**SafetyMiles**

**Software Requirements Specification and Analysis Report**

**Submitted By**

**Busrat Jahan sabiha [BSSE-1311]**

**Nahida Sultana [BSSE-1318]**

**Supervised by**

**Dr. Emon Kumar Dey**

Professor,

Institute of Information Technology

University of Dhaka

**Date of Submission**

27 th March,2024

[SafetyMiles: Safe Walking Navigation App](#_wttkhfy7xxek) 5

[**1. Inception Report**](#_yivorngp59td) **7**

[1.1 Introduction](#_gk85t2s4kvp6) 7

[1.2 Inception Of this project](#_k4yx10mrfdfu) 8

[1.3 Identifying The Stakeholders](#_ilt0xk3g0xth) 9

[2. Elicitation of Safety Miles](#_jfn26e89okfe) 10

[2.1 Quality Function Deployment (QFD)](#_xln9amm94yyv) 10

[**2.2 User Scenario**](#_2dz1866zqqtw) **12**

[**3. Scenario Based Modeling**](#_61ip0ramjml7) **16**

[The Purpose of Use Case Diagram](#_v47bfdb8yh3k) 17

[Use Case Diagram : SafetyMiles](#_1vqbxy8gx2d9) 18

[Level 0:](#_7r7l3nfkih8f) 18

[Level 1:](#_h9h38vkccitz) 19

[Description of the Account Creation and Management Process](#_h941st7kxffe) 20

[Action-Replay Scenario](#_uhv9u3hotz6u) 20

[Action: User Registers an Account](#_dhdw7dwmsqx9) 20

[Action: User Verifies Email Address](#_f7hwo6lrimk4) 21

[Action: User Forgets and Recovers Password](#_1c92tet0flyj) 21

[Action: User Resets Password](#_gnhrpcoz9z9a) 21

[Description of the Location and Destination Input Process](#_mgp5mjzeefo2) 21

[Action-Reply Scenario](#_512dskayhks5) 22

[Action: User Opens the App to Plan a Trip](#_z67vy4oxunpx) 22

[Action: User Chooses Automatic Location Detection](#_95aul7e2dkx4) 22

[Action: User Begins Typing Destination](#_56u9wd1xl6sy) 22

[Action: User Selects Destination from Suggestions](#_7aq9hym6uex) 22

[Description of the Path Display and Safety Information Process](#_4zr9m9e0i2yi) 23

[Action -Reply Scenario](#_a5jt4hq28aq6) 23

[Action: User Inputs a Destination](#_b5d9klzenph8) 23

[Action: Display of Paths and Safety Information](#_d5vmu6g0o971) 23

[Action: User Reviews Safety Data and Selects a Path](#_mlzy4f45tg37) 24

[Action: User Commences Navigation](#_mijy27cz0vol) 24

[Description of the Submitted Safety Data Process](#_vypgndgkwhzx) 24

[Action- Reply Scenario](#_d6vak34cokcp) 25

[Action: User Encounters a Safety Incident](#_pdzisyssj374) 25

[Action: Admin Reviews the Submission](#_6q84u3tamwi7) 25

[Action: Report Verified and Database Updated](#_5huehcuaijff) 25

[Description of the Admin Verification and Database Update Process](#_2gi69lvblpq0) 26

[Action -Reply Scenario](#_sgzhd38qufpu) 26

[Action: Admin Receives a New Safety Report](#_k45phs55yruh) 26

[Action: Verification and Database Update](#_75o3xwh9qnsv) 27

[Action: Sending a Safety Advisory Alert](#_igft0lwzpq51) 27

[Description of the User Feedback Mechanism](#_8wsihdthti2q) 27

[Action- Reply Scenario](#_r2pfxnkmytmo) 28

[Action: User Submits Feedback](#_x5ns9y1auhqz) 28

[System Acknowledgment](#_vvjwgzm89zrh) 28

[Action: Development Team Addresses Feedback](#_slp82iuli64z) 28

[User Notification on Updates](#_rgl4hcjytajr) 28

[Level 1.1:](#_pdmnnllcau7w) 32

[Name: Account Management](#_rlyup0799iwk) 32

[Primary actor: User,Admin](#_v40allpzb92t) 32

[Secondary actor: Mail system,database.](#_8yggynwrhakw) 32

[Level 1.2:](#_xbcn7sufzdfn) 33

[Name: Input &Output](#_1rysj3sqc92i) 33

[Primary actor: User](#_o14ko6l78b2g) 33

[Secondary actor: Database,Open street map,auto tracking system,mail system.](#_xawcnhsnzy38) 33

[Level 1.3:](#_7fpjcrjaz7vn) 34

[Name: Safety Data Management](#_olbu87vde45y) 34

[Primary actor: User,Admin](#_4i43zrevfrpv) 34

[Secondary actor: Database,mail system.](#_jvgymsb8doq5) 34

[Level 1.4:](#_7qmpyfhlsq86) 35

[Name: Dashboard](#_rms952jn7zgh) 35

[Primary actor: User,Admin](#_i9izup23d0fo) 35

[Level 1.5:](#_fzv4ajhfjmqm) 36

[Name: Feedback](#_g66roypgvh4g) 36

[Primary actor: User,Admin](#_81yuk181f3ay) 36

[Secondary actor: Database.](#_6cb46evxkyt8) 36

[4.What is an Activity Diagram?](#_4ubcsavshu8e) 37

[The Purpose of Activity Diagram](#_v4dqybieiai8) 38

[Activity Diagram: Safety Miles](#_mj0mduy715rw) 38

[Level 1.1](#_23hcy7kjf3rf) 38

[5.What is a Swimlane Diagram?](#_4qx04dmbyiqi) 47

[The Purpose of Swimlane Diagram](#_9r6lw0p2xb62) 47

[Swimlane Diagram: Safety Miles](#_zdgm9igydxg1) 47

[**6.Database Modeling**](#_w6miwhm8v96h) **53**

[What is Database Modeling?](#_yusi08fzjaw8) 53

[ThePurpose of Database Modeling](#_94wlst6wr73z) 53

[List of data Object :](#_w660oaf1iox3) 59

[Relational Diagram](#_w50wzcop1mka) 64

[ER Diagram](#_u7gxso8xix5r) 65

[Relational Schema](#_namzztbbi85o) 66

[**7. Class Based Modeling**](#_kkhd6erwsqxs) **68**

[What is Class Based Modeling?](#_qdeyk7mh5o9j) 68

[Class Based Modeling:Safety Miles](#_3jxu353jm6xu) 69

[List of Nouns and Related Verb](#_rp60k3m4m2yy) 69

[General Classifications: Candidate classes are categorized based on the seven general classification. The analysis classes manifest themselves in one of the following ways:](#_xhopw4n7dzpt) 72

[Selection Criteria](#_rwaln3y5mv3n) 75

[Selected Classes](#_vp0a4v14hmu6) 76

[**8. Behavioral Modeling**](#_dv963lpwn36z) **86**

[What is Behavioral Modeling?](#_2y7v4ukeyz6h) 86

[State Transition Diagrams (State Diagrams)](#_hd6y0063lwma) 86

[**9. Data Flow Diagram (DFD)**](#_gm7skrwad7l1) **105**

[Level 0](#_lsi846i4572m) 106

[Level 1](#_gqha1pyfavx0) 107

[Level 2](#_ff0xtytixr26) 108

**10. Change log**………………………………………………………………………………………...113

**SRS DOCUMENTATION**

**Introduction:**

## **SafetyMiles: Safe Walking Navigation App**

SafetyMiles is a web application designed to help users navigate their surroundings safely, with a focus on walking routes.

Here are the key features:

* **Safe Routes:** It displays multiple walking route options with estimated distances.
* **Police Data Integration:** Uniquely, SafetyMiles integrates police data on incidents like theft and robbery, allowing users to choose the safest path.
* **Community-Driven Safety:** Users can report safety incidents, which admins verify before updating the database.
* **User-Friendly Interface:** The app offers features like location detection, search suggestions, and user feedback options.

SafetyMiles aims to empower users with safety information and foster a sense of community by encouraging participation in reporting incidents.

**Purpose :**

The purpose of the SafetyMiles app is to provide safe walking navigation by offering multiple routes with police data on incidents and allowing users to report safety concerns.

Unlike traditional mapping applications, SafetyMiles integrates real-time police data to highlight potential safety risks such as incidents of snatching, theft, and robbery. By offering users comprehensive safety information alongside route options, SafetyMiles empowers users to make informed decisions about their travel routes, ultimately enhancing their safety and security. Additionally, the app fosters a sense of community by allowing users to contribute safety incident reports and providing mechanisms for user feedback, creating a collaborative environment focused on improving road safety awareness and practices.

**Intended Audience :**

SafetyMiles targets a specific audience within urban environments who prioritize safety when walking:

* **Pedestrians who are concerned about their safety:** This includes people who might feel vulnerable walking alone, especially at night or in unfamiliar areas.
* **People who are new to an area:** SafetyMiles can be a valuable tool for those unfamiliar with the layout and potential safety issues of a new neighborhood or city.
* **People who walk for exercise or commuting:** This app caters to walkers who want to choose the safest route while getting their daily steps or reaching their destination efficiently.

**Conclusion :**

SafetyMiles offers a unique approach to safe walking navigation, combining route suggestions with police data and user-reported incidents. This empowers walkers, particularly those in unfamiliar areas or prioritizing safety, to make informed choices. With further development, SafetyMiles has the potential to become a go-to app for safe urban walking.

# **1. Inception Report**

## **1.1 Introduction**

Safety is a primary concern for pedestrians navigating urban environments. Existing navigation apps often lack a focus on safety, leaving users unaware of potential risks along their routes. SafetyMiles addresses this gap by offering a web-based application specifically designed to promote safe walking.

## **1.2 Inception Of this project**

**Problem Statement**

Pedestrians, especially those in unfamiliar areas or concerned about security, lack a user-friendly tool to identify safe walking routes. Traditional navigation applications prioritize distance and traffic information, neglecting safety considerations.

**Proposed Solution**

SafetyMiles is a web application that provides safe walking navigation for urban environments. Key features include:

* Integration of police data on incidents like theft and robbery with route suggestions.
* User-friendly interface for location detection, route selection, and safety information display.
* A community-driven safety reporting system allowing users to report incidents and contribute to a safer walking environment.

**Target Audience**

SafetyMiles targets pedestrians in urban areas who prioritize safety while walking, including:

* Individuals concerned about walking alone, particularly at night or in unfamiliar areas.
* People new to a city or neighborhood seeking safe walking paths.
* Walkers for exercise or commuting who want safe and efficient routes.

SafetyMiles has the potential to transform urban walking experiences by prioritizing safety and fostering a community-driven approach. By empowering users with informed route choices and promoting a culture of safety reporting, SafetyMiles can create a safer and more walkable environment for everyone.

## **1.3 Identifying The Stakeholders**

Identifying stakeholders is crucial for understanding who is involved or impacted by the project and what their roles and interests are. Here's a breakdown of the stakeholders involved in the "SafetyMiles" application:

**Stakeholders:**

**Users:** Users are individuals who utilize the SafetyMiles application to plan their journeys and ensure safety during travel. They create accounts, input their locations and destinations, access safety information, submit safety reports, provide feedback, and utilize the app's features for a secure journey. Users have a vested interest in the accuracy and reliability of the safety information provided by the app.

**Admin:** The admin plays a central role in managing various activities within the SafetyMiles application. They are responsible for user management, safety data verification, database management, route and safety information management, responding to user feedback, and sending notifications. Admins ensure the integrity and reliability of the app's information and features, enhancing user trust and satisfaction.

**Law Enforcement Agencies:** Since your app will incorporate real-time safety data, law enforcement agencies become important stakeholders. They provide this data, which could include crime reports, accidents, or other safety-related incidents. Collaborating with them ensures access to accurate and timely information, benefiting both users and the community.

Each stakeholder plays a distinct role in the development, operation, and utilization of the SafetyMiles application, contributing to its functionality, reliability, and effectiveness in enhancing user safety during travel. Understanding and addressing the needs and interests of these stakeholders is essential for the success and sustainability of the project.

## **2. Elicitation of Safety Miles**

The elicitation process for the Swimming Pool Management System involves systematically gathering requirements from stakeholders. Through interviews, surveys, observations, and collaborative sessions, we aim to understand user needs, identify pain points, and envision improvements. This comprehensive approach ensures that the final system is tailored to address the diverse requirements of users. The elicitation process is iterative, allowing for continuous refinement based on feedback and evolving insights, ultimately contributing to the successful development and implementation of an efficient and user-friendly map system solution.

## **2.1 Quality Function Deployment (QFD)**

Quality Function Deployment (QFD) is a structured approach to defining customer needs or requirements and translating them into specific plans to produce products to meet those needs. The “voice of the customer” is the term to describe these stated and unstated customer needs or requirements.

We talked to many people and got the following requirements for our automated system:

**Normal Requirements:**

**Safety Information Display:** Providing relevant safety data such as police incidents along routes.

**User-Friendly Interface**: Ensuring ease of use for all users, including account management and route selection.

**Real-Time Updates:** Delivering up-to-date information on safety incidents and route options.

**Secure Account Management:** Implementing robust measures for user authentication and data protection.

**Integration with Location Services:** Allowing for automatic detection of the user's current location.

**Expected Requirements :**

**Customizable Route Preferences:** Allowing users to set preferences for safety alerts and route options.

**Account Management :** Design a system to manage all users, verify accounts and authenticate them.

**User-friendly Interface :** Designing a user friendly interface so that one can easily use it.

**Database management system:** Maintain a database to store all path related information.

**Auto tracking system :** Including gps/ other tracking system to automatically detect the user's current location.

**Route Display :** Using open street maps to display possible routes.

**Exciting Requirements:**

1**.User submitted Safety data** : Providing a form and enabling users to share real-time safety updates and warnings with the community.

2.**Add favorite** : User can mark route as favorite for faster access.

**3. Feedback mechanism :** Users can provide feedback based on the performance of the application. They can provide ratings out of 5 stars and leave any comment.

**4. Notifications:** If there is any update on the features of the app, or any new incident has happened ,the user will be notified of that incident.

# **2.2 User Scenario**

***SafetyMiles***

“SafetyMiles” is a web based application software developed to provide a secure journey. Google Maps provides the layout of roads, the locations of cities and towns, state boundaries, geographical features, restaurant reviews and satellite images. Google provides the Street View perspective, allowing us to see houses, storefronts and points of interest from a driver’s point of view. In addition, Google Maps has indoor maps of some airports, museums and other facilities. But what about safety!!

We are concerned about the user's security. Therefore, “SafetyMiles” provides users with possible routes along with police data in order to ensure a safe and comfortable journey. This project only shows the walking distance.

As a user of SafetyMiles, he can navigate from one location to another safely, with an informed understanding of the safety landscape along with route.

**Account Creation and Management**

Initially, a user can create an account on SafetyMiles using his email address or mobile number and a password. After registering, he will receive an email to verify his address, ensuring his account is secured and activated. There will be an option for password recovery. In this system there is an admin to manage various activities. He logs in to the system providing personal information.

**Location and Destination Input**

When planning a trip, the app allows a user to input his current location manually or conveniently detects it automatically. As a user's type of destination, the app offers helpful suggestions, making the process of finding his/her destination faster and reducing the chance of input errors.

**Path Display and Safety Information**

Upon entering the destination, SafetyMiles impresses the user by displaying all possible paths he can take, each with an estimated walking distance. What sets SafetyMiles apart is its display of relevant police data for incidents like snatching, theft, and robbery along or near each path. This data is not only up-to-date but also clearly marked with dates, allowing me to make an informed decision about the safest route to take.

**Submitted Safety Data**

One can find value in the community aspect of SafetyMiles, where he can contribute by reporting new safety incidents or concerns. The process is user-friendly, with a submission form that prompts him for the incident type, location, date, time, and a brief description.The form looks like:

**Safety data submission form:**

| Username/email | Incident type | Location | Date | Time | Brief Description |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

Knowing that his report will be verified by an admin before updating the database ensures trust in the accuracy of the information provided on the app. Besides, the admin can input real-time data in the database.

**Admin Verification and Database Update**

It's reassuring to know that admins play a critical role in maintaining the integrity of SafetyMiles. They verify user-submitted reports, ensuring that only authentic safety information updates the paths displayed on the app. This verification process keeps the app reliable and trustworthy. Admin can send alerts or important information to all users, such as updates about new app features or advisories about specific areas based on recent safety data.

**User Feedback Mechanism**

The app values user’s feedback, offering a simple mechanism for him to share his experiences regarding the accuracy of safety data and the overall app experience. Users can rate this app according to its performance out of five (5)stars and can comment also.

**User Dashboard:**

* **Account Management:**
  + View/Edit Profile: Allows users to update their personal information such as name, email, and password.
  + Password Recovery: Option to recover forgotten passwords through email or mobile number verification.
* **Current Location & Destination:**
  + Manual Input: Users can manually input their current location or allow the app to detect it automatically.
  + Destination Input: Provides a search feature for users to input their destination conveniently, with suggestions for faster input and reduced errors.
* **Safety Data Submission Form:**
  + Username/Email: Input field for user identification.
  + Incident Type: Dropdown menu to select the type of incident (snatching, theft, robbery, etc.).
  + Location: Text field for specifying the incident location.
  + Date & Time: Fields to enter the date and time of the incident.
  + Brief Description: Text area for providing additional details about the incident.
* **Route Selection & Safety Info:**
  + Display of Possible Paths: Shows all available routes with estimated walking distances.
  + Safety Information: Alongside each path, displays relevant police data for incidents like snatching, theft, and robbery, marked with dates.
* **Feedback and Rating:**
  + Rating: Allows users to rate the app's performance on a scale of five stars.
  + Comment Section: Enables users to provide detailed feedback and suggestions for improvement.
* **History & Favorites:**
  + History: Records user's past routes and safety information accessed.
  + Favorites: Allows users to bookmark preferred routes for quick access.

**Admin Dashboard:**

* **User Management:**
  + View/Edit Users: Admin can access and manage user accounts, including profile details.
  + Account Verification: Option to verify user accounts to ensure authenticity.
* **Safety Data Verification:**
  + Review Submitted Reports: Admin can verify user-submitted safety incidents before updating the database.
  + Database Update: Ensures only authentic safety information is added to the app.
* **Database Management:**
  + Data Integrity: Admins maintain the integrity of SafetyMiles' database, ensuring accurate and up-to-date information.
  + Regular Updates: Admins can input real-time safety data and keep the database current.
* **Route and Safety Information Management:**
  + Path Updates: Admins manage route options and safety information displayed to users.
  + Ensure Accuracy: Verifies that safety information presented to users is reliable and relevant.
* **Respond to Feedback:**
  + Feedback Review: Admins can review user feedback and respond to concerns or suggestions.
  + Address Issues: Promptly address any issues or complaints raised by users.
* **Send Notifications:**
  + Important Alerts: Admins can send notifications to all users regarding app updates, new features, or safety advisories based on recent data.

Through these features and commitments, SafetyMiles not only enhances user’s safety but also empowers him with the knowledge to make informed decisions about his travel routes. The app stands out as a vital tool for navigating the complexities of urban environments, fostering a sense of community and shared responsibility for road safety.

# 

# **3. Scenario Based Modeling**

The success of a computer-based system or product is measured in many ways but user satisfaction resides at the top of the list. Understanding how actors want to interact with a system will lead the software team to properly characterize requirements and build meaningful analysis and design models. Hence, requirements modeling begins with the creation of scenarios in the form of use cases, activity diagrams, and swimlane diagrams.

**What is a Use Case Diagram?**

A use case is a written description of how users will perform tasks on your website. It outlines, from a user's point of view, a system's behavior as it responds to a request. Each use case is represented as a sequence of simple steps, beginning with a user's goal and ending when that goal is fulfilled. Use cases specify the expected behavior (what), and not the exact method of making it happen (how). Use cases once specified can be denoted both textual and visual representation (i.e. use case diagram). A key concept of use case modeling is that it helps us design a system from the end user's perspective. It is an effective technique for communicating system behavior in the user's terms by specifying all externally visible system behavior.

**Primary Actor**

Primary actors interact to achieve required system function and derive the intended benefit from the system. They work directly and frequently with the software.

**Secondary Actor**

Secondary actors support the system so that primary actors can do their work. They either produce or consume information.

## **The Purpose of Use Case Diagram**

The reasons why an organization would want to use case diagrams include:

● Represent the goals of systems and users.

● Specify the context a system should be viewed in.

● Specify system requirements.

● Provide a model for the flow of events when it comes to user interactions.

● Provide an outside view of a system.

● Shows external and internal influences on a system.

## **Use Case Diagram : SafetyMiles**

### **Level 0:**

**Name:** Safety Miles

**Primary Actor:** User, Admin.

**Secondary Actor:** Mail system, Open street map, database, auto tracking system.

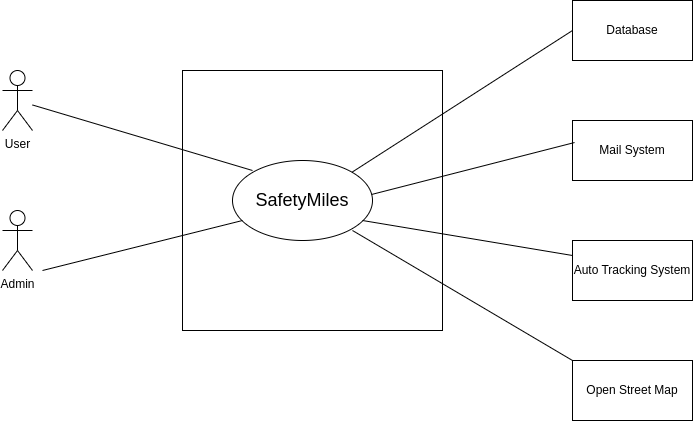


Figure 1:( Level- 0) Safety Miles

### **Level 1:**

**Name:** SafetyMiles details

**Primary Actor:** User,Admin

**Secondary Actor:** Mail system, Open street map, database, auto tracking system.

****

Figure 2: (Level -1) Safety Miles Details

### **Description of the Account Creation and Management Process**

1. **Account Creation:** A new user begins by providing a unique email address or mobile number along with a chosen password. This step initiates the creation of an account on SafetyMiles, a platform likely dedicated to promoting safety, possibly in transportation or travel contexts.
2. **Email Verification:** Once the account creation form is submitted, the system automatically sends a verification email to the provided email address. This email contains a link or a verification code. The purpose of this step is to confirm the ownership of the email address, ensuring it's valid and accessible by the person attempting to create the account. It adds a layer of security by verifying that the email address is not being used without consent.
3. **Account Activation:** The account is activated when the user clicks on the verification link or inputs the verification code into the platform. This step completes the registration process, confirming that the user has access to the email address used for registration. It prevents the creation of accounts with false or unauthorized email addresses.
4. **Password Recovery:** This feature allows users to recover or reset their password in case it is forgotten. Typically, the password recovery process involves the user requesting a password reset link, which is then sent to their registered email address. The user can follow this link to create a new password, thus regaining access to their account.

