

Software Design Document

Section 1 - Project Description

1.1 Project

UniSell

1.2 Description

The UniSell software functions by allowing Fresno State students to list their textbooks for sale, search for specific course materials, communicate with peers, and ensure accessibility for all users while offering a user-friendly interface and cross-device compatibility. Some features may include chatrooms, search by filtering courses/instructors, and a friendly user interface that promotes a satisfying experience.

1.3 Revision History

| Date | Comment | Author |
|------------|--|--|
| 10/08/2023 | Initial additions | UniSell |
| 10/31/2023 | Midpoint update | UniSell |
| 11/16/23 | QA Inputs, | Alan Vang, |
| 11/17/23 | Updated navigation flow and activity diagrams, also updated section 7.1 details regarding the UI design, requirements, More QA Inputs and wireframe, interface | Grace Salazar, Alan Vang, Izaiah Benavides, Nathan |
| 12/3/2023 | Inserted time estimates in hours | Izaiah Benavides |

Contents

[Software Design Document Section 1 - Project Description](#)

[1.1 Project](#)

[1.2 Description](#)

[1.3 Revision History](#)

[Section 2 - Overview](#)

[2.1 Purpose](#)

[2.2 Scope](#)

[2.3 Requirements](#)

[2.3.1 Estimates](#)

[2.3.2 Traceability Matrix](#)

[Section 3 - System Architecture](#)

[Section 4 - Data Dictionary](#)

[Section 5 - Software Domain Design](#)

[5.1 Software Application Domain Chart](#)

[Section 6 – Data Design](#)

[6.1 Persistent/Static Data](#)

[6.1.1 Dataset](#)

[6.1.2 Static Data](#)

[6.1.3 Persisted data](#)

[6.2 Transient/Dynamic Data](#)

[6.3 External Interface Data](#)

[6.4 Transformation of Data](#)

[Section 7 - User Interface Design](#)

[7.1 User Interface Design Overview](#)

[7.1.1 Design Philosophy](#)

[7.1.2 Minimum UI Requirements](#)

[7.1.3 Wireframe](#)

[7.2 User Interface Navigation Flow](#)

[7.2.1 Navigation Flow Diagram](#)

[7.2.2 User Journey Navigation Flow](#)

[7.3 Use Cases / User Function Description](#)

[Section 8 - UML Diagrams](#)

[8.1 Class Diagram](#)

[8.2 Sequence Diagram](#)

[8.3 Activity Diagrams](#)

[Section 9 - Other Interfaces](#)

[9.1 Firebase](#)

[9.2 Android Studio](#)

[Section 10 - Extra Design Features / Outstanding Issues](#)

[10.1 Outstanding Issues](#)

[Section 11 – References](#)

[Section 12 – Glossary](#)

Section 2 - Overview

2.1 Purpose

Higher education comes with a cost. With the increase in textbook pricing as well as tuition, according to the National Center of Education Statistics, the average cost of books and academic supplies for students in 2020 – 2021 was 1,226 dollars per year. According to the National Retail Federation, students also spend roughly another 1,200 dollars per year on dorm supplies and electronics. Yet the majority of these supplies and books are usable for 1 – 2 semesters before students upgrade or pass their classes. Nowadays, students usually sell their materials and supplies to fellow students, friends, and family at a lower cost. However, this is usually done by word of mouth or through social media, which can be inefficient when it comes to selling items promptly. This project proposes a mobile application platform that streamlines and centralizes the selling and purchasing power strictly for Fresno State students at certain universities.

The purpose of UniSell is to simplify Fresno State Student's experiences of buying and selling college materials. Students who seek affordable academic resources will use this platform to find supplies for any course. Sellers will be able to create listings with photos and descriptions to reach potential buyers within campus. UniSell aims to have an interface where students can easily manage their school equipment.

Fresno State students will benefit from this as they can buy used textbooks, materials, and supplies from local students so they can reduce shipping and traveling costs and find specific textbooks for certain classes. The success of the project will be determined by a working prototype on a local machine with 2-3 features allowing students to upload contact information and images. Fresno State could later officially improve and continue the project to upload the application for all students, building a stronger community at Fresno State.

2.2 Scope

UniSell offers numerous benefits and primarily aims to reduce the financial burden of a college education. The objective is to create a user-friendly application that streamlines the buying and selling of college and academic materials. This will make it easier for students to find and sell secondhand textbooks and supplies. The project's goal includes building a sense of community amongst students and expanding the application's reach campus-wide to promote affordable and accessible resources for students.

2.3 Requirements

Design Requirements:

- User-Friendly Interface: The application must have an intuitive and easy-to-navigate user interface to ensure that both buyers and sellers can use it effectively without extensive training.
- Responsive Design: UniSell should be accessible and functional on a variety of devices, including smartphones and tablets, to accommodate the diverse preferences of its student users.
- Search and Filtering: The software must include robust search and filtering capabilities, allowing users to find specific items efficiently by course, category, price, or location.
- User Accounts: A system for creating and managing user accounts with features like profile customization, transaction history, and messaging functionality must be integrated.
- Listing Creation: Sellers should be able to create detailed listings for their items, including photos, descriptions, prices, and contact information.
- Item Tagging: The ability to tag listings with relevant keywords and categories to enhance search results and organization is essential.
- Security: Strong security measures should be in place to protect user data, including personal information and transaction records.

- Scalability: The design should allow for future scalability to accommodate a growing user base and additional features.

Graphics Requirements:

- User-Friendly Visual Design: The application should employ an appealing and user-friendly visual design with clear icons, buttons, and intuitive layouts.
- Image Upload: The software must support image uploads for item listings, ensuring that users can provide clear and informative visuals of their products.
- Logo and Branding: Incorporation of a UniSell logo and branding elements to establish a consistent and recognizable visual identity.

Operating System Requirements:

- UniSell should be compatible with Android
- Minimum OS Versions: The application must specify minimum supported OS versions to ensure a consistent and stable user experience.

