Design Specification

Bluetooth Low Energy Tracker (BLU)

Version 1.0
Fei Hoffman
CECS 491A Sec 02: Senior Project I
16th October 2024

The BLU Team:
Raquel Hernandez
John Pauly
Samantha Preciado
Luke Trinh
Nicholas Tsimerekis

Table of Contents

Abstract.....3

Behavior: Activity Diagrams.....4

- Activity 1: Create User Profile.....4
- Activity 2: Pairing Device.....5
- Activity 3: Downloading Data.....6
- Activity 4: Access Map.....8
- Activity 5: Database Lookups.....10

Architecture.....12

- Class Diagram.....12
- Site Map.....14

Logical Database Design.....15

ABSTRACT

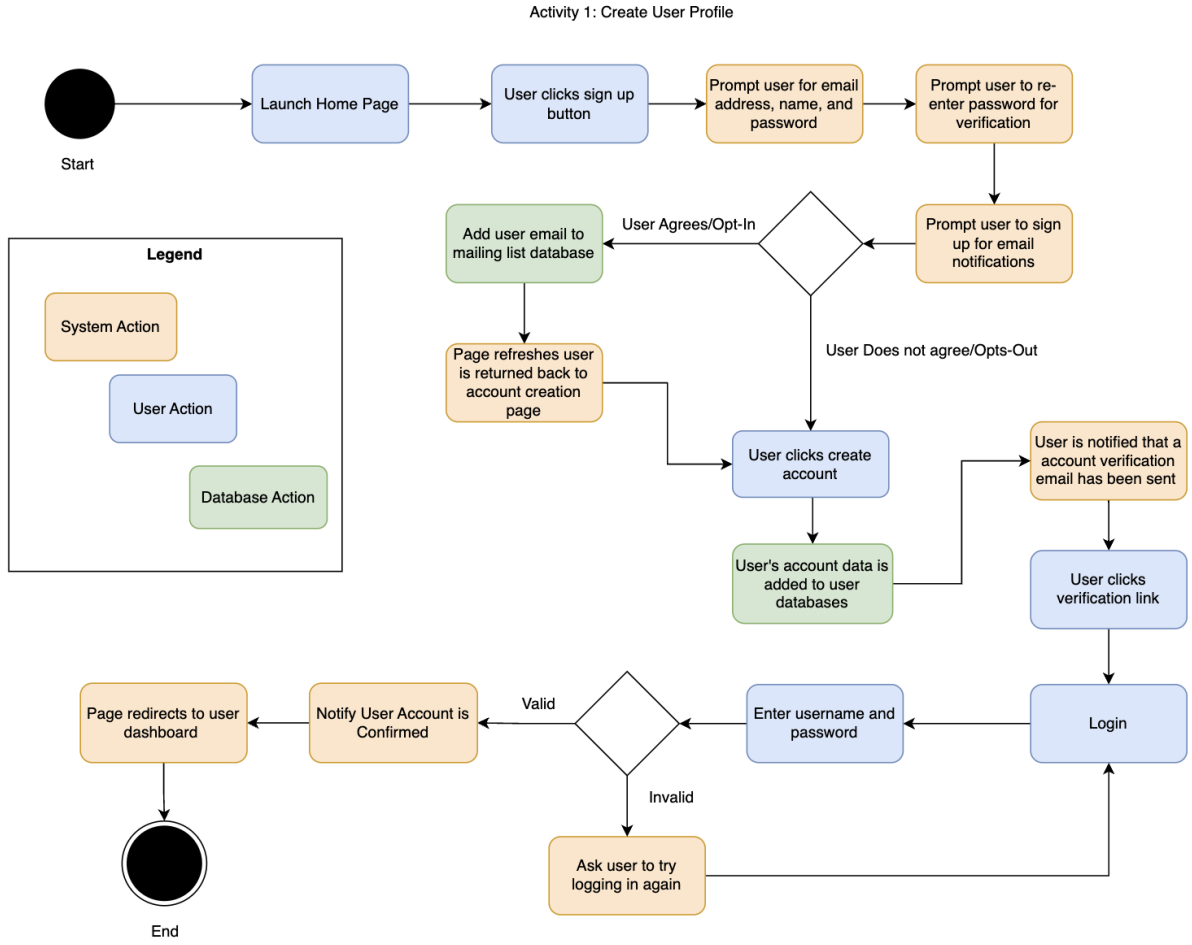
This document details the specific architecture and software designs of the BLU web application. This will go over the decisions of the most important features of BLU as well as the basic architecture of the application and the database. For the activities, the team chose the profile creation, pairing device, data download, access mapping, and database lookups to be the most important features to highlight when developing BLU. Along with the activities will be visualizations of the activities through activity diagrams. Afterwards is the class diagram which will explain and show the basic architecture of the web application's design and how it will be utilized. Lastly is the logical database design which will further breakdown the architecture of the web application and focus on the data structures, relationships, as well as the constraints of the web application.

