Foodstabook

High Level Design

Revision Number: 1.1

Last date of revision: March 23, 2022

Team: ***The Cool Team***

John.Bower@student.csulb.edu

Bich-Tram.Pham01@student.csulb.edu

dishant.sarvaiya@student.csulb.edu

Spencer.Breding@student.csulb.edu

kevin.garciajulca@student.csulb.edu

Project Overview

Product Perspective:

Foodstabook is an Android application that consists of several different components. Majority of the components will utilize Firebase - a Google's Backend as a Service application development software . Firebase will help with: authentication, security, database, deployment, and data analytics. Android Studio is used as the official IDE. We will mainly use it for designing the application's front-end while integrating with Firebase's tools for the back-end. Users need to log into the system to interact with the application.

Tools used:

Firebase Realtime & Firestore Databases

Firebase Auth (Oauth 2.0)

Firebase Hosting

Cloudflare Application Services

Google Analytics

Tailwind CSS

Android Studio IDE (xml, Java)

Design Details

Security

- **Application:**

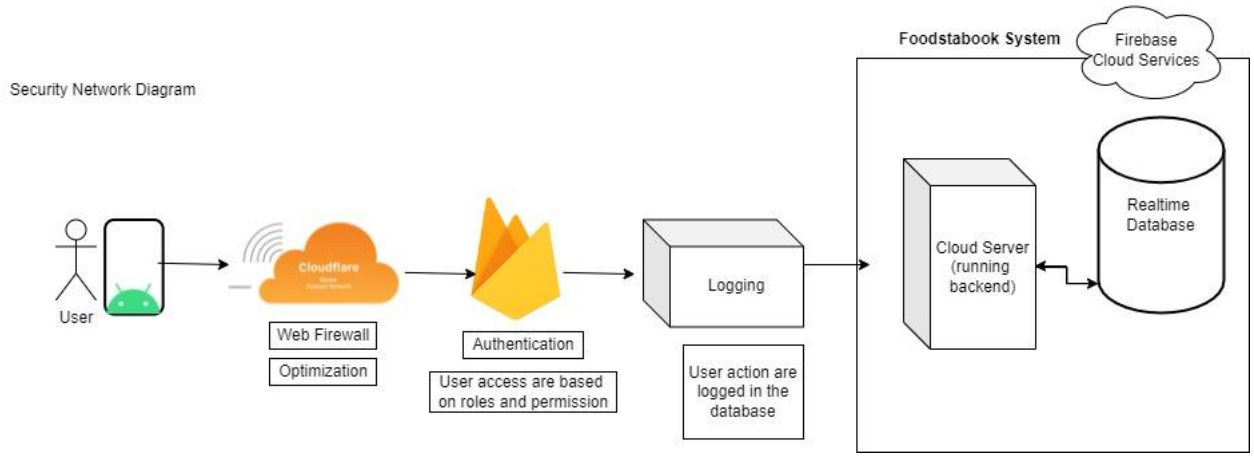
  For application security, we will be using authentication, encryption, and logging. Users are able to access the screens generated by the front end of our app. Our password policy will include a minimum length of eight characters. It will also require at least one uppercase letter, one lowercase letter, one number, and one special character. Passwords will never expire, but we will require users to reset their password if someone attempts to access their account with an incorrect password more than 5 times in a row.

- **Data:**

  We will be collecting location data, emails, names, usernames, passwords, and user food preferences. Any data in transit, such as location data when the users are actively using the app, will be encrypted using HTTPS. Password data will be encrypted using a one-way hash function and then stored. Emails and real names will also be encrypted using triple DES. Any sensitive or personally identifying data will also adhere to A.C.I.D. set properties. We do believe that the data we're collecting is subject to any specific security standards.

- **Network:**

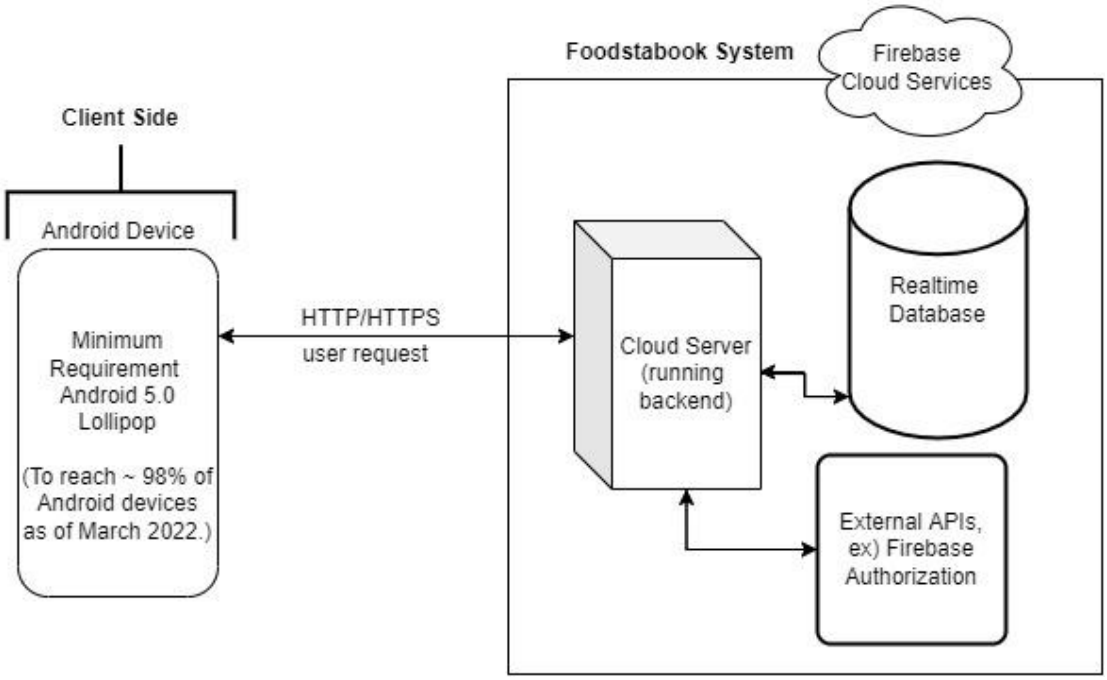
  While not exactly the same, Firebase's security rules will act as a sort of firewall for our application. We will use Cloudflare for a web application firewall and DDoS protection. User access will be limited depending on the type of account they have, with regular users only being able to access screens generated by our app's frontend. Moderator accounts will be able to execute special actions like issuing bans but will not have access to the backend. Admin accounts will have all the powers of moderator accounts and will additionally have access to the backend.