2.3.1 Estimates

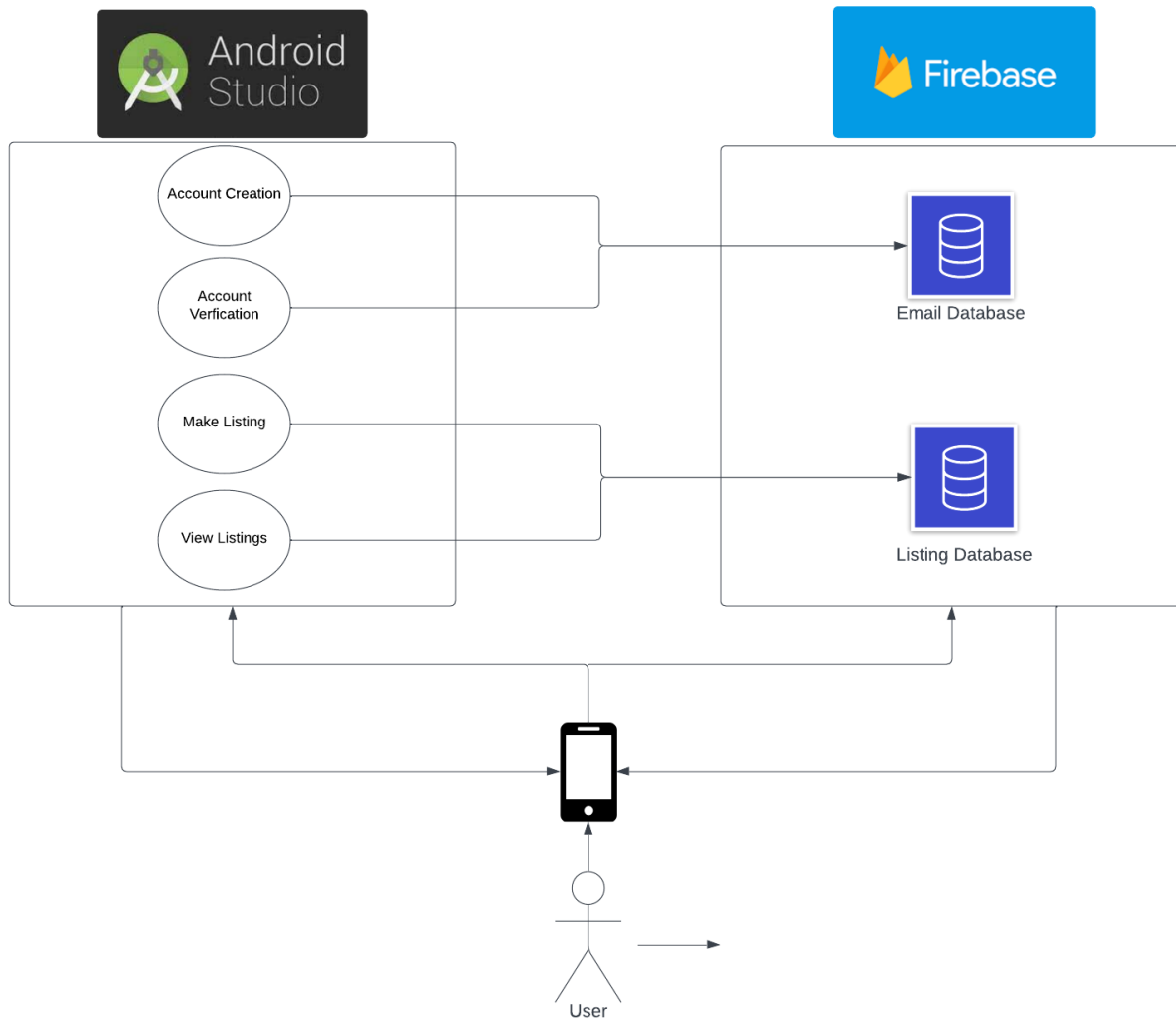
| # | Description | Hrs. Est. |
|---|---|-----------|
| 1 | Setting up Github Repository with all the branches | 1 hr |
| 2 | Welcome Screen/Login/Sign up screen/Firebase Auth | 4 hrs |
| 3 | Create a Listing Screen | 5 hrs |
| 4 | View Listings (Home screen) | 2 hrs |
| 5 | Search Functionality for all screens | 1 hr |
| 6 | Categories screens with backtracking | 4 hrs |
| 7 | Description Screen and backtracing for each item and category | 6 hrs |
| 8 | Profile screen and Icon updating | 3 hrs |
| | TOTAL: | 26 hrs |

2.3.2 Traceability Matrix

Cross reference this document with your requirements document and link where you satisfy each requirement

| SRS Requirement | SDD Module |
|-----------------|--------------------------------------|
| Req 1 | 5.1.1 (link to module), 5.1.2 (link) |
| | |
| | |

Section 3 - System Architecture



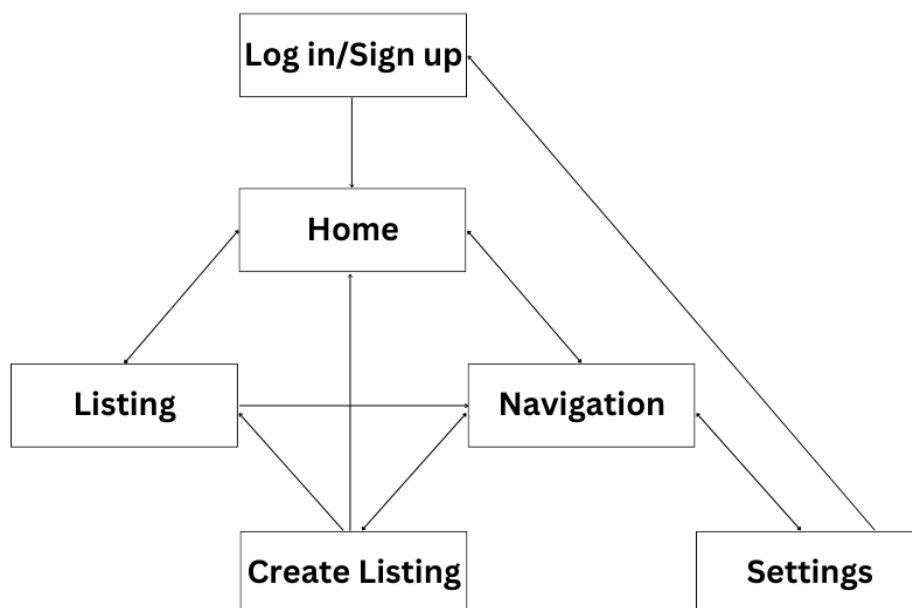
Section 4 - Data Dictionary

| User | | |
|----------|-----------------------------|--------|
| Field | Notes | Type |
| UID | Unique identifier for Table | UID |
| email | The email of user | string |
| name | Account name of user | string |
| password | Password of user | string |
| phone | Phone number of user | string |

