**System Specifications:**

**Flutter** :

Flutter is an open-source UI software development kit (SDK) created by Google. It is used to build natively compiled applications for mobile, web, and desktop from a single codebase. Flutter uses the Dart programming language and provides a rich set of pre-designed widgets, tools, and libraries to create beautiful, fast, and responsive user interfaces.

**1. Core Concepts of Flutter**

1. **Widgets**:
   * Everything in Flutter is a widget, from structural elements (like buttons and text) to layout elements (like rows, columns, and grids).
   * Widgets are categorized into:
     + **StatelessWidget**: Immutable widgets that do not change over time.
     + **StatefulWidget**: Widgets that can change dynamically based on user interaction or data changes.
2. **Dart Programming Language**:
   * Flutter uses Dart, a modern, object-oriented language developed by Google.
   * Dart is optimized for building UIs and supports features like async/await, garbage collection, and strong typing.
3. **Hot Reload**:
   * Flutter’s hot reload feature allows developers to see changes in the app instantly without restarting it, making the development process faster and more efficient.
4. **Platform-Specific Adaptations**:
   * Flutter provides platform-specific widgets and APIs to ensure apps look and feel native on both Android and iOS.

**2. Flutter Architecture**

1. **Layered Architecture**:
   * Flutter is built on a layered architecture, with each layer providing a specific set of functionalities:
     + **Framework Layer**: Contains widgets, animations, and rendering logic.
     + **Engine Layer**: Handles rendering, input, and platform-specific integrations.
     + **Embedder Layer**: Integrates Flutter with the host operating system.
2. **Rendering Pipeline**:
   * Flutter uses Skia, a 2D rendering engine, to draw widgets directly onto the canvas, ensuring high performance and smooth animations.
3. **Reactive Programming**:
   * Flutter follows a reactive programming model, where the UI updates automatically in response to changes in the app’s state.

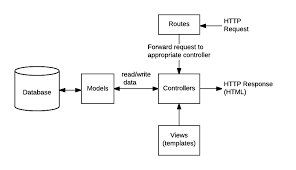
****

**Express.js** is a fast, unopinionated, and minimalist web framework for **Node.js**. It is widely used for building web applications and APIs (Application Programming Interfaces) due to its simplicity, flexibility, and scalability. Express.js provides a robust set of features for building single-page, multi-page, and hybrid web applications.

Here’s an overview of the **content of Express.js**:

**1. Core Concepts of Express.js**

1. **Middleware**:
   * Middleware functions are the backbone of Express.js. They have access to the request (req), response (res), and the next middleware function in the application’s request-response cycle.
   * Middleware can perform tasks like logging, authentication, error handling, and more.
2. **Routing**:
   * Express.js allows you to define routes for handling HTTP requests (GET, POST, PUT, DELETE, etc.).
   * Routes are defined using methods like app.get(), app.post(), app.put(), and app.delete().
3. **Request and Response Objects**:
   * **Request (req)**: Contains information about the HTTP request (e.g., query parameters, headers, body).
   * **Response (res)**: Used to send a response back to the client (e.g., sending JSON, rendering a view).
4. **Error Handling**:
   * Express.js provides built-in and custom error-handling mechanisms to manage errors in your application.
5. **Template Engines**:
   * Express.js supports template engines like **EJS**, **Pug**, and **Handlebars** for rendering dynamic HTML pages.

****

**MongoDB** is a popular **NoSQL database** that stores data in a flexible, JSON-like format called **BSON** (Binary JSON). It is designed for scalability, performance, and ease of development. MongoDB is widely used in modern web applications, especially those built with Node.js, due to its flexibility and ability to handle large volumes of unstructured or semi-structured data.

Here’s an overview of the **content of MongoDB**:

### ****1. Core Concepts of MongoDB****

1. **Document-Oriented Database**:
   * MongoDB stores data in **documents**, which are JSON-like objects with key-value pairs.
   * Example of a document:

json

{

"\_id": "12345",

"name": "John Doe",

"age": 30,

"email": "john.doe@example.com"

}

1. **Collections**:
   * Documents are grouped into **collections**, which are analogous to tables in relational databases.
   * Example: A collection named users can store multiple user documents.
2. **Schemaless**:
   * MongoDB is schemaless, meaning documents in the same collection can have different structures.
3. **BSON**:
   * MongoDB uses **BSON** (Binary JSON) for storing and transferring data. BSON supports additional data types like dates, binary data, and object IDs.
4. **Indexes**:
   * Indexes improve query performance by allowing faster data retrieval. MongoDB supports single-field, compound, and multi-key indexes.
5. **Replication**:
   * MongoDB provides high availability through **replica sets**, which are groups of MongoDB instances that maintain the same data.
6. **Sharding**:
   * MongoDB supports horizontal scaling through **sharding**, which distributes data across multiple servers.