### **Action-Replay Scenario**

#### **Action: User Registers an Account**

* **User Action:** The user visits the SafetyMiles website or app, chooses to create a new account, enters their email address and a password, and submits the form.
* **System Reply:** "Thank you for registering with SafetyMiles! A verification email has been sent to [user's email address]. Please click on the verification link to activate your account."

#### **Action: User Verifies Email Address**

* **User Action:** The user checks their email, finds the verification email from SafetyMiles, and clicks on the verification link.
* **System Reply:** "Your email has been successfully verified, and your account is now active! Welcome to SafetyMiles."

#### **Action: User Forgets and Recovers Password**

* **User Action:** Sometime later, the user forgets their password and selects the "Forgot Password" option on the login page, entering their email address for recovery.
* **System Reply:** "A password reset link has been sent to your registered email address. Please follow the link to reset your password."

#### **Action: User Resets Password**

* **User Action:** The user clicks on the reset link received in their email, enters a new password, and submits it.
* **System Reply:** "Your password has been successfully reset. You can now log in to your account with your new password."

### **Description of the Location and Destination Input Process**

1. **Current Location Input:**
   * **Automatic Detection:** Upon permission from the user, the app uses the device’s GPS to automatically detect and set the current location. This feature is convenient, saving time and improving accuracy for the user’s starting point.
   * **Manual Input:** Users also have the option to manually enter their starting location. This is useful in scenarios where the automatic detection may not be accurate due to GPS signal limitations, or when planning a trip from a different location than the user's current one.
2. **Destination Input with Suggestions:**
   * As the user begins typing a destination, the app dynamically offers suggestions based on the input. This smart suggestion feature is designed to accelerate the search process, reduce typing effort, and minimize errors by helping users quickly find and select their intended destination from a list of possible matches. The suggestions are likely powered by a comprehensive database of locations, landmarks, and addresses, ensuring a wide range of destinations.

### **Action-Reply Scenario**

#### **Action: User Opens the App to Plan a Trip**

* **User Action:** The user launches SafetyMiles and chooses to plan a new trip.
* **System Reply:** "Let’s get started on your trip! Would you like us to use your current location or would you prefer to enter it manually?"

#### **Action: User Chooses Automatic Location Detection**

* **User Action:** The user selects the option for automatic location detection.
* **System Reply:** "We’re detecting your location… Location found! You’re starting at [User’s Current Location]. Where would you like to go?"

#### **Action: User Begins Typing Destination**

* **User Action:** The user starts typing the name of their destination.
* **System Reply:** As the user types, a dropdown menu appears with suggested destinations that match the user’s input. "Do you mean any of these locations?" followed by a list of suggestions.

#### **Action: User Selects Destination from Suggestions**

* **User Action:** The user selects their desired destination from the list of suggestions.
* **System Reply:** "Great choice! You’re headed to [Selected Destination]. Let’s plan your trip."

### **Description of the Path Display and Safety Information Process**

1. **Path Options Display:**
   * After the user inputs their destination, the app calculates and displays all possible routes to reach the destination. Each route is presented with an estimated walking distance, providing users with multiple options to choose from based on distance or personal preference.
2. **Integration of Police Data:**
   * Unique to SafetyMiles is its integration of up-to-date police data concerning safety incidents such as snatching, theft, and robbery that have occurred along or in the vicinity of each path. This data is meticulously organized and presented with relevant dates, offering a transparent overview of the safety record of each route.
3. **Informed Decision-Making:**
   * By combining route options with detailed safety information, users are empowered to make well-informed decisions about which path to take. This approach not only enhances the practical utility of the app for route planning but also instills a sense of confidence and security in users by prioritizing their safety.

### **Action -Reply Scenario**

#### **Action: User Inputs a Destination**

* **User Action:** After choosing their current location, the user enters their desired destination into SafetyMiles.
* **System Reply:** "Calculating routes to [Destination]. Please wait a moment."

#### **Action: Display of Paths and Safety Information**

* **User Action:** The system finishes calculating and displays the routes.
* **System Reply:** "Here are the possible paths to [Destination], each with an estimated walking distance. For your safety, we’ve also included recent police data on incidents like snatching, theft, and robbery near these paths. Please review to choose the safest route."

#### **Action: User Reviews Safety Data and Selects a Path**

* **User Action:** The user reviews the provided safety data for each path, noting the dates and types of incidents reported.
* **System Reply:** (Upon selection) "You’ve selected [Path Name] as your route to [Destination]. Safe travels! Remember, safety is paramount. Keep an eye on your surroundings and report any suspicious activity."

#### **Action: User Commences Navigation**

* **User Action:** The user starts their journey along the chosen path, possibly utilizing real-time navigation features of SafetyMiles.
* **System Reply:** "You are on your way to [Destination] via [Path Name]. Estimated walking distance: [X] minutes. Follow the path shown on your screen and stay alert."

### **Description of the Submitted Safety Data Process**

1. **User-Friendly Submission Form:**
   * Users who wish to report a safety incident can easily do so through a streamlined submission form available within the SafetyMiles app or website. This form is designed to be intuitive, encouraging user participation by simplifying the data entry process.
2. **Form Fields:**
   * The form requests essential details about the incident, including the user's username or email (for verification and follow-up purposes), the type of incident (e.g., theft, snatching, etc.), the exact location, date and time of the incident, and a brief description. This structured format ensures that all reports are comprehensive and useful for the community.
3. **Verification Process:**
   * To maintain the integrity and reliability of the data, each submission is reviewed by an admin. This verification process is critical to ensuring that the information provided is accurate and relevant before it is added to the SafetyMiles database. It helps build trust among users in the accuracy and timeliness of the safety information shared within the app.
4. **Real-Time Admin Input:**
   * Apart from user submissions, admins have the capability to input real-time data into the database, ensuring that the safety information on SafetyMiles is as current and comprehensive as possible.

### **Action- Reply Scenario**

#### **Action: User Encounters a Safety Incident**

* **User Action:** After witnessing or experiencing a safety incident, the user decides to report it via SafetyMiles. They access the safety data submission form and fill it out with all the required details.
* **System Reply:** "Thank you for submitting your report. Your contribution is invaluable to us and the SafetyMiles community. We will review the details for verification."

#### **Action: Admin Reviews the Submission**

* **Admin Action:** An admin reviews the submitted report to ensure it meets the verification standards and contains accurate and relevant information.
* **System to Admin Reply:** If the information is verified to be accurate, "The report has been verified and will now be updated in the database." If there are issues or more information is needed, "Please contact the user for further clarification."

#### **Action: Report Verified and Database Updated**

* **System Action:** Once the report is verified, it is added to the SafetyMiles database.
* **System to User Reply:** "Your report on [Incident Type] at [Location] on [Date] and [Time] has been verified and added to our database. Thank you for helping keep the SafetyMiles community informed and safe."

### **Description of the Admin Verification and Database Update Process**

1. **Verification of User-Submitted Reports:**
   * Admins receive notifications of user-submitted safety reports and begin the verification process. This involves cross-referencing the reported information with available data sources or reaching out to the user for additional details if necessary. The goal is to confirm the authenticity and accuracy of the report.
2. **Database Update:**
   * Once a report is verified, the admin updates the SafetyMiles database to reflect the new safety data. This update immediately impacts the paths displayed to users, ensuring the information about potential safety risks along certain routes is current.
3. **User Alerts and Information Broadcasts:**
   * Admins can send alerts or broadcast important information to all users of the app. These communications can range from notifications about new app features or enhancements to advisories about specific areas based on recent safety data or emerging threats.
4. **Maintaining Reliability and Trust:**
   * The diligent work of admins in verifying reports and updating the database is central to maintaining the app's reliability. It ensures that users can trust the safety information provided by SafetyMiles, making informed decisions about their travel routes.

### **Action -Reply Scenario**

#### **Action: Admin Receives a New Safety Report**

* **Admin Action:** A new user-submitted safety report is flagged for review.
* **System to Admin Reply:** "A new safety report has been submitted for verification. Please review the details and verify the authenticity of the report."

#### **Action: Verification and Database Update**

* **Admin Action:** After verifying the report's details, the admin updates the database to reflect the new safety information.
* **System Reply:** "The safety report has been verified and the database updated. Users will now see the updated safety information along their routes."

#### **Action: Sending a Safety Advisory Alert**

* **Admin Action:** Given a recent uptick in incidents in a specific area, the admin decides to send out a safety advisory to all users.
* **System to User Broadcast:** "Safety Advisory: Due to recent reports of increased incidents in [Area/Route], we advise all users to exercise heightened caution. Consider alternative routes if possible. Stay safe, and keep the SafetyMiles community informed."

### **Description of the User Feedback Mechanism**

1. **Rating System:**
   * Users can rate their experience with SafetyMiles on a scale from one to five stars. This straightforward rating system allows users to quickly convey their level of satisfaction with the app, offering a quantifiable measure of its performance.
2. **Comment Section:**
   * Beyond numerical ratings, users are encouraged to leave written comments. This section is designed for more detailed feedback, where users can share specific experiences, suggest improvements, or highlight features they find particularly useful or in need of refinement.
3. **Improvement and Development:**
   * The feedback collected through this mechanism is regularly reviewed by the SafetyMiles team. Insights gained from user ratings and comments are crucial for prioritizing development tasks, enhancing app features, and addressing any reported issues.
4. **Community Engagement:**
   * By valuing and acting upon user feedback, SafetyMiles reinforces its commitment to its user base. This ongoing dialogue between users and the development team strengthens trust and encourages a collaborative effort towards creating a safer navigation experience.

### **Action- Reply Scenario**

#### **Action: User Submits Feedback**

* **User Action:** After using the app for several weeks, a user decides to submit feedback. They rate the app 4 stars and leave a comment: "Really appreciate the safety data accuracy. However, I've encountered some issues with the app's responsiveness on my device."

#### **System Acknowledgment**

* **System Reply:** "Thank you for your feedback and for rating us 4 stars! We're glad to hear that you find the safety data accurate. Your comment regarding app responsiveness has been noted and will be forwarded to our technical team for review. We're committed to improving your experience."

#### **Action: Development Team Addresses Feedback**

* **Development Team Action:** The feedback about app responsiveness is evaluated and identified as a priority for the next update cycle.

#### **User Notification on Updates**

* **System to User Broadcast (After Update):** "We've made some improvements! Based on user feedback, our latest update includes enhancements to app responsiveness. Update your SafetyMiles app to enjoy a smoother experience. Thank you for helping us make SafetyMiles better for everyone."

**Description of the User Dashboard Scenario :**

***User Dashboard:***

**1. Account Management:**

* *Action:* User updates their email address in their profile.
* *Reply:* The user successfully edits their profile information, and a confirmation message is displayed.

**2. Current Location & Destination:**

* *Action:* User manually inputs their current location.
* *Reply:* The app detects the user's location and displays it on the map interface.

**3. Safety Data Submission Form:**

* *Action:* User submits a safety incident report regarding a theft near their neighborhood.
* *Reply:* The submission form is successfully completed, and the user receives a confirmation email acknowledging their report.

**4. Route Selection & Safety Info:**

* *Action:* User selects a route to their destination.
* *Reply:* Safety information along the chosen route is displayed, including recent police data, helping the user make an informed decision.

**5. Feedback and Rating:**

* *Action:* User rates the app with four stars and leaves a positive comment.
* *Reply:* The user's feedback is recorded, and a thank-you message is displayed, acknowledging their input.

**6. History & Favorites:**

* *Action:* User checks their past routes in the history section.
* *Reply:* The user views their recent routes, along with safety information accessed during those journeys.

**Admin Dashboard Action Replay Scenario:**

***Admin Dashboard:***

**1. User Management:**

* *Action:* Admin verifies a newly registered user's account.
* *Reply:* The user's account is successfully verified, and an email notification is sent confirming the verification.

**2. Safety Data Verification:**

* *Action:* Admin reviews a submitted safety report about a recent incident.
* *Reply:* After verification, the safety incident is approved, and the database is updated with the new information.

**3. Database Management:**

* *Action:* Admin updates the database with real-time safety data.
* *Reply:* The database is successfully updated, ensuring users have access to the latest safety information.

**4. Route and Safety Information Management:**

* *Action:* Admin adds a new route option for users.
* *Reply:* The new route is added to the app, providing users with additional navigation choices.

**5. Respond to Feedback:**

* *Action:* Admin responds to a user's feedback regarding a technical issue.
* *Reply:* The user receives a prompt response addressing their concern, along with steps to resolve the issue.

**6. Send Notifications:**

* *Action:* Admin sends a safety advisory notification to all users regarding a recent increase in theft incidents in a specific area.
* *Reply:* All users receive the notification, alerting them to exercise caution in the mentioned area.