| Listing | | |
|-------------|---|--------|
| Field | Notes | Type |
| UID | Unique identifier for Table | UID |
| category | Category that an item falls under, for filtering purposes | string |
| date | Date listing was created | string |
| description | The description of the listing the user gives | string |
| img_url | The image of the product the user wants to sell | string |
| price | Price of the product | string |
| time | Time the listing was created | string |
| title | Title of the product, what the product is | string |
| userName | The user-name of the user who posted the listing | string |
| userPhone | The user's phone number so a buyer can contact a seller | string |

Section 5 - Software Domain Design

5.1 Software Application Domain Chart



Section 6 – Data Design

The data that is needed for the project account data and listing data. We need data, email, and password data to determine that the account is valid. Items being sold in the app are going to be stored with numerous fields. You would need to have the location, price, name of the item, image, and who is selling the item to have a listing

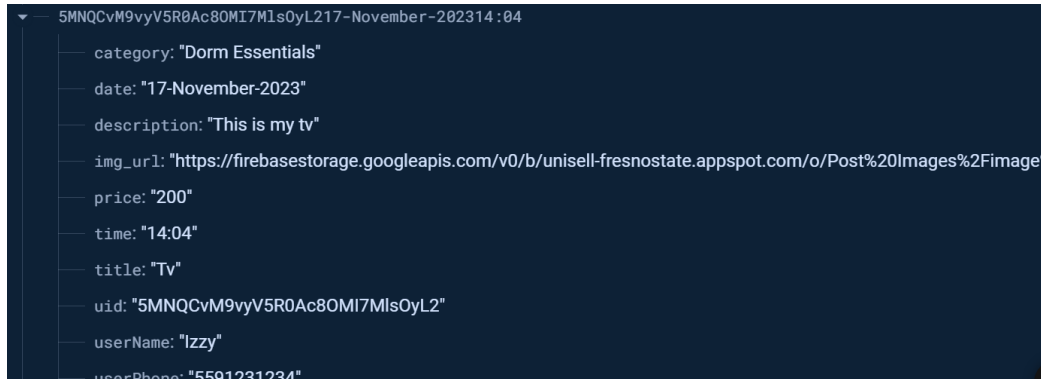


Figure 6.1 Reference to a specific listing in the real-time database

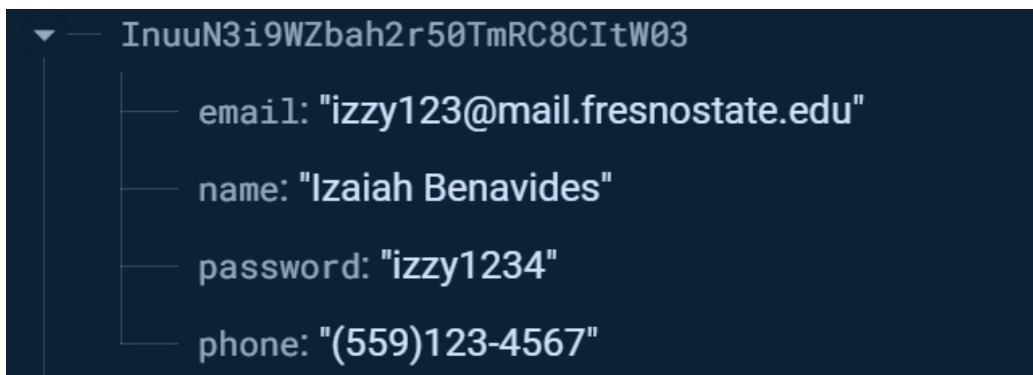


Figure 6.2 Reference to a specific user in the authentication database







| <input type="checkbox"/> | Name | Size | Type | Last modified |
|--------------------------|--|-----------|------------|---------------|
| <input type="checkbox"/> | raw/ | — | Folder | — |
| <input type="checkbox"/> |  164403769516-November-202317:34.png | 141.95 KB | image/jpeg | Nov 16, 2023 |
| <input type="checkbox"/> |  image:1251916-November-202316:40.png | 327.85 KB | image/jpeg | Nov 16, 2023 |
| <input type="checkbox"/> |  image:2715-November-202300:41.png | 141.95 KB | image/jpeg | Nov 15, 2023 |
| <input type="checkbox"/> |  image:2715-November-202300:44.png | 141.95 KB | image/jpeg | Nov 15, 2023 |
| <input type="checkbox"/> |  image:2717-November-202310:21.png | 141.95 KB | image/jpeg | Nov 17, 2023 |
| <input type="checkbox"/> |  image:2717-November-202310:29.png | 141.95 KB | image/jpeg | Nov 17, 2023 |

Figure 6.3 Reference to the images stored by users in the storage database

6.1 Persistent/Static Data

The Data Model we are going to use is a model where we assign the post to an account so it is easy to identify which account posted each listing. For example, a post will need to be linked to the account that posted the selling item. It would need the name of the item, the price of the item, and the location the item will be transferred to the buyer. The name of the item, price of the item, and the location will be with the post itself but the account will need the name and phone number attached to the post to know who it is.

6.1.1 Dataset

How we will connect data to other data is by referencing them. A post references the account to show that this is the person who made the listing.

6.1.2 Static Data

The static data would be the Unique ID that the account and post have which will always be consistent. This ID cannot be changed by anyone.

6.1.3 Persisted data

The persistent data would be the name, prices, location, etc on posts or accounts that are being used. The reason is that sometimes we make mistakes so we would like to edit the post without making new posts.

