

**Final Documentation**

Scott Lam, Leo Ras, Thomas Simmons

**Introduction**

As technology continues to advance, we see new and innovative solutions emerging to tackle various problems. One of the challenges that many people face is finding the perfect outfit to wear. Whether it's for a special occasion or everyday wear, it can be a struggle to find clothes that fit our style and preferences. Additionally, fashion trends are constantly changing, and it can be expensive to keep up with the latest styles. To address these challenges, we have developed V-Closet, an online virtual wardrobe that provides users with a range of features to make their fashion choices easier and more affordable. V-Closet allows users to create an account and upload pictures of their clothing items, which can be categorized by type, color, or any other criteria. This provides users with a convenient and organized way to keep track of their wardrobe and easily find specific items when needed.

**Background**

Closet organization applications currently exist to help users control their wardrobe and plan their outfits. These applications, though, focus mainly on styling, recycling, and finding outfit inspirations. So far, a wardrobe organizer that plans outfits in accordance to weather data does not exist.

In creating V-Closet, we gathered information on various geolocation and weather APIs that would support our intended use. We incorporated these APIs in our project to provide a unique service that we believe many users will benefit from. By automating daily outfits and ensuring its compliance with forecasted weather conditions, V-closet can help eliminate the difficulty of making outfit decisions, promote the reuse and recycling of clothes, and bring our user the confidence of dressing correctly for any occasion, at any time, and during any type of weather.

One of the standout features of V-Closet is the personalized recommendations it provides based on the weather. By using up-to-date weather information, V-Closet can suggest outfits that are appropriate for different weather conditions, such as rain, cold, or hot weather. This helps users to stay comfortable and stylish, regardless of the weather outside.

In addition to personalized recommendations, V-Closet also functions as a marketplace, allowing users to buy, sell, and trade clothing items with other users. This feature provides users with a cost-effective way to expand their wardrobe and experiment with new styles without having to spend a lot of money on new clothes. With V-Closet, users can explore a wide range of fashion options and find unique pieces that they may not have been able to find otherwise.

Overall, V-Closet provides users with a comprehensive solution for managing their wardrobe, finding new styles, and staying on top of fashion trends. By combining personalized recommendations, weather updates, and marketplace functionality, V-Closet is a must-have tool for anyone looking to elevate their fashion game. The following block diagram in Figure 1 showcases the main features of our system.

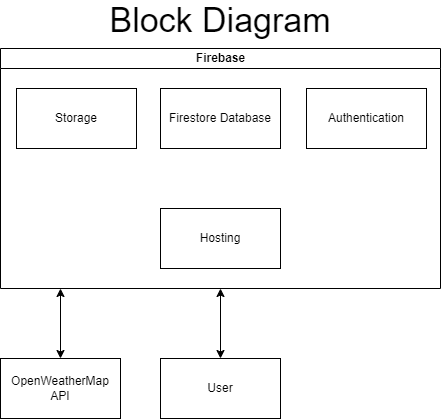


Figure 1: Block Diagram

**Technology**

We created V-closet as a web-based application and so one of our initial decisions focused on browser compatibility. For our internet browser, we decided to use Google Chrome, a web browser developed by Google, and Firefox, a web browser developed by the Mozilla Foundation. The designs and layouts will be optimized specifically for Google and Firefox. This means some elements of the project may not work as intended with other browsers. These browsers were easy to work with, because we have been using them for many years.

Our project will primarily be developed on the Windows 10 and 11 operating systems. These Windows versions are ideal for our project needs, given their rich set of features and capabilities. Windows 10, a widely used operating system, is renowned for its stability and ease of use, and Windows 11, the latest Windows version, comes with an array of new features and enhancements. We did not have much trouble with Windows. There was a little trouble setting up some of the emulators due to some security features, but we were able to get around them.

We have opted to utilize Visual Studio Code exclusively for our project's code writing. Our decision was informed by several considerations, including the user-friendly interface, adaptability, and comprehensive functionalities provided by VS Code. Moreover, VS Code's extensibility offers an additional advantage. Its extensive library of extensions and add-ons enables us to tailor our development environment to our specific needs and preferences, enabling us to work with greater efficiency and effectiveness, unconstrained by the default environment's features and capabilities. Lastly, we value the open-source aspect of VS Code, which provides numerous benefits, such as community support, prompt bug fixes and updates, and the ability to contribute to the project's development. We all had previous experience working in Visual Studio Code, so learning the ins and outs was not an issue.

Our project will be mainly focused on web development which utilizes three critical languages, namely HyperText Markup Language (HTML), Cascading Style Sheets (CSS), and JavaScript (JS) to build interactive and visually appealing web pages. HTML defines the page structure, CSS styles the page, and JavaScript adds functionality. For the V-Closet website, we used these three languages to develop an immersive and visually appealing user experience.

For version control and managing progress, we have chosen to use GitHub as our primary development platform. With GitHub, we can easily manage and track the progress of our development process, including any code changes, bug fixes, and feature enhancements. We can also work collaboratively with our team members, sharing code snippets, reviewing code, and providing feedback in real time.

For the library used for the development of this project, we decided to utilize the power of React to develop the website’s user interface. With React, we can create reusable UI components that can be easily combined to create complex and responsive layouts. Another advantage of using React is the extensive tools and libraries available. We can take advantage of third-party plugins and modules to streamline our development process.

After some research, we decided to use the OpenWeatherMap as it was suitable for our application’s weather data needs. The API is user-friendly and the basic service is available at no cost, although a paid, premium service is also available should we need to scale the application. This API provides us with the weather information we need to enable V-Closet’s recommendation component, which includes a brief description of the current weather conditions, high and low temperatures, and a weather icon.