### **Level 1.1:**

### **Name:** Account Management

### **Primary actor:** User,Admin

### **Secondary actor:** Mail system,database.

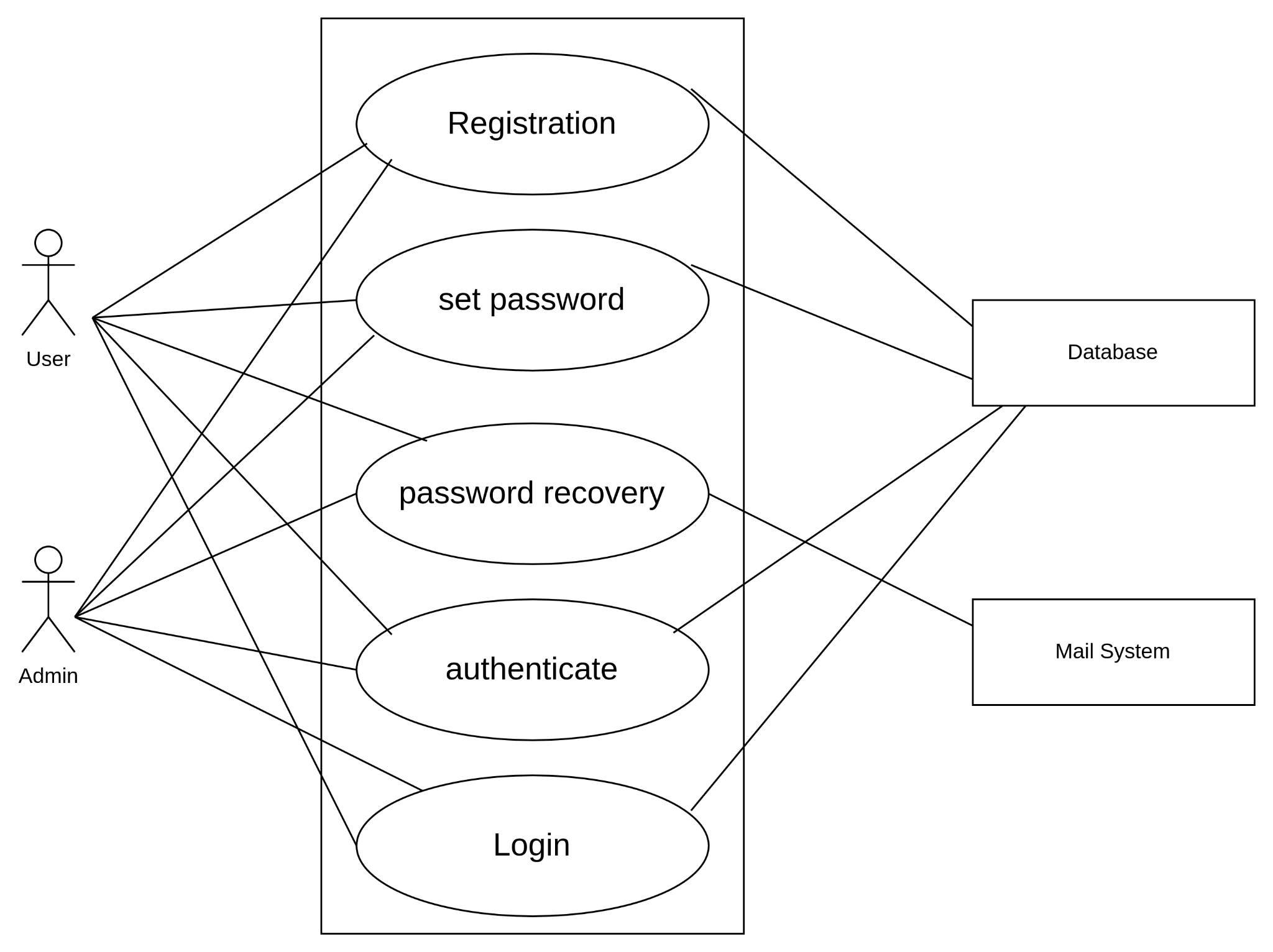


Figure 3 : (Level 1.1) Account Management module

### **Level 1.2:**

### **Name:** Input &Output

### **Primary actor:** User

### **Secondary actor:** Database,Open street map,auto tracking system,mail system.

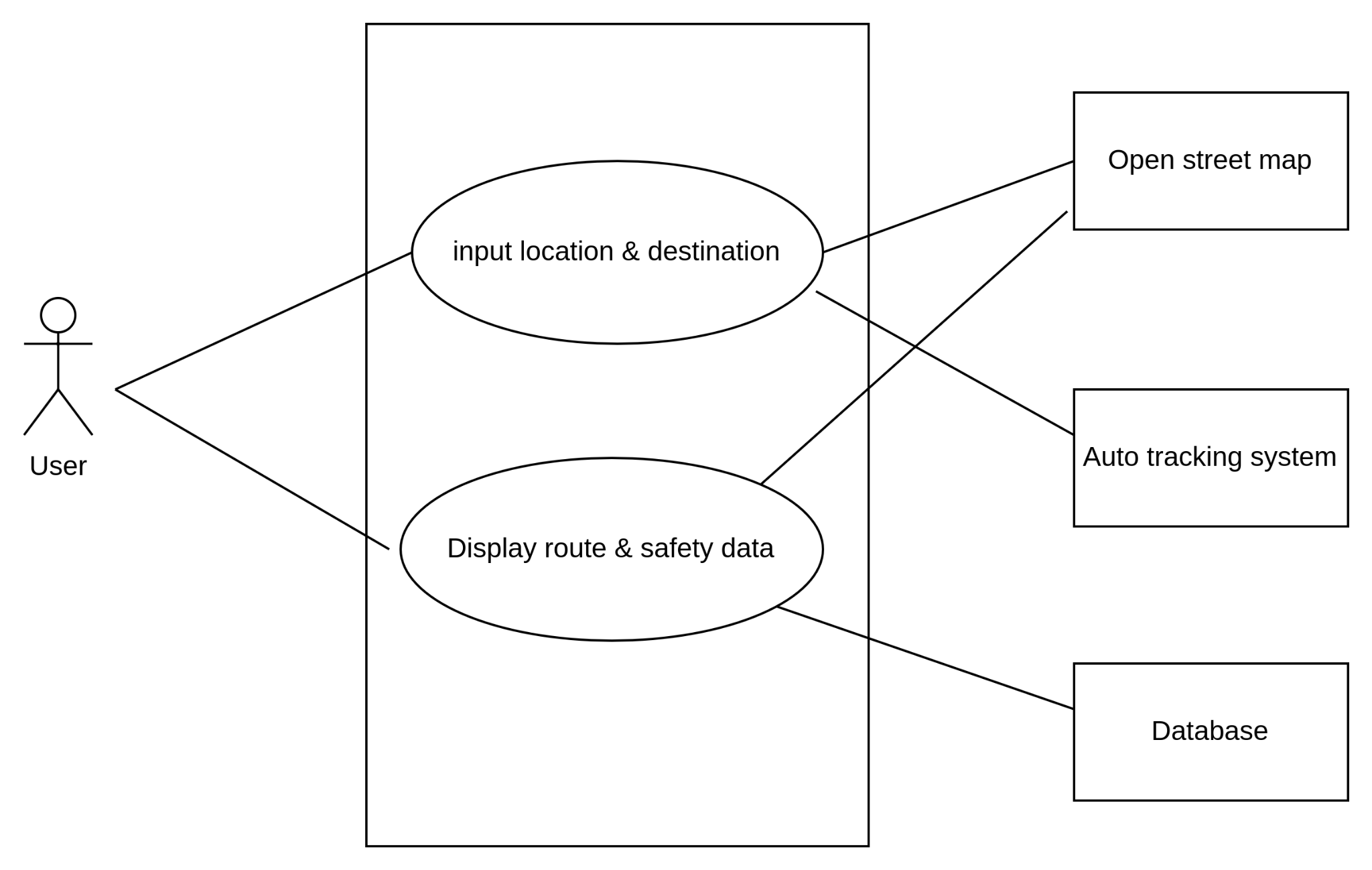


Figure -4 : (Level 1.2) Input & Output Module

### **Level 1.3:**

### **Name:** Safety Data Management

### **Primary actor:** User,Admin

### **Secondary actor:** Database,mail system.

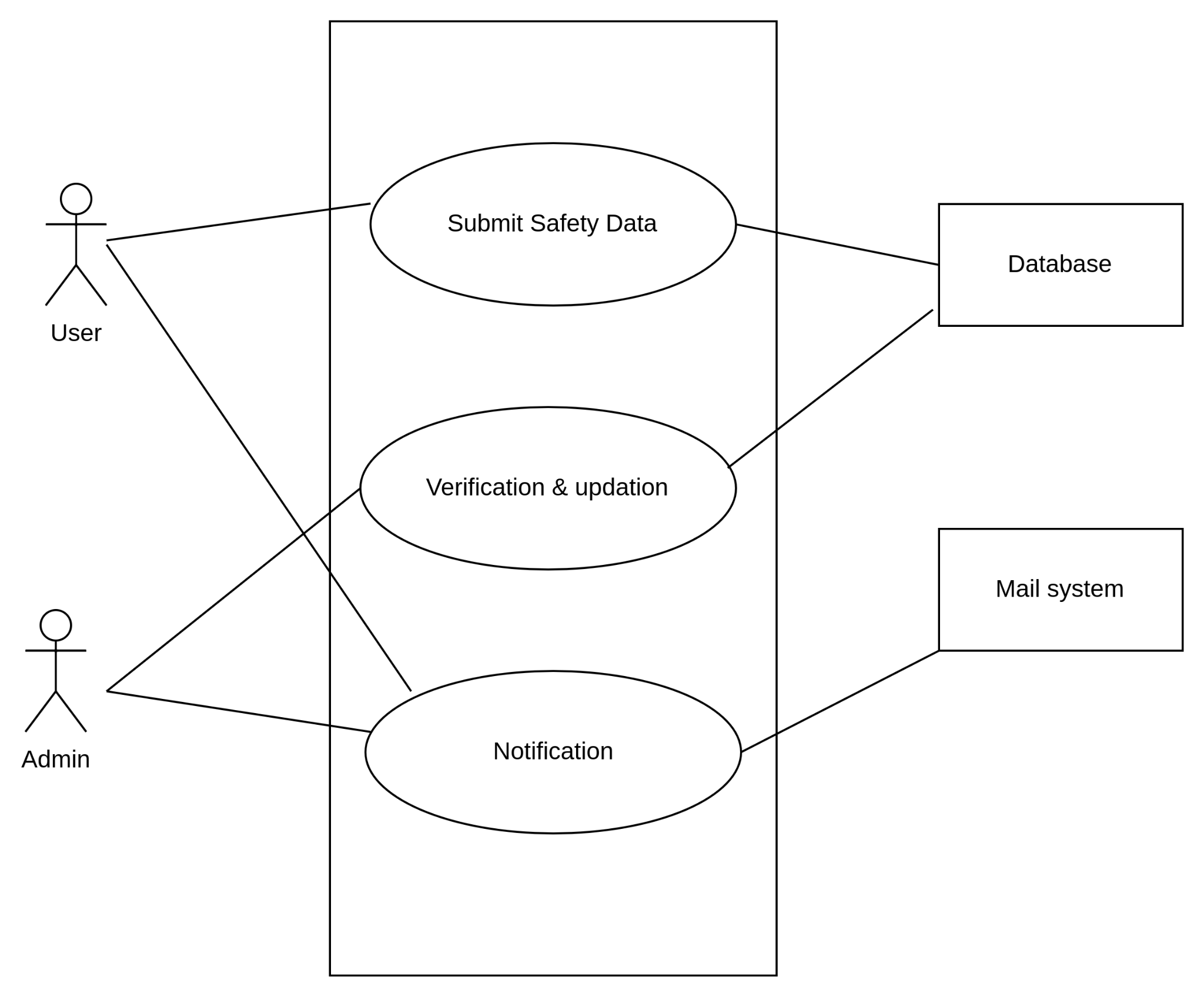


Figure 5 : (Level 1.3) Safety Data Management

### **Level 1.4:**

### **Name:** Dashboard

### **Primary actor:** User,Admin

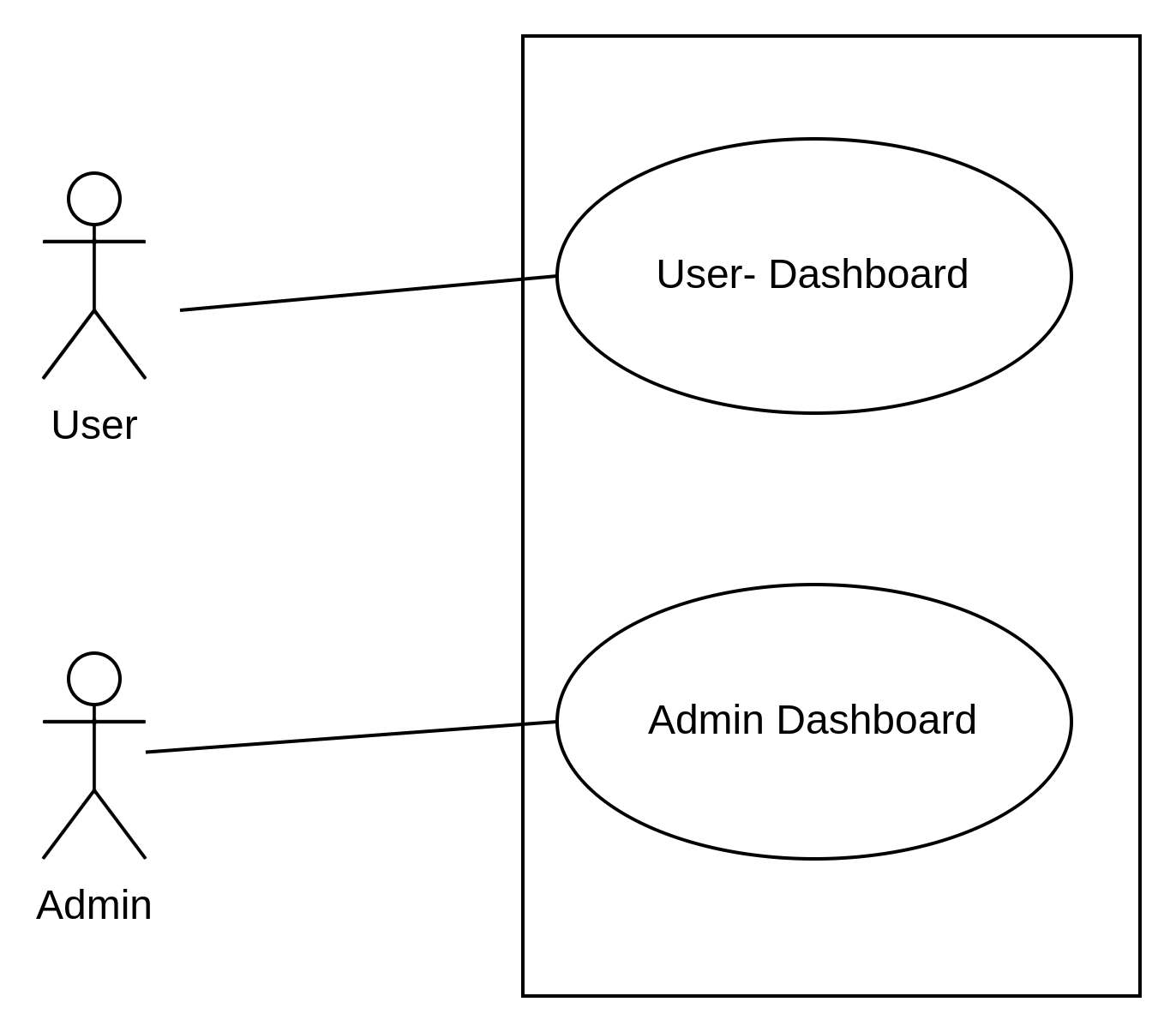


Figure 6: (Level 1.4) Dashboard

### **Level 1.5:**

### **Name:** Feedback

### **Primary actor:** User,Admin

### **Secondary actor:** Database.

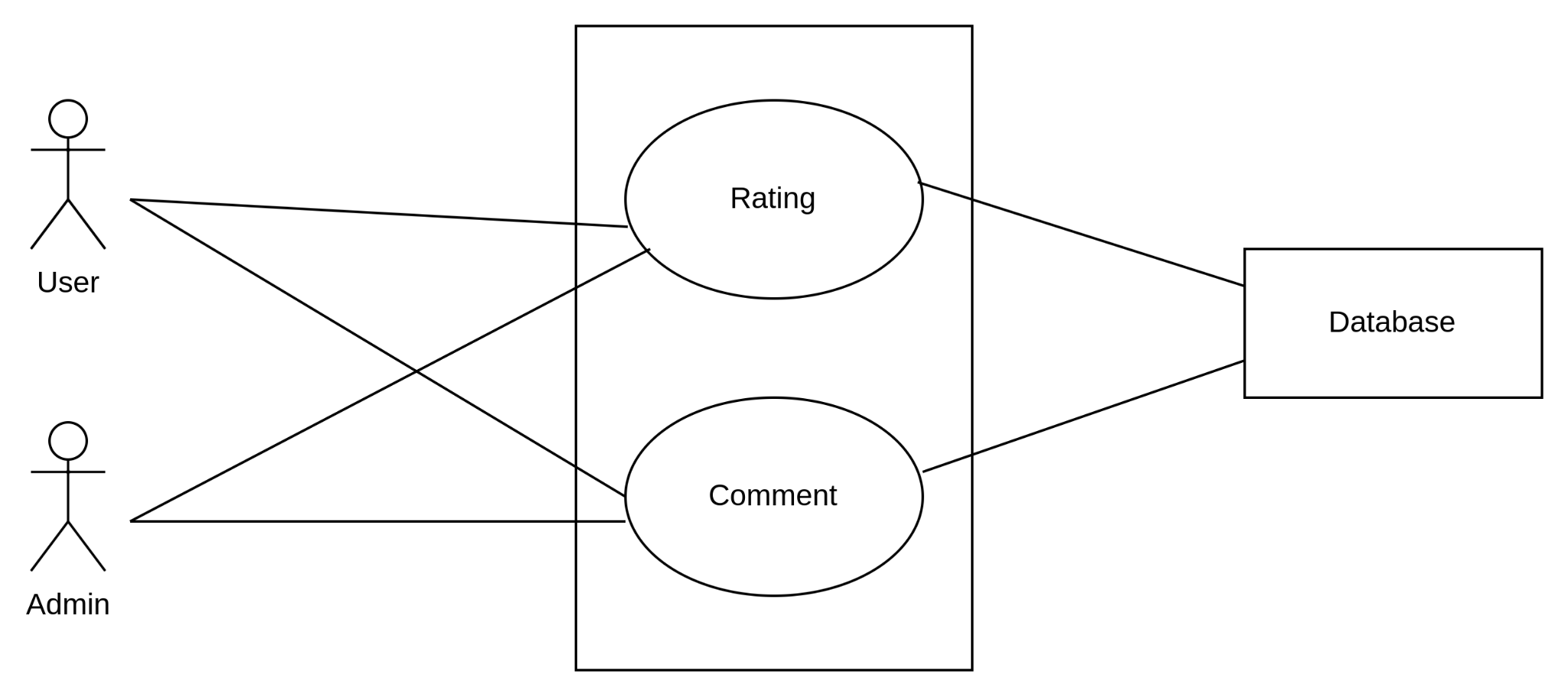


Figure 7 : (Level 1.5) Feedback

## **4.What is an Activity Diagram?**

An activity diagram is a type of diagram used in software engineering and requirements engineering to model the flow of activities and actions within a system or a business process. It provides a visual representation of the dynamic aspects of a system, showing how different activities interact and the sequence in which they occur.

## **The Purpose of Activity Diagram**

An activity diagram can prove to be useful to a software engineer for various reasons, such as:

* Drawing the activity flow of a system.
* Describing the sequence from one activity to another.
* Describing the parallel, branched and concurrent flow of the system.

## **Activity Diagram: Safety Miles**

To understand each of the processes involved in every module of the SafetyMiles, we have taken the help of activity diagrams. We have created the activity diagrams for the Activity diagram based on the Use Case Diagrams and the information we obtained.

### 

### **Level 1.1**

**Name:** Account Management

**Reference:** Use case diagram( level - 1.1)

### 

Figure 8 : Account Management

**Level 1.2**

**Name:** Input & Output

**Reference:** Use case diagram( level - 1.2)

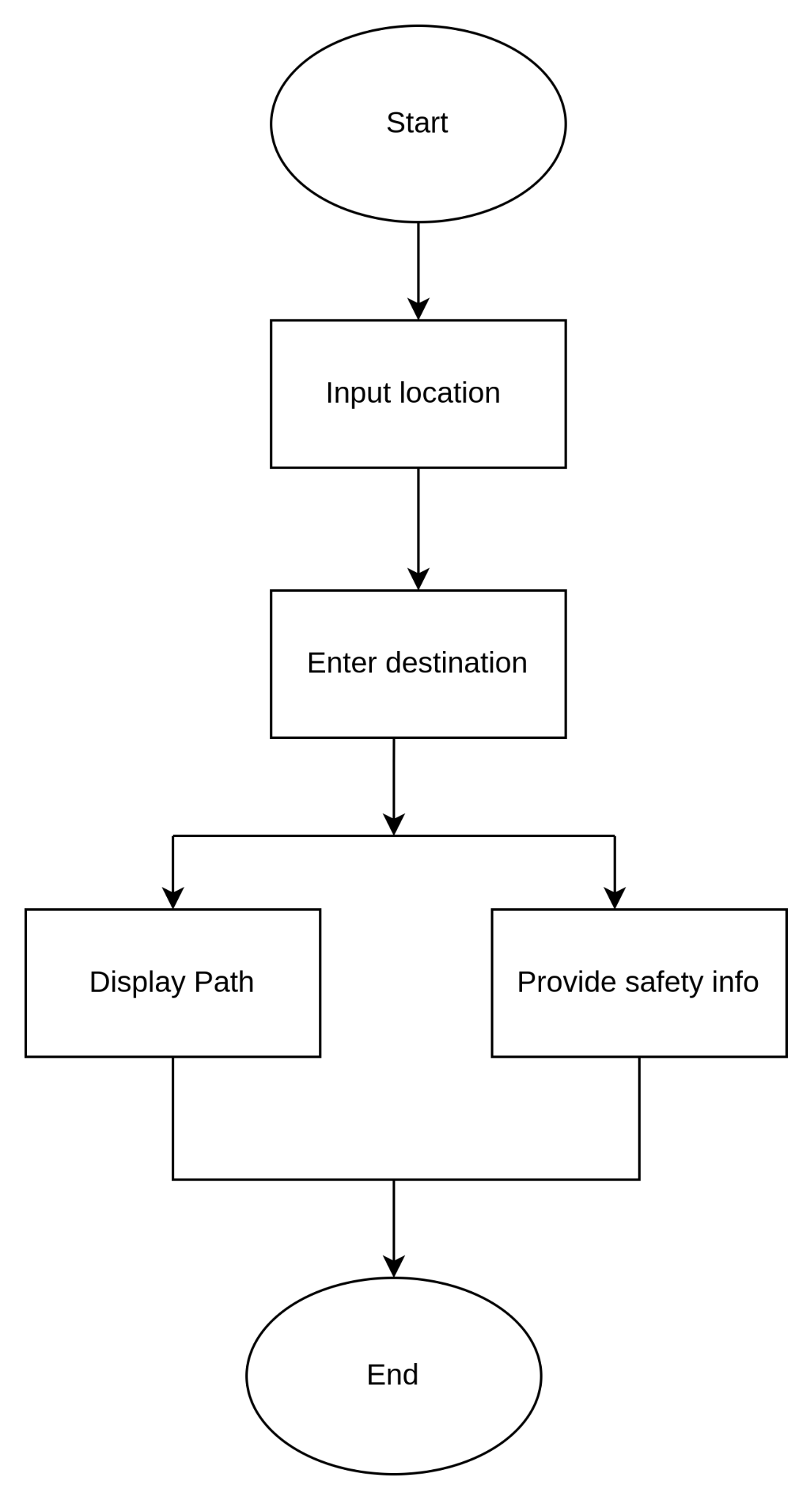


Figure 9: Input & Output

**Level 1.3**

**Name:** Safety Data Management

**Reference:** Use case diagram( level - 1.3)

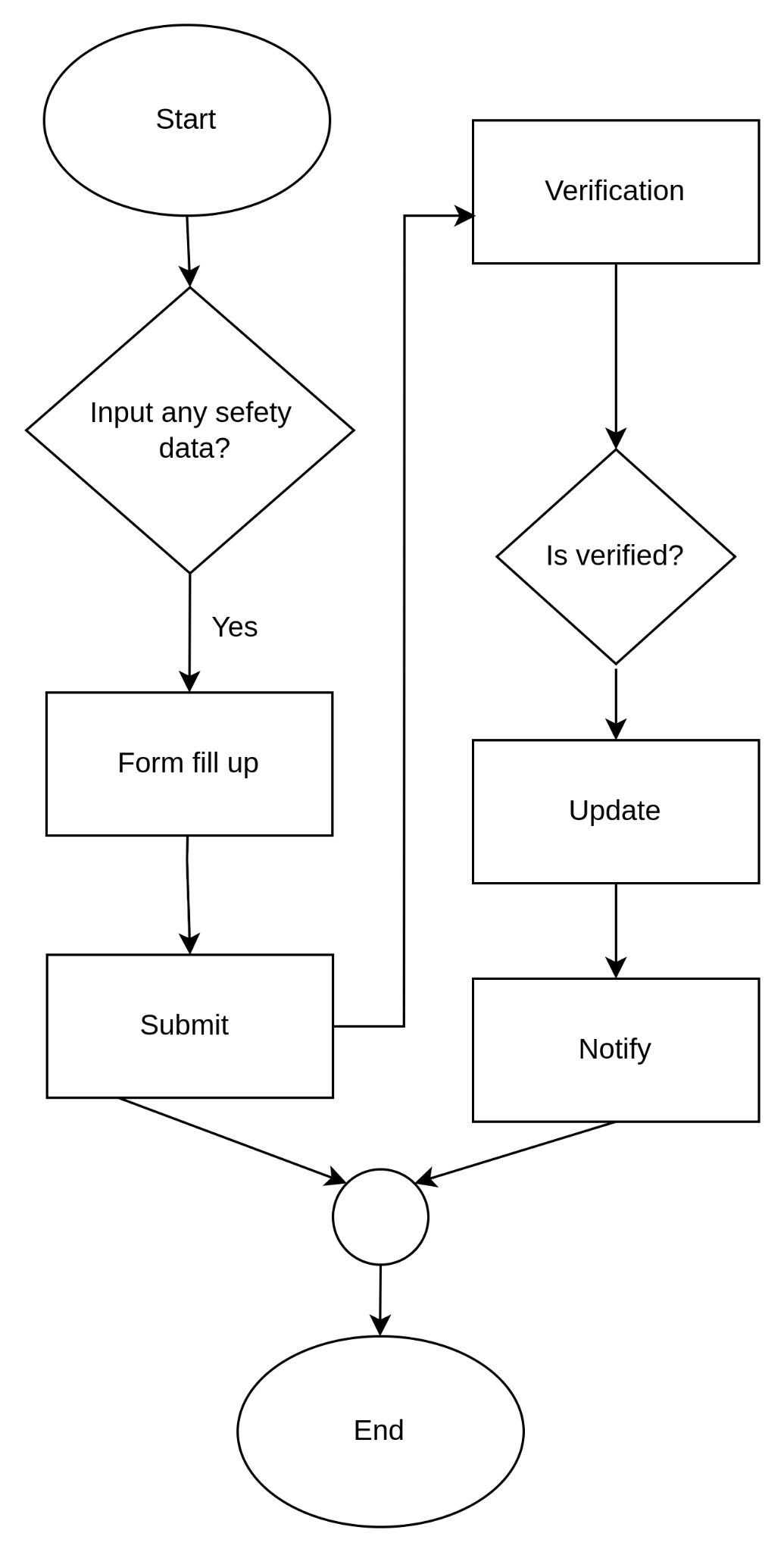
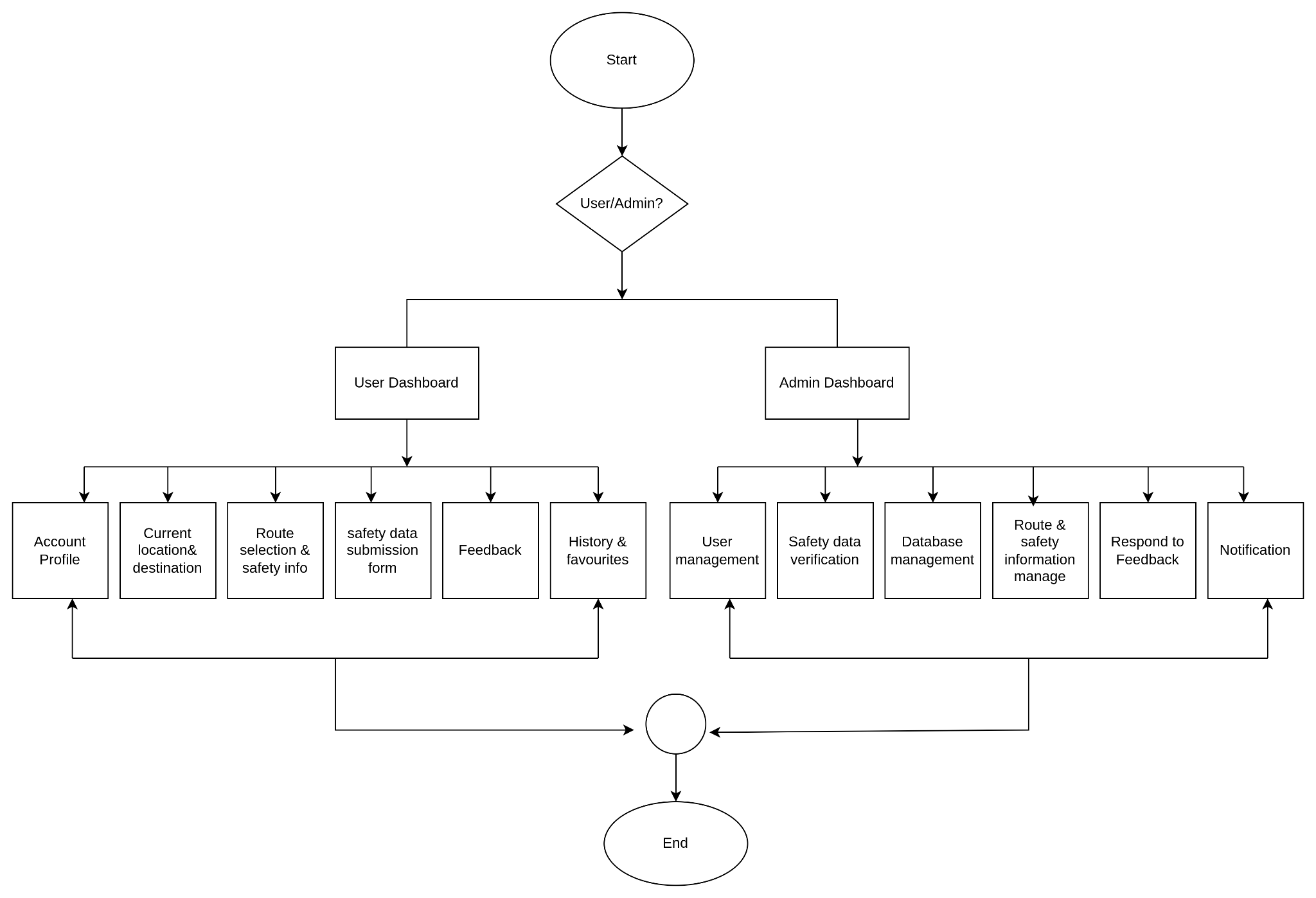


Figure 10 : Safety Data Management

**Level 1.4**

**Name:** Dashboard

**Reference:** Use case diagram( level - 1.4)



IFigure 11 : Dashboard

**Level 1.5**

**Name:** Feedback

**Reference:** Use case diagram( level - 1.5)



Figure 12 : Feedback

## **5.What is a Swimlane Diagram?**

A swimlane diagram is a type of flowchart, which diagrams a process from start to finish, but it also divides these steps into categories to help distinguish which departments or employees are responsible for each set of actions. It is based on the analogy of lanes in a pool, as it places process steps within the horizontal or vertical “swimlanes” of a particular department, work group or employee, thus ensuring clarity and accountability.