6.2 Transient/Dynamic Data

The dynamic data is the time the post has been posted. This dynamic data will always be updated so clients can see how old the post is and see if the item is still there or not. This is done through Firebase as it can make real-time updates when new accounts are made or authenticated as well as when new listings are made and called from the view listing use case.

6.3 External Interface Data

We use the API of FireBase to send our data from one user to another. Listing data is stored in a database so that users can view other user's listings.

6.4 Transformation of Data

If we need to add an item to the database, we will have to make a form that has all the necessary data to be added. Once it fulfills the requirements, we send that data to the database and this will then post it into the home screen.

Section 7 - User Interface Design

7.1 User Interface Design Overview

7.1.1 Design Philosophy

Our design philosophy for the UniSell app centers on simplicity and ease of use. We aim to develop an interface that is straightforward and intuitive, avoiding unnecessary complexity. Each design aspect is carefully considered to eliminate ambiguity, allowing users to navigate and interact without confusion. We prioritize clear, readable text and visuals to enhance accessibility for a diverse user base. The journey a user takes to perform any task on the app should be direct and uncomplicated. Overall, our design blends aesthetic simplicity and functional clarity, meeting user needs without overwhelming them.

with excessive details.

7.1.2 Minimum UI Requirements

Eight pages

- **Welcome**
 - Login and register buttons
 - UniSell logo/name displayed
- **Log in**
 - Email and password input fields
 - Login button
- **Sign up**
 - Email, password, phone, and account name input fields
 - Sign up button
- **Home**
 - Search bar for users to enter and submit text
 - Search button
 - Photo grid displaying listing photos
 - OR, buttons to various pages such “recent listings” or “recommended listings” that will display photo grid of listings
 - UniSell logo/name displayed
- **Search results**
 - User input search text displayed
 - Photo grid of resulting listings
- **Create listing**
 - Title text field
 - Upload/take a photo to submit
 - Description text field
 - Category selection menu
 - Drop down menu OR some way of displaying
 - Price input field
 - Post button
- **View listing**
 - Display:
 - Title text
 - Category
 - Photo
 - Price
 - Description
 - Contact information
- **Profile**
 - Display account information
 - Email/username
 - Logout button
- Constant access to navigation menu
 - Home button
 - Make Listing button
 - Profile button
- Vertical scrolling on pages

- OR, numbered pages
- Optional pages
 - Recent listings
 - Recommended listings
 - Categories
 - Favorite listings
 - Mission statement page

7.1.3 Wireframe

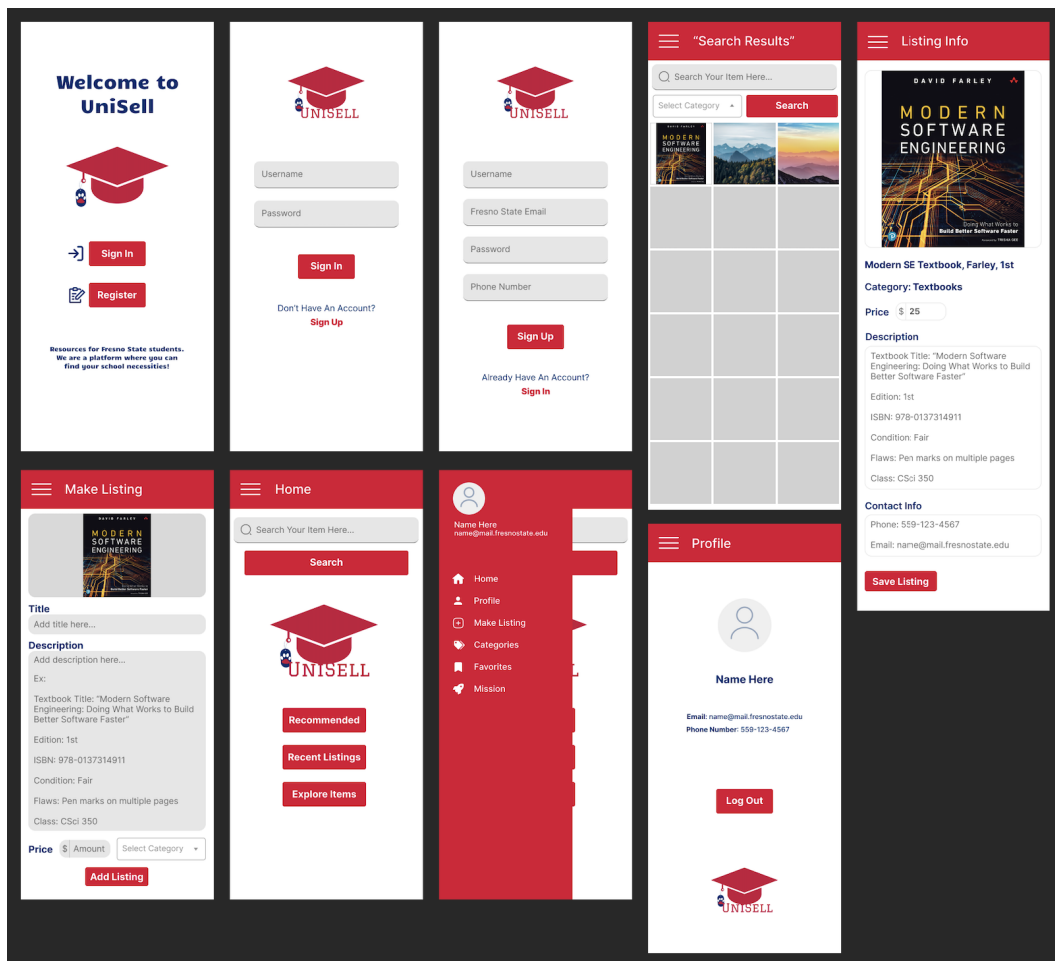
Low/Mid-Fidelity:



1st frame: Registration page, 2nd frame: Login page, 3rd frame: Listing page, 4th frame: Create Listing, 5th frame: View Listing, 6th frame: Logout, 7th frame: Menu

The main focus for the low-fidelity wireframe is to plan and illustrate the general UI design. In the image above, the text input fields, such as “username” and “password” are indicated on the necessary frames. The buttons and text to be displayed are also indicated, as well as any other additional elements.