We also needed a geolocation API to provide geographic coordinates for the weather API. This service is provided by GeoDB Cities from RapidAPI. This API offers a beginner-friendly interface for determining the user’s geographic coordinates and is customizable based on the developer’s needs. Using this free-to-use API allows us to maximize the weather function of the application while minimizing any additional costs.

The majority of our web application has been developed using Firebase, a comprehensive and robust mobile and web application development platform owned by Google. Firebase offers a diverse range of tools and services that enable developers to create high-quality applications with ease. For efficient management of documents and clothing photos, we have leveraged two Firebase products: Storage and Firestore.

Storage facilitates the storage of clothing photos submitted by our users on Google's secure cloud infrastructure, while also providing mechanisms to regulate access to the images. Firestore manages documents that contain information about each clothing item, such as color, category, and links to the corresponding photos stored in Storage. An Entity Relationship Diagram can be viewed in Figure 2. Firebase Authentication has been set up to handle user sign-up and login processes on our web app. Our current web app setup supports email/password authentication and third-party authentication with Google. Each user is granted access only to their own photos and documents, ensuring data privacy. To make our web app accessible on the internet, we will be utilizing Firebase Hosting for deployment. To ensure secure communication, all traffic to our site will be encrypted through HTTPS, as facilitated by Firebase Hosting.

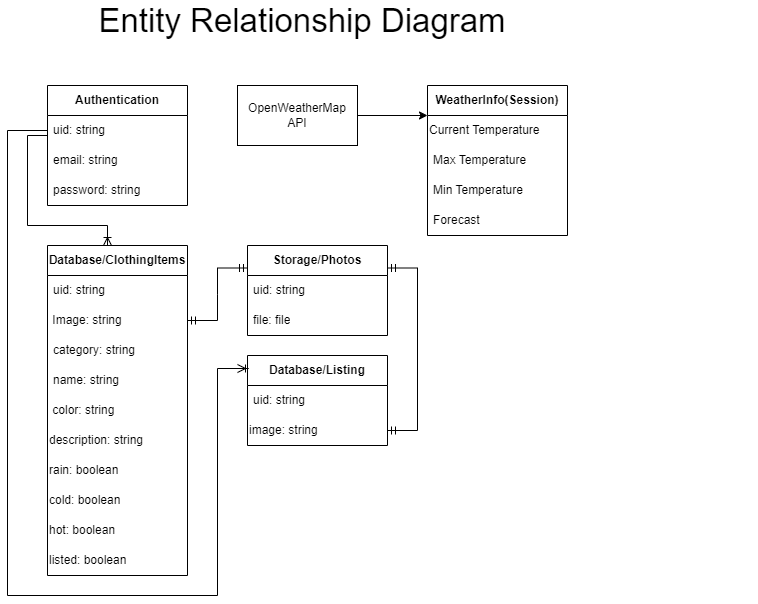
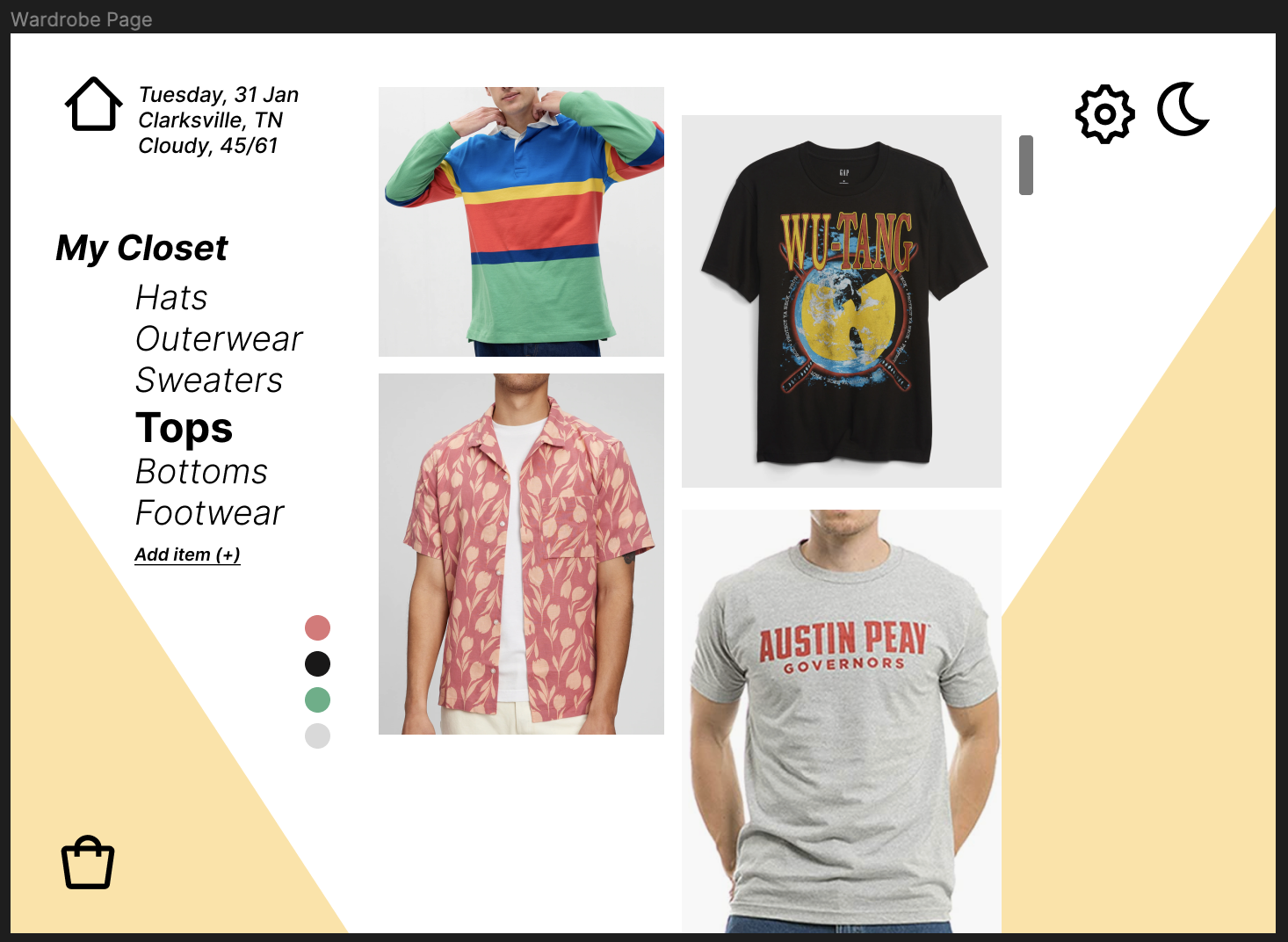