The target audience of the document are mostly the developers, designers, and testers of BLU. The goal is to create a medium to connect Bluetooth Low Energy (BLE) devices and use triangulation to make a tracker comparable to GPS with the main advantage of being accurate under indoor environments. This document will further detail and better visualize the use cases of BLU to establish a much clearer goal to make an optimize tracker.

BEHAVIOR: ACTIVITY DIAGRAMS

Activity 1: Create User Profile

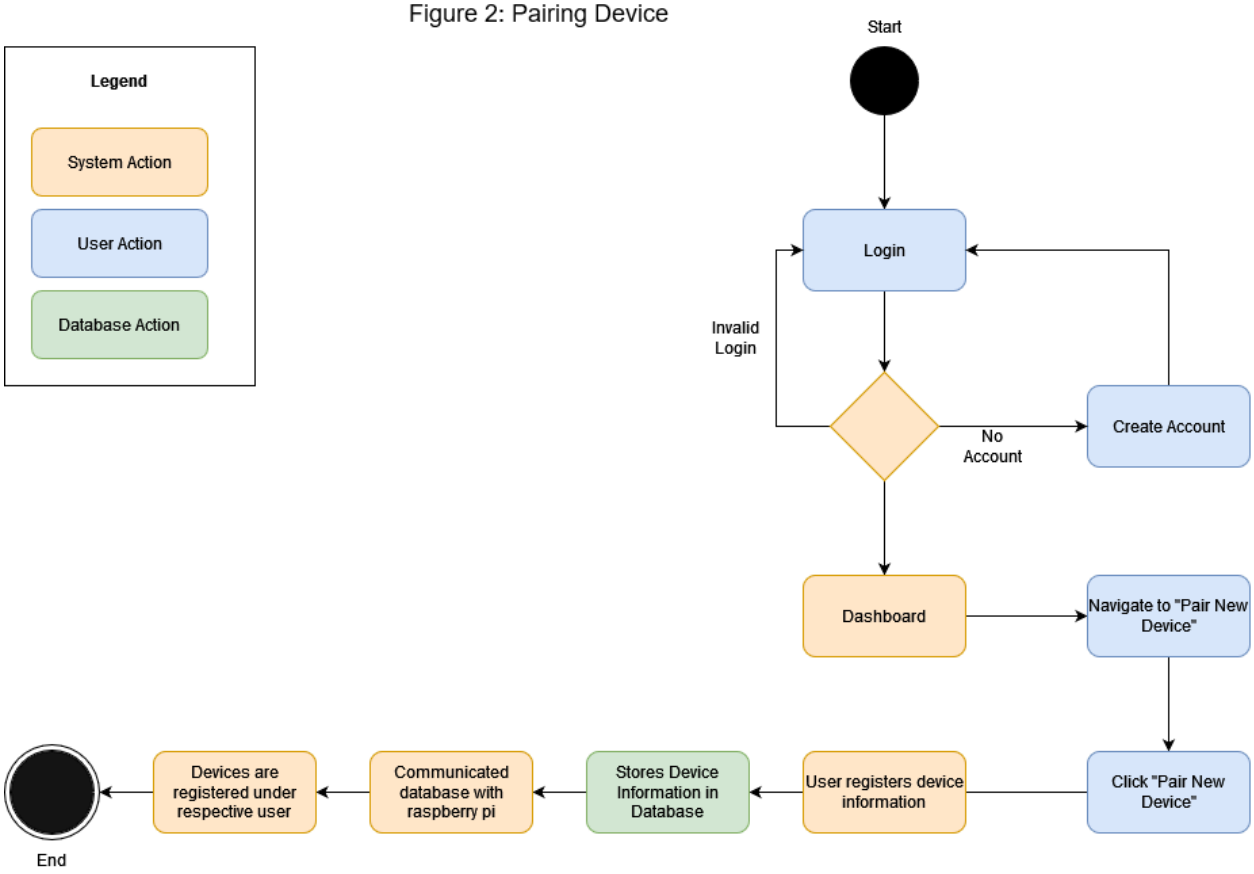
The create user profile feature is a fundamental part of our application, as without it the user is unable to access any other features that are part of the application. As a result, a user's experience when creating a new account puts as a priority a seamless process while also maintaining security. This is achieved by automatic redirects within the application triggered by specific user actions such as when the user verifies their account via email. All user information is stored safely within a database where our application can quickly retrieve a user's account for future use when a user logs in to our system.