Hardware

We are using Firebase's cloud capabilities for our app, so we should not have to manage any servers ourselves. Our app is only planned to be accessible by Android phones currently.

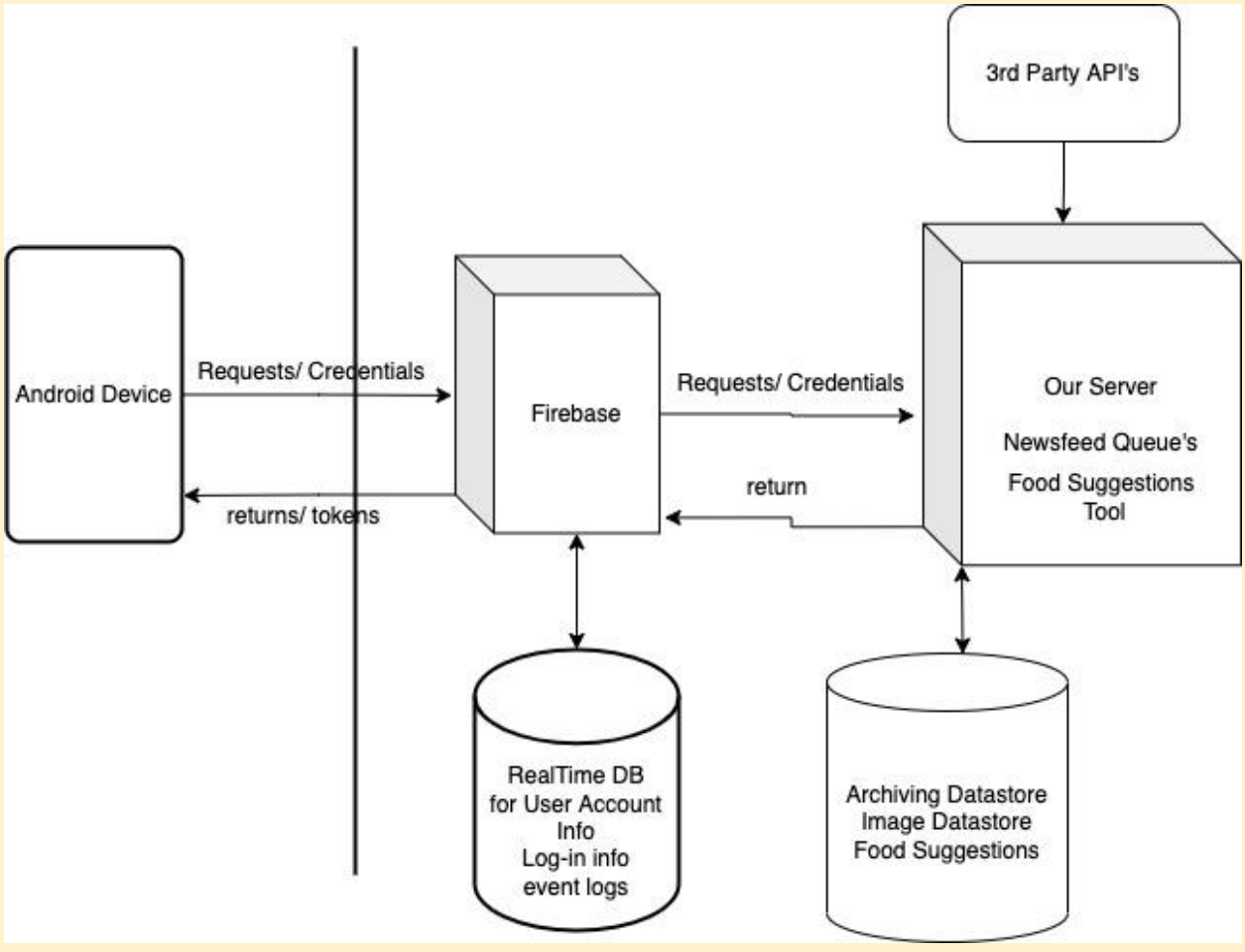
High Level Hardware Design
Foodstabook Mobile App
John.Bower@student.csulb.edu



External Interfaces

- Spoonacular- An API that provides us with information on recipes, ingredients, and restaurant items.
- Firebase Hosting REST API - API used to manage Firebase hosting services.
- Firebase Database REST API - API used to manage Firebase realtime database services.
- Firebase Auth REST API - API used to manage Firebase Auth services.
- Google Cloud Storage API - Used for Firebase cloud storage.
- Logpacker - An API that collects and analyzes log files such as server logs, crash logs and javascript error logs