## **The Purpose of Swimlane Diagram**

A swimlane diagram provides various facilities to a software engineer, such as

* The separate lanes of the diagram make it easy to delineate responsibilities belonging to certain actors. This helps to clarify complex processes within the software.
* Visualizing processes in this way provides a more thorough overview of an actor’s roles within an organization and helps to reduce bottlenecks, redundancies, and extraneous steps.
* Ensures that everyone knows their specific role and avoids collisions.
* Helps to standardize work processes and record them in highly shareable formats that people within the organization can quickly refer to if needed.

## **Swimlane Diagram: Safety Miles**

**Level 1.1**

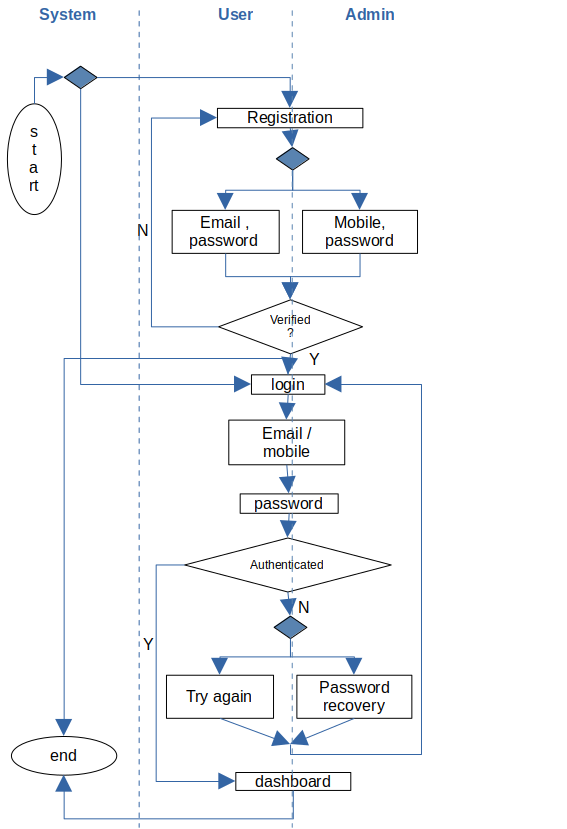


Figure 13 : Account management

**Level 1.2**

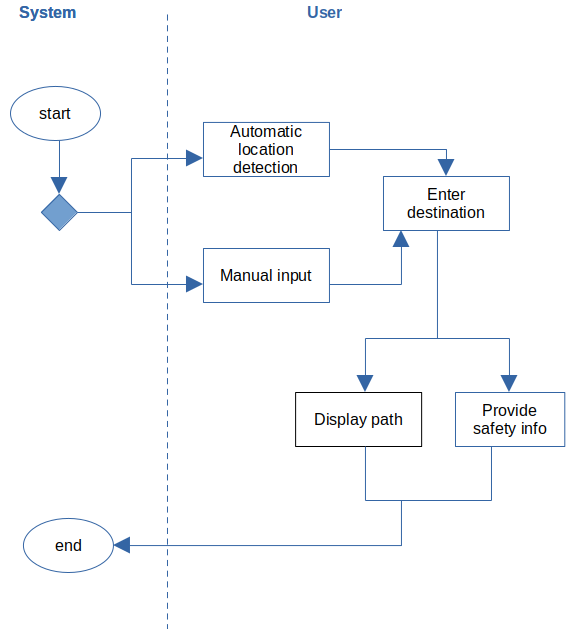


Figure 14 : Input & Output

**Level 1.3**

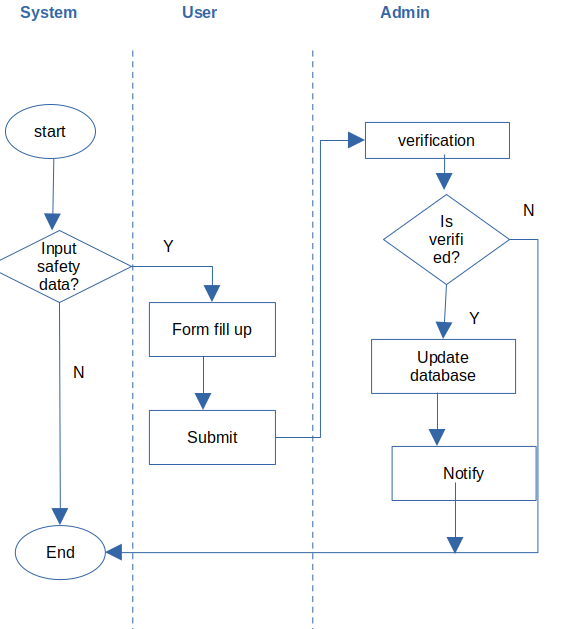
****

Figure 15: Submitted safety Data

**Level 1.4**

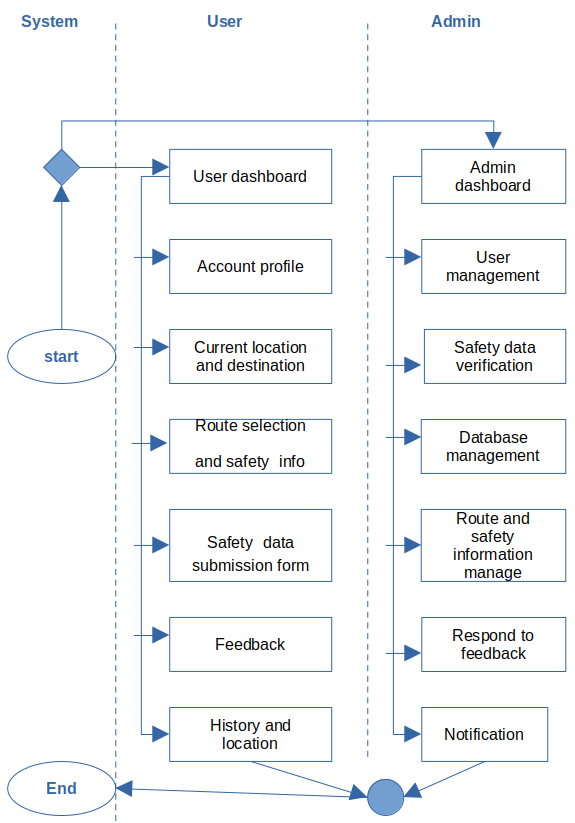
****

Figure 16 : Dashboard

**Level 1.5**

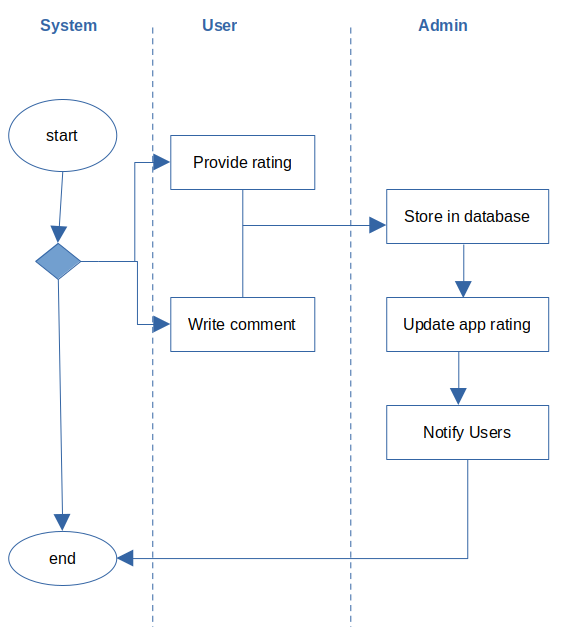
****

Figure 17 : Feedback

# **6.Database Modeling**

## **What is Database Modeling?**

Database modeling is the process of creating a detailed representation of a database structure. It involves defining the structure of the database, including tables, fields, relationships, and constraints. The goal of database modeling is to organize and structure data in a way that supports efficient storage, retrieval, and management of information. It serves as a blueprint for database development, helping developers design databases that meet the specific needs of an application or system. Database modeling can be visualized through diagrams such as Entity-Relationship Diagrams (ERD) that illustrate the entities, attributes, and relationships within the database.

## **ThePurpose of Database Modeling**

The purpose of data modeling can be summarized in several key points:

* Data modeling aids communication by visually representing how data is organized and related in a system.
* It serves as a design guide for developing databases, defining the structure and relationships among data elements.
* Data modeling clarifies business requirements, ensuring the database aligns with organizational needs.
* Ensures data integrity by setting constraints and rules for maintaining accurate and consistent information.

**Database Modeling: Safety Miles**

**Noun Listing :**

| No | Nouns | Attributes | P/S |
| --- | --- | --- | --- |
| 1 | SafetyMiles |  | P |
| 2 | Web |  | P |
| 3 | Application |  | P |
| 4 | Software |  | P |
| 5 | Journey |  | P |
| 6 | Google |  | P |
| 7 | Map |  | P |
| 8 | Layout |  | P |
| 9 | Road |  | P |
| 10 | Location | 11,13,30,44,112 | S |
| 11 | city |  | S |
| 12 | Town |  | P |
| 13 | State |  | S |
| 14 | Boundary |  | P |
| 15 | Feature |  | P |
| 16 | Restaurant |  | P |
| 17 | Review |  | P |
| 18 | Satellite |  | P |
| 19 | Image |  | P |
| 20 | Street |  | P |
| 21 | Houses |  | P |
| 22 | Driver |  | p |
| 23 | Indoor |  | p |
| 24 | Airport |  | p |
| 25 | Museum |  | p |
| 26 | Facility |  | p |
| 27 | Safety |  | S |
| 28 | User | 35,36,38,39,40,42,71,84,110,111 | S |
| 29 | Security |  | p |
| 30 | Route | 27,33,12,113 | S |
| 31 | Police data |  | p |
| 32 | Project |  | p |
| 33 | Walking distance |  | S |
| 34 | Landscape |  | p |
| 35 | Account |  | S |
| 36 | Creation |  | S |
| 37 | Management |  | p |
| 38 | Email address |  | S |
| 39 | Mobile no. |  | S |
| 40 | Password |  | S |
| 41 | Registration |  | p |
| 42 | Password recovery |  | S |
| 43 | Trip |  | p |
| 44 | Destination |  | S |
| 45 | Suggestions |  | p |
| 46 | Error |  | p |
| 47 | Path |  | p |
| 48 | Incident | 49-51,58-63,114,71 | S |
| 49 | Snatching |  | S |
| 50 | Theft |  | S |
| 51 | Robbery |  | S |
| 52 | Data |  | p |
| 53 | Up-to-date |  | p |
| 54 | Decision |  | p |
| 55 | Value |  | p |
| 56 | Community |  | p |
| 57 | user-friendly |  | p |
| 58 | Submission form |  | S |
| 59 | Incident type |  | S |
| 60 | Incident location |  | S |
| 61 | Incident date |  | S |
| 62 | Incident time |  | S |
| 63 | Incident’s description |  | S |
| 64 | Admin | 38,39,40,42,115 | S |
| 65 | Database | 87,108,111,115 | S |

| 66 | Trust |  | p |
| --- | --- | --- | --- |
| 67 | Accuracy |  | p |
| 68 | Information |  | p |
| 69 | App |  | p |
| 70 | Real-time-data |  | p |
| 71 | Verification |  | S |
| 72 | Updation |  | p |
| 73 | Integrity |  | p |
| 74 | Report |  | p |
| 75 | Alert |  | p |
| 76 | Area |  | p |
| 77 | Feedback | 80,83,110,111,116 | S |
| 78 | Mechanism |  | p |
| 79 | Experience |  | p |
| 80 | Rating |  | S |
| 81 | Performance |  | p |
| 82 | Stars |  | p |
| 83 | Comment |  | S |
| 84 | Profile |  | S |
| 85 | Menu |  | p |
| 86 | Text field |  | p |
| 87 | Details |  | S |

| 88 | Scale |  | p |
| --- | --- | --- | --- |
| 89 | Section |  | p |
| 90 | History | 110,111,113,119 | S |
| 91 | Favorites | 110,113,120 | S |
| 92 | Past routes |  | p |
| 93 | Bookmark |  | p |
| 94 | Access |  | p |
| 95 | Option |  | p |
| 96 | Authenticity |  | p |
| 97 | Issues |  | p |
| 98 | Complains |  | p |
| 99 | Notification | 110,111,117,118 | S |
| 100 | App updates |  | p |
| 101 | New features |  | p |
| 102 | Safety advisories |  | p |
| 103 | Commitment |  | p |
| 104 | Tool |  | p |
| 105 | Environment |  | p |
| 106 | Sense |  | p |
| 107 | Responsibility |  | p |
| 108 | Update Id |  | S |
| 109 | Safety data submission | 58-63,110,111,,112,121 | S |
| 110 | User Id |  | S |
| 111 | Date |  | S |
| 112 | Location Id |  | S |
| 113 | Route Id |  | S |
| 114 | Incident Id |  | S |
| 115 | Admin Id |  | S |
| 116 | Feedback Id |  | S |
| 117 | Notification Id |  | S |
| 118 | Message |  | S |
| 119 | History Id |  | S |
| 120 | Favorite Id |  | S |
| 121 | Submission Id |  | S |

### **List of data Object :**

| Data Object | Attributes |
| --- | --- |
| User | - UserID (Primary Key)  - Username/Email  - Mobile Number  - Password  - Account Verification Status  - Account Creation Date  - Last Login Date  - Profile Details (Name)  - Security Settings (Password Recovery Options) |
| Location | - LocationID (Primary Key)  - Latitude  - Longitude  - Address  - City  - State  - Country  - Type (Current, Destination, Incident Location)  - Route |
| Route | - RouteID (Primary Key)  - Start Location ID (Foreign Key)  - End Location ID (Foreign Key)  - Distance  - Estimated Time  - Safety Score |
| Incident | - IncidentID (Primary Key)  - Incident Type  - Location ID (Foreign Key)  - Date  - Time  - Description  - Verified Status  - Admin Notes |
| Admin | - AdminID (Primary Key)  - Username  - Password  - Email  - Mobile Number |
| Feedback | - FeedbackID (Primary Key)  - User ID (Foreign Key)  - Date  - Rating  - Comment |
| Notification | - NotificationID (Primary Key)  - User ID (Foreign Key)  - Date  - Message |
| History | - HistoryID (Primary Key)  - User ID (Foreign Key)  - Route ID (Foreign Key)  - Date |
| Favorite | - FavoriteID (Primary Key)  - User ID (Foreign Key)  - Route ID (Foreign Key) |
| Safety\_Data\_Submission | - SubmissionID (Primary Key)  - User ID (Foreign Key)  - Incident Type  - Location ID (Foreign Key)  - Date  - Time  - Description  - Verification Status |

**Analysis :**

Here we have added UserId,Location Id,Route Id,Incident Id,Admin Id,Feedback Id,Notification Id,History I'd,Favorite Id,Submission Id.

1.**User (UserID, Username/Email, Mobile Number, Password,:** This class stores user information, including their identification details, contact information, security credentials, and account management data. The primary key (UserID) uniquely identifies each user.

**2.Location (LocationID, Latitude, Longitude, Address):** This class manages location data, including latitude, longitude, address details, and location type (current, destination, incident location). It's linked to other tables using foreign keys to establish relationships.

**3.Route (RouteID, Start Location ID, End Location ID, Distance):** This class tracks route information, including start and end locations, distance, estimated time, and safety score. Routes are associated with specific locations and are essential for providing navigation guidance to users.

**4.Incident (IncidentID, Incident Type, Location ID, Date, Time ):** This class stores data related to safety incidents reported by users. It includes incident details such as type, location, date, time, description, and verification status. Admins verify these incidents before updating the database.

**5.Admin (AdminID, Username, Password, Email)**: This class manages admin accounts responsible for overseeing SafetyMiles operations. Admins can review user-submitted data, update the database, respond to feedback, and send notifications. Each admin has a unique identifier (AdminID).

**6.Feedback (FeedbackID, User ID, Date, Rating, Comment**): This class records user feedback, including ratings and comments on the app's performance and user experience. Feedback helps improve the app's functionality and user satisfaction.

**7.Notification (NotificationID, User ID, Date, Message**): This class handles notifications sent to users regarding app updates, safety advisories, or other important information. Notifications are associated with specific users and contain relevant messages.

**8.History (HistoryID, User ID, Route ID, Date**): This class maintains a record of users' past routes and accessed safety information. It helps users track their navigation history and facilitates quick access to previously used routes.

**9.Favorite (FavoriteID, User ID, Route ID**): This class allows users to bookmark preferred routes for quick access. It establishes a relationship between users and their favorite routes.

**10.SafetyDataSubmission (SubmissionID, User ID, Incident Type, Location ID, Date**): This class captures user-submitted safety incidents, including incident type, location, date, time, description, and verification status. Admins review and verify these submissions before updating the database.

### **Relational Diagram**

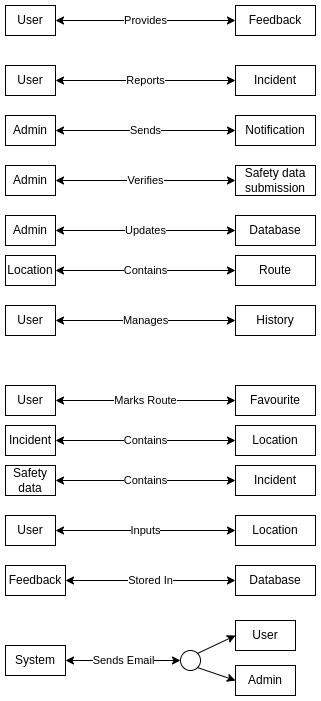


Figure 18 : Relational diagram

### **ER Diagram**

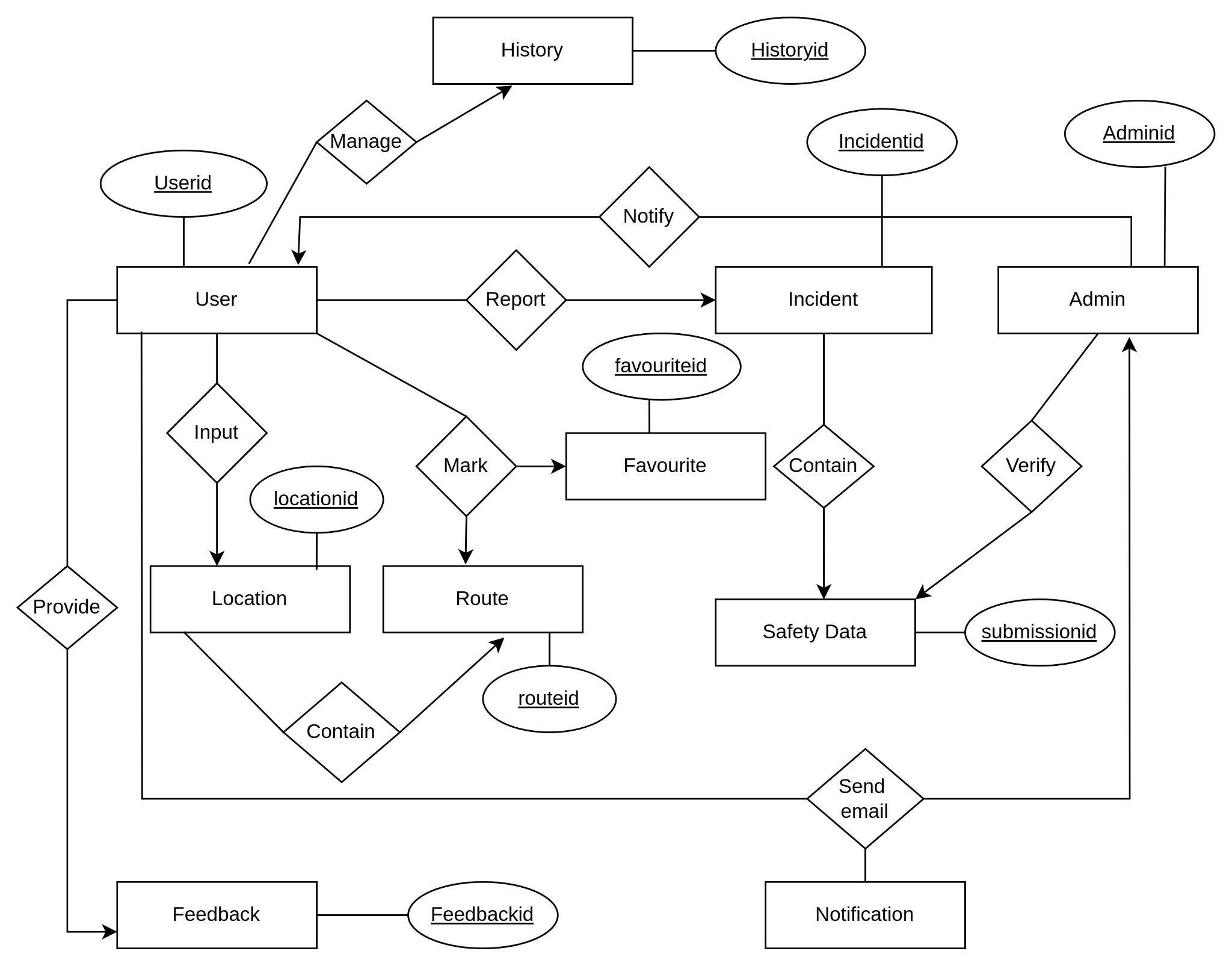


Figure 19 : ER Diagram

### **Relational Schema**

| Object | Attributes and Type |
| --- | --- |
| User | - UserID (Primary Key)(Integer)  - Username/Email(Varchar)  - Mobile Number(Integer)  - Password(Varchar)  - Account Verification Status(Boolean)  - Account Creation Date(Date)  - Last Login Date(Date)  - Profile Details (Name)(Varchar)  - Security Settings (Password Recovery Options) |
| Location | - LocationID (Primary Key)(Integer)  - Latitude(Number)  - Longitude(Number)  - Address(Varchar)  - City(Varchar)  - State(Varchar)  - Country(Varchar)  - Type (Current, Destination, Incident Location)(Varchar)  - Route (Route class type) |
| Route | - RouteID (Primary Key)(Integer)  - Start Location ID (Foreign Key)(Integer)  - End Location ID (Foreign Key)(Integer)  - Distance(Integer)  - Estimated Time(Number)  - Safety Score(Number) |
| Incident | - IncidentID (Primary Key)(Integer)  - Incident Type(Varchar)  - Location ID (Foreign Key)(Integer)  - Date(Date)  - Time(Number)  - Description(Varchar)  - Verified Status(Boolean)  - Admin Notes(Varchar) |
| Admin | - AdminID (Primary Key)(Integer)  - Username(Number)  - Password(Varchar)  - Email(Varchar)  - Mobile Number(Number) |
| Feedback | - FeedbackID (Primary Key)(Integer)  - User ID (Foreign Key)(Integer)  - Date(Date)  - Rating(Number)  - Comment(Varchar) |
| Notification | - NotificationID (Primary Key)(Integer)  - User ID (Foreign Key)(Integer)  - Date(Date)  - Message(Varchar) |
| History | - HistoryID (Primary Key)(Integer)  - User ID (Foreign Key)(Integer)  - Route ID (Foreign Key)(Integer)  - Date(Date) |
| Favorite | - FavoriteID (Primary Key)(Integer)  - User ID (Foreign Key)(Integer)  - Route ID (Foreign Key)(Integer) |
| Safety\_Data\_Submission | - SubmissionID (Primary Key)(Integer)  - User ID (Foreign Key)(Integer)  - Incident Type(Varchar)  - Location ID (Foreign Key)(Integer)  - Date(Date)  - Time(Number)  - Description(Varchar)  - Verification Status(Boolean) |