Activity 2: Pairing Device

In order to create paths, users will first need to be able to register their devices. Figure 2 shows how users will be able to register their device under their account. After logging in, users will navigate on the dashboard to “Pair New Device”. After clicking on that page, users will be instructed to register information of their raspberry pi that communicates with the nRF boards. Information includes IP and MAC address. Afterwards the device gets registered under the database under the user’s profile.

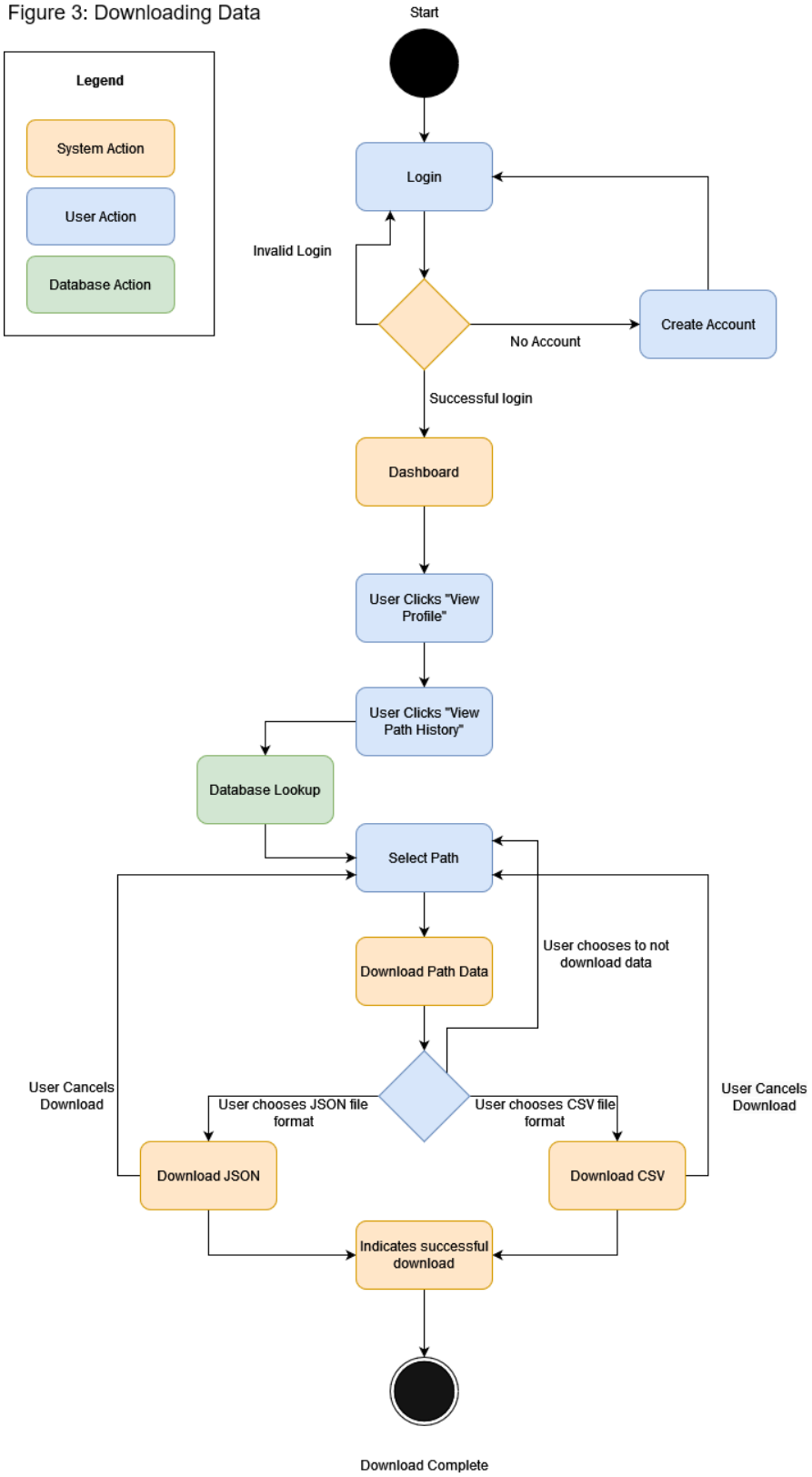
Figure 2: Pairing Device



Activity 3: Downloading Data

