WCU Banking Web App Analysis

CS 380 Group Project

Alex Ryse, Garrett Kraxberger, Victoria Cendejas Garibay

Table of Contents

[Functional & Non-Functional Requirements 4](#_Toc197343244)

[Use Case Descriptions & Diagrams 6](#_Toc197343245)

[User Authentication 6](#_Toc197343246)

[Display Account Info 6](#_Toc197343247)

[Withdraw/Deposit 8](#_Toc197343248)

[Trasnfer/Receive Funds 9](#_Toc197343249)

[ATM Locator 10](#_Toc197343250)

[UML Diagram 11](#_Toc197343251)

[11](#_Toc197343252)

[Sequence Diagrams 12](#_Toc197343253)

[User Authentication 12](#_Toc197343254)

[Display Account Info 14](#_Toc197343255)

[Withdraw/Deposit 15](#_Toc197343256)

[Transfer/Receive Funds 16](#_Toc197343257)

[ATM Locator 17](#_Toc197343258)

[17](#_Toc197343259)

[API Selection 18](#_Toc197343260)

# Functional & Non-Functional Requirements

**Business Requirements**

* The web app must follow the project timeline and project plan, with key milestones achieved no later than their scheduled dates.
* The web app must be fully functional, passing all functional test cases, and ready for deployment by the final project deployment date of June 6, 2025.

**User requirements**

* The user authentication, dashboard, ATM locator, and fund management pages must each have a clean, responsive for user satisfaction and ease of access.
* The system must allow users to instantly access available funds, ensuring completion times (for viewing balances, deposits, and withdrawals) are within 3 seconds under normal network conditions.
* All UI pages must be loaded in less than 3 seconds in normal network conditions.
* All notifications detailing transaction history must display 100% accurate, up-to-date information, within 5 seconds after any transaction is completed.

**Regulatory Systems**

* All user passwords must be encrypted before being stored in the database, ensuring that plaintext passwords are never stored or encrypted.
* Passwords must meet the following requirements: at least 1 uppercase, 1 number, 1 special character, along with a length of at least 5 characters.
* Users must provide a valid email address in the format local-part@domain (ex: example@domain.com), verified using input validations before account creation can proceed.
* Users must provide a valid U.S. phone number in the format (XXX) XXX-XXXX or XXX XXX-XXXX (9 digits, with a dash between), verified using input validation before account creation can proceed.
* The system must require a valid SSN in the format XXXXXXXXX (9 digits) during account creation.
* SSNs must be 100% unique across all users; if a duplicate SSN is entered during account creation, an error message must be displayed within 2 seconds preventing account creation.
* Input validation must be implemented and tested to ensure only valid, sanitized inputs are accepted, with 100% error handling coverage.

**User Authentication**

* A maximum of 5 failed login attempts must be allowed before the account is locked for 10 minutes, with an appropriate error message displayed to the user.

**Fund Management**

* All fund management features (deposit, withdraw, transfer) must load in under 3 seconds under standard network conditions.
* Upon successful fund transaction, the user should receive a confirmation notification within 5 seconds.
* All transaction details should reflect 100% real-time data with no manual refresh required.
* Users should be able to view their lifetime transaction history with no page load exceeding 3 seconds under normal network conditions.
* Transaction amount input fields must only accept numeric values between $0.00 and $500.00 inclusive.

**Withdraw/Deposit**

* A user can perform no more than 3 deposits and 3 withdraws per day, enforced by server-side validation.
* A user cannot exceed $500 total per deposit or per withdraw per day, enforced by backend rules.

**Transfer funds/receive**

* Users must not be allowed to transfer funds to or receive funds from their own account, enforced by the transfer validation stage.

**ATM locator**

* Upon navigating to the ATM locator page, the map automatically defaults the view to Ellensburg, WA within 2 seconds.
* Clicking on an ATM pin must display full ATM details (address, hours, and services) within 2 seconds.
* The app must pinpoint the user’s current location within a 100-foot radius of their actual GPS location, as determined by the device’s location services.

**Account info display**

* Account details must load dynamically within 3 seconds on the dashboard.
* Users must be able to refresh their account details manually, with updates reflecting within 2 seconds after refresh.

# Use Case Descriptions & Diagrams

## User Authentication

A black screen with white text

AI-generated content may be incorrect.