Figure 2: Entity Relationship Diagram(ERD)

It is important to be able to collaborate on our work outside of the classroom setting and so we used a few web applications to support this need. For word processing and collaboration, we used Google Docs. This allowed us to create, edit, and store our presentations, papers, diagrams as a group. Google Drive was used for file sharing and for backing up our files throughout the planning and development process. We also used the popular instant messaging platform Discord to be able to communicate with each other during this process.

Figma is a cloud-based design and prototyping tool used by designers, developers, and product teams to create and collaborate on digital designs. It is a powerful tool that can be used to create visual designs and interactive prototypes for a wide range of projects. Using Figma allowed us to streamline our planning and design processes leading to a smooth transition into development. Figure 3 shows a version of our prototype design.



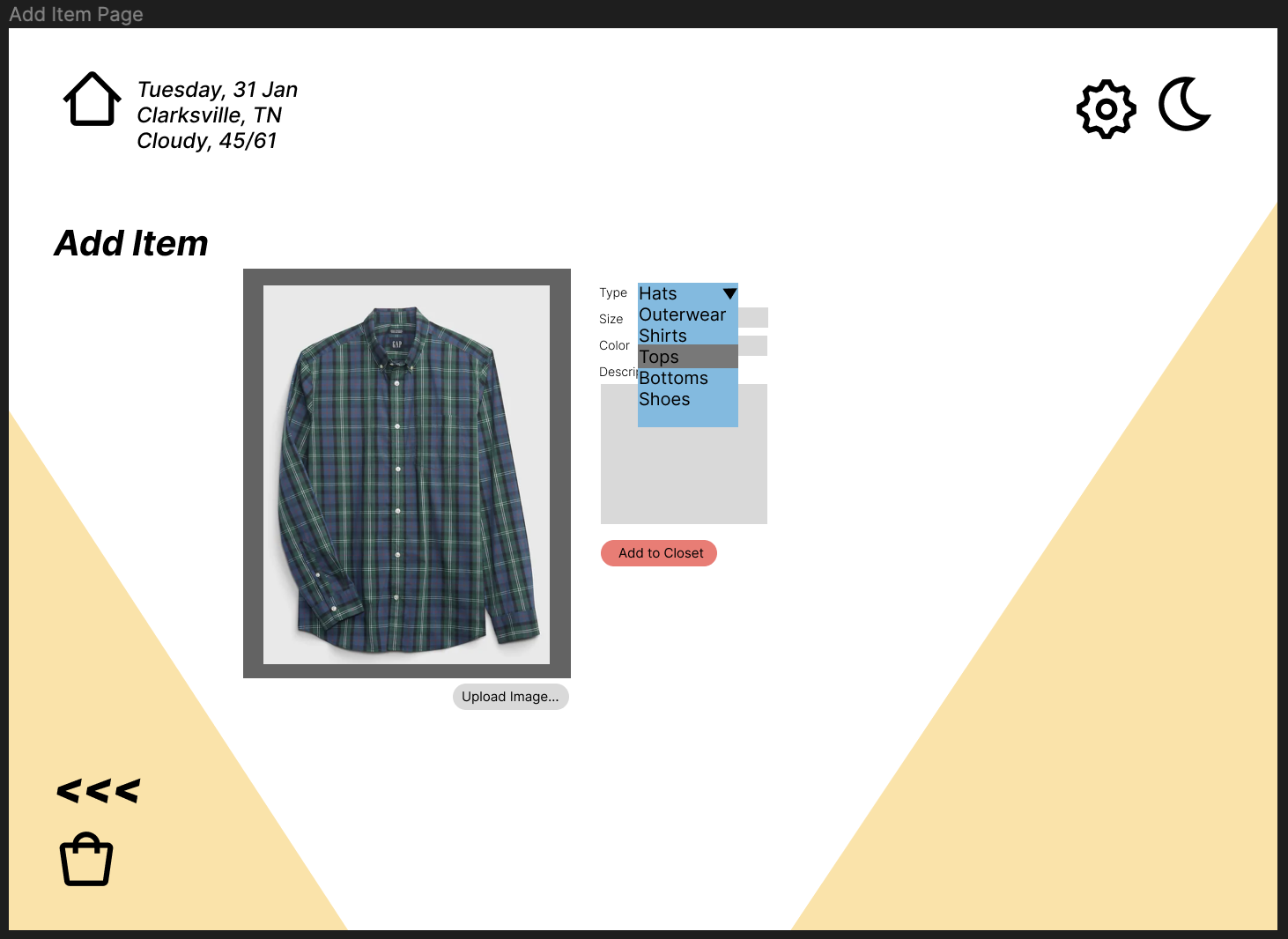
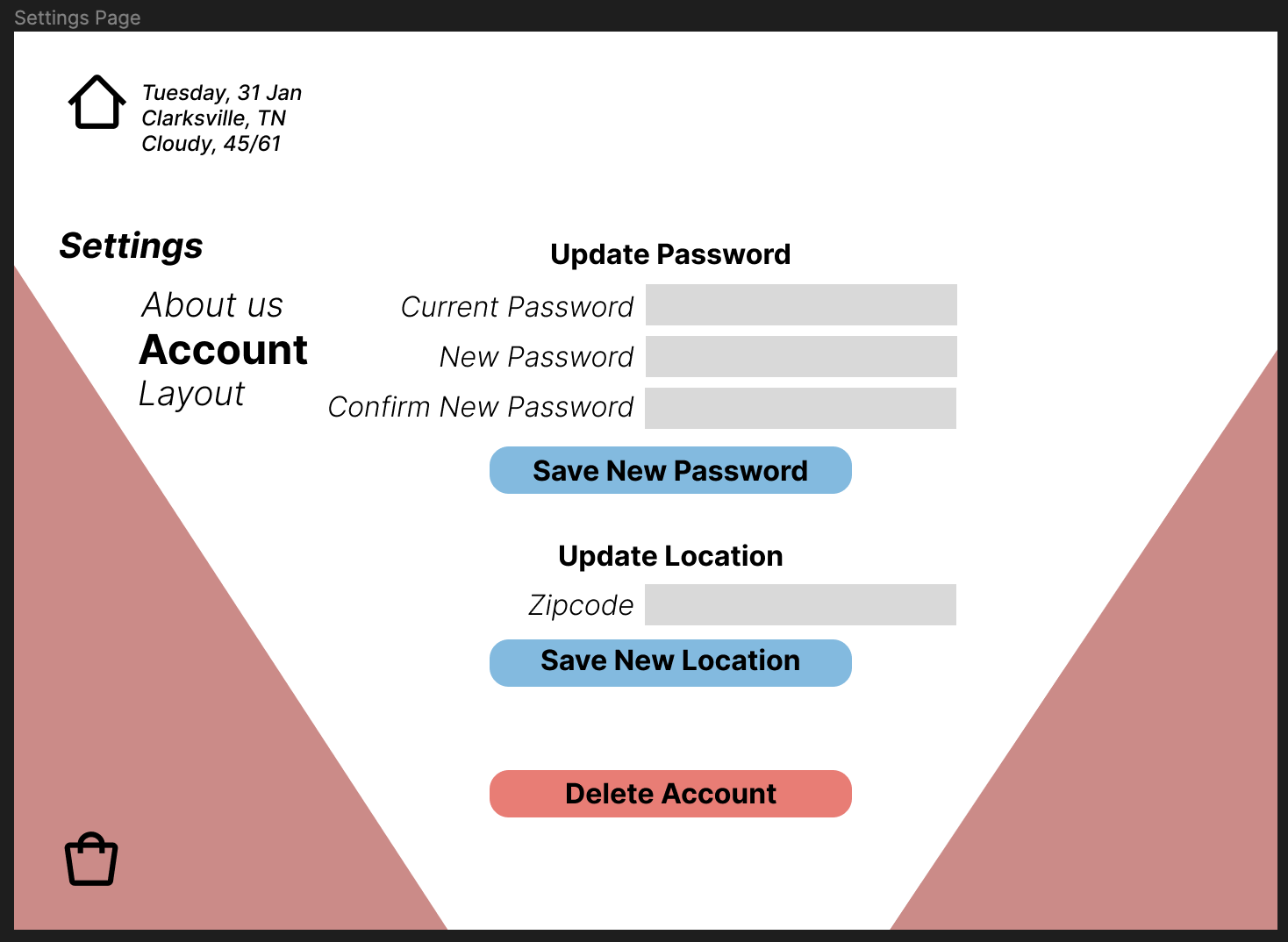


Figure 3: Prototype design of V-Closet pages created on Figma

Our team possessed a broad knowledge of software design and architecture, front-end and back-end web development, and user experience that was largely gained in an academic setting. We had varying experiences with a few of the technologies we employed, and even less with the others. For example, most of us had not worked with Firebase or React. As with any new technology, the learning curve can almost alway be overcome with a few Youtube tutorials or general web searches. The learning curve with Firebase, though, is as steep as they come and we can collectively agree that this was the most difficult to configure of the technologies we used in our project. Learning how to use the different APIs took a lot of time, especially because there was a recent update on how they are used. This made learning from examples made over a year ago impossible.

The technologies we used throughout development certainly made development easier. For one, being able to configure Firebase to handle most back-end functions, including the database, user authentication, and hosting, saved us time and allowed us to focus on other aspects of the application. The ability to create, combine, and reuse React components allowed us to quickly add functionality to pages wherever needed. One example of this was how we added an authentication component to each page that took care of verifying whether a user was logged in or not. The Node Package Manager allowed us to easily install any dependencies we needed such as Firebase, icons, and other React components.

We stuck with most of our design from our original proposal. However, we saw an opportunity to use a different weather API during the development phase. We replaced our original plan of using the National Weather Service weather API with another popular, free to use, weather API from OpenWeatherMap. We made the decision to switch as we had no use for the extensive amount of weather information that the original API provided. We also found it to be more cumbersome than we expected. The latter, however, provided us with a more simpler interface while still being reliable and free. The latter being a popular weather API also meant that there were plenty of resources to reference when building the component. The decision ultimately allowed us to save time and effort in development, and minimized the amount of data that the application needed.