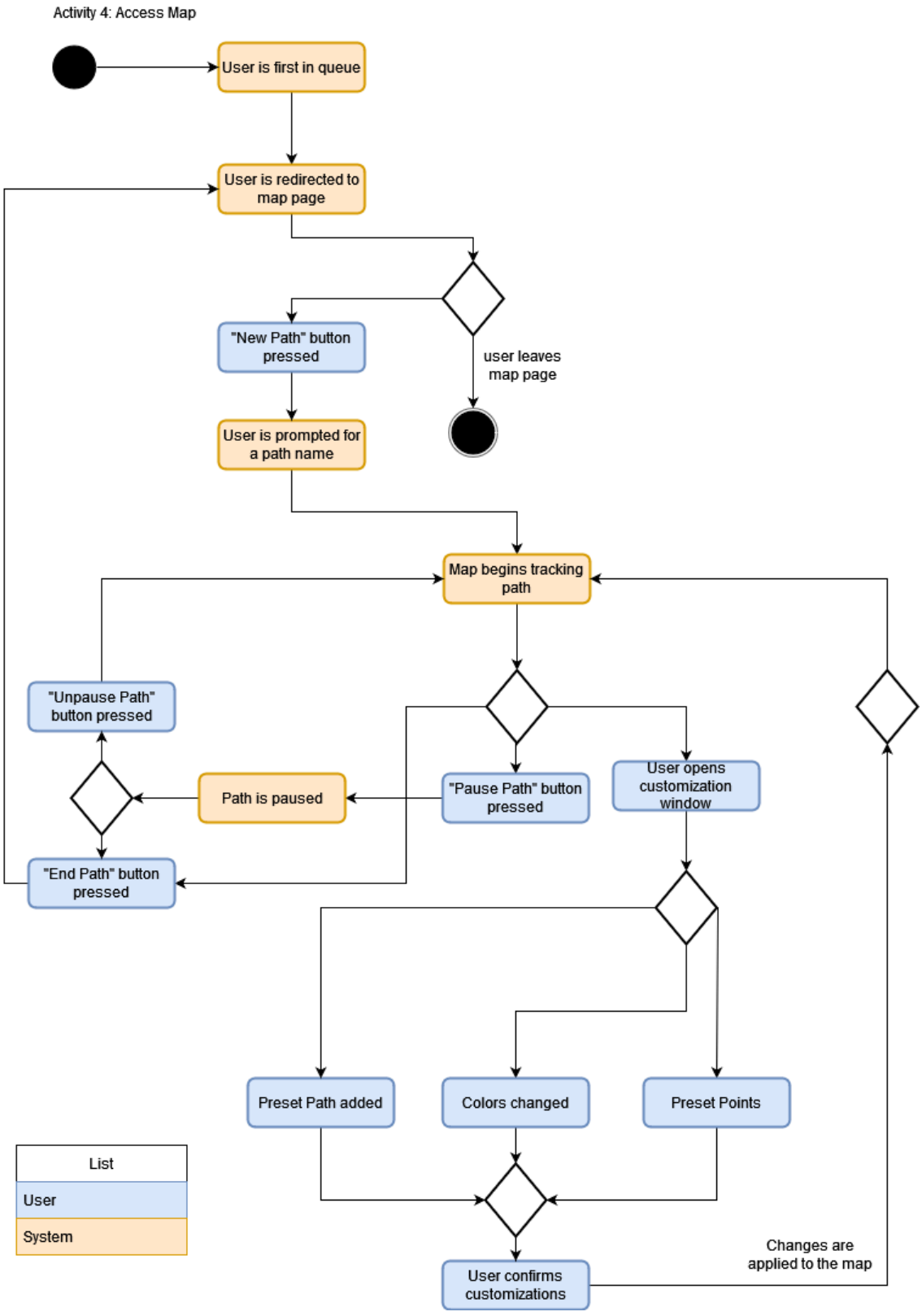
Users will have the ability to download their data at any time. This makes users less reliant on the web server and keeps data offline. Figure 3 will show how users will be able to download their data after logging in. After the correct login credentials, users can view their profile, access their path history, and select a previously done path. Afterward, users are given the choice to download path data as a JSON file, or a CSV file, or cancel the download. After making their decision, the user will download the path data and can still cancel anytime. There will be a follow-up message once the download is complete.