# **7. Class Based Modeling**

## **What is Class Based Modeling?**

Class-based modeling identifies classes, attributes, and relationships that the system will use. It represents the object. The system manipulates the operations. The elements of the class-based model consist of classes and objects, attributes, operations, and class – responsibility - collaborator (CRC) models. Classes are determined using underlining each noun or noun clause and entering it into the simple table. Attributes are the set of data objects that are defining a complete class within the context of the problem. The operations define the behavior of an object.

## **Class Based Modeling:Safety Miles**

### **List of Nouns and Related Verb**

| **Nouns** | **Verbs** |
| --- | --- |
| Location | Offer suggestion,display, |
| city |  |
| State |  |
| Safety |  |
| User | create,register,receive,recover,input,report,submit,rate |
| Route |  |
| Walking distance |  |
| Account |  |
| Creation |  |
| Email |  |
| Mobile no. |  |
| Password |  |
| Password recovery |  |
| Destination |  |
| Incident |  |
| Snatching |  |
| Theft |  |
| Robbery |  |
| Submission form | Is offered |
| Incident type |  |
| Incident location |  |
| Incident date |  |
| Incident time |  |
| Incident’s description |  |
| Admin | verifies,updates,maintain, |
| Database Update |  |

| Verification | Is done,keep |
| --- | --- |
| Feedback | Is provided |
| Rating | Is given |
| Comment | Is done |
| Profile | Is updated |
| Details | Is provided |

| History | Is recorded |
| --- | --- |
| Map | Is used |
| Favorites | Is maintained |
| Notification | Is sent |
| Update Id |  |
| Safety data submission | Is done |
| User Id |  |
| Date |  |
| System |  |
| Location Id |  |
| Route Id |  |
| Incident Id |  |
| Admin Id |  |
| Feedback Id |  |
| Notification Id |  |
| Message |  |
| History Id |  |
| Favorite Id |  |
| Submission Id |  |

### **General Classifications:** Candidate classes are categorized based on the seven general classification. The analysis classes manifest themselves in one of the following ways:

1. External entities

2. Things

3. Events

4. Roles

5. Organizational units

6. Places

7. Structures

A candidate class is selected for special classification if it fulfills three or more characteristics.

| **Nouns** | **General Classification** |
| --- | --- |
| Location | 2,6,7 |
| city | 1,6 |
| State | 1,6 |
| Safety | 2 |
| User | 4,5,7 |
| Route | 2,4,7 |
| Walking distance | 2 |
| Account | 2 |
| Creation | 4,7 |
| Email | 1,2 |
| Mobile no. | 2 |
| Password | 2 |
| Password recovery | 3 |
| Destination | 2 |
| Incident | 2,6,7 |
| Snatching | 2 |
| Theft | 2 |
| Robbery | 2 |
| Submission form | 2,6,7 |
| Incident type | 2 |
| Incident location | 6 |
| System | 3,4,7 |
| Incident date | 2 |
| Incident time | 2 |
| Incident’s description | 2,7 |
| Admin | 4,5,7 |
| Database Update | 3,4,7 |

| Verification | 3,7 |
| --- | --- |
| Feedback | 2,3,7 |
| Rating | 2 |
| Comment | 2 |
| Profile | 2,4,7 |
| Details | 2 |

| History | 2,3,7 |
| --- | --- |
| Map | 1 |
| Favorites | 2,3,7 |
| Notification | 2,3,7 |
| Update Id | 2 |
| Safety data submission | 2,4,7 |
| User Id | 2 |
| Date | 2 |
| Location Id | 2 |
| Route Id | 2 |
| Incident Id | 2 |
| Admin Id | 2 |
| Feedback Id | 2 |
| Notification Id | 2 |
| Message | 2 |
| History Id | 2 |
| Favorite Id | 2 |
| Submission Id | 2 |

### **Selection Criteria**

The candidate classes are then selected as classes by six Selection Criteria:

1. Retain information

2. Needed services

3. Multiple attributes

4. Common attributes

5. Common operations

6. Essential requirements

A candidate class generally becomes a class when it fulfills around three characteristics.

| **Potential Classes** | **Selection Criteria** |
| --- | --- |
| Location | 1,2,3 |
| User | 1,2,3,4,5 |
| Route | 1,2,3 |
| Admin | 1,2,3,4,5 |
| Submission form | 1,2 |
| Incident | 1,2,3 |
| Database update | 1,2,3 |
| History | 1,2,3,6 |
| Profile | 2,3 |
| Feedback | 1,2,3 |
| Notification | 1,2,3,6 |
| Favorite | 1,2,3 |
| System | 1,2,6 |
| Safety Data Submission | 1,2,3,6 |

### **Selected Classes**

1.User

2.Admin

3.Route

4.Feedback

5.History

6.Safety Data submission

7.Favourite

8.Notification

9.Database Update

10.Incident

11.Location

12.System

**Class-Card**

**1.User**

| **Attributes** | **Methods** |
| --- | --- |
| - UserID (Primary Key)(Integer)  - Username/Email(String)  - Mobile Number(Integer)  - Password(String)  - Account Verification Status(Boolean)  - Account Creation Date(Date)  - Last Login Date(Date)  - Profile Details (Name)(String)  - Security Settings (Password Recovery Options) | * createUserProfile(username: String, email: String, password: String): void * updateProfileDetails(userId: Integer, name: String, email: String, password: String): void * verifyAccount(userId: Integer): void * recoverPassword(emailOrMobile: String): void |

| **Responsibility** | **Collaborator** |
| --- | --- |
| Creating user profile | Database,System |
| Updating profile details | database,system |
| Verifying Account | Database,system |
| Recovering password | Database,system |

**2.Admin**

| Attributes | Methods |
| --- | --- |
| - AdminID (Primary Key)(Integer)  - Username(Number)  - Password(String)  - Email(String)  - Mobile Number(Number) | * adminLogin(username: String, password: String): boolean * verifySafetyIncident(submissionId: Integer): void * updateDatabase(adminId: Integer, description: String): void * sendNotificationToUsers(message: String): void |

| Responsibility | Collaborator |
| --- | --- |
| Logging in to system | Database,system |
| Verifying safety incident | user |
| Updating Database | database |
| Sending notification | User,Notification |

**3. Location**

| **Attributes** | **Methods** |
| --- | --- |
| - LocationID (Primary Key)(Integer)  - Latitude(Number)  - Longitude(Number)  - Address(Varchar)  - City(Varchar)  - State(Varchar)  - Country(Varchar)  - Type (Current, Destination, Incident Location)(String)  - Route (Route class type) | * getCurrentLocation(): Location * getDestinationLocation(): Location * getIncidentLocation(): Location |

| **Responsibility** | **Collaborator** |
| --- | --- |
| Showing current location | System |
| Showing destination location | System |
| Track incident location | System,Route |

**4.Route**

| **Attributes** | **Methods** |
| --- | --- |
| - RouteID (Primary Key)(Integer)  - Start Location ID (Foreign Key)(Integer)  - End Location ID (Foreign Key)(Integer)  - Distance(Integer)  - Estimated Time(Number)  - Safety Score(Number) | * calculateRoute(startLocation: Location, endLocation: Location): Route * displayPossiblePaths(route: Route): void * updateSafetyScore(route: Route, safetyScore: Number): void |

| **Responsibility** | **collaborator** |
| --- | --- |
| Calculating route | system,location |
| Displaying possible paths | system(mapping) |
| Updating safety score | database,system |

**5.Feedback**

| **Attributes** | **Methods** |
| --- | --- |
| - FeedbackID (Primary Key)(Integer)  - User ID (Foreign Key)(Integer)  - Date(Date)  - Rating(Number)  - Comment(string) | * submitFeedback(userId: Integer, rating: Number, comment: String): void * viewFeedback(userId: Integer): Feedback[] |

| **Responsibility** | **Collaborator** |
| --- | --- |
| Submitting feedback | database,system,user |
| Viewing feedback | database,admin |

**6.Favourite**

| **Attributes** | **Methods** |
| --- | --- |
| - FavoriteID (Primary Key)(Integer)  - User ID (Foreign Key)(Integer)  - Route ID (Foreign Key)(Integer) | * addFavorite(userId: Integer, routeId: Integer): void * removeFavorite(userId: Integer, routeId: Integer): void * viewFavorite(userId: Integer): Route[] |
|  |  |

| **Responsibility** | **Collaborator** |
| --- | --- |
| Adding favorite | database,system |
| Deleting favorite | database,system |
| Viewing favorites | database,Route |

**7.Safety Data Submission**

| **Attributes** | **Methods** |
| --- | --- |
| - SubmissionID (Primary Key)(Integer)  - User ID (Foreign Key)(Integer)  - Incident Type(Varchar)  - Location ID (Foreign Key)(Integer)  - Date(Date)  - Time(Number)  - Description(String)  - Verification Status(Boolean) | * submitSafetyIncident(userId: Integer, incidentType: String, locationId: Integer, date: Date, time: Number, description: String): void * viewSubmittedIncidents(userId: Integer): Incident[] |
|  |  |

| **Responsibility** | **Collaborator** |
| --- | --- |
| Submitting safety data | database,system,user |
| View submitted incidents | database,Admin,system,Incidents |

**8.Database Update**

| **Attributes** | **Methods** |
| --- | --- |
| - UpdateID (Primary Key)(Integer)  - Admin ID (Foreign Key)(Integer)  - Date(Date)  - Description(String) | * updateDatabase(adminId: Integer, description: String): void * viewDatabaseUpdates(): DatabaseUpdate[] |
|  |  |

| **Responsibility** | **Collaborator** |
| --- | --- |
| Updating database | Database,system,Admin |
| Viewing database updates | database,Admin,system |

**9.Incident**

| **Attributes** | **Methods** |
| --- | --- |
| - IncidentID (Primary Key)(Integer)  - Incident Type(Varchar)  - Location ID (Foreign Key)(Integer)  - Date(Date)  - Time(Number)  - Description(Varchar)  - Verified Status(Boolean)  - Admin Notes(string) | * verifyIncident(incidentId: Integer): void * addAdminNotes(incidentId: Integer, notes: String): void * viewIncidentDetails(incidentId: Integer): Incident   + Collaborator: Database |

| **Responsibility** | **Collaborator** |
| --- | --- |
| Verify incident | database,system,admin |
| Adding admin notes | User,Notification |
| Viewing incident details | Database,Incident,admin |

**10.Notification**

| **Attributes** | **Methods** |
| --- | --- |
| - NotificationID (Primary Key)(Integer)  - User ID (Foreign Key)(Integer)  - Date(Date)  - Message(string) | * sendNotification(userId: Integer, message: String): void * viewNotifications(userId: Integer): Notification[] |

| **Responsibility** | **Collaborator** |
| --- | --- |
| sending notification | System,admin,User |
| Viewing notification | User,admin |

**11.History**

| **Attributes** | **Methods** |
| --- | --- |
| - HistoryID (Primary Key)(Integer)  - User ID (Foreign Key)(Integer)  - Route ID (Foreign Key)(Integer)  - Date(string) | * viewUserHistory(userId: Integer): History[] * clearHistory(userId: Integer): void |

| **Responsibility** | **Collaborator** |
| --- | --- |
| View user history | Admin,user |
| Clear history | User,database |

**12.System**

| **Attributes** | **Methods** |
| --- | --- |
| **-**emailAddress(String)  - trackingSystem(Structure)  -message(string)  - map(Structure) | * sendEmail(emailAddress: String, message: String): void * displayMessage(message: String): void |

| **Responsibility** | **Collaborator** |
| --- | --- |
| Sending email | Admin,user,Notification |
| Display message | database,Notification |

**CRC Modelling :**

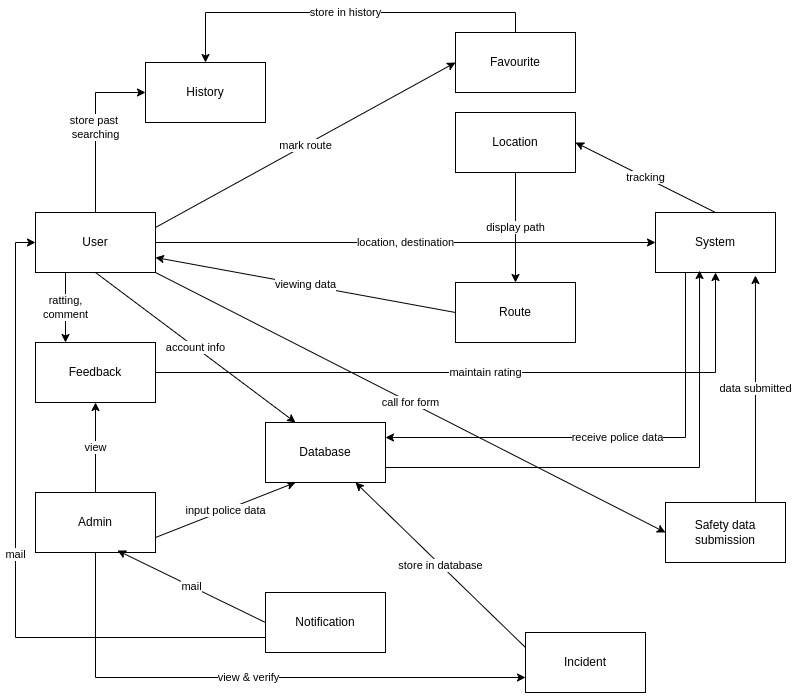
****

Figure 20 : CRC Modelling

# **8. Behavioral Modeling**

## **What is Behavioral Modeling?**

Behavioral modeling in the context of a Software Requirements Specification (SRS) is an approach that focuses on specifying how a software system behaves or should behave in response to various events, inputs, or conditions. It aims to capture the dynamic aspects of the system—how it operates over time, reacts to inputs, and transitions between different states. This type of modeling is crucial for understanding system functionality from an external point of view and for specifying functional requirements. Two commonly used behavioral modeling techniques are **State Transition Diagrams** and **Sequence Diagrams.** Each serves a unique purpose in illustrating the system's behavior and helping stakeholders understand the system's operations.

### **State Transition Diagrams (State Diagrams)**

**Definition:** State Transition Diagrams, also known as State Diagrams, are a form of behavioral modeling that depicts the states an entity (such as an object or a system) can occupy and the transitions between those states caused by events. These diagrams are particularly useful for modeling objects that have a limited number of states but can undergo significant changes in state in response to events.

**Purpose:**

* Modeling life cycles of objects.
* Understanding how the system reacts to sequential events.
* Specifying system behavior for different states, which is crucial for testing and implementation.

**Key Components:**

* **States:** Represented by rectangles or circles, indicating different conditions or statuses of an object/system.
* **Transitions:** Arrows that connect states, showing how the system moves from one state to another in response to events or conditions.
* **Events:** Triggers that cause transitions from one state to another.
* **Initial and Final States:** Special symbols indicating the start and end points of the system's life cycle.

**State Transition Diagrams: Safety Miles**

**List Of Events**

| **Initiator** | **Events** | **Collaborator** |
| --- | --- | --- |
| User | Create an account | Database, System |
| User | Submits email address or mobile number and a password | Databas, System |
| Admin | logs in to the system providing personal information. | Database, System |
| User | Verifies his account | Database, System |
| User | requests password recovery. | System |
| User | sends password reset link | System |
| User | View and update profile | System, database |
| User | Inputs current location | Location,  System |
| User | inputs destination, and the app offers suggestions | History,Location,  System |
| History | Records user's past routes and safety information accessed. | User |
| Database | displays possible paths with safety information | Location,Route |
| User | submits a safety incident report | System, Location |
| Admin | verifies Safety Incident | User, Incident |
| Admin | updates the database with safety data | Database |
| Admin | inputs real-time safety data | Database |
| User | submits feedback and ratings for the app | Feedback |
| User | Marks route as favorite | Favorite,  History |
| Favorite | Allows users to bookmark preferred routes for quick access. | User |
| Admin | reviews user feedback and addresses any issues | Feedback |
| Admin | ends alerts or information to users | Notification |