Architecture



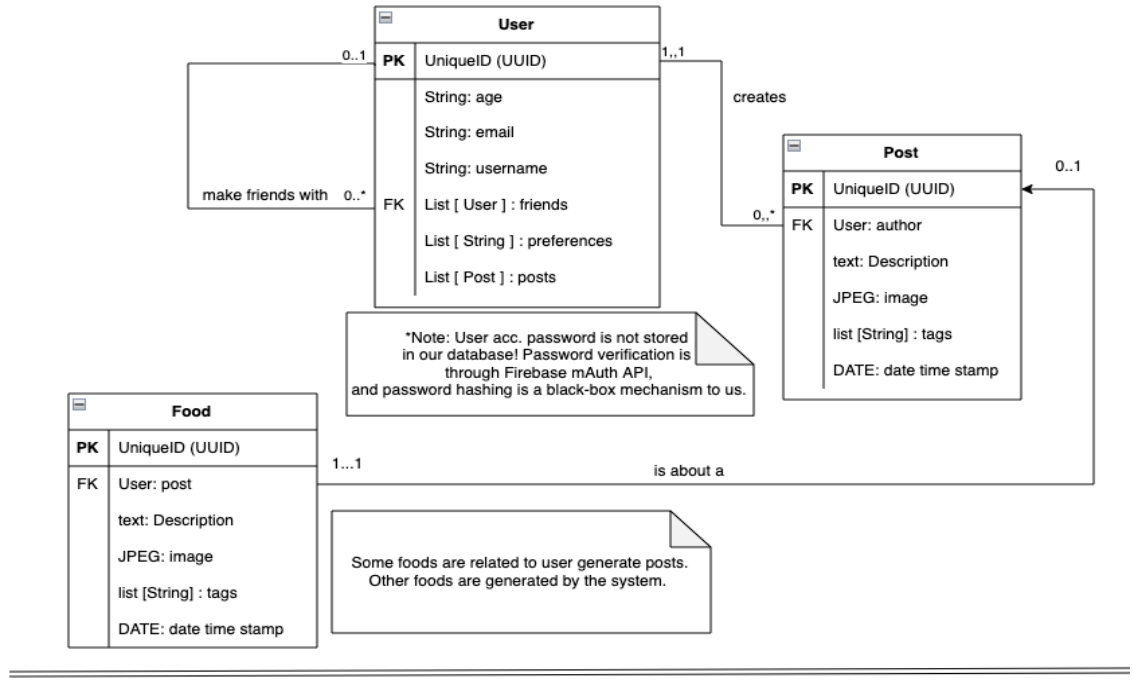
Update 9/28/22 - John.Bower@student.csulb.edu

Database

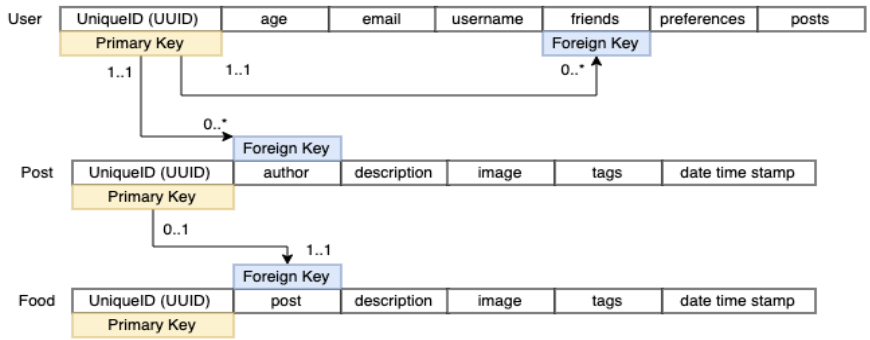
-We will be using Firebase's Realtime Database, A realtime database that is based on JSON objects that will allow us to access our data in real time. This will allow us to have direct and secure access to our database from the client-side code. This is based on NoSQL giving us the flexibility to add or change our schema if we would like to expand it and add more features in the future.
Also it will help our queries to process faster since we are using a Document type structure to store our data.

Real Time Database
UML Class Diagram
and Relational Database Scheme
John.Bower@student.csulb.edu