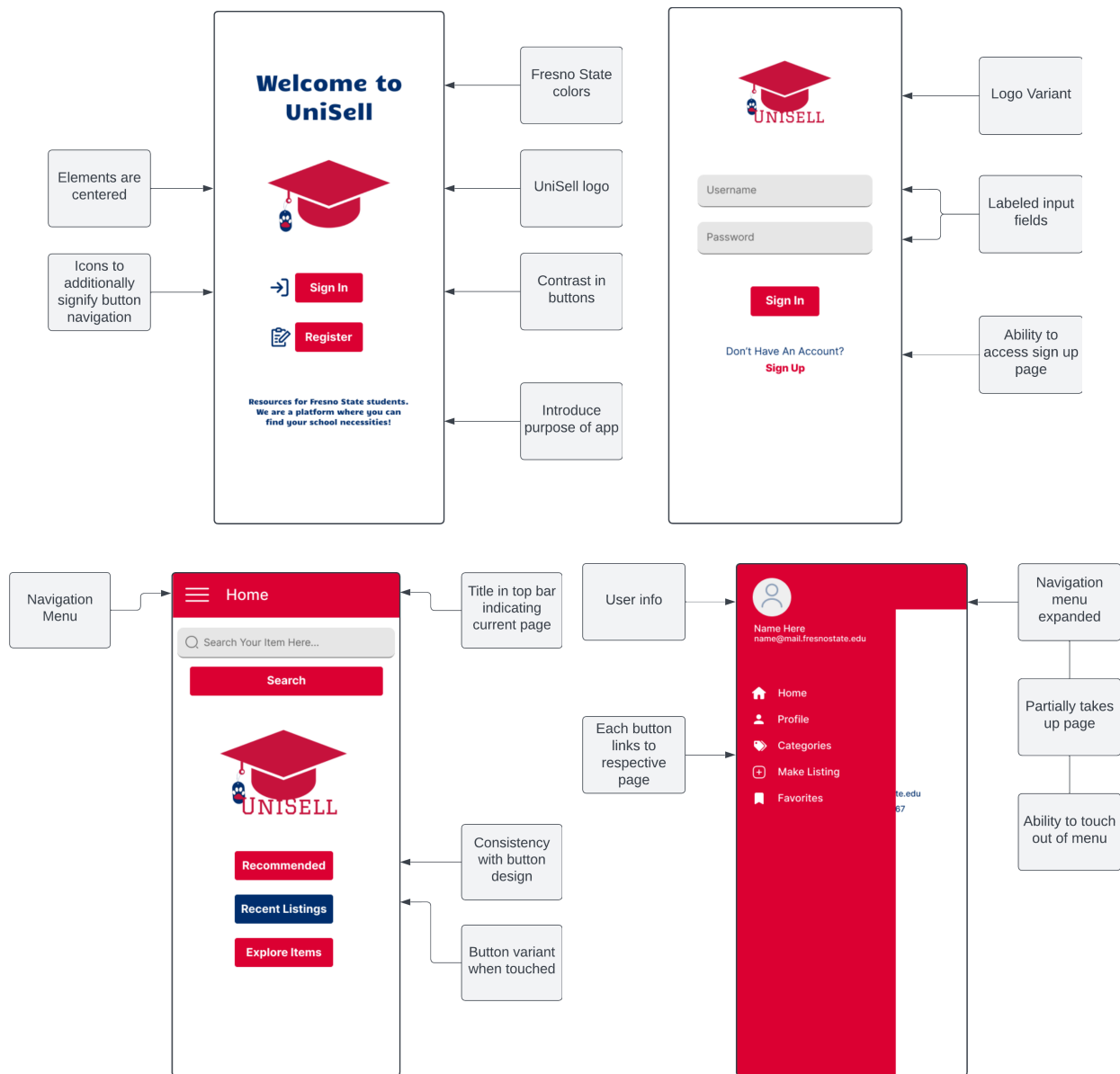
High-Fidelity, from Figma Prototype:

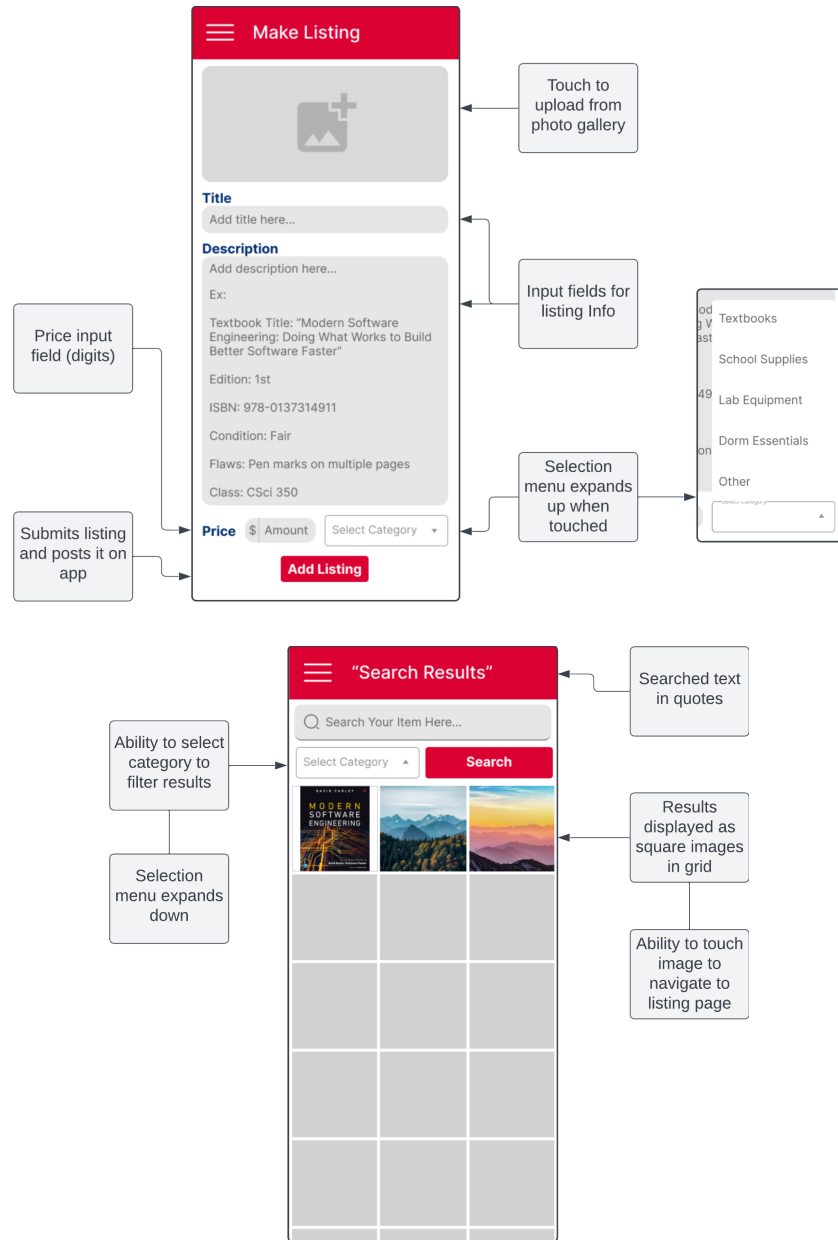


(Left-right, top-bottom), 1st frame: Welcome page, 2nd frame: Login page, 3rd frame: Sign-up page, 4th frame: Search results page, 5th frame: View listing page, 6th frame: Make listing page, 7th frame: Home page, 8th frame: Home page w/ navigation menu open, 9th frame: Profile page

The high-fidelity wireframe is more detailed than the previous one and is interactive. The same things are indicated on each frame as before, but expands into additional frames and is more specific with styled elements. The Fresno State color palette was used since the app serves Fresno State students. The navigation flow diagram in section 7.2.1 is consistent with the Figma prototype. The description in section 7.2.2 illustrates the navigation flow based on interaction with the UI for different user journeys. A link to the interactive prototype is placed in the [References](#) section.

7.1.3.1 Figma UI Design Details



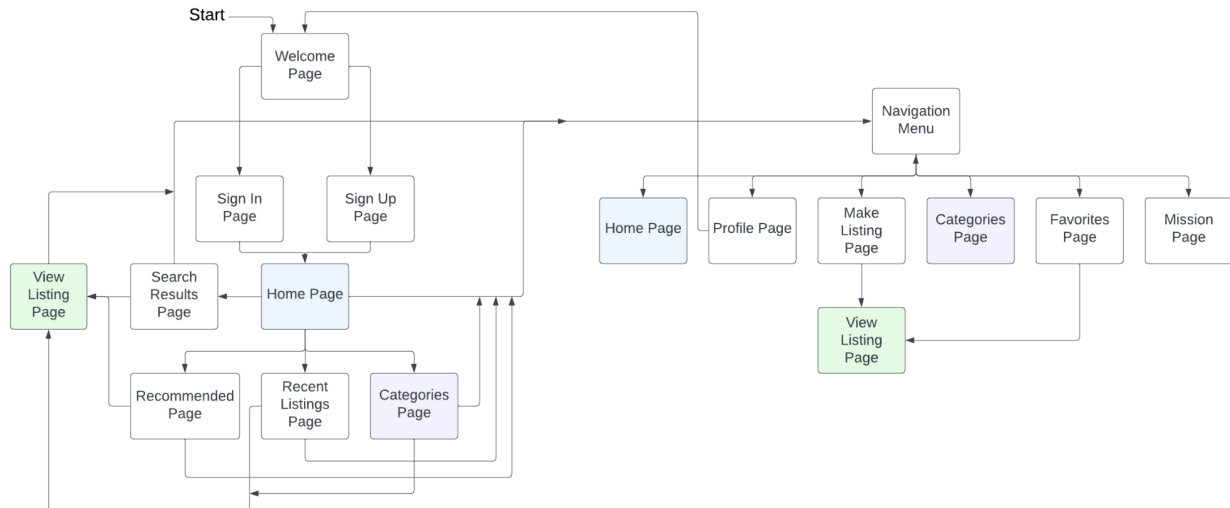


Here are some design elements of the prototype in detail.

| Color Scheme (Hex) | Icons |
|---|---|
| White: #FFFFFF Red: #DB0032 Blue: #002E6D | https://iconduck.com/ |

7.2 User Interface Navigation Flow

7.2.1 Navigation Flow Diagram

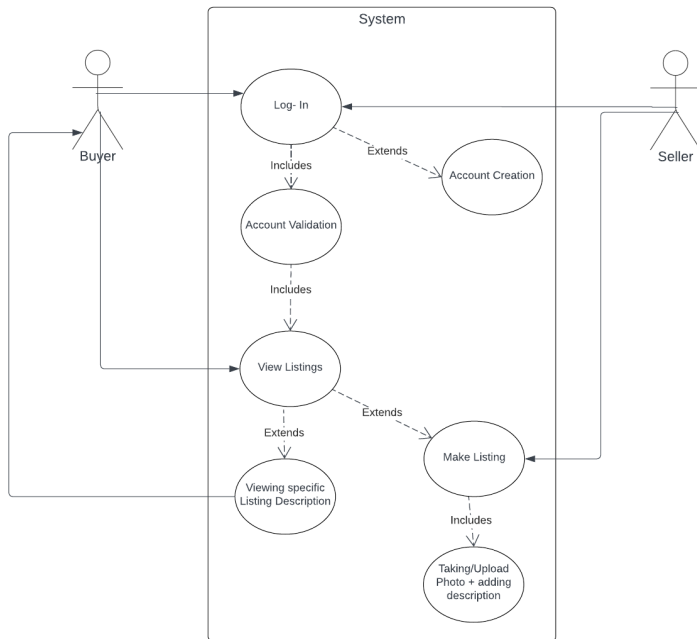


Note: The color-filled blocks indicate the page flow from that page was already defined. For example, the flow from the “Home Page” was already defined on the left side of the diagram, but appears again on the right side.