**State Transition Diagram :**

**Name : User**

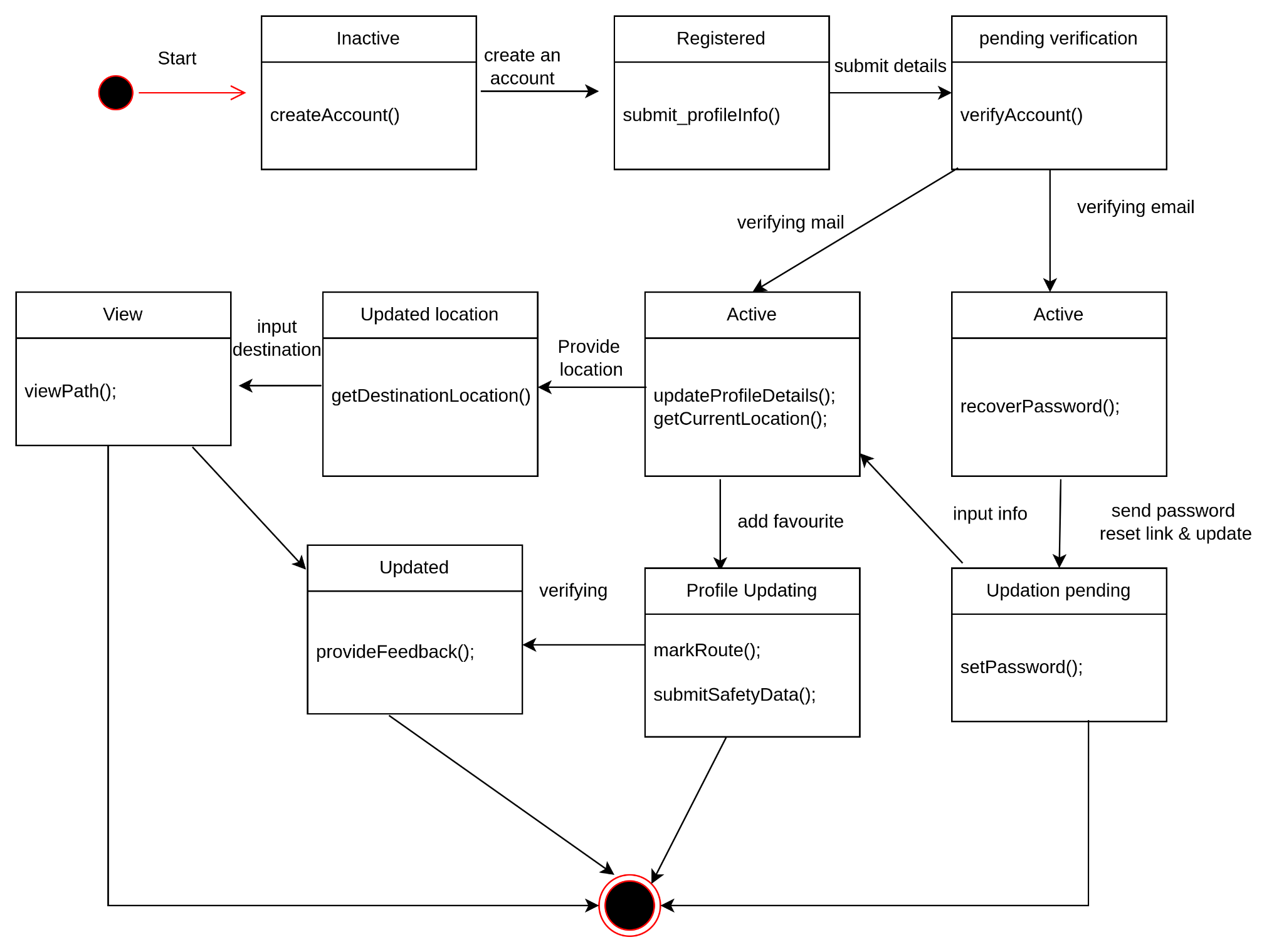
****

Figure 21 : User state transition diagram

**Name : Admin**

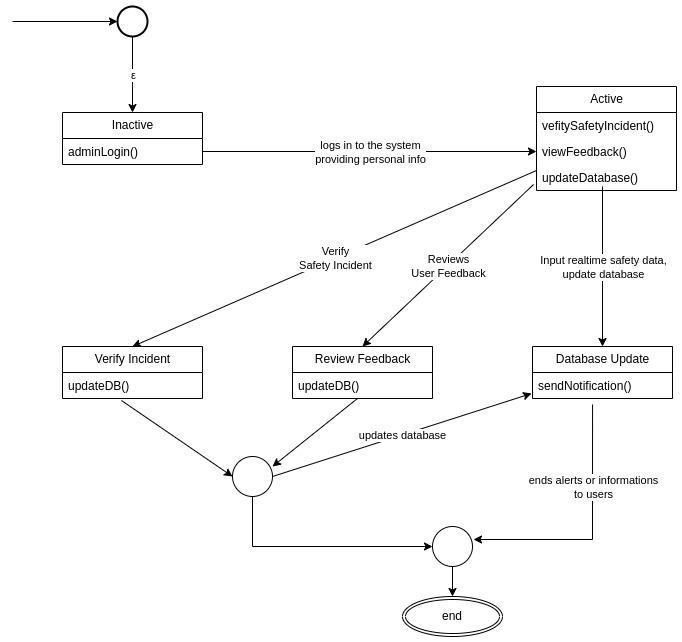
****

Figure 22 : Admin state transition diagram

**Name : Location**

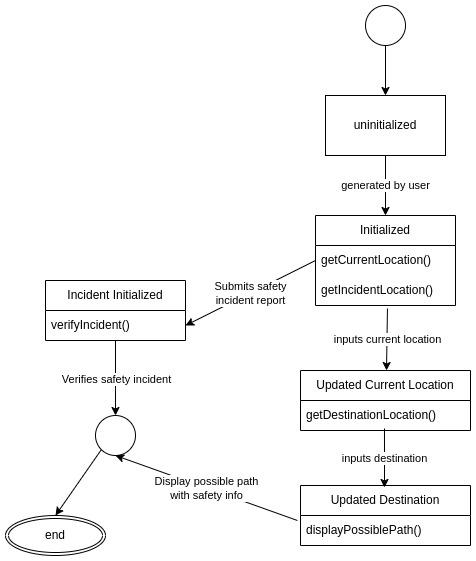
****

Figure 23 : Location state transition diagram

**Name : Route**

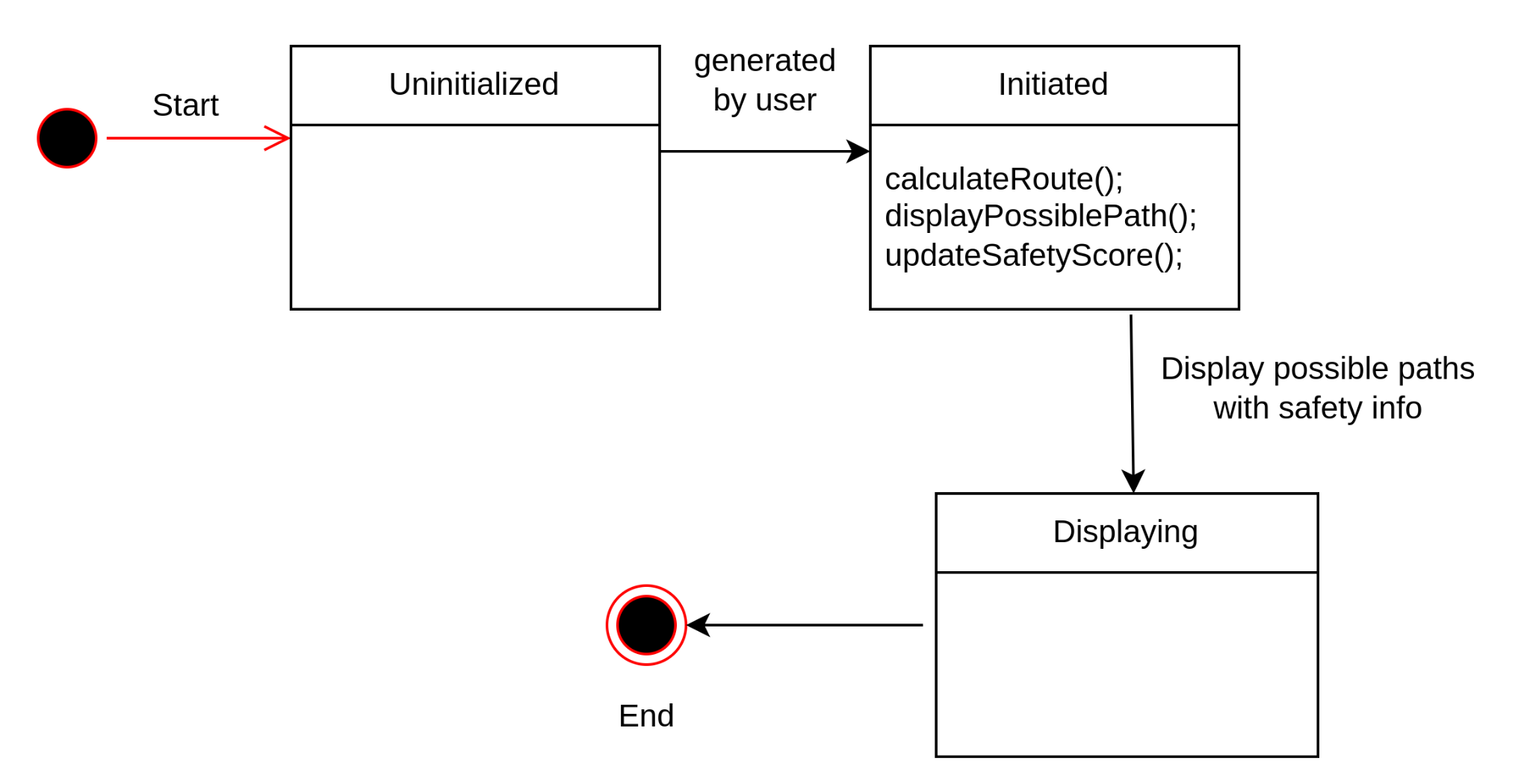
****

Figure 24 : Route state transition diagram

**Name : History**

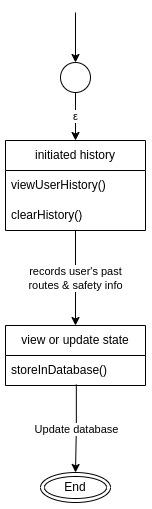
****

Figure 25 : History state transition diagram

**Name : Favorite**

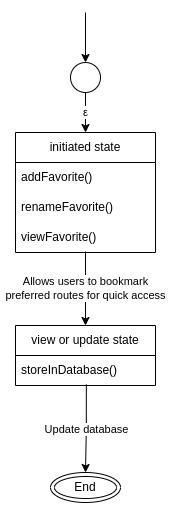
****

Figure 26 : Favorite state transition diagram

**Name : Incident**

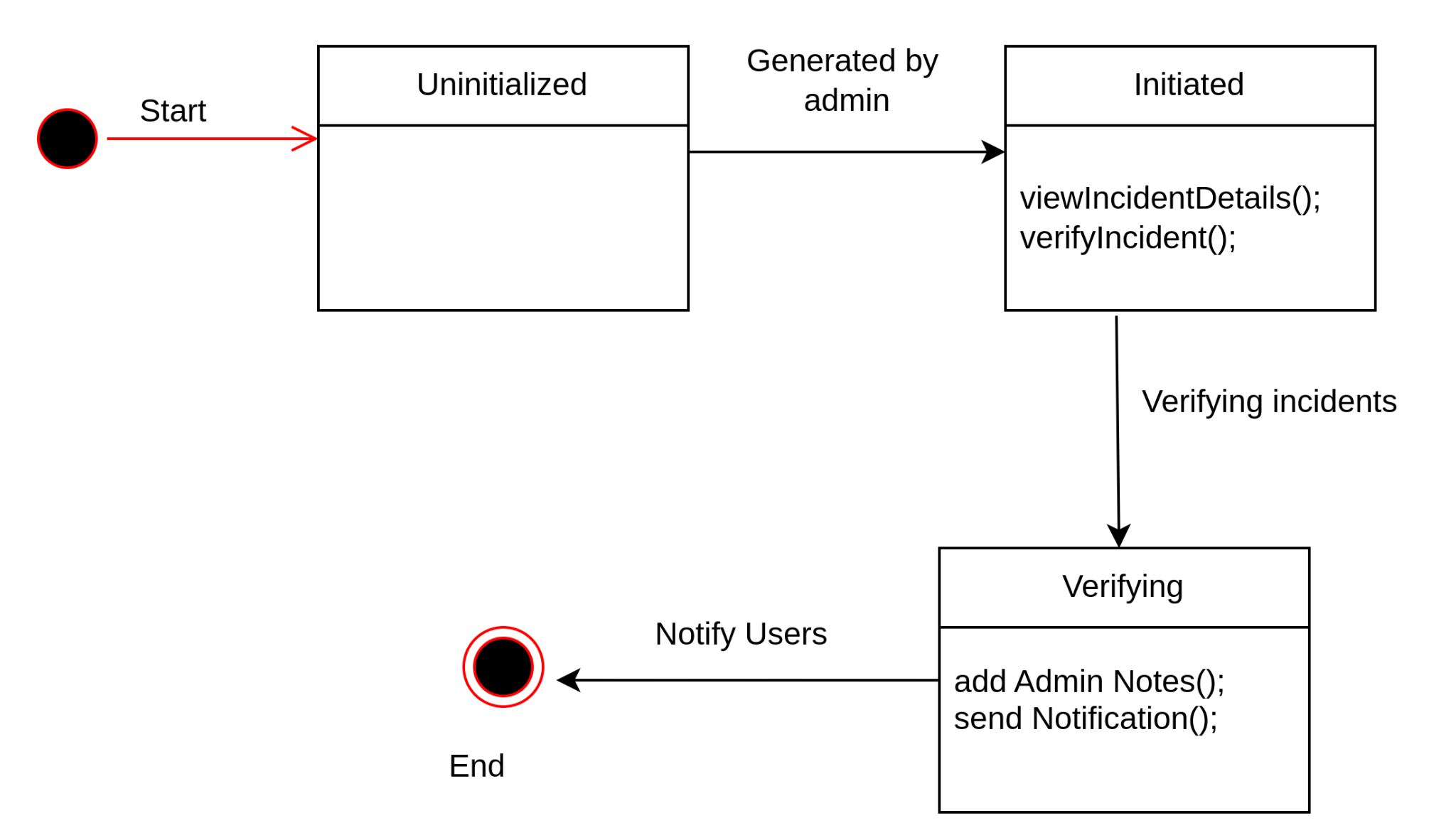
****

Figure 27 : Incident state transition diagram

**Name : Database**

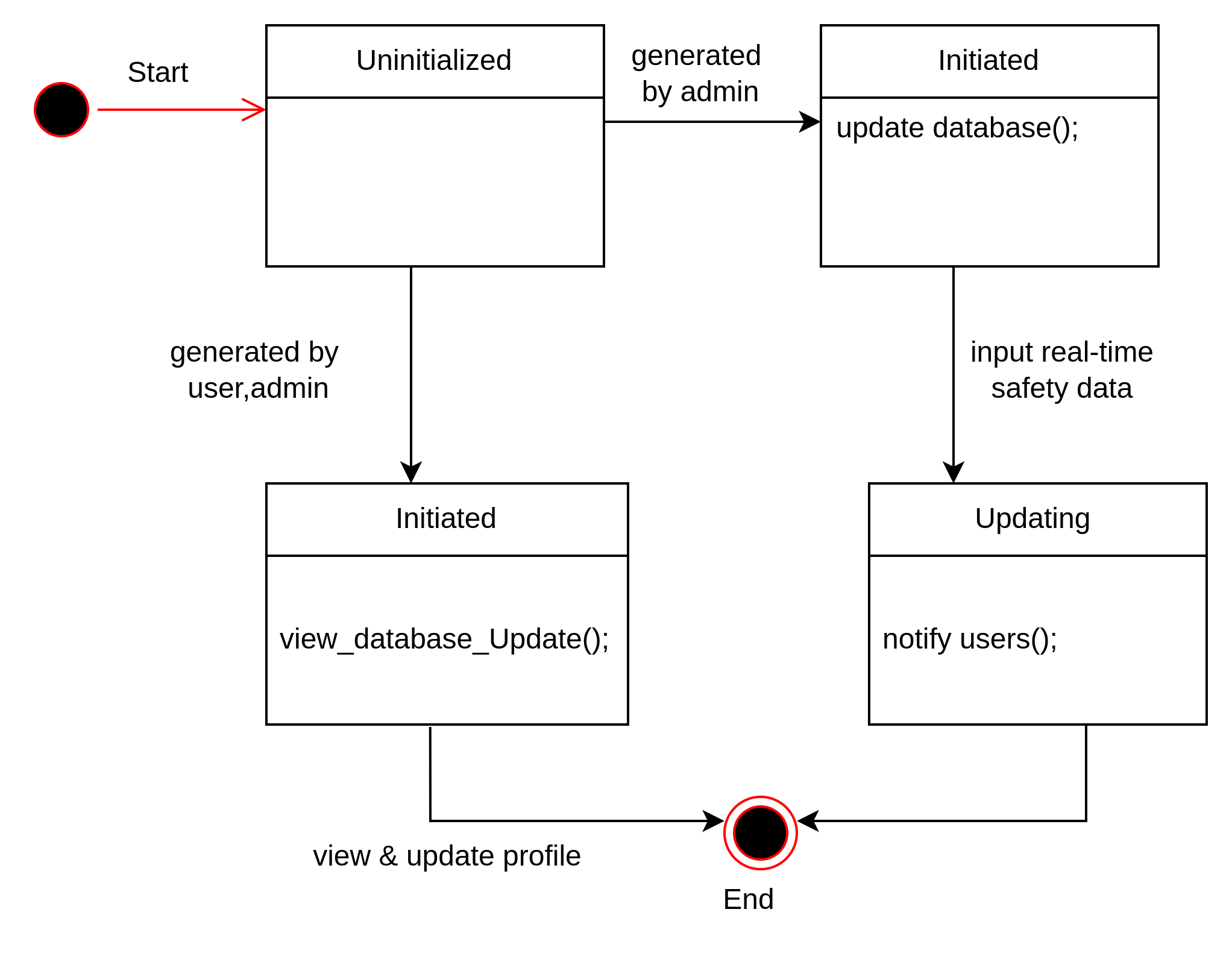
****

Figure 28 : Database state transition diagram

**Name : Notification**

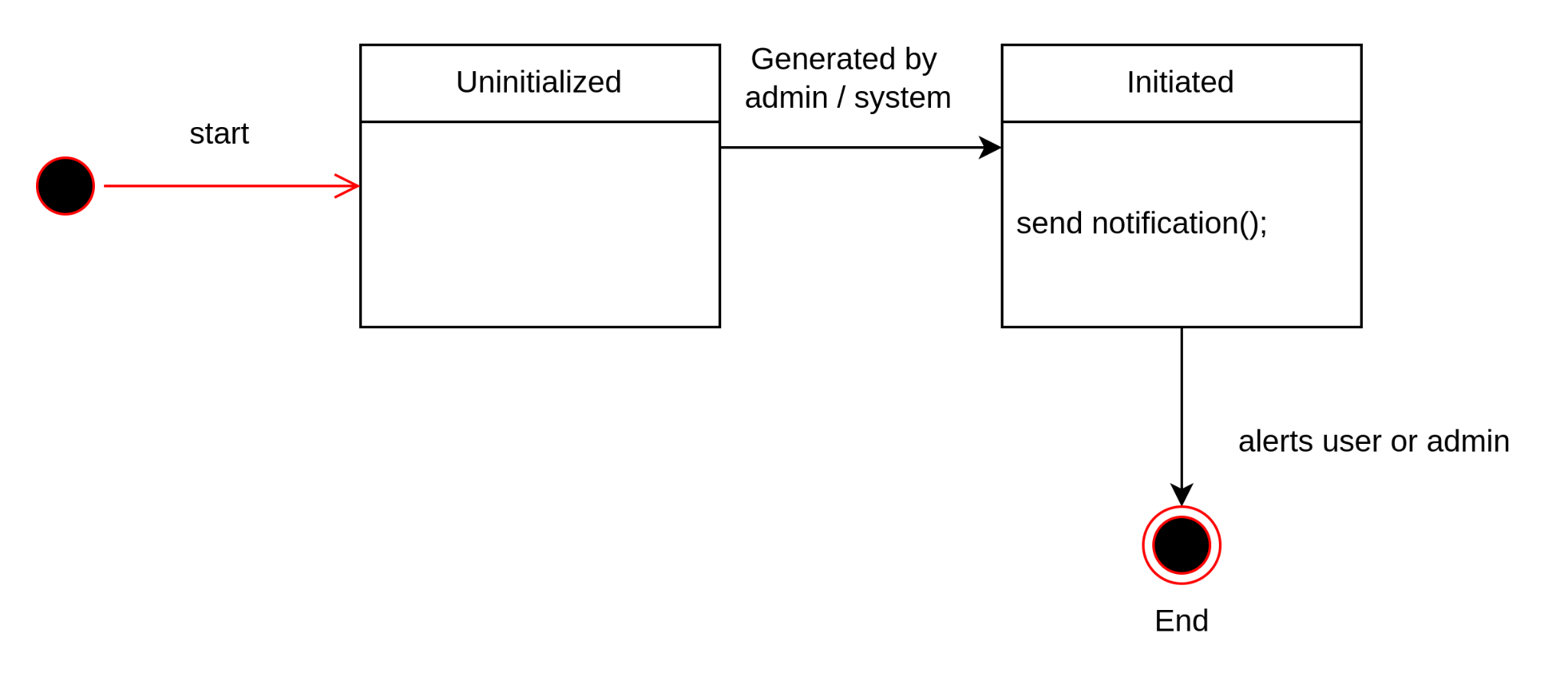
****

Figure 29 : Notification state transition diagram

**Name : Feedback**

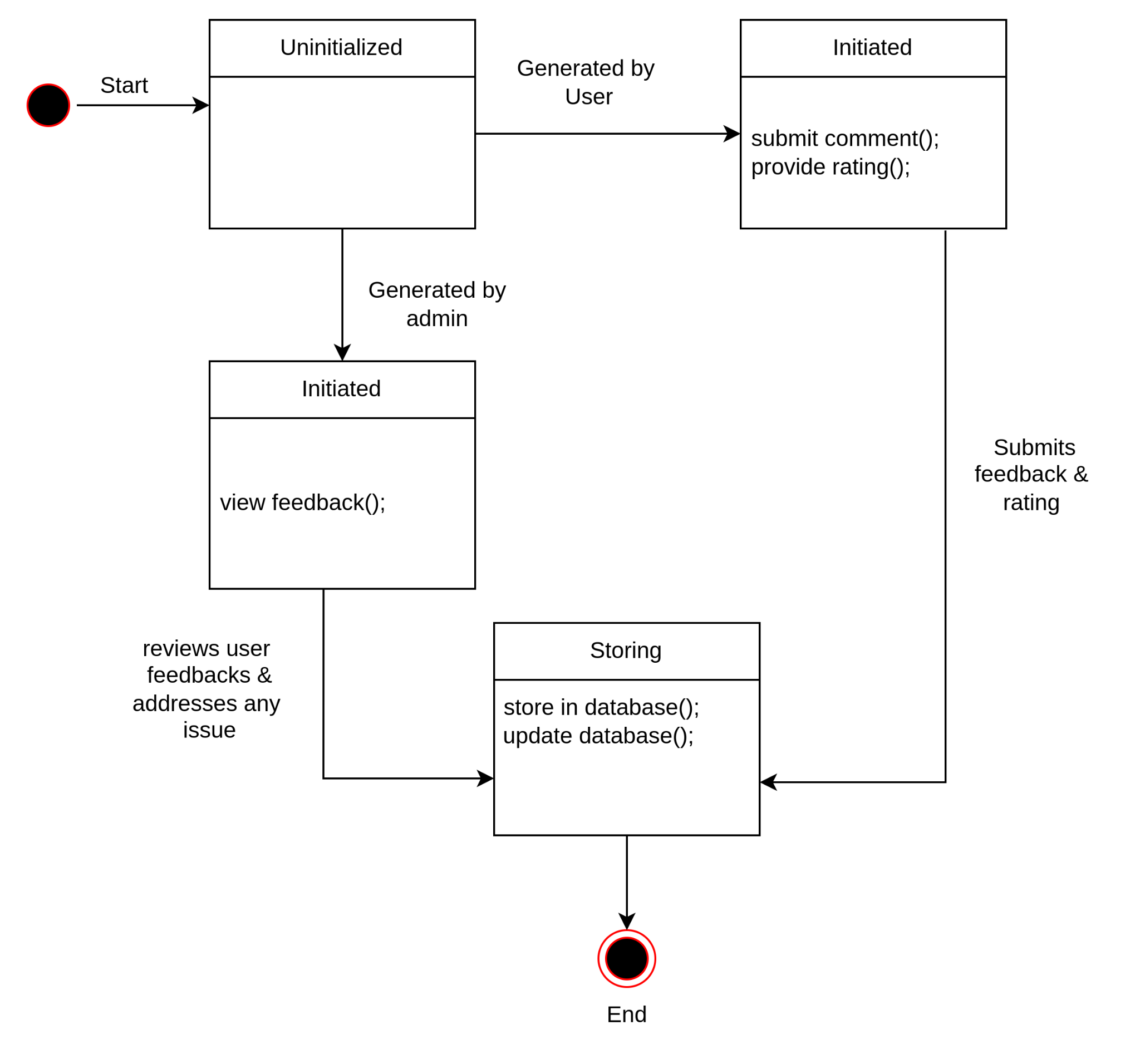
****

Figure 30 : Feedback state transition diagram