UML Class Diagram



Entity Relationship Schema



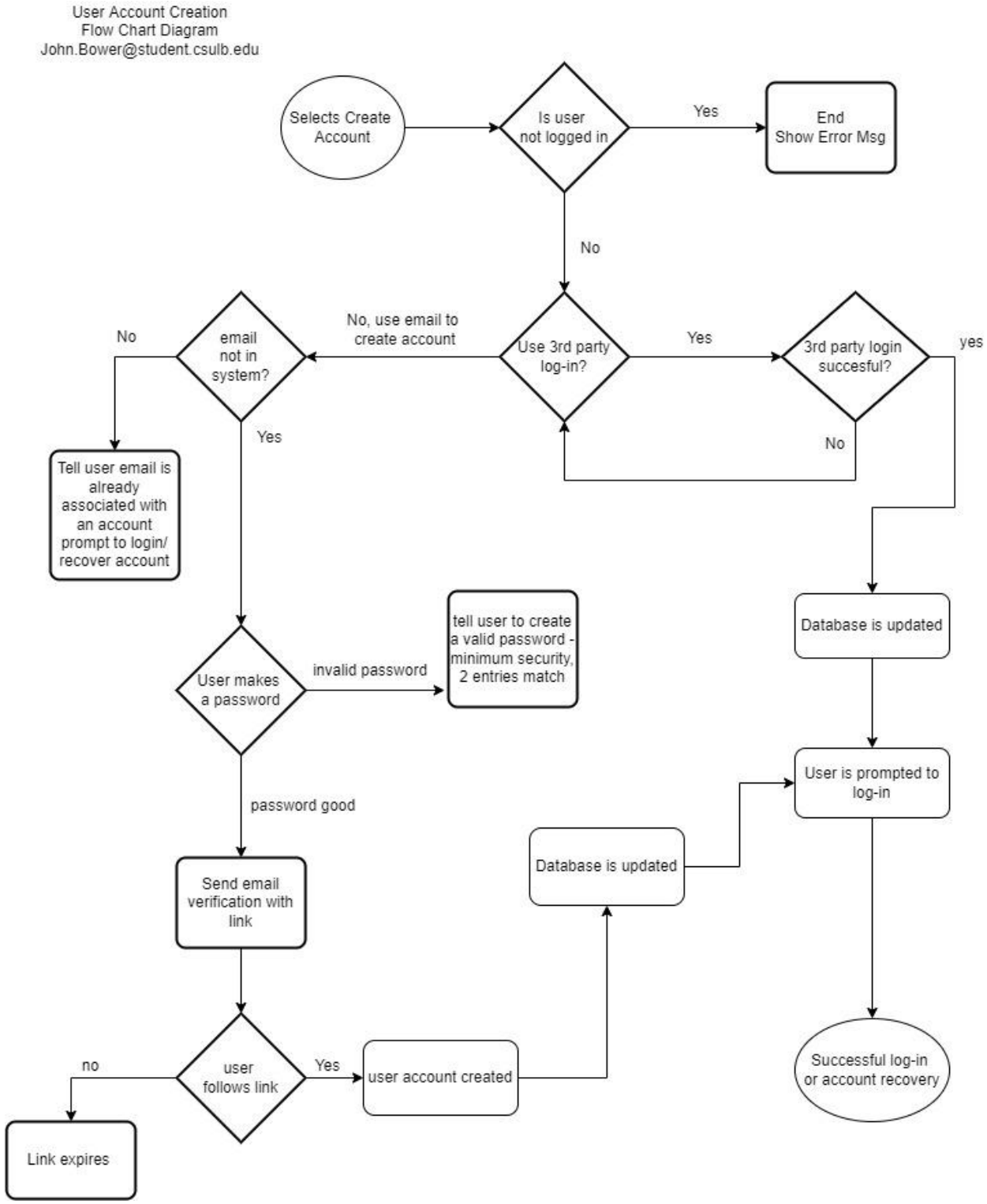
Logging Rules

- We will collect logs from our users by either using logcat command or Logpacker api(Logpacker api can collect and analyze log files such as server logs, crash logs and javascript error logs.) Our app will collect two main sets of logs from the users. The first set of logs will have all the user logins and logout data from recent to oldest specified