**Design**

The V-Closet project is built using a component-based approach using React, where pages serve as the primary components (Figure 4). Each page comprises various components and functions that are tailored to its specific purpose. The project encompasses several pages, such as Landing, Login, Sign Up, Home, Wardrobe, Adding Item, Marketplace, and Settings, which are made possible to render and navigate through using the index.js file. Upon visiting the website, users are automatically directed to the Landing Page, which showcases the brand's name, motto, and two clickable buttons: Login and Sign Up. The website's design is minimalistic, trendy, and chic, featuring a black and white color scheme. Clicking on the Sign Up button will redirect users to the Sign Up Page.

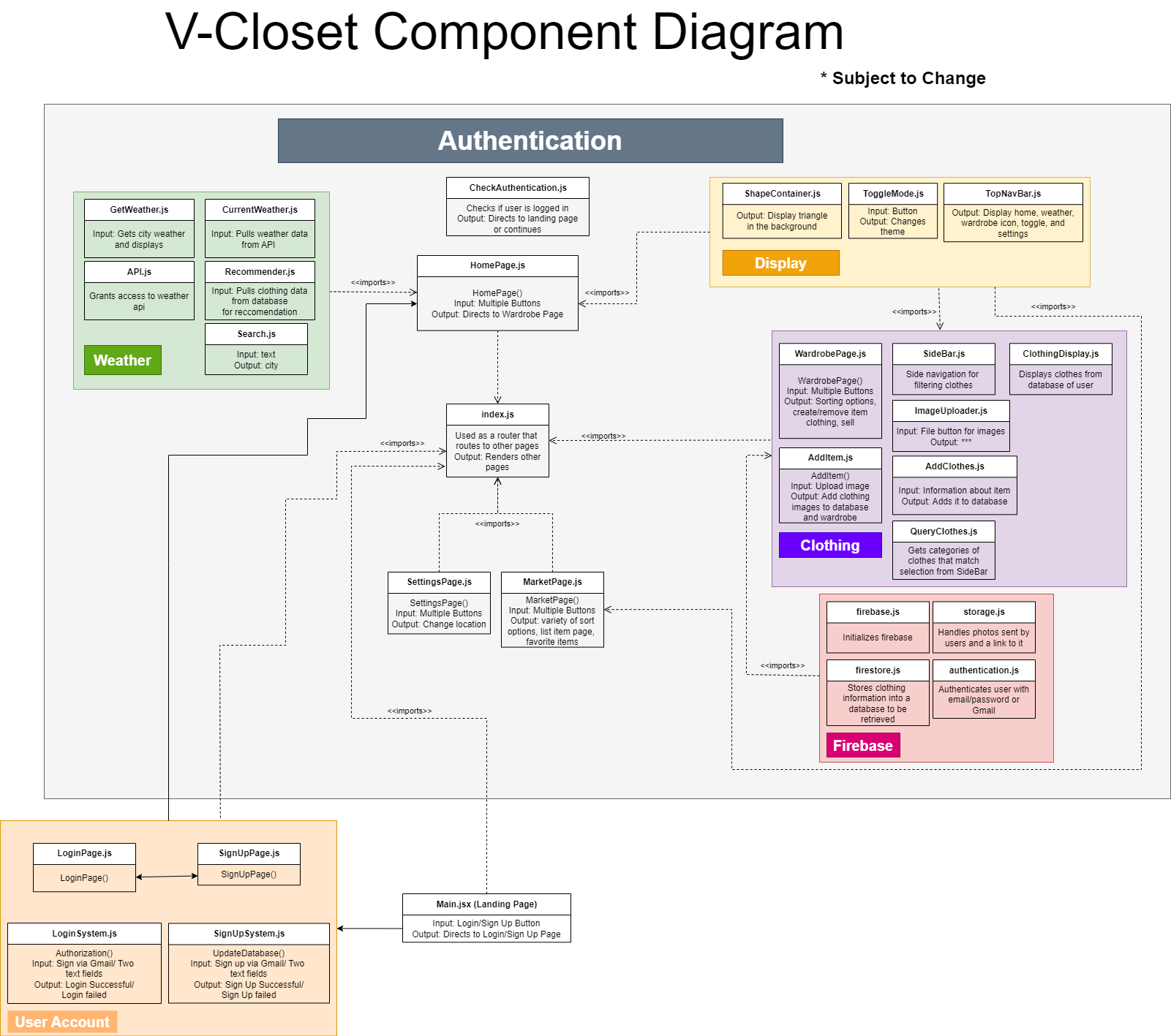


Figure 4: V-Closet Component Diagram

The Sign Up page serves as a registration form for users interested in creating a V-Closet account (Figure 5). Without an account, users cannot access the website's features. The page exclusively consists of the SignUpSystem.js component, which functions as the registration form. The component includes fields for entering email and password or using Gmail to register. Additionally, the component includes error-checking features to ensure that the user enters matching passwords.

If the user fails to sign up correctly, the form resets, requiring the user to re-enter their information. Below the input fields are the two clickable buttons mentioned previously - Sign Up and Gmail. Beneath these buttons is a hyperlink to the Login Page, in case the user mistakenly clicks the wrong button. If the user registers correctly or uses Gmail to sign up, the website adds their information to the database and automatically redirects them to the Home Page.