**Name : System**

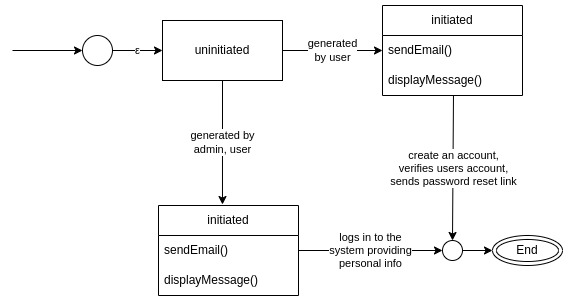
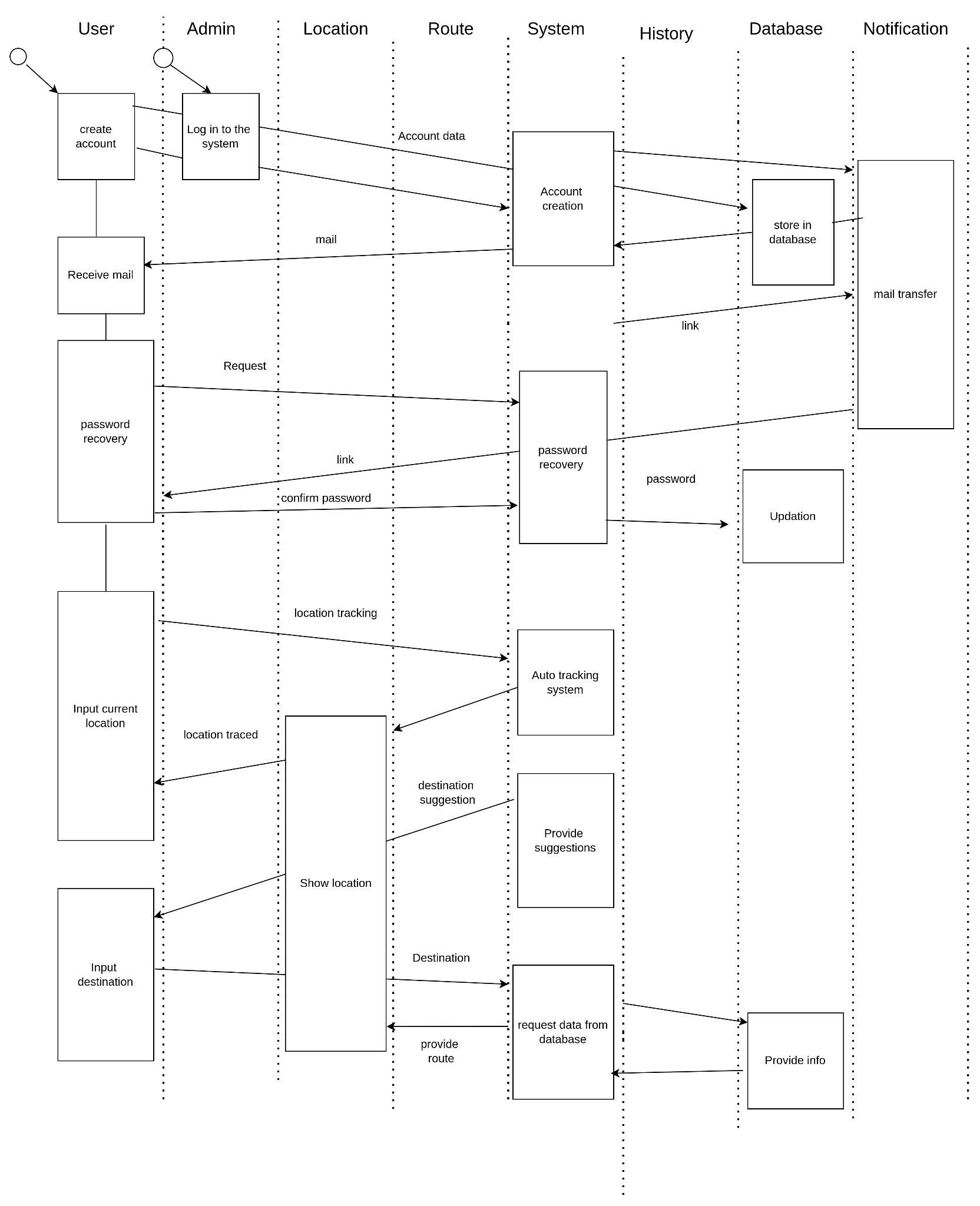
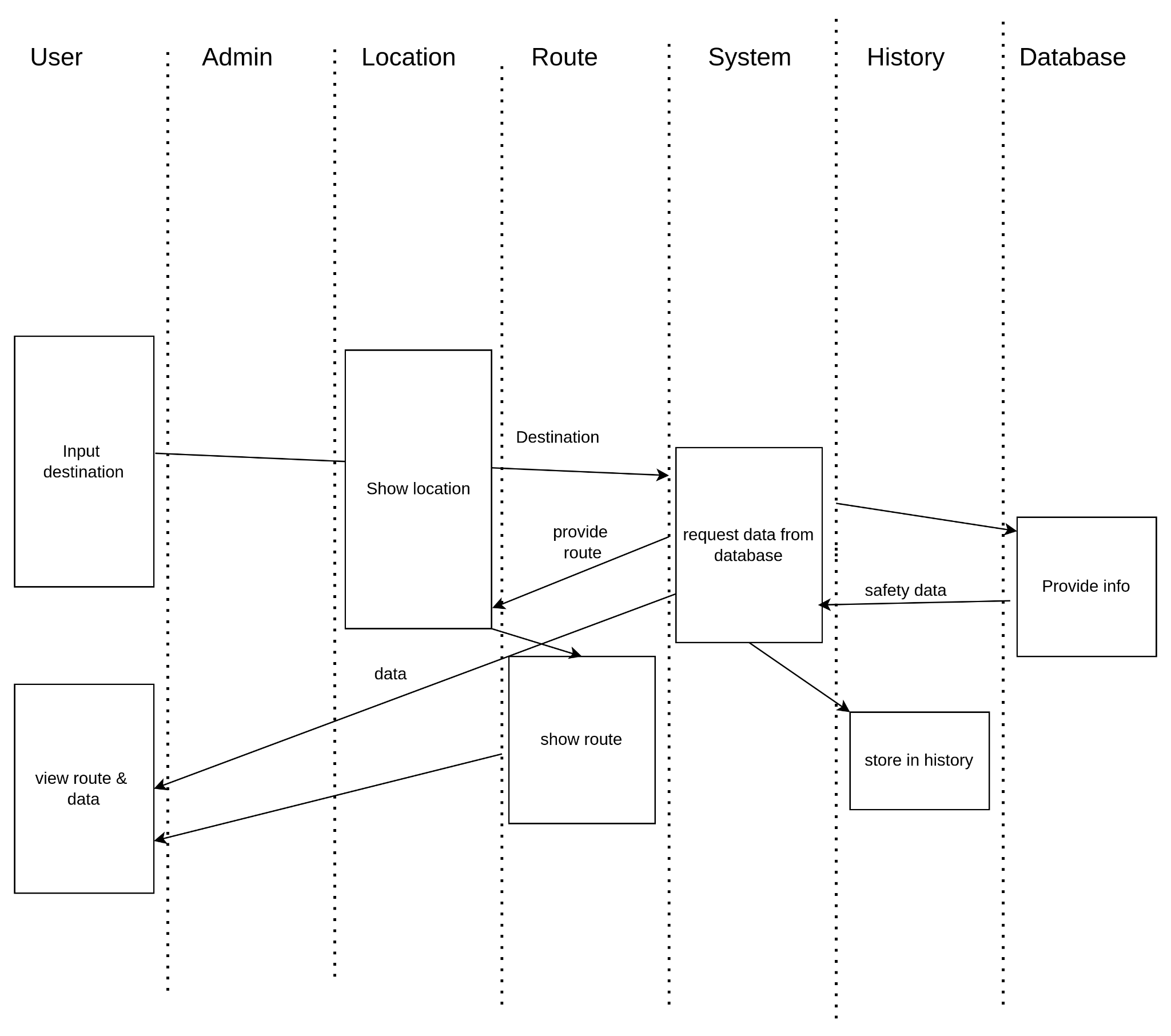
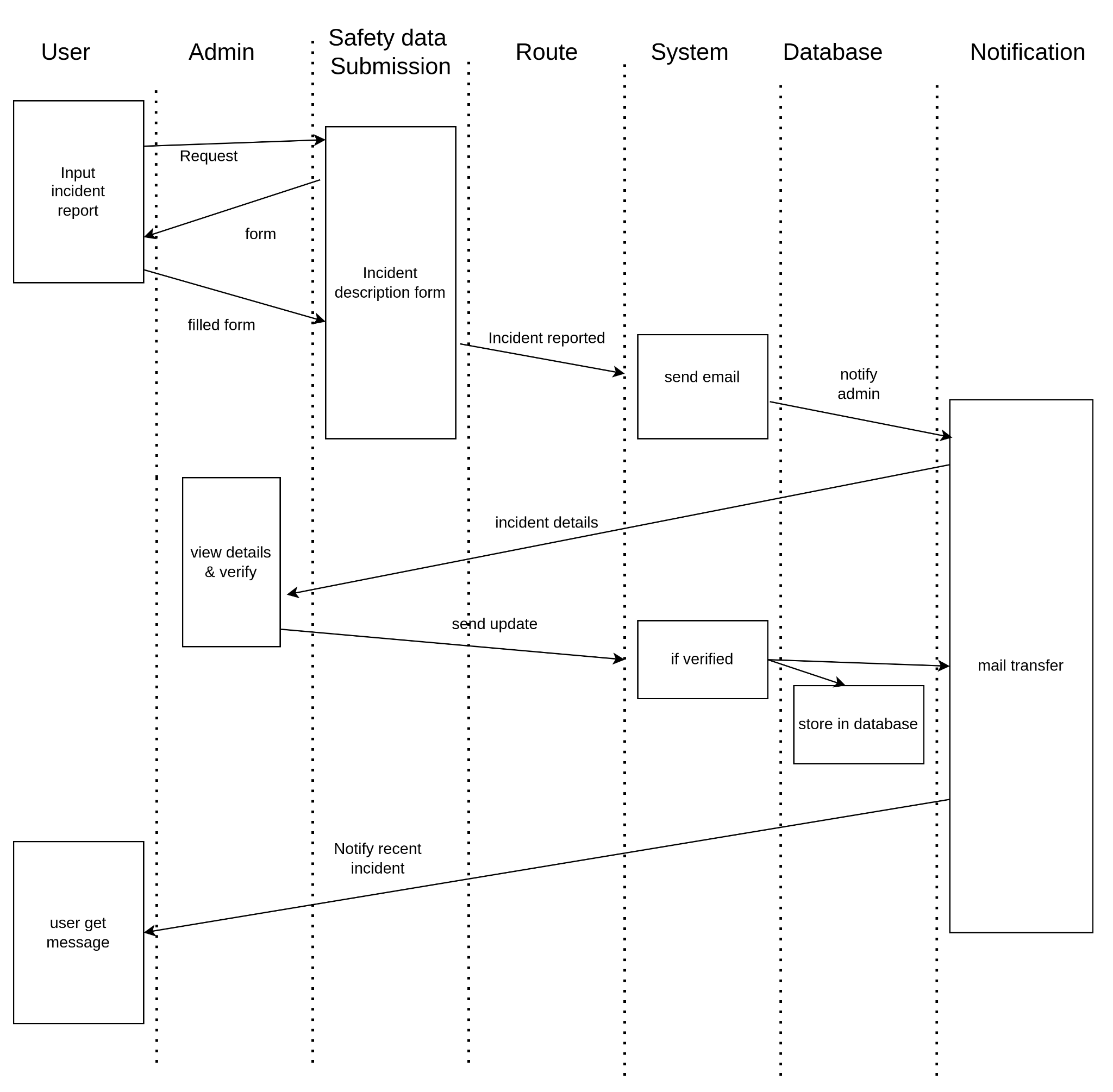
****

Figure 31 : System state transition diagram

**Sequence diagram :**

****

****

****

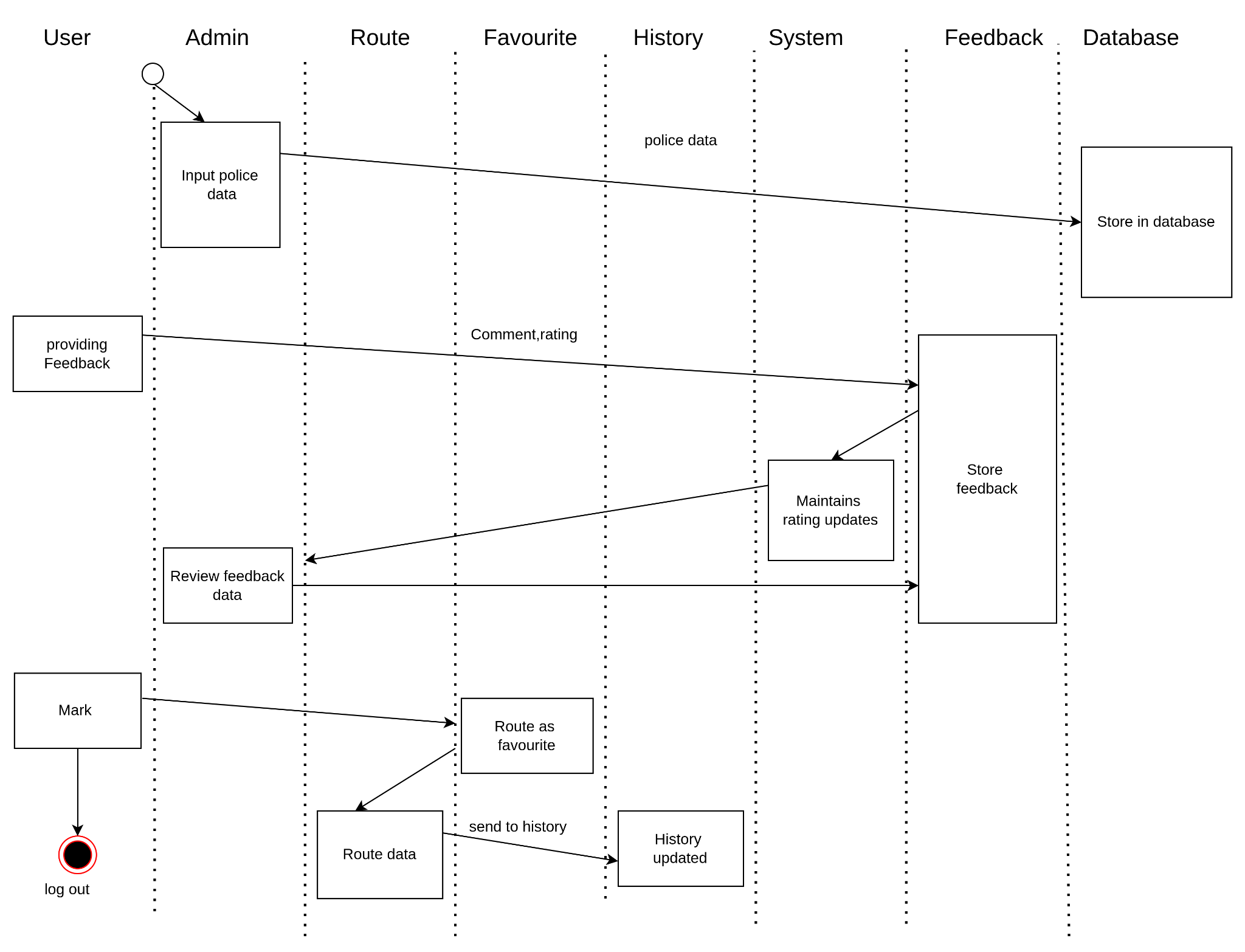
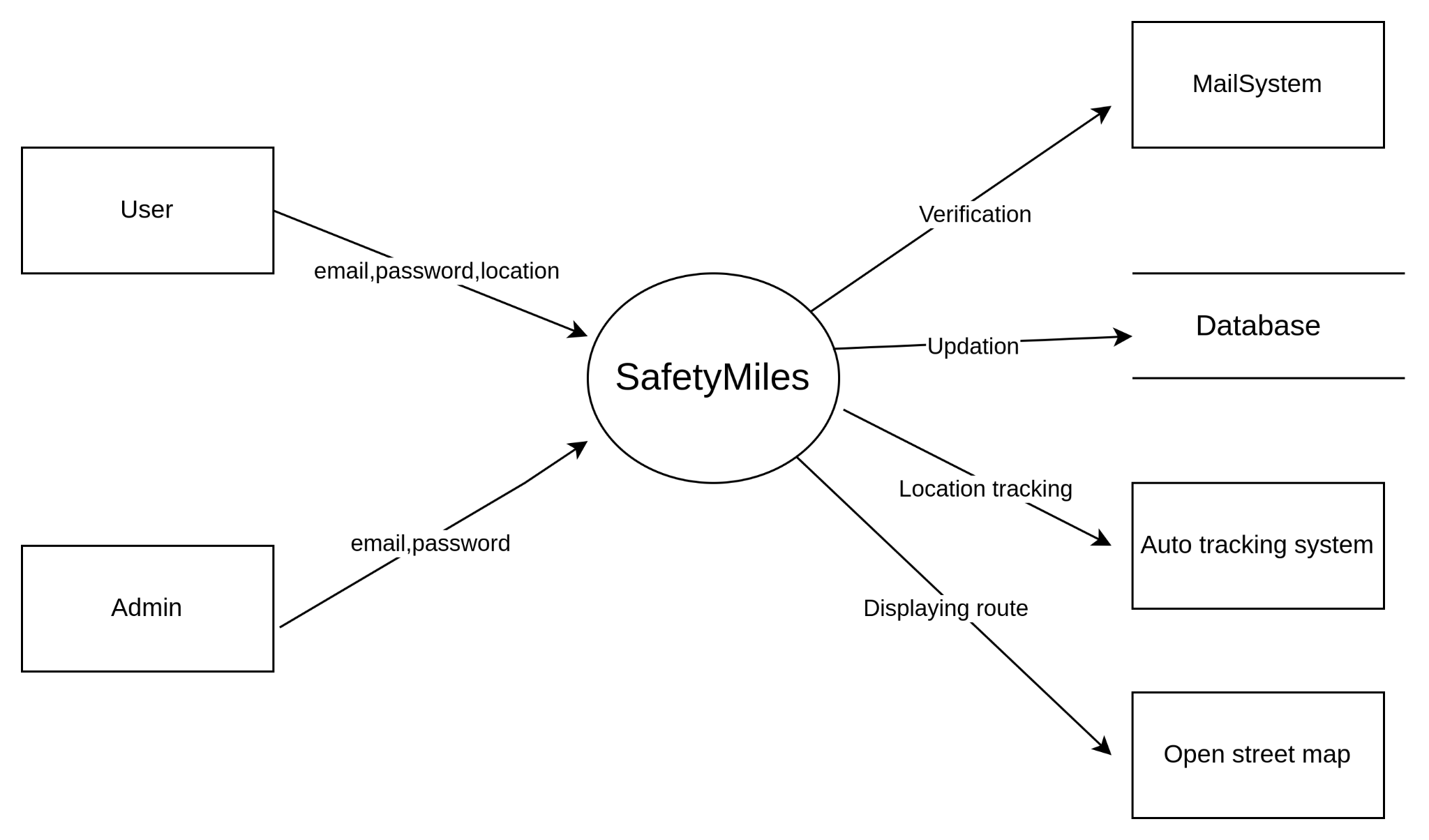
****

Figure 32 : Sequence diagram

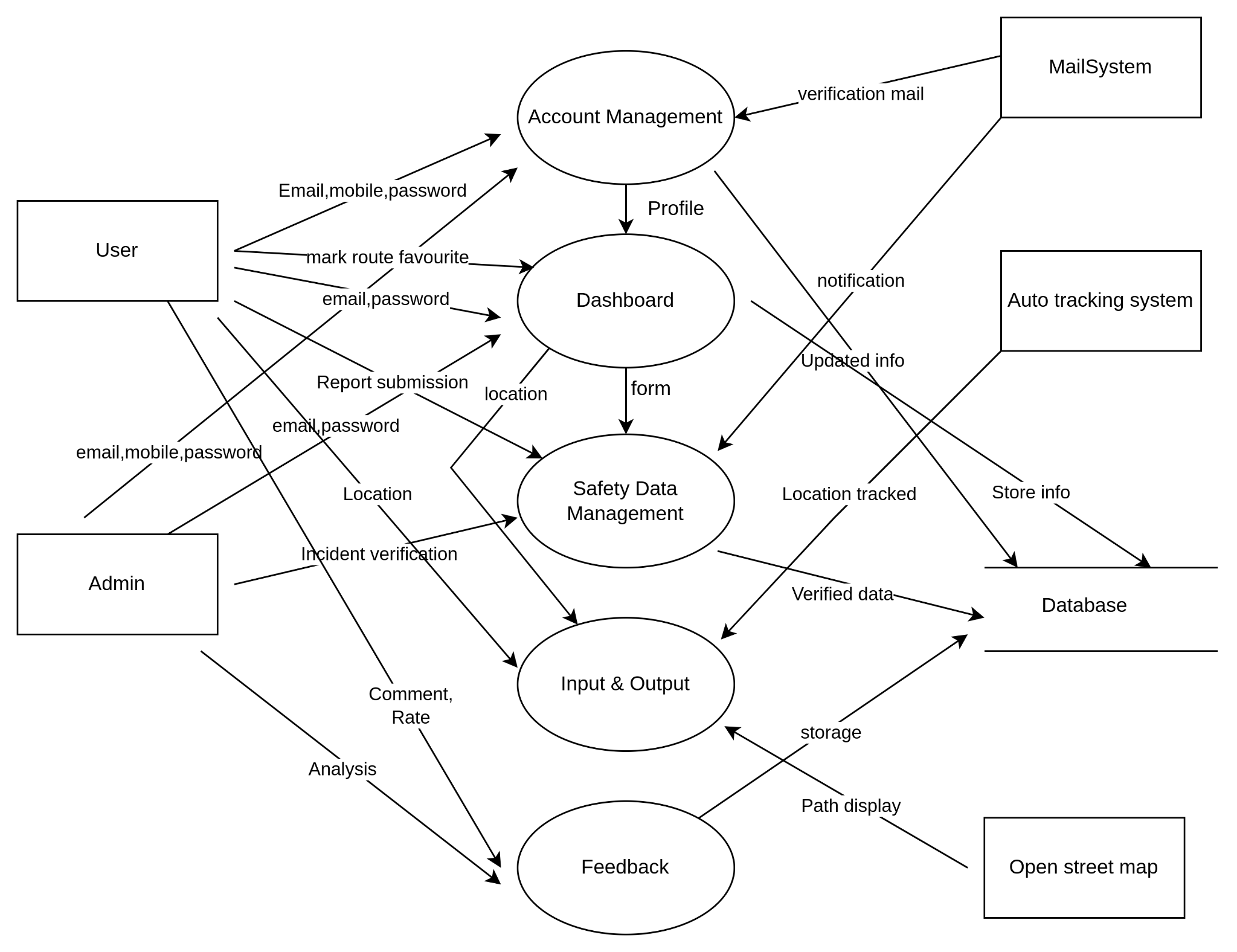
# **9. Data Flow Diagram (DFD)**

DFD, or Data Flow Diagram, is a visual representation that illustrates the flow of information within a system. It employs various symbols to depict processes, data stores, data flows, and external entities, providing a clear and concise overview of how data moves through a system. DFDs are essential in system analysis and design as they help stakeholders, including analysts and end-users, to comprehend the information flow and interactions within a complex system. By breaking down a system into manageable components and highlighting data transformations, DFDs facilitate effective communication, aid in identifying potential bottlenecks or areas for improvement, and serve as a foundation for designing and implementing efficient information systems.