by date and time. The first set of logs will save users' last 60 day data in the database. After 60 days the old data will be archived.

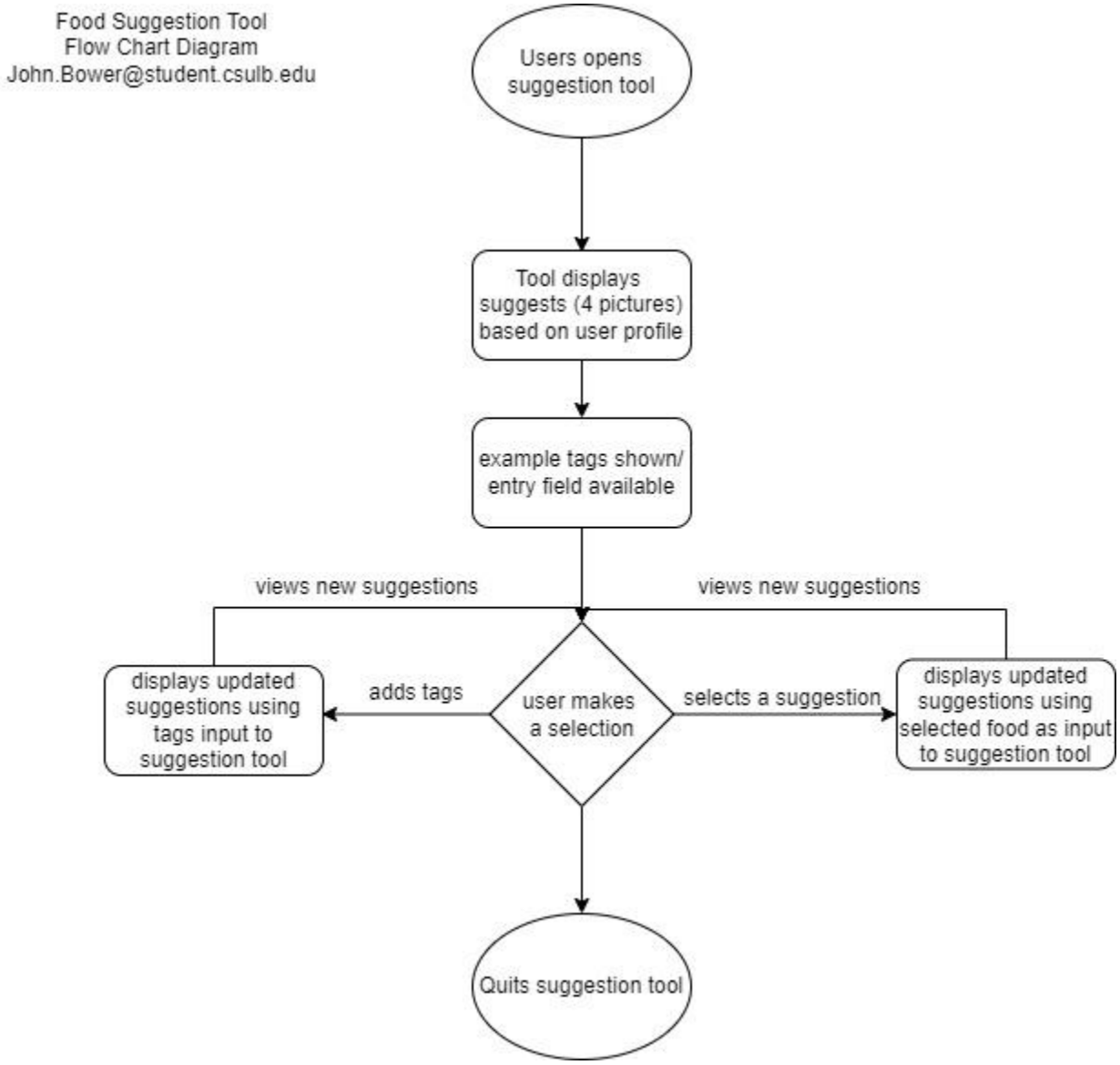
- The second set of logs will have all the error logs, crashes logs, and reports which will be saved as a .txt file or .json file in our database. These data will either be saved for six month or developers will have the option to delete the log files once they go over them.