7.2.2 User Journey Navigation Flow

- The user opens the UniSell Application and first goes to the login screen. The user has three choices to navigate.
 - The User can log into an already existing account to go to the home screen
 - The User can sign up for a new account and go to the home screen
 - The User can go to the settings to what the user prefers his application to do in certain cases.
- The user goes to the home screen.
 - The user can look in the listings to find deals they might want to buy from other students.
 - The user can go look for specific listings the user is looking for.
 - The user can create a brand new listing selling an item for others to buy in cash.
- The user goes to the listings
 - The user can go back to the home screen
 - The user can't find what they are looking for so they go to the navigation to see if anyone is selling what they are looking for.
- The User goes to the navigation menu
 - The user goes to create a listing to sell their books to other students
 - The user can go to settings for more options or log out.
- The User goes to the create listings
 - This listing goes to both the listings for other users to see.
 - The user goes back to the navigation menu to cancel the listing

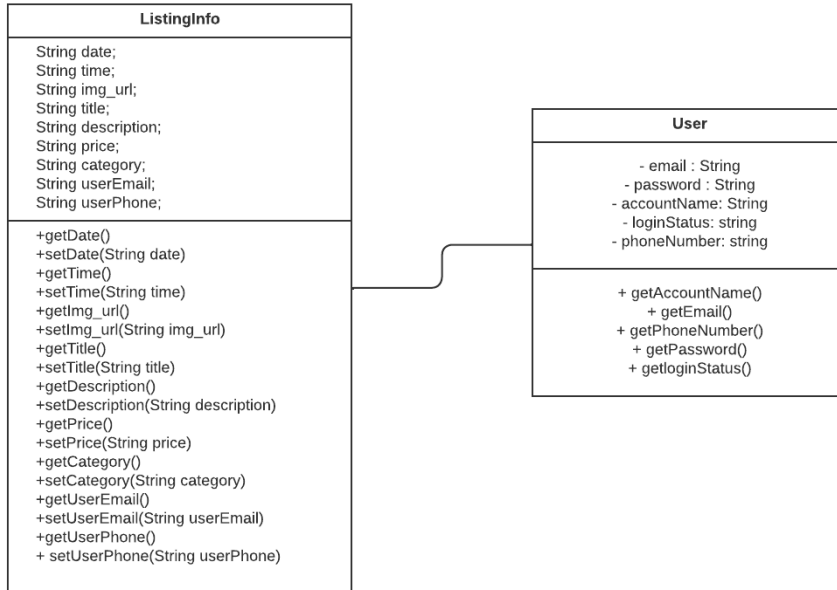
7.3 Use Cases / User Function Description



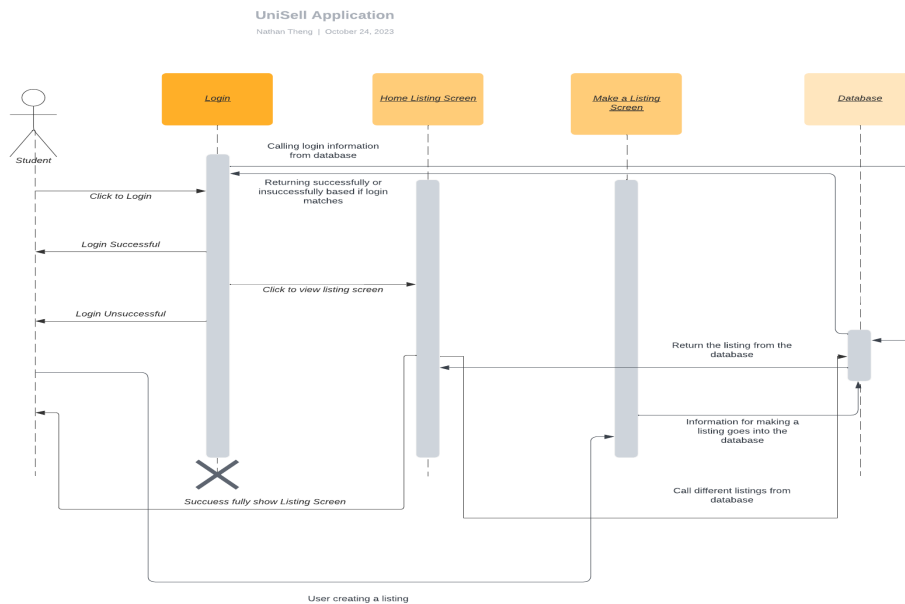
- Student is logging in
 - Enter username and password
 - Authenticates if a user has an account
- Student is browsing listings
 - Scroll through the home page
 - Click a listing photo and view listing
 - Go back to the home page and repeat
- Student wants to create a listing
 - Click the plus button on navigation bar to create a listing
 - Enter the requested information in each field
 - Provide contact information (phone number/email) in respective field
 - Click the post button
 - View the listing just posted