**JSON WEB TOKEN**

**JSON Web Token (JWT)** is an **open standard (RFC 7519)** for securely transmitting information between parties as a **JSON object**. The information is digitally **signed**, ensuring **authenticity and integrity**.

JWT is commonly used for:  
 **Authentication** (e.g., login sessions, API access)  
 **Authorization** (e.g., role-based access control)  
 **Data exchange** (secure information sharing)

## **Structure of a JWT**

A JWT consists of **three parts**, separated by dots (.):

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VySWQiOjEwMiwicm9sZSI6ImFkbWluIiwiaWF0IjoxNjg0ODg4ODAwfQ.r9gULAfzO1HVOf6xFqjJGdFmAH4JAt6rKNfuLb3mvmA

### ****1️ Header (Metadata)****

* Specifies the **token type** (JWT) and **signing algorithm** (HS256, RS256, etc.).

json

{

"alg": "HS256",

"typ": "JWT"

}

### ****2 Payload (Claims)****

* Contains **user data** (e.g., user ID, role) and **token metadata** (e.g., expiration time).

json

{

"userId": 102,

"role": "admin",

"iat": 1684888800, // Issued At timestamp

"exp": 1684892400 // Expiration timestamp

}

### ****3 Signature (Security)****

* Ensures data integrity using a **secret key or private key**.
* Example signing process (HS256 algorithm):

HMACSHA256(

base64UrlEncode(header) + "." + base64UrlEncode(payload),

secret

)

* Verifies that the **JWT hasn't been tampered with**.

## **How JWT Works?**

1️ **User Logs In**

* User enters credentials (email/password).
* Server **verifies** credentials and generates a **JWT**.

2️ **Server Sends JWT to User**

* The token is returned as a response and stored **client-side** (e.g., **localStorage, sessionStorage, HTTP cookies**).

3️ **Client Uses JWT for API Requests**

* The client includes JWT in the Authorization header:

http

Authorization: Bearer <JWT>

4️ S**erver Verifies JWT**

* Extracts the token, **validates the signature**, and **decodes the payload**.
* If valid, **grants access** to protected resources.

5️ **Token Expiration & Refresh**

* JWTs **expire** (exp claim), and a **refresh token** can be used to obtain a new one.

**MODULE DESCRIPTION**

## **1. User Management Module**

This module handles the **authentication, role management, and user profiling** functionalities.

**User Registration & Authentication**

* Users can register using **email, phone number, or social login (OAuth for Google, Facebook, etc.)**
* Uses **JWT-based authentication** for secure session management.
* Passwords are stored using **bcrypt hashing** to prevent data breaches.

**Role-Based Access Control (RBAC)**

* The system supports different roles:
  + **User:** Can report cybercrimes and track their progress.
  + **Admin:** Manages reports, scam alerts, and users.
  + **Authority:** Law enforcement agencies that investigate cybercrime.

**User Profile Management**

* Users can update their **contact details and preferences.**

**Security Measures**

* **Multi-Factor Authentication (MFA)** can be enabled for added security.
* **Session timeout** for inactive users.

### ****Expected Outcomes:****

Secure login system  
Prevent unauthorized access  
 Manage different types of users efficiently

## **2. Cybercrime Incident Reporting Module**

This module allows users to **report cybercrimes**, upload **evidence**, and track the **progress** of their case.

**Incident Submission**

* Users submit detailed information about the incident, including **date, time, location, and description**.
* **Incident categories:**
  + **Phishing Attacks**
  + **Online Fraud (Scams, Fake Websites, etc.)**
  + **Identity Theft**
  + **Ransomware Attacks**

**Evidence Upload**

* Users can attach **screenshots, videos, PDFs, or logs** as proof.

**Incident Tracking & Status Updates**

* Users can track the status of their reported case (Pending, Under Review, Resolved).
* Notifications are sent to **keep users updated** on the progress.

### ****Expected Outcomes:****

Allows citizens to report fraud easily

Helps law enforcement agencies manage cybercrime reports efficiently  
 Ensures **secure evidence collection**

## **3. Fraud Detection & AI-Based Analysis Module**

This module leverages **Artificial Intelligence (AI) and Machine Learning (ML)** to **analyze and detect fraudulent activities** in reports and alerts.