Figure 3: Downloading Data



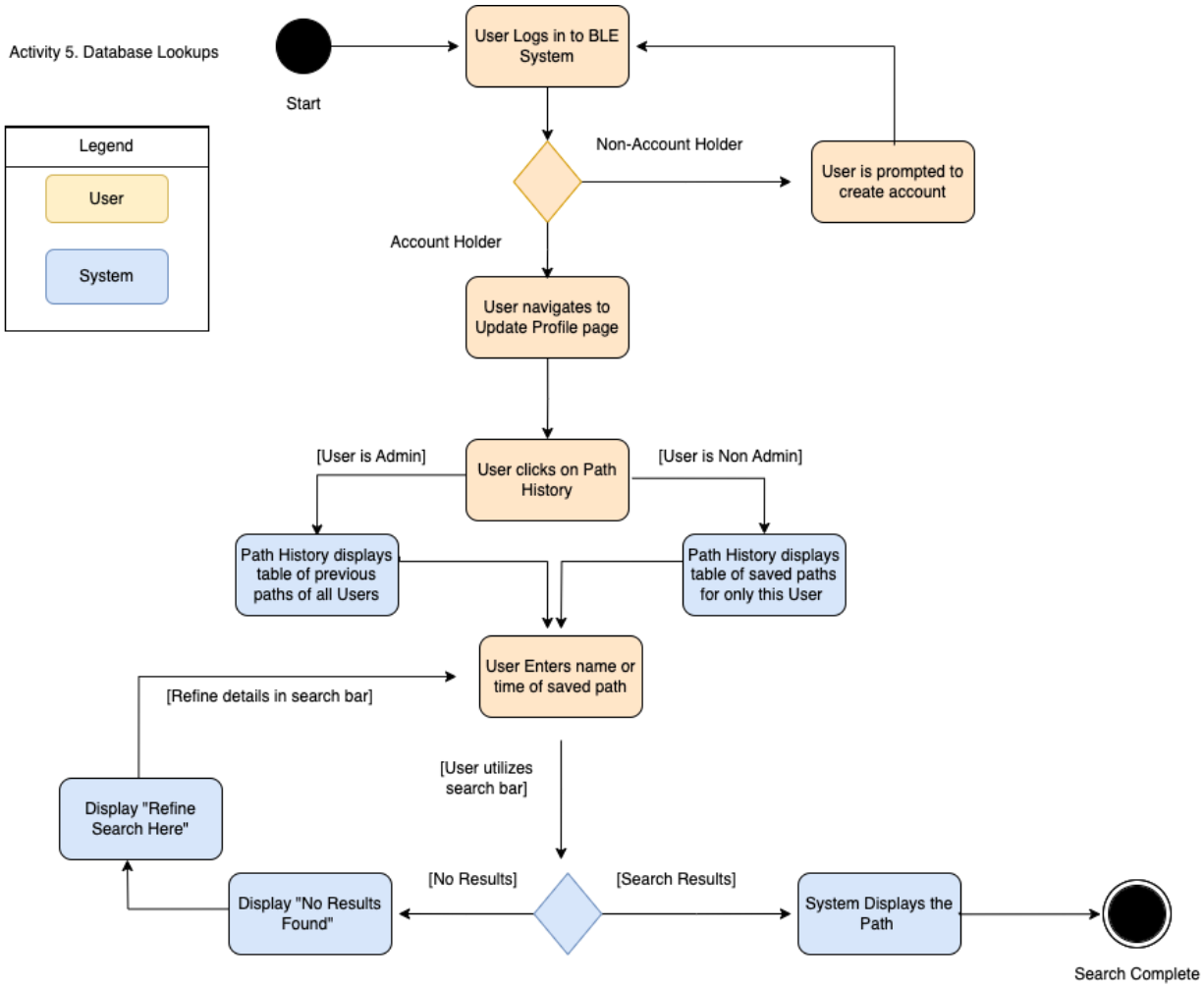
Activity 4: Access Map

Once the user is at the front of the queue, they will be automatically redirected to the map page. The user can begin using the map by pressing a button, and the map will begin updating the path. The user can use the customization menu to make changes to the map. The changes will be applied after the user presses a button for confirmation. The user can pause the maps tracking, and unpause at any time. The map will continue tracking location until the user presses the button to end the path, at which point the user can exit the page.