Additional Sequence, Flowchart, Use Case Diagrams

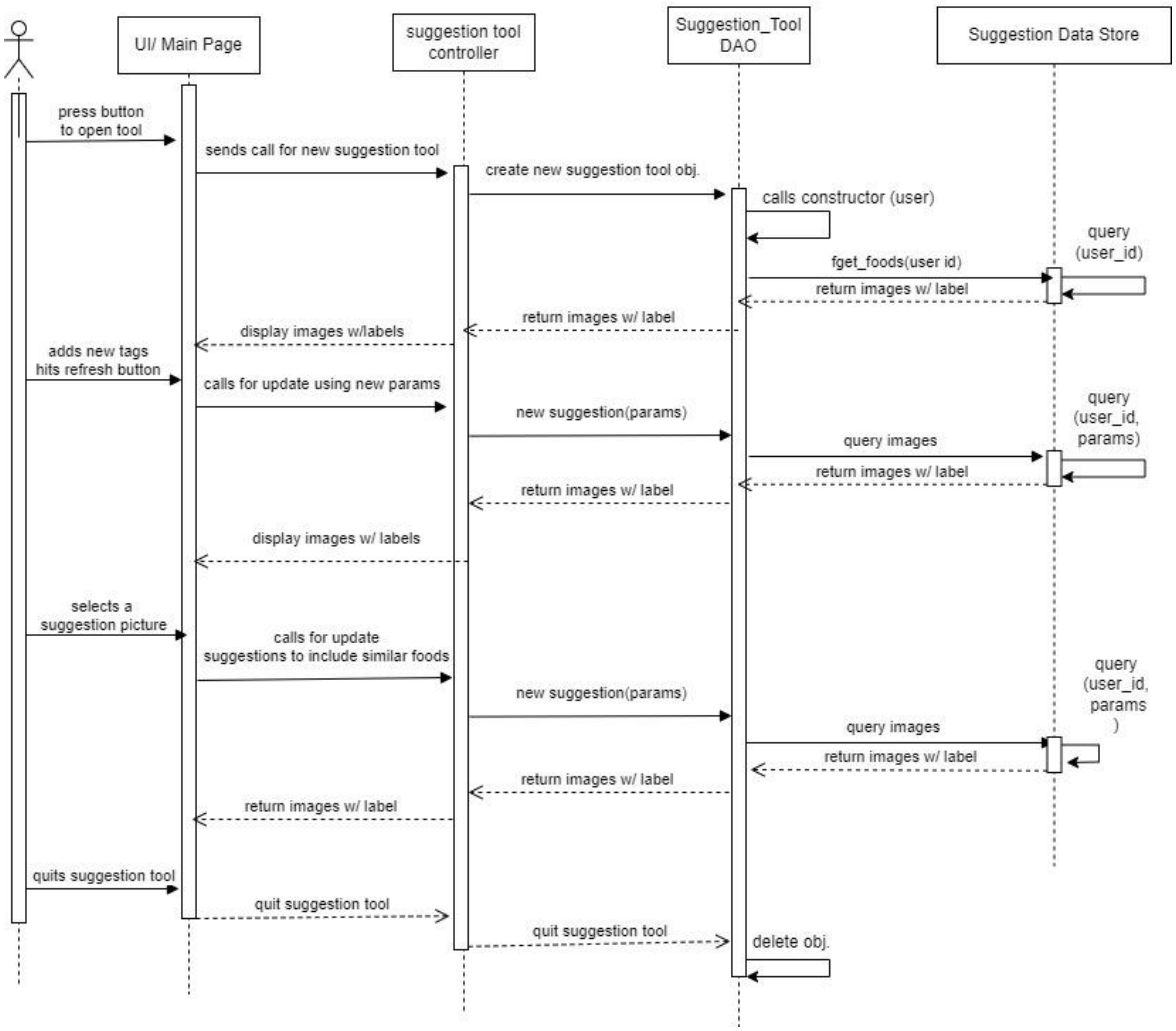
- ## Account Creation



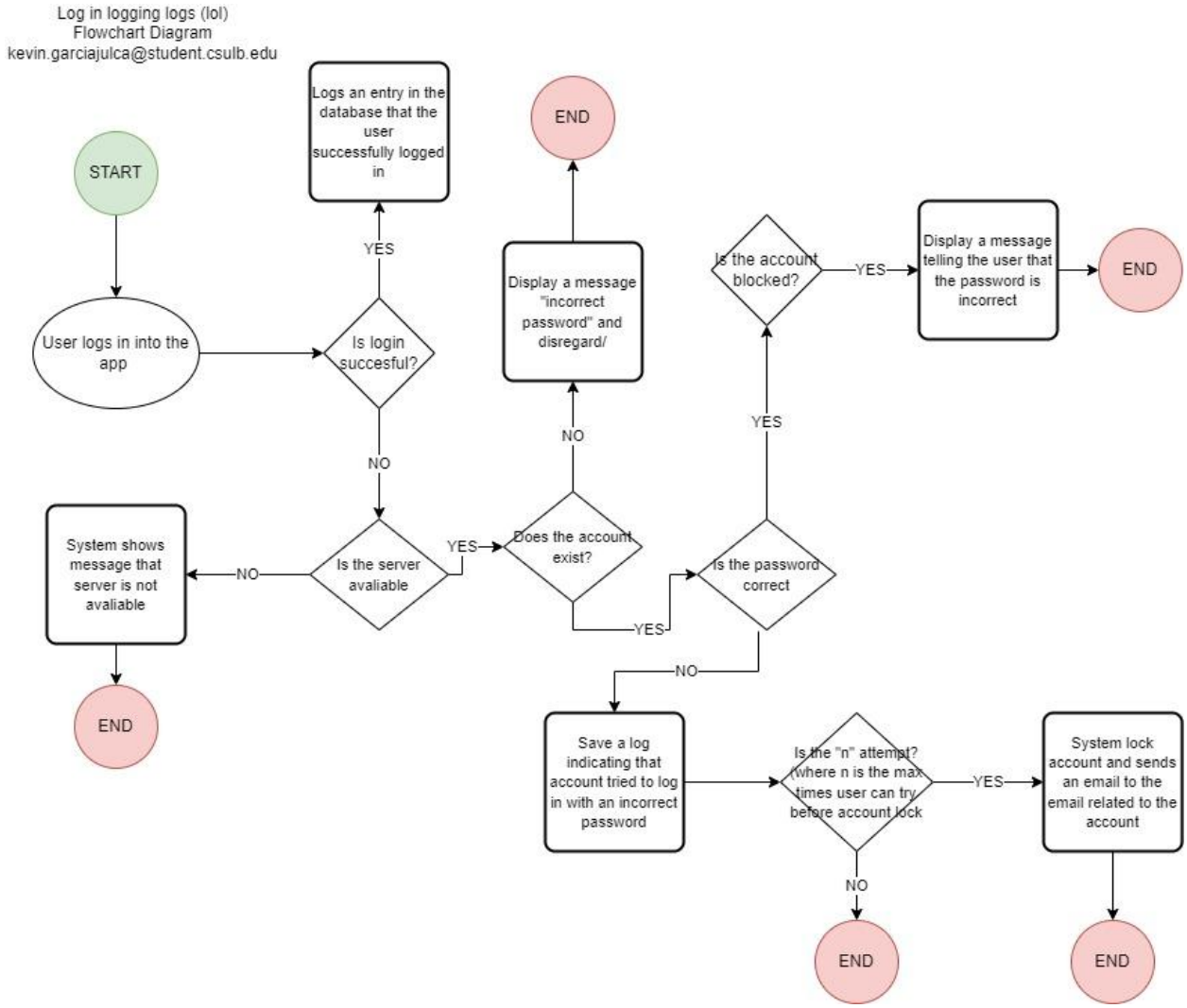
- **Food Suggestion Tool**



Food Suggestion Tool
Sequence Diagram
John.Bower@student.csulb.edu

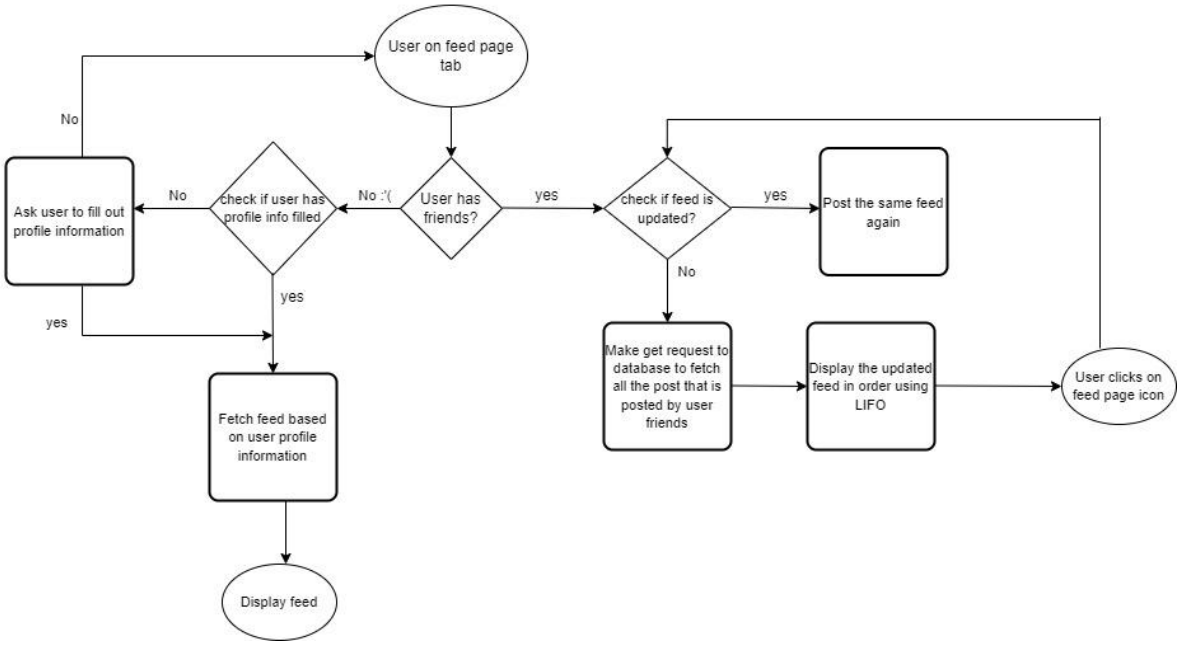


- **Logging:**


Log in logging logs (lol)
Flowchart Diagram
kevin.garciajulca@student.csulb.edu
START
User logs in into the app
Is login succesful?
YES
Logs an entry in the database that the user successfully logged in
END
NO
Display a message "incorrect password" and disregard/
END
Is the server avaliable
NO
System shows message that server is not avaliable
