this implementation is 'useEffect', which listens for changes to the 'memoryID'. The 'memoryID' usually changes when the value of the current memoryID is modified through the 'useParams' hook. If a memoryID exists, the 'useEffect' hook will automatically update all the data in the 'temp data' object.

3.2.4 Updating temp data: Given the large number of fields in a memory template, it's best to avoid invoking the 'setState' function in the 'useState' hook all the time with React. Instead, we can simply modify the key-value pairs in the 'temp data' object, which is just a JavaScript object. To change a specific field, we only need to modify the corresponding key in the 'temp data' object, such as 'reminder date time' or 'task1', and provide a new value. In the 'create memory' screen, the key in the 'temp data' object would be a string. This approach eliminates the redundancy of repeatedly typing the 'setState' function and the spread operator that JavaScript would otherwise require when updating only one value in the object.

3.2.5 Authenticating with user ID: The 'created memory' screen also utilizes a custom hook called 'useAuth', which was implemented by the We2Link employees. This hook makes it easy to obtain the user's ID without having to write boilerplate code. We need this ID when writing data to Firebase because we need to specify the path location. For example, the path location string would be 'users/{userId}/memories', where 'userId' is the user's unique ID. In addition, we need the user's ID to access that location, which is done behind the scenes by the 'useAuth' method.

3.2.6 Saving Memory to Firebase: Saving data to Firebase is quite simple. The React application is already connected to Firebase when the website is loaded. To get started, we need to import the database at the top of the file as 'db'. We also need to know which part of the database we will be working with and how to reference it. This is called a collection, and in the create screen page, we will be working with the 'memories' collection. Before we create a new document or update an existing one, we must check if the memory ID already exists in the collection. If not, we can simply create a new document using the temporary data and the new memory ID generated by Firebase, and then upload it to Firebase. Given the new memory ID the application will automatically reload itself using the id in the parameters.

3.3 Implementation

The project was split into three sections to allow efficient development: User Interface and Experience, React Framework, Firebase. Each section plays a key role in presenting the progression of the project.

3.3.1 User Interface and Experience:

The user Interface and Experience is one of the most important things we focus on because as mentioned in 1.3 design benefits, we want to provide simplicity. We always strive to make the app's user interface as simple and effortless for the user to have a good usage experience. With this said, for every component that we work with or we create, we avoid any nesting where a user can get lost or find it complicated to maneuver through the app.

3.3.2 React Framework

We used React for the reason of saving us, as developers, time and energy. React provides many tools, such as virtual DOM(Document Object Model), build complex and dynamic user interfaces, and many more. One of the biggest perks is that React allows us to create and reuse components that we can apply to other ideas we want to add to the app. We frequently use this because of the ease of applying it to different and new components of our work. It allows us to be flexible and use less files to become more efficient in how the app works and runs.

3.3.3 Firebase

When it comes to our testing data we use Google firebase. Firebase is a mobile and web application development platform. It provides many tools but one of the main tools we took into play throughout our work was its real-time database and hosting. If we wanted to test changes to a component or test a new component we use firebase to be able to upload data and retrieve it back to make sure our work is working properly and the way we expected it to run. It gives us the satisfaction of how it would run or what problems might appear if a user were to be using the app.

4. Conclusions

4.1. Results

This project brought to the Cal State LA Senior Design team has resulted in the adaptation of a mobile app to a web app to help people with cognitive impairments and their caregivers. A new base for the Want2Remember application has been built and optimized for web use, heavily relying on React.js for a smooth UI experience. The functionality of the mobile app has been used to guide the design and usage pattern of the web app so the two are not hugely different. Certain architecture patterns have also persisted into the web app to more easily migrate the mobile components or adapt new components to the correct design pattern.

This web app is connected to Google Cloud Services using Google Storage and Firebase to manage data. As well as Google Analytics to track performance and usage. User data is stored locally and in the cloud so a user will always have access to their memories.

A login and sign up page has been created for more secure entry to a user's profile. Want2Remember also allows users to import and export their data which includes memories, contacts, files, etc.

The web application also has been designed with many quality of life features. Users can customize their memory colors to any variation of color blindness, or even to custom colors. Filters and sorting have been implemented on the main page for memories so a user can more easily find or search for a category of memory.

Caregiver support has been added which allows a person to add another user as a caregiver to their profile. Caregivers may be permitted to view, edit, or create memories for the person they are caring for allowing for better assistance in managing memories.

Allow Users to Import/Export User Data. Added Caregiver Support. Customize Memory Colors. Filter Memory Categories. Sorting and Filter functionality. User Sign In and Sign Up. Contact Management

4.2. Future

The Want2Remember web app currently has most of the features from mobile implemented as discussed earlier in 4.1. The web app version has been deployed using firebase hosting and will allow mobile users to access their memories from more devices. While the Want2Remember web app has a lot of features from mobile currently implemented as discussed in 4.1, there are still improvements and features to add. The following are some of the features that are currently planned:

- Accessibility Features: While color palettes are currently stored in firebase and are being used for memories, the UI will need to be updated across all screens to use the users custom color palette. Fonts will also need to be customizable improve accessibility.
- Memory Customization: Want2Remember currently has 11 memory templates, and with memory customization, the user can create custom memory templates suited for them.
- Improved Caregiver features: Some of the features that will help caregivers include gps tracking of supported users and automatic role assignments.
- Medication Tracking: Users will be able to set up reminders for medication and medication administration and will be able to save things like proofs of prescriptions.
- Machine Learning/AI features: This can include things like using AI to improve interactivity with the app or implementing smart reminders.

that can be added to further support people with cognitive impairments.