Activity 4: Access Map
User is first in queue
User is redirected to map page
"New Path" button pressed
user leaves map page
User is prompted for a path name
Map begins tracking path
"Unpause Path" button pressed
Path is paused
"Pause Path" button pressed
User opens customization window
"End Path" button pressed
Preset Path added
Colors changed
Preset Points
List
User
System
User confirms customizations
Changes are applied to the map

Activity 5: Database Lookups

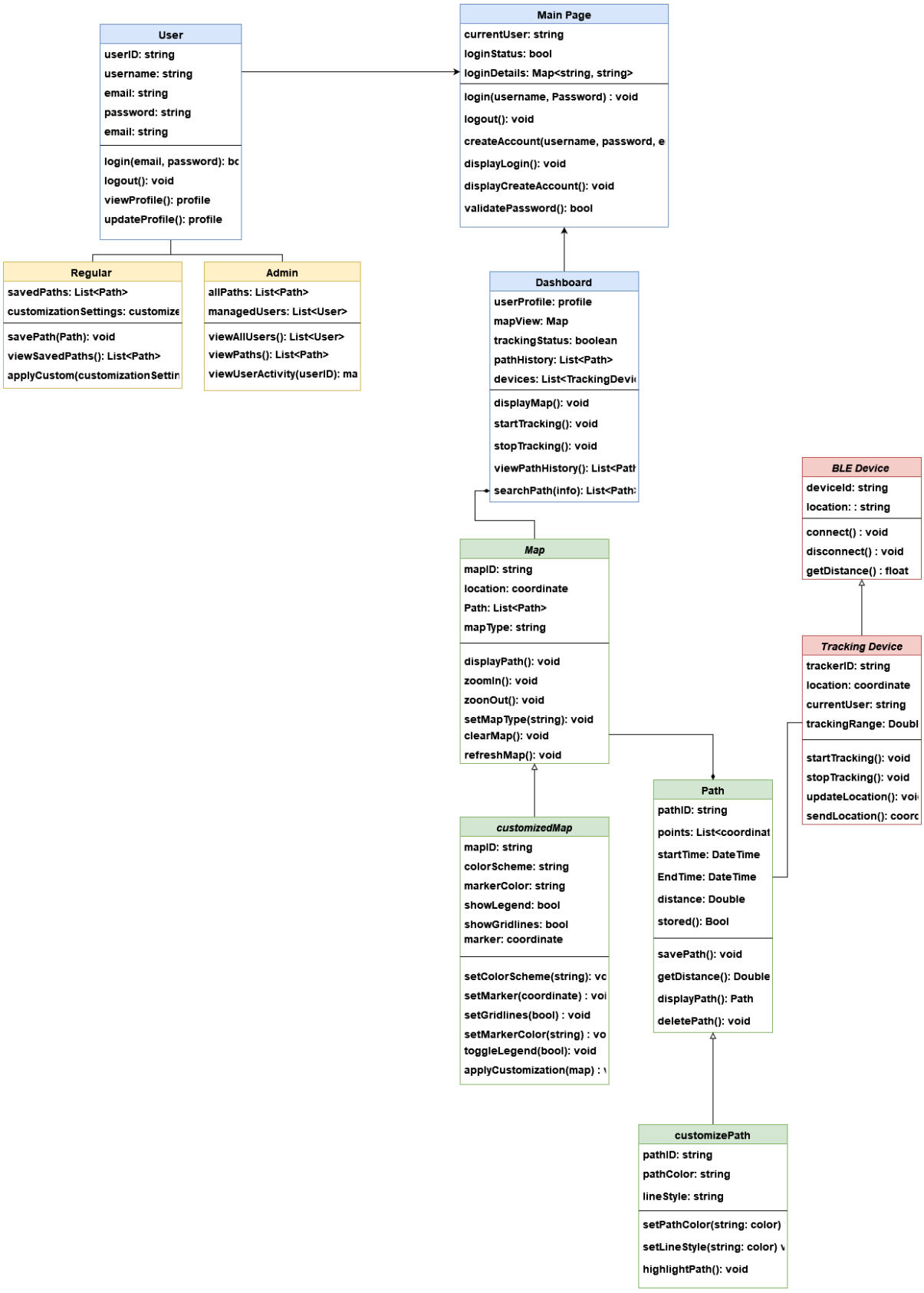
The Database Lookups feature enables users to access their saved paths, which are displayed under the Update Profile section of our web application, specifically on the Path History subpage. Admin users will have enhanced privileges, allowing them to view all saved paths across all user accounts. In contrast, regular users can only access the paths they have created and saved. If a user has not saved any paths, the table will be empty, but they will have the option to start a new path directly from the dashboard and save it afterward. When utilizing the search bar, users can enter either the name they assigned to the path upon saving or search by the timestamp to locate specific entries.