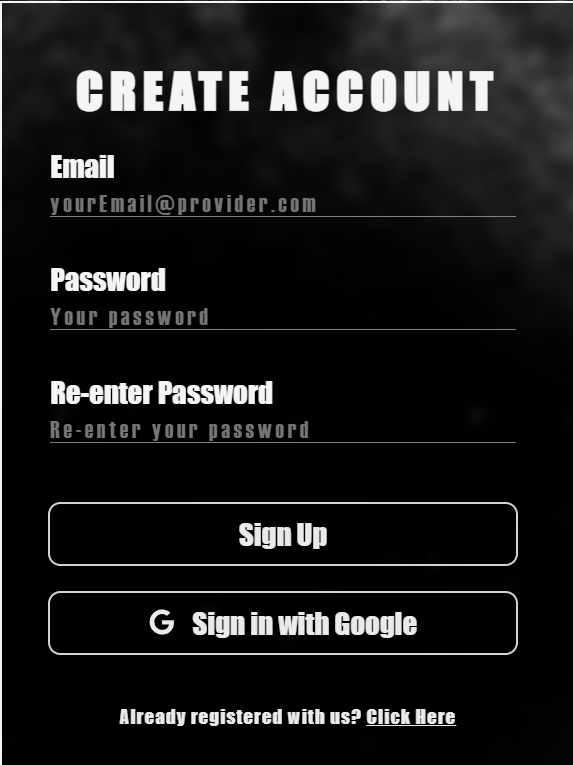


Figure 5: Create Account Page

When a user clicks on the Login button on the Landing Page, they are directed to the Login Page, which is similar in appearance to the Sign Up Page, but is designed specifically for user login. The Login Page includes the LoginSystem.js component, which functions as a form and contains two fields for users to enter their login information: Username and Password. Users can also log in using their Gmail account. Like the Sign Up Page, the Login Page includes two clickable buttons: Login and Sign in with Gmail. Additionally, there is a hyperlink to the Sign Up Page below the buttons. If the user enters the correct login information, they are automatically redirected to the Home Page.

The Home Page is the main hub of V-Closet. The home page serves as the landing page once the user is authenticated. The user can see the recommended clothing items that have been uploaded to V-Closet. The Home Page includes the Recommender.js components. The home page is meant to serve the user with a recommendation for that day based off of the weather in the location the user sets. Some other components that include TopNavBar.js, ShapeContainer.js, and CheckAuthentication.js that will persist throughout the other pages.

The Recommender component is responsible for providing the user their daily outfit recommendation based on the current weather conditions. This component works by checking if weather data is available, building an array of clothes from the user’s wardrobe based on the temperature range, searching the description for keywords that pertain to precipitation, and then calling a function to display each clothing item. We developed a simple decision flowchart to guide this feature, although we are considering adding several more decision layers based on comments from the Innovation Experience (Figure 6).

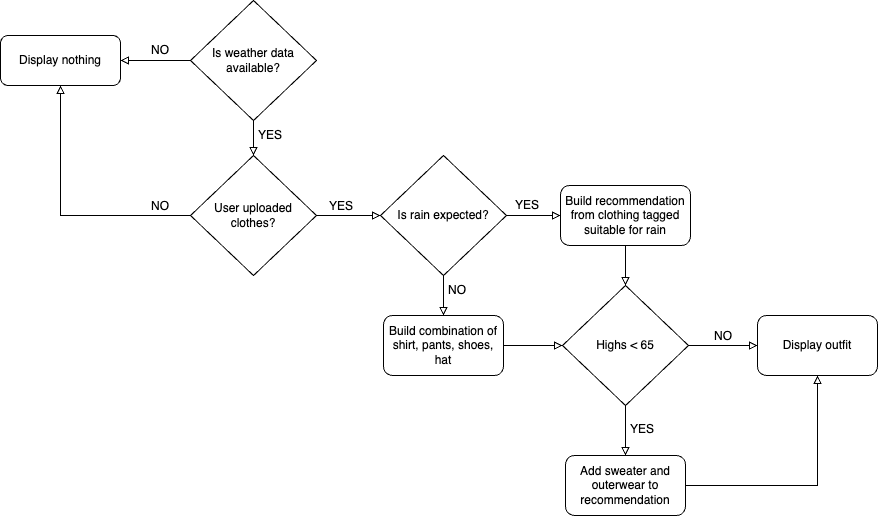


Figure 6: Clothing recommendation flowchart

One of the key components of the V-Closet project is the TopNavBar, which provides users with easy access to other pages within the website. This component includes five clickable icons: home, wardrobe, shop, dark and light mode toggle, and gear. Clicking on the home icon will direct users to the Home Page, regardless of their current location on the website. The wardrobe icon will take users to their personal wardrobe page, while the shop icon will direct them to the marketplace page. The toggle icon toggles the theme of the website dark or light theme that will persist throughout the application. The gear icon provides access to the Settings Page. Additionally, located next to the home icon is a weather display that reflects the user's preferred weather settings. This feature is designed to provide users with real-time weather information as they browse the website.

The CheckAuthentication component is responsible for verifying whether a user is logged into the V-Closet website before granting access to its features. This component renders every time a user attempts to navigate to a new page. To minimize the amount of content visible to non-logged-in users, a loading animation is displayed until authentication is confirmed. It is important to note that there is currently a bug related to this feature that is still being addressed.

The ShapeContainer component is designed purely for display purposes, and it is used to provide the website with an upside-down triangle shape in certain pages. This component was created to work in conjunction with the ToggleMode feature, as aligning the shape with the theme was initially difficult.

The Wardrobe Page is designed to display a user's inventory of clothing items, and it includes the SideBar.js and QueryClothes.js components. The SideBar component functions as a navigation bar, offering users various categories to sort their inventory. These categories may include shirts, pants, headwear, and sweaters, among others. When a user clicks on a category, the corresponding value is sent to the QueryClothes component, which will search for any clothing items that match the selected category and create an array to display those items. The ClothingDisplay.js component is responsible for displaying the clothing items on the page and allowing users to interact with them. When a user clicks on an item's image, they are presented with two options: delete the item or list it on the marketplace. It is important to note that these features are currently in development and may not be fully functional by the deadline.