Section 8 - UML Diagrams

8.1 Class Diagram



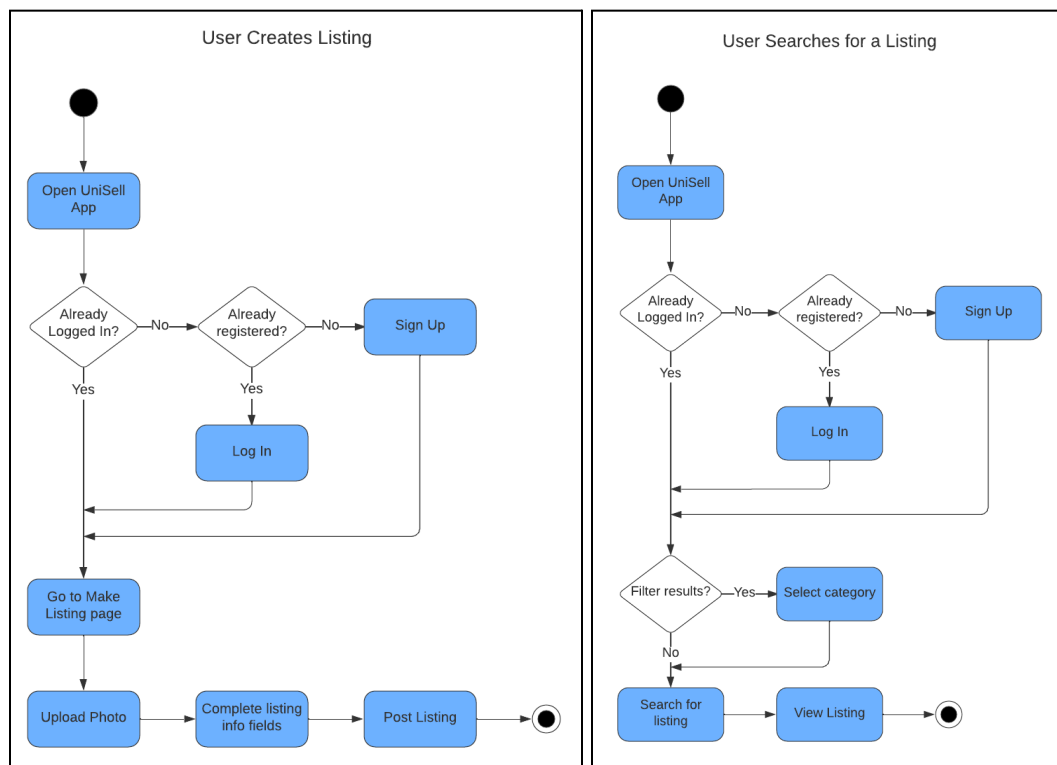
8.2 Sequence Diagram



- The user opens the UniSell Application and first goes to the login screen.

- The user inputs the details of their account information. This information goes to the database to see if this account information is correct.
 - The account information is right so we update the user to the home listing screen
 - The account information is wrong so we will have to stay in the login screen until there is valid account information.
- Assuming that we do get to the home listing screen we call the database again to give us listings
 - The user can browse through the listings and we constantly update the user with new listings as the user searches
 - The user can request a new listing
 - The user can log out the application
- The user requesting a new listing now has to send the proper information to the database.
 - This in turn will update other users that this new listing has been created for people to see.

8.3 Activity Diagrams



Section 9 - Other Interfaces

External interfaces such as Firebase and Android Studio play a critical role in the functionality of UniSell:

- **Firebase:** Firebase Is the chosen relational database management system for storing and managing data within the software. The database stores information such as user profiles, listing information such as descriptions and images
- **Android Studio:** Developer tool

9.1 Firebase

Firebase Authentication: Systems provide an easy-to-use and secure way to authenticate users which support various IDs such as email, password, social identity providers. Developers can utilize Firebase Authentication for UniSell's interface to create accounts, sign into accounts, and manage their accounts.

Firebase Realtime Database: This database is a NoSQL database that allows developers to store and sync up data in real-time. The mobile application can interact with this database through the Firebase SDK to read and write the data as well as receive real-time updates.

Cloud Firestore: This database is a scalable NoSQL database that provides a more powerful and flexible data model than the Realtime Database. It can do this as Firestore utilizes the Firestore SDK to perform operations such as create, read, update, and delete and will wait and listen for real-time updates.

Firebase Cloud Storage: This allows the developers to store and return user-generated content such as images, audio, and video files. Through the applications, these files are downloaded uploaded, and accessed securely through Firebase Authentication

Firebase Analytics: This feature of the Firebase interface helps the developers understand user behavior and app performance. This will allow the mobile application to log custom events, track user demographics, and utilize the analytics dashboard to gain further insights into how users interact with the application to improve it

Firebase Remote Config: Remote Configuration allows the developers and engineers to modify the appearance and behavior without publishing an app update. This will allow the application to fetch the specific configuration parameters from the Firebase console and update them to personalize the app for different user segments.

9.2 Android Studio

Android Emulator: Android Studio includes an emulator that lets the developers run the test application on a virtual device. This allows developers to see if their code and programming are aligned with the purpose of the

Android Debug Bridge(ADB): A command-line tool that facilitates communication between a developer's machine and Android device

Version Control Integration: Android Studio can integrate with version control systems like Git. By doing this it will commit changes, switch branches, and collaborate with team members on a shared codebase.

Firebase Integration: Android Studio has built-in support for integrating Firebase services into Android applications. By doing this it connects Firebase to the application without the use of API. The Firebase projects can be managed directly from Android Studio.

Section 10 - Extra Design Features / Outstanding Issues

When testing I first get the most recent GitHub Build and install the application on a FairPhone 4 that is Jailbroken and running on a custom version of Android based on Android 12. The reason for using a Jailbroken Android phone is so I can modify my phone to use different configurations of what other users might use. For example, maybe I can limit my phone to use 2 cores instead of the usual 8 core design to

simulate how the phone would perform in a low-end phone.

Once the application is installed, I sign in to an account that exists to find out what works, what changed, and what is currently broken from the current build.

10.1 Outstanding Issues

- General Issues
 - There is Currently No Functionality of...
 - Settings is just a button with no pages at all.
- Account Issues
 - Currently no checks to see if the email is real or not. So you can just make an account with xxxx@mail.fresnostate.edu for example and that would work.
- Small issues
 - Login Page input for accounts does not disable autocorrect or does not use the remember account autocorrect system. So when inputting for example GearsyFox@mail.fresnostate.edu, your phone could autocorrect to GearsyFox@mail.fresnostate.edu. Currently annoying but the account system does work.

Section 11 – References

Wireframe:

<https://www.figma.com/proto/iEwNIjCeLuLqOBvVY18hHz/UniSell?type=design&node-id=126-67&t=QTKJsHw2yro0Cdzl-1&scaling=scale-down&page-id=0%3A1&starting-point-node-id=126%3A67&mode=design>

Github Repository:

<https://github.com/CSCI150-LAB01/UniSell>

Section 12 – Glossary

API - Application Programming Interfaces

HTTP - HyperText Transfer Protocol Secure

ADB - Android Debug Bridge

SDK - Software Development Kit

SQL - Structured Query Language