Activity 5. Database Lookups
Start
User Logs in to BLE System
Non-Account Holder
User is prompted to create account
Account Holder
User navigates to Update Profile page
Legend
User
System
[User is Admin]
User clicks on Path History
[User is Non Admin]
Path History displays table of previous paths of all Users
Path History displays table of saved paths for only this User
User Enters name or time of saved path
[Refine details in search bar]
[User utilizes search bar]
Display "Refine Search Here"
[No Results]
[Search Results]
Display "No Results Found"
System Displays the Path
Search Complete

ARCHITECTURE

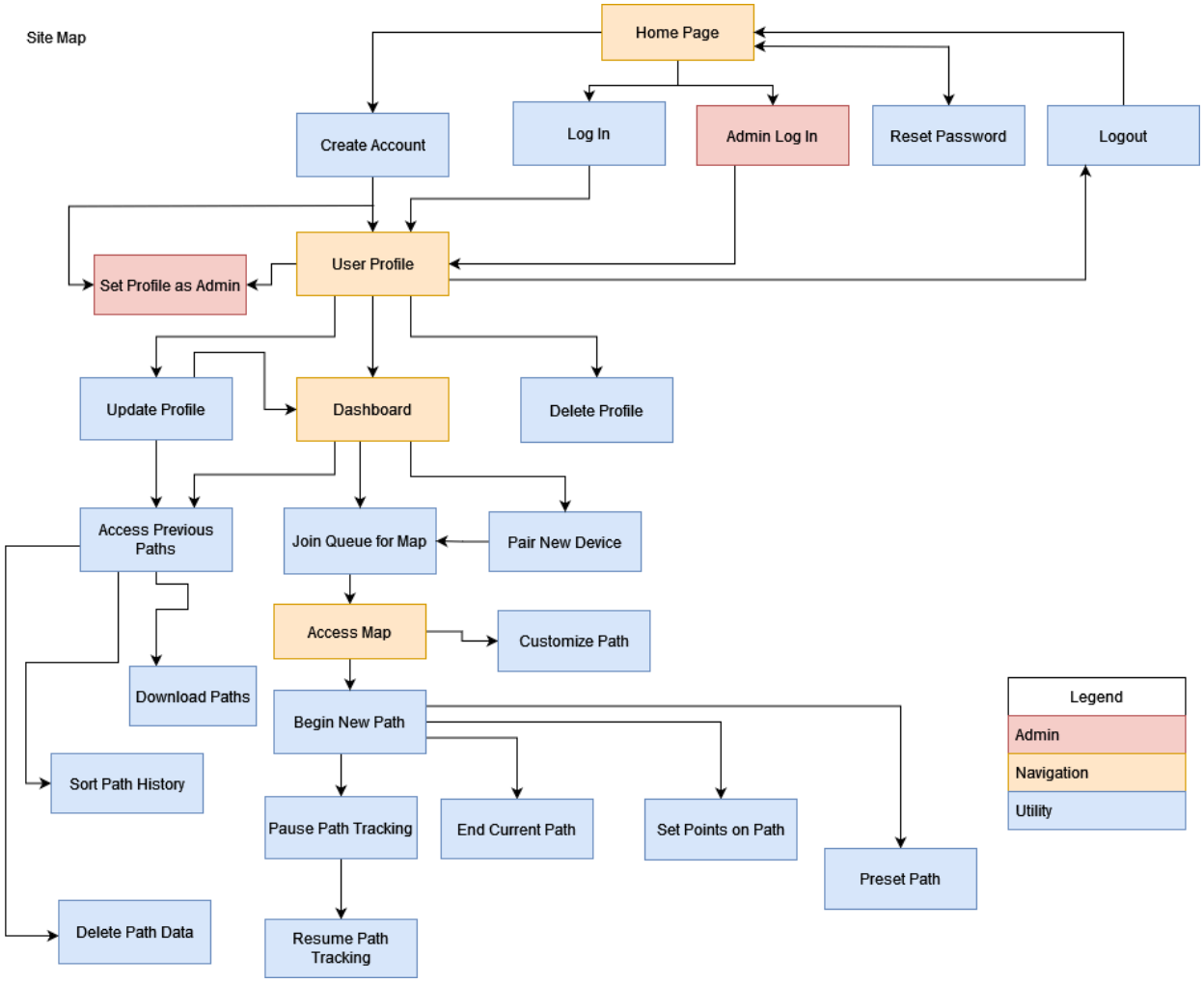
CLASS DIAGRAM

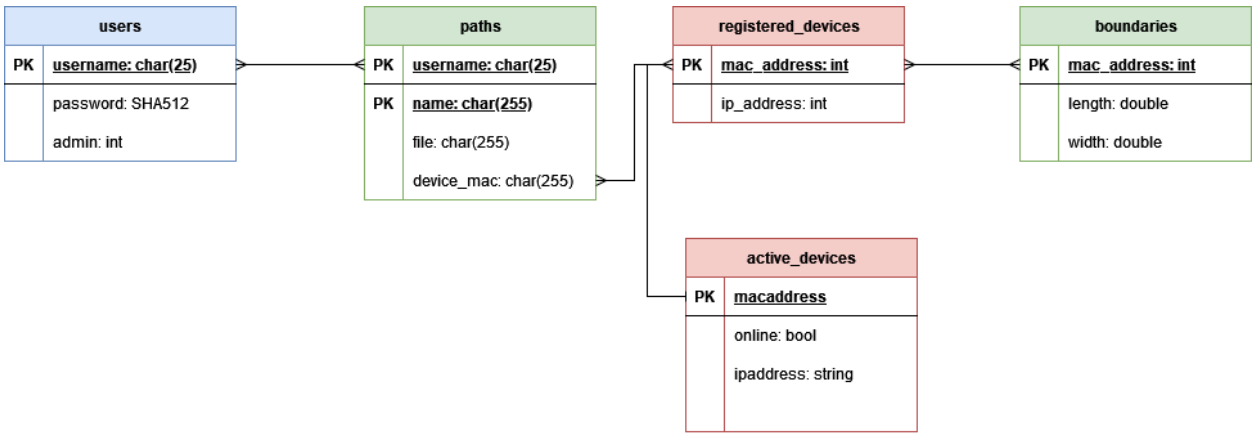
The class diagram contains information about our system design. Color coded we have blue which refers to all objects relating to users and the webpage itself, yellow which refers to system objects like permission structures, green for anything relating to paths, and red for registered devices. The class diagram visualizes the relationships of all the classes and how they interact with one another. The class also shows both points of views of the admin and the normal user.


User
userID: string
username: string
email: string
password: string
email: string
login(email, password): bc
logout(): void
viewProfile(): profile
updateProfile(): profile
Main Page
currentUser: string
loginStatus: bool
loginDetails: Map<string, string>
login(username, Password) : void
logout(): void
createAccount(username, password, e
displayLogin(): void
displayCreateAccount(): void
validatePassword(): bool
Regular
savedPaths: List<Path>
customizationSettings: customize
savePath(Path): void
viewSavedPaths(): List<Path>
applyCustom(customizationSettin
Admin
allPaths: List<Path>
managedUsers: List<User>
viewAllUsers(): List<User>
viewPaths(): List<Path>
viewUserActivity(userID): ma
Dashboard
userProfile: profile
mapView: Map
trackingStatus: boolean
pathHistory: List<Path>
devices: List<TrackingDevi
displayMap(): void
startTracking(): void