The Add Item Page is where users can upload their clothing items before they can interact with them (Figure 7). It includes the AddClothes.js component which allows users to add their clothing item details to the database. Users can upload an image of the item, select a category, color, suitability, and add a description. The suitability option is a boolean that allows the system to recommend clothing suitable for specific weather conditions like rain, hot or cold. This recommendation is done through the Recommender.js component. The image uploading process is handled by a separate component called ImageUploader.js.



Figure 7. Adding a clothing item

The Market Page is the least developed among all the other pages. Currently, it is only set to display a loading animation, and there is no functionality available for users to interact with. It is possible that further development may be required in the future to fully implement this page's features.

The Settings Page allows users to customize their account, location, and access information about V-Closet. The page is designed with a tab layout to provide a modern and practical user experience. The first tab, Account, enables users to change their nickname and delete their account. However, a password changing feature is still under development. The second tab, Location, allows users to update their current or preferred location. The third tab, About, provides information about V-Closet's mission, vision, and history for curious users.

Finally, the last tab, Log Out, logs the user out of the website. So far, we've covered the various components and pages of the V-Closet website from a user's perspective. Now, let's delve into the technical details of what happens behind the scenes to make the website function properly.

Authentication.js defines functions to be called when a user is attempting to login or sign in. There are four functions defined: signUpWithEmailAndPassword, googleSignIn, signInWithEmailAndPassword, and logout. signUpWithEmailAndPassword handles registering a user with V-Closet. The function takes in a username and password that are then recorded in Firebase’s Authentication service using an API. A unique user ID is given to each new user. The googleSignIn function uses an OAuth API to sign in users to V-Closet using their pre-existing Google account. If it is their first time signing in with Google, a user is created by the Authentication service and is given a unique user ID. The signInWithEmailAndPassword function takes in a username and password and checks them against the existing users using an API. All three of these functions, upon success, update the authentication state of the user to signal that they are authenticated. The final function, logout, changes the auth state of the user to unauthenticated using an API.

Firebase.js configures Firebase using their Software Development Kit(SDK). The SDK initializes the Firebase configuration with our unique apiKey: “AIzaSyB8kRGwhXfMUspInMbSdyAyq407FTx3CfQ”, authDomain: "v-closet-f9736.firebaseapp.com", projectId: "v-closet-f9736", storageBucket: "v-closet-f9736.appspot.com", messagingSenderId: "482526602179", appId: "1:482526602179:web:5698a2afbd93bee85dd376", and measurementId: "G-T8YZENSRRP". Firebase.js also initializes 3 additional SDKs: Firestore, Storage, and Authentication. The Firebase.js file is essential to establishing the connection to our Firebase project and initializing all the important services we need.

Firestore.js defines the functions that send and retrieve queries to our Firestore Database. The functions defined are addClothingItem, getClothingItem, getClothingItemsForHomepage, deleteClothingItem, addToMarket, and getAllMarketItems. The function addClothingItem adds a clothing item to Firestore in a collection labeled after the user’s unique ID. It does this by taking in the clothing item information as parameters and passes them in an API. The function getClothingItem handles the process of querying Firestore for clothing items of a selected category. The function takes in the user ID and category to retrieve the right clothes, and two useState functions setClothingItems and setIsLoadingClothes. A snapshot is taken of the query and all documents in the snapshot are stored in an array. The setClothingItems sets the clothingItems state on the wardrobe page to the array while the setIsLoadingClothes sets isLoadingClothes on the wardrobe page to false. The function getClothingItemsForHomepage does much of the same as getClothingItem but instead takes all clothing items and stores them in an array in accordance with its category. These arrays are then set as the useState variables on the home page using the useState functions passed as parameters. The isLoadingClothes variable is set as false on the home page as well. The function deleteClothingItem is called when a user clicks the delete button on any of their clothing items in the wardrobe page. The function deletes the item by passing the user ID and document ID to the delete API. The function addToMarket sets the boolean ‘isListed’ of the relevant clothing item to true so that it will be listed on the marketplace. The function getAllMarketItems acts similarly to getClothingItems but instead queries for items with the ‘isListed’ boolean as true.

Storage.js has two functions called uploadImageToStorage and getStorageDownloadURL. The function uploadImageToStorage handles the upload of images to the Firebase Storage service. An upload task begins and ensures that either all of the data is stored or none at all. The path to the image in storage is stored in the image’s clothing item document in Firestore.The function getStorageDownloadURL is used to display the image. It takes the pathname as a parameter.

**How to deploy or build our application**

Users wishing to set up their V-Closet website will have to first set up a firebase project of their own. Go to the Firebase Console website: <https://console.firebase.google.com/> and sign in to your Google account or create a new one. Click on the "Add project" button to create a new Firebase project. Enter a project name and select your country/region from the drop-down menu.

Click on the "Create project" button to create your project. Once your project is created, you will be taken to the Firebase project dashboard. From the dashboard, you can choose which Firebase services you want to use for your project. For setting up V-Closet, you will need to enable Authentication, Firestore, Storage, and Hosting. Firebase will walk you through setting each of these up. The access rules for Firestore and Storage will need to be set up as well. Figures 8 and 9 show what the rules should look like for Firestore and Storage respectively.