## Display Account Info

|  |  |
| --- | --- |
| **Use Case Name:** | **Display Account Information** |
| **Related Requirement:** | **The system must display users account information, including personal and banking details** |
| **Goal in Context:** | **Allow users to view their personal and banking account details** |
| **Preconditions:** | 1. **The user must be logged into the banking web app** 2. **The system must be online and able to fetch user data** |
| **Successful End Condition:** | **The system displays all users’ account information, where the user can access** |
| **Failed End Condition:** | **The system is unable to display the users’ account information due to database fetch errors or network connection issues** |
| **Primary Actor(s):** | **Customer** |
| **Secondary Actor(s):** | **N/A** |
| **Trigger:** | **The user logs into the app** |
| **Main Flow:** | 1. **The user logs into the app** 2. **The system redirects to the account information page** 3. **The system fetches the users’ personal and banking information from the database** 4. **The system displays the users’ personal and banking account information** |
| **Extensions:** | **2a. The system is unable to load the account information page due to the system being offline or a network connection issue: system shows an error message** |
|  | **3a. The system is unable to fetch the users’ personal and banking account information due to a database error: system shows an error message and logs the error for admin review** |
| **Assumptions:** | 1. **Users can access the banking web app online** 2. **Users have valid personal and banking account details** 3. **The database and backend services are online and working** |
| **Open Issues:** | 1. **What is expected if partial data is fetched, where personal info loads but banking data does not?** 2. **How often should the account information on this page be refreshed or updated?** 3. **What kind of account details should there be?** |

## Withdraw/Deposit

|  |  |
| --- | --- |
| **Use Case Name:** | **Withdraw/Deposit** |
| **Related Requirement:** | **The system must perform user withdraw and deposit functionality with their banking funds securely** |
| **Goal in Context:** | **Allow users to withdraw and deposit their banking funds on their own using a secure and dynamic process** |
| **Preconditions:** | 1. **The user must be logged into the banking web app** 2. **The system must be online and able to update user banking data** |
| **Successful End Condition:** | **The user withdraws or deposits their banking funds, and the system updates the user banking account information in both the server and client side** |
| **Failed End Condition:** | **The user is unable to withdraw or deposit their banking funds due to invalid input or database issues** |
| **Primary Actor(s):** | **Customer** |
| **Secondary Actor(s):** | **N/A** |
| **Trigger:** | **The user clicks on either the “Withdraw” or “Deposit” button** |
| **Main Flow:** | 1. **The user clicks on either the “Withdraw” or “Deposit” button** 2. **The user inputs the desired amount of banking funds (to either withdraw or deposit)** 3. **The user clicks on the “OK” button** 4. **The system validates the user input** 5. **The system updates the users’ banking account information** 6. **The system displays a confirmation message** |
| **Extensions:** | **4a. The user input is invalid: prompt the user to enter valid details**  **5a. The system fails to update the users’ banking account information due to database issues: display an error message** |
| **Assumptions:** | 1. **Users have an active and valid banking account with sufficient funds for withdrawals** 2. **The system validates the amount entered before proceeding with the transaction** 3. **There is a minimum and maximum amount allowed per transaction (system enforces this)** 4. **Users can only input numeric values** |
| **Open Issues:** | 1. **Should users receive confirmation after each transaction (email or in-app notification)** 2. **Are there any daily/weekly/monthly withdrawal and deposit limits?** |

## Trasnfer/Receive Funds

|  |  |
| --- | --- |
| **Use Case Name:** | **Transfer/Receive Funds** |
| **Related Requirement:** | **The system must allow users to transfer and receive banking funds securely** |
| **Goal in Context:** | **Allow users to transfer and receive banking funds to and from other active users using a secure process, offering real-time updates** |
| **Preconditions:** | 1. **The user must be logged into the banking web app** 2. **The system must be online and able to access the database** |
| **Successful End Condition:** | **The user transfers or receives funds to or from another user, and the system updates both users banking account information in both the server and client side** |
| **Failed End Condition:** | **The user is unable to transfer or receive funds to or from another user due to invalid input or database issues** |
| **Primary Actor(s):** | **Customer A (sender)** |
| **Secondary Actor(s):** | **Customer B (receiver)** |
| **Trigger:** | **The user clicks the “Transfer” button** |
| **Main Flow:** | 1. **The user clicks the “Transfer” button** 2. **The system fetches all the users from the database and displays each user on the transfer/receive page** 3. **The user clicks on an available user to transfer funds to** 4. **The user inputs the amount of funds to transfer** 5. **The user clicks the “OK” button** 6. **The system validates the user input** 7. **The system updates the selected users’ banking funds with the transfer amount** 8. **The system updates the current users’ banking funds by deducting the transfer amount** 9. **The system displays a confirmation message to the sender** 10. **The system sends a confirmation notification to the receiver** |
| **Extensions:** | **2a. The system is unable to fetch all the users due to a database issue: display an error message**  **6a. The user input is invalid: prompt the user to enter valid details**  **7a, 8a. The system is unable to update either/both users banking funds due to a database issue: display an error message** |
|  | **10a. The confirmation notifications fail: the system logs the error** |
| **Assumptions:** | 1. **Both users (sender and receiver) have active accounts** 2. **The sender has enough funds to cover the transfer amount** 3. **The system prevents users from transferring funds to themselves** 4. **All users are uniquely identified in the system** |
| **Open Issues:** | 1. **Should there be a limit to daily/weekly/monthly transfers?** 2. **Will the system display any transaction history for the users?** |