END
YES
Does the account exist?
NO
Display a message "incorrect password" and disregard/
END
YES
Is the password correct
YES
Is the account blocked?
YES
Display a message telling the user that the password is incorrect
END
NO
Save a log indicating that account tried to log in with an incorrect password
Is the "n" attempt? (where n is the max times user can try before account lock
YES
System lock account and sends an email to the email related to the account
END
NO
END

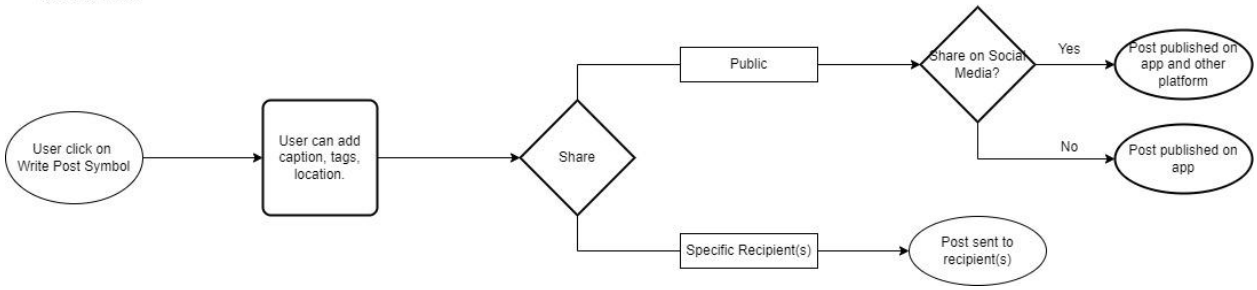
- **Updating/Refreshing Newsfeed**

Process of refreshing/updating feed
Flowchart Diagram
dishant.sarvaiya@student.csulb.edu