## **Level 0**

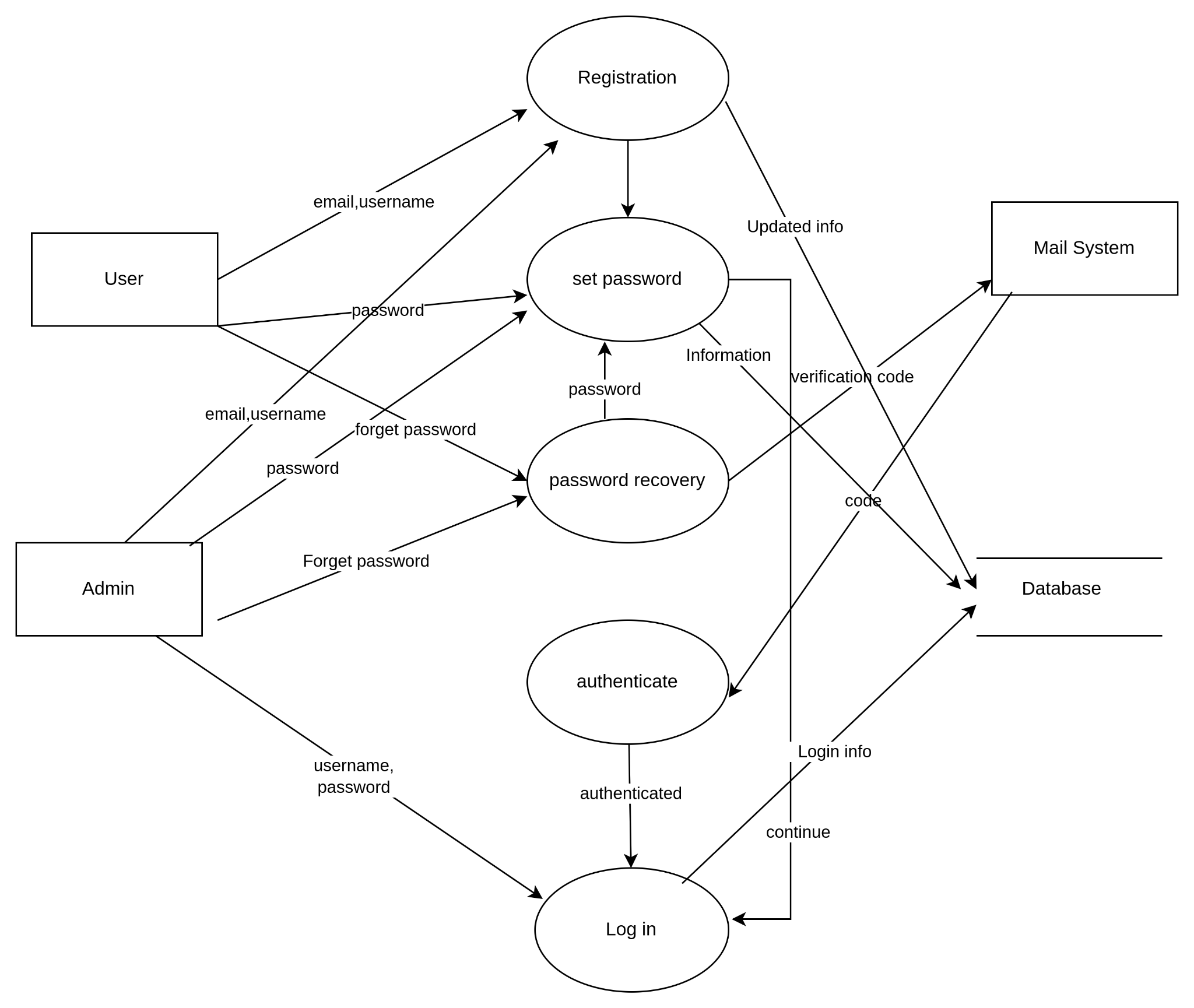


## **Level 1**

****

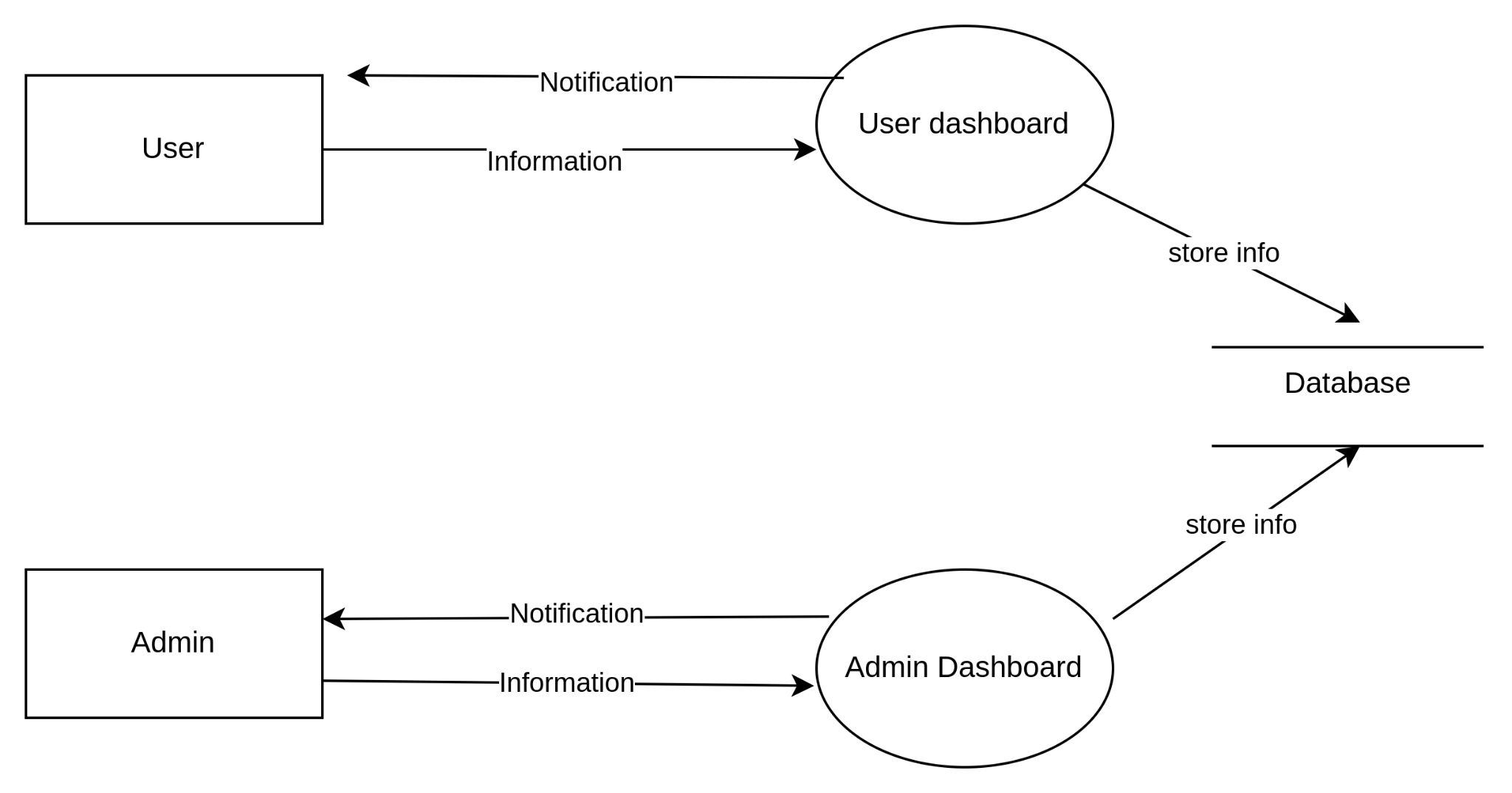
## **Level 2**

**Account Management Module :**

****

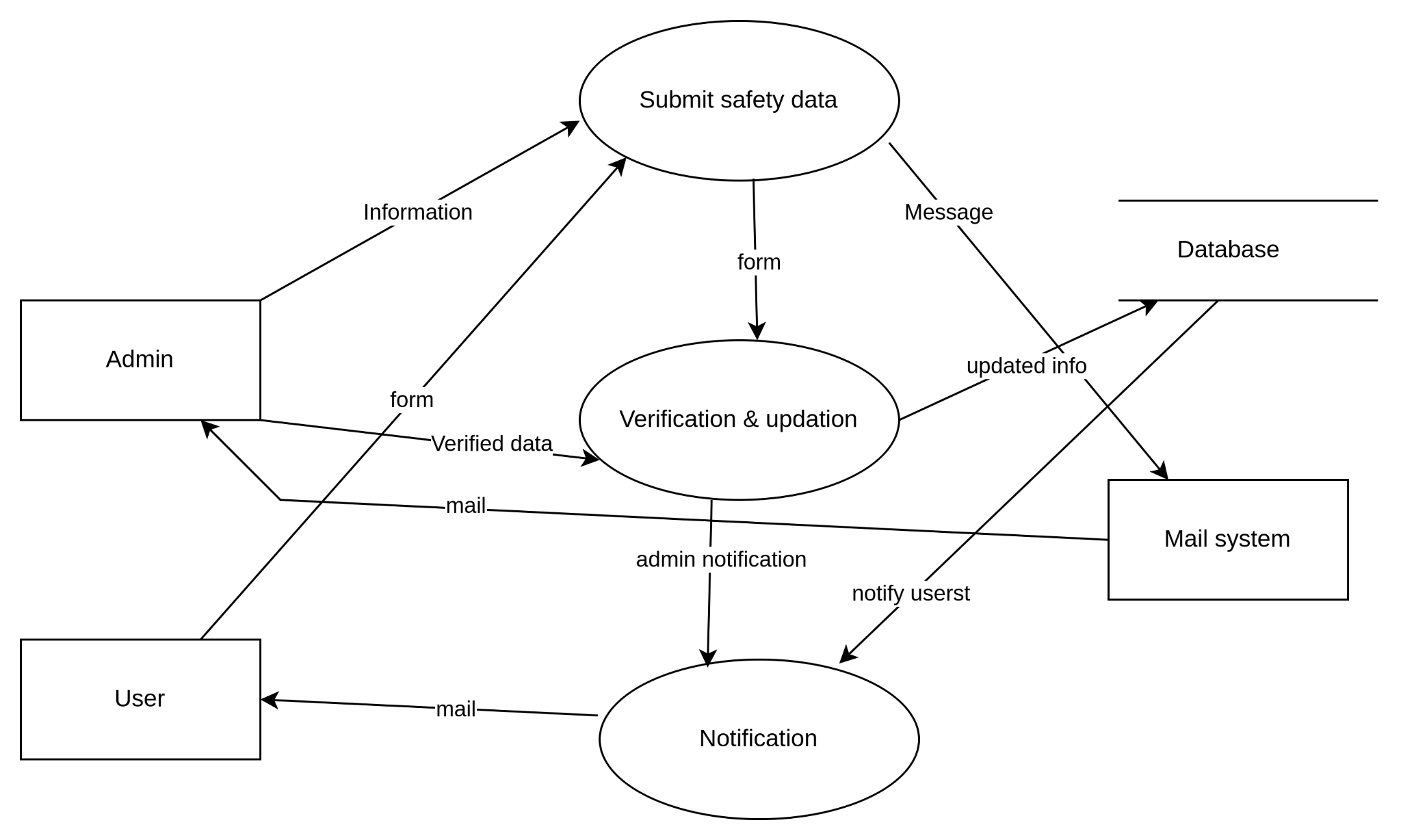
## 

**Dashboard Module:**

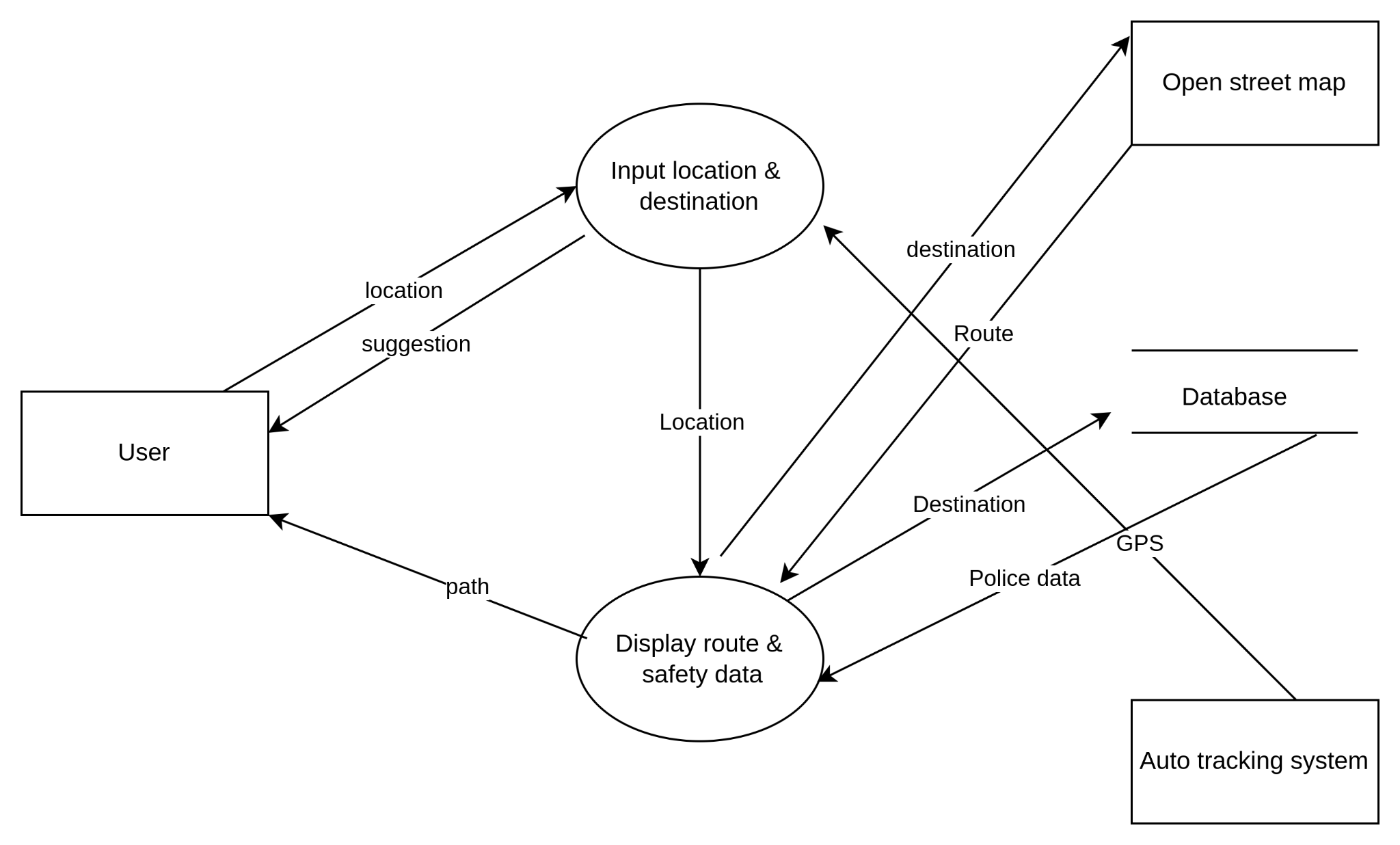
****

## 

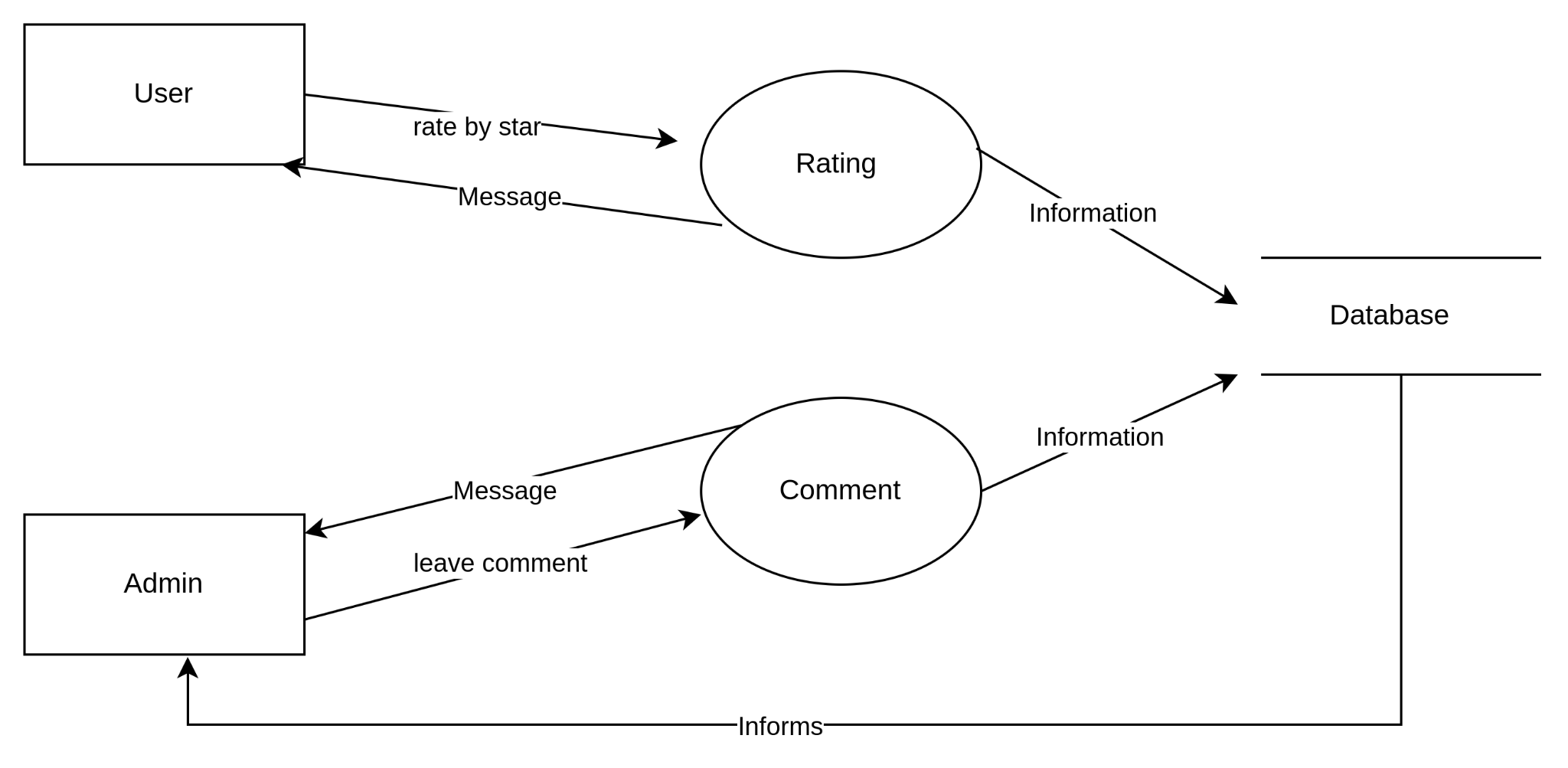
**Submit Safety Data**

****

**Input & Output**

****

**Feedback**

****

**10.Change LOG**

**10.1 Stakeholder identification**

**Users:** Users are individuals who utilize the SafetyMiles application to plan their journeys and ensure safety during travel. They create accounts, input their locations and destinations, access safety information, submit safety reports, provide feedback, and utilize the app's features for a secure journey. Users have a vested interest in the accuracy and reliability of the safety information provided by the app.

**Admin:** The admin plays a central role in managing various activities within the SafetyMiles application. They are responsible for user management, safety data verification, database management, route and safety information management, responding to user feedback, and sending notifications. Admins ensure the integrity and reliability of the app's information and features, enhancing user trust and satisfaction.

**Law Enforcement Agencies:** Since your app will incorporate real-time safety data, law enforcement agencies become important stakeholders. They provide this data, which could include crime reports, accidents, or other safety-related incidents. Collaborating with them ensures access to accurate and timely information, benefiting both users and the community.

**10.2 Modification in ER-Diagram**