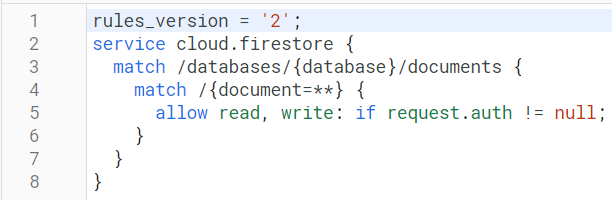


Figure 8: Firestore Database Rules

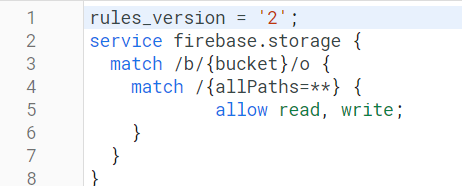


Figure 9: Storage Firebase Rules

After you have finished setting up the firebase project and each required service, head to the V-Closet github at <https://github.com/Scott-projects/V-Closet>. Download a zip of the main branch by clicking the green “< > Code” button, and then extract it. Next. the IDE will need to be set up. Download visual studio code at <https://code.visualstudio.com/> and node.js at <https://nodejs.org/en>. Open the extracted folder in Visual Studio Code and run the command ‘npm install’ in the terminal. While the necessary packages are being installed, open the firebase.js file and change the firebase config object so that it contains your api key and other relevant information instead of ours. This can be found in your Firebase project’s settings and is shown in figure 10.

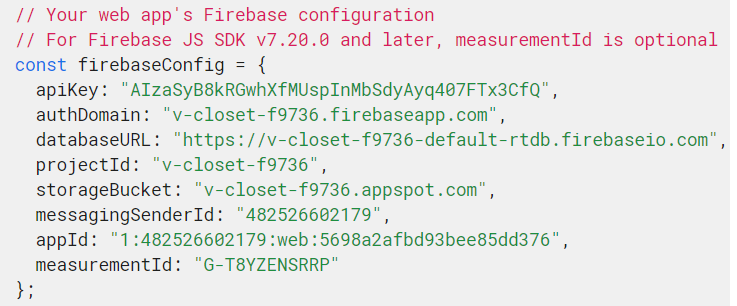


Figure 10: Firebase Config Information

Finally, we can start to deploy the project. First off, run the command ‘npm run build’. This will create a build folder which will be the code that should be deployed to hosting. Next, login and initialize the Visual Studio Code project folder as a Firebase project. Type the command ‘firebase login’ to login to your google account associated with your Firebase project. Next, type the command ‘firebase init’. This will take you through the initialization process. Make sure you check all the necessary services. The deploy directory should be specified as the “build” folder. Finally, run the command ‘firebase deploy’ to have firebase host the build that was created in the build folder.

**Known bugs**

Currently, we have identified a bug wherein if a user enters a link, such as the homepage, while not logged in, the page may momentarily display. Although the root cause of this issue is unclear, we have managed to mitigate the problem by incorporating a loading page animation to reduce the time it takes for the user to access the webpage.

A minor issue that has been identified is a result of using localhost for an extended period. Some of the test accounts that were created no longer function once we migrated to hosting. The resolution for this problem is straightforward and involves removing all prior users, collections, and Storage files.

We have identified another bug where the logout button occasionally requires two clicks to function correctly. Unfortunately, the root cause of this issue is currently unknown to us.

**Future Work**

In version 1 of our website, we focused on meeting our basic needs and building a solid foundation. We wanted to ensure that users could easily upload their clothing items, display them, and receive personalized recommendations based on their preferences. While we were able to achieve these goals, there were a few features that we had to cut due to time constraints. However, for version 2, we have a clear vision of what we want to accomplish and are excited to implement some of our long-awaited ideas.

Our top priorities for version 2 include enabling users to delete their clothing items and list them in the marketplace. We understand the importance of being able to remove unwanted items and want to ensure that our users have the ability to do so. Additionally, we promised to include a marketplace in our initial release, but were unable to deliver on this promise. This time around, we are determined to make it a reality and give users the opportunity to buy and sell clothing items with ease.

Another area we want to focus on is improving the security of our website. While users can currently create an account with an email and password, we want to implement measures to prevent scammers and bots from accessing the platform. This includes implementing email verification and allowing users to change their password. Additionally, we want to give users the ability to delete their account if they choose to do so.

A major upgrade we want to make in version 2 is to increase the social media aspect of our website. Our initial idea was to create a platform where users can connect and engage with each other, but we had to put this on hold for version 1. In version 2, we want to add a profile feature where users can upload their profile picture, showcase their clothing items, and create posts. Additionally, we want to implement a followers/following system to allow users to keep in touch and feel more connected. Adding a likes and comments system is also on our to-do list.

Finally, we want to add an event calendar feature to our website. This would allow users to keep track of special occasions like weddings, birthdays, and other events where they may need to dress up. By including this feature, we hope to make it easier for users to plan their outfits and stay on top of their fashion game.

Overall, we have a clear roadmap for version 2 of our website and are excited to see our vision come to life. We believe that by implementing these features, we can create a more engaging and user-friendly platform that will meet the needs of our users.

**Sources**

V-Closet: <https://v-closet-f9736.web.app/>

Github: <https://github.com/Scott-projects/V-Closet>

Learning React: <https://react.dev>

Firebase: <https://firebase.google.com>

OpenWeatherMap API: <https://openweathermap.org>

JavaScript: <https://www.w3schools.com/js/>

Visual Studio Code: <https://code.visualstudio.com/>

node.js: <https://nodejs.org/en>

**Acknowledgment**

We would like to thank Dr. Karen Meisch for her support of students in the College of Science, Technology, Engineering & Mathematics, and Dr. Leong Lee for his support of students in the Department of Computer Science and Information Technology. We would also like to thank Dr. Nicholson for advising us throughout the development process.