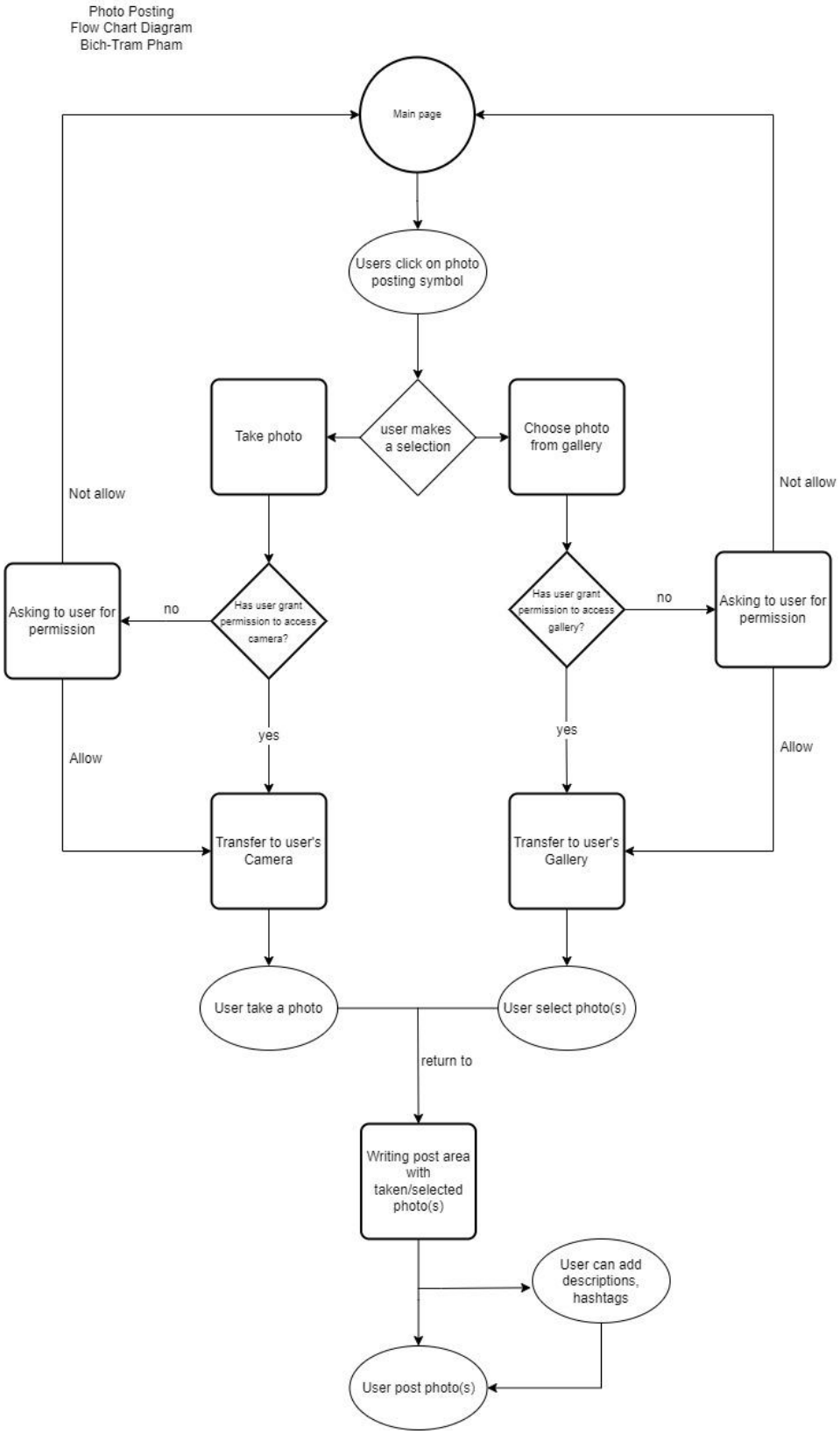


- **User post writing/publishing**

Users writing post
Flow Chart Diagram
Bich-Tram Pham



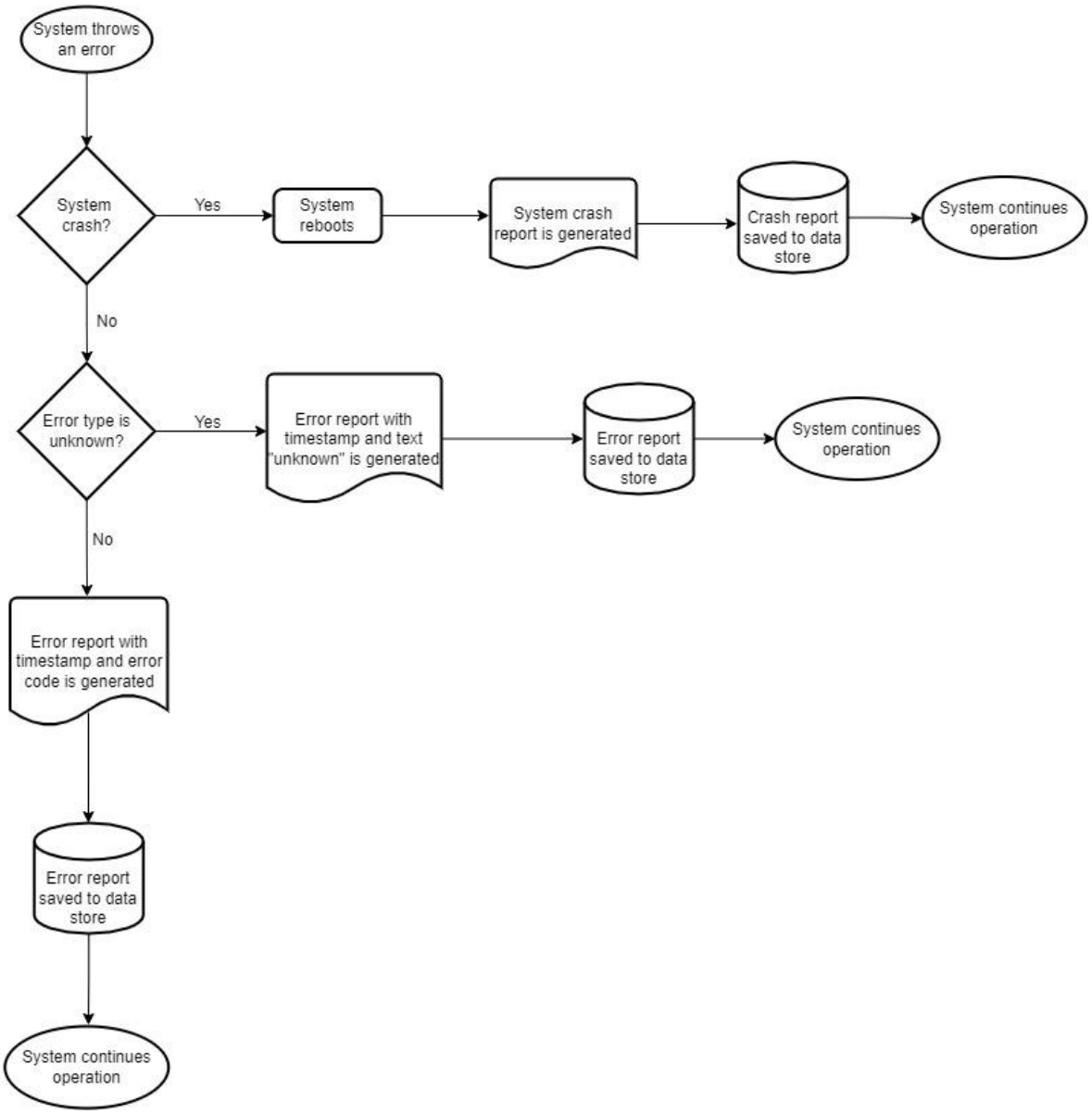
- **User Take/Publish Photos**



Error Handling:

- ## Generating Report

Generate Error Report
Flowchart Diagram
Spencer.Breding@student.csulb.edu



- **Notify Users**

Display Error Message to User
Flowchart Diagram
Spencer.Breding@student.csulb.edu