stopTracking(): void
viewPathHistory(): List<Path
searchPath(info): List<Path:
BLE Device
deviceId: string
location: : string
connect() : void
disconnect() : void
getDistance() : float
Map
mapID: string
location: coordinate
Path: List<Path>
mapType: string
displayPath(): void
zoomIn(): void
zoonOut(): void
setMapType(string): void
clearMap(): void
refreshMap(): void
Tracking Device
trackerID: string
location: coordinate
currentUser: string
trackingRange: Doubl
startTracking(): void
stopTracking(): void
updateLocation(): voi
sendLocation(): coorc
customizedMap
mapID: string
colorScheme: string
markerColor: string
showLegend: bool
showGridlines: bool
marker: coordinate
setColorScheme(string): vc
setMarker(coordinate) : voi
setGridlines(bool) : void
setMarkerColor(string) : vo
toggleLegend(bool): void
applyCustomization(map) : \
Path
pathID: string
points: List<coordinat
startTime: DateTime
EndTime: DateTime
distance: Double
stored(): Bool
savePath(): void
getDistance(): Double
displayPath(): Path
deletePath(): void
customizePath
pathID: string
pathColor: string
lineStyle: string
setPathColor(string: color)
setLineStyle(string: color) \
highlightPath(): void

SITE MAP

Below is the site map for BLU. The site is largely organized to navigate to two main pages: Access Previous Paths and Access Map. These have the most utility for a given user, so the site was designed to make it easy to access either one once logged on. The site map also displays paths that only admins can access.



LOGICAL DATABASE DESIGN

This is our database design. It stores information about users, paths, and devices. There are two device tables: registered devices and active devices. All active devices are registered devices, we have them in a separate table to easily query which devices are active at any given time. Users have sha hashes where their passwords are stored. In production these will be properly salted. Paths contain references to path files that store information about a path. They are associated with registered devices.