**Fraud Pattern Analysis**

* The system **analyzes past incidents** to detect patterns in cyber fraud.
* It uses **Machine Learning (ML) models** to predict **high-risk** fraud cases.

**Anomaly Detection**

* Uses **Natural Language Processing (NLP)** to scan scam messages, emails, and reports.
* Detects **suspicious trends** in user activity (e.g., multiple reports from the same IP address).

**Risk Score Assignment**

* Each reported incident is given a **risk score** based on severity and likelihood of fraud.
* High-risk cases are flagged for **manual review by authorities.**

### ****Expected Outcomes:****

Automates fraud detection  
Helps authorities focus on **high-priority cases**  
 Reduces false reports

## **4. Scam Awareness & Security Alerts Module**

This module helps spread **awareness about cyber scams** and **sends alerts** to users.

**Admin-Controlled Scam Alerts**

* Admins can create **public scam alerts** based on new cybercrime trends.
* Alerts include **detailed information on how to avoid scams.**

**Real-Time Security Notifications**

* Users receive **alerts on new cyber threats** via push notifications and emails.
* Notifications include details of **ongoing scams, phishing attempts, and fraudulent activities.**

**Severity Levels**

* Each scam alert is assigned a severity level (Low, Medium, High).
* **High-severity alerts** are sent **immediately** to all users.

### ****Expected Outcomes:****

Helps users avoid scams **before they happen**  
 Reduces cybercrime by educating the public  
 Provides **real-time scam alerts**

## **5. Authority Integration & Legal Actions Module**

This module **allows law enforcement agencies** to **review reported cybercrime cases and take action.**

**Forwarding Reports to Authorities**

* Admins can **assign cases to external agencies** such as **Cybercrime Police, FBI, or Interpol.**
* Authorities can **investigate, update the status, and provide legal support.**

**Legal Documentation & Case Management**

* Secure handling of **legal documents and case progress**.
* Integration with **court proceedings and law enforcement databases**.

**Closed-Case Management**

* Once an incident is resolved, it is **archived** in the system.

### ****Expected Outcomes:****

Enables seamless law enforcement integration  
 Ensures **cybercriminals are tracked legally**  
 Enhances efficiency in resolving **complex cybercrime cases**

## **6. Reports & Visualization Module**

This module provides **statistical reports and data visualization** to **monitor cybercrime trends**.

**Graphical Reports on Cybercrime Trends**

* Shows **charts, graphs, and maps** of reported cybercrimes.
* Helps detect **geographical cybercrime hotspots**.

**Incident Analytics for Admins**

* Admins can track **phishing, malware, and fraud incidents** in real-time.

**Export & Download Reports**

* Reports can be **downloaded as PDFs, Excel files, or shared with legal teams.**

### ****Expected Outcomes:****

### Helps identify **patterns in cyber fraud** Assists authorities in making **data-driven decisions** Improves the **prevention of cybercrimes**

## **7. Notification & User Alerts Module**

This module sends **real-time alerts and system notifications** to users.

**Incident Status Updates**

* Users are notified when their case is updated (Under Review, Resolved).

**Security Alerts & Warnings**

* Users receive alerts on **new fraud detection patterns**.

**Push & Email Notifications**

* Users can choose to receive **push notifications, SMS, or email alerts**.

### ****Expected Outcomes:****

Keeps users informed of case progress  
 Reduces the risk of cyber fraud by **alerting users early**

**TABLE DESIGN:**

**1.USER Table**

| **Column Name** | **Data Type** | **Constraints** | **Description** |
| --- | --- | --- | --- |
| user\_id | SERIAL | PRIMARY KEY | Unique identifier for users |
| name | VARCHAR(100) | NOT NULL | User's full name |
| email | VARCHAR(100) | UNIQUE, NOT NULL | User's email (must be unique) |
| phone\_number | VARCHAR(15) | UNIQUE, NOT NULL | User's phone number |
| gender | VARCHAR(10) | NULLABLE | User's gender |
| password\_hash | TEXT | NOT NULL | Hashed password |
| role | ENUM | DEFAULT 'user' | User role (user/admin/authority) |

**2. Incidents Table**

| **Column Name** | **Data Type** | **Constraints** | **Description** |
| --- | --- | --- | --- |
| incident\_id | SERIAL | PRIMARY KEY | Unique incident identifier |
| user\_