## ATM Locator

|  |  |
| --- | --- |
| **Use Case Name:** | **ATM Locator** |
| **Related Requirement:** | **Provide an ATM locator function for the customer.** |
| **Goal in Context:** | **Provide an accessible way for customers to locate their nearest ATM to facilitate deposits, withdraws, or transfer of funds.** |
| **Preconditions:** | * **Customers must have an existing bank account.** * **Customers must have a linked and approved portal account.** |
| **Successful End Condition:** | **ATM locations based on the user’s input are displayed with travel time and distance information.** |
| **Failed End Condition:** | **ATM location information is unable to be displayed.** |
| **Primary Actor(s):** | **User: customer** |
| **Secondary Actor(s):** |  |
| **Trigger:** | **Customers logs into their portal account.** |
| **Main Flow:** | 1. **Customers logs into their portal account.** 2. **They click the “ATM locator” option listed on their account dashboard.** 3. **Google Maps will be displayed with points of interest near the CWU.** 4. **A list will be displayed sorted by increasing distance based on current location.** 5. **The user will click the marker on the map that they would like directions to.** 6. **Directions to the ATM location will be displayed.** |
| **Extensions:** | **3a. Location input is invalid or not found: error message for invalid input. 4a. No ATM location within range of location input: error message for no ATM locations available currently.** |
| **Assumptions:** | - **The customer is within CWU Ellensburg campus.** |
| **Open Issues:** | - **Are there any additional location details added to specific ATM locations?** |

## UML Diagram

## A diagram of a bank AI-generated content may be incorrect.

# Sequence Diagrams

## User Authentication

A diagram of a user

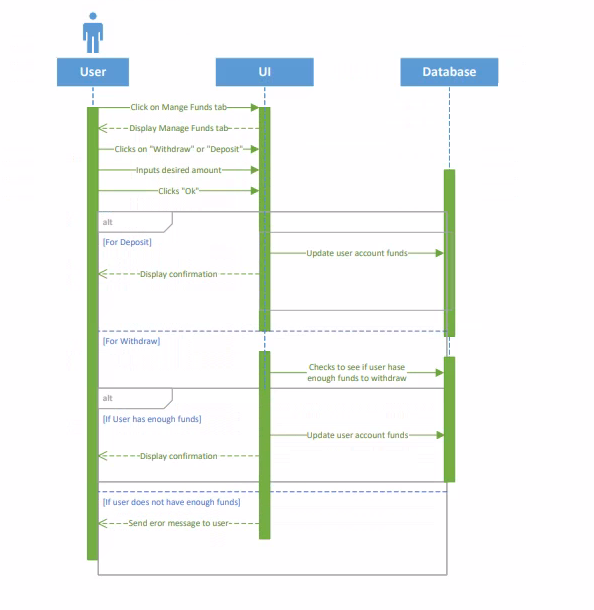
AI-generated content may be incorrect.

A diagram of a user application

AI-generated content may be incorrect.

## A screenshot of a computer screen AI-generated content may be incorrect.Display Account Info

## Withdraw/Deposit



## A screenshot of a computer screen AI-generated content may be incorrect.Transfer/Receive Funds

## ATM Locator

## A diagram of a user and user map AI-generated content may be incorrect.

# API Selection

For our Wildcat Credit Union (WCU) Banking Web App we are implementing the Google Maps JavaScript API. This API was chosen to develop our ATM Locator feature, which allows users to locate, view, and get information for our WCU ATMs around Ellensburg, WA. Google Maps offers access to accurate and up-to-date geographic information, which is essential for our banking application. On top of that, it supports features like real-time location, custom markers, and responsive zooming – enchaning the user’s ability to find nearby ATMs easily and interactively. Since our front end is planning to built using JavaScript/React, the Google Maps API provides a well-documented and supported interface that works directly with our tech stack. Not only this, as a top tech company, Google APIs are reliable, scalable, and perfect for production environments, which aligns with our goals of this app. Lasty, choosing the Googla Maps API supports both our functonal and non-functional requirements to offer usability, performance, and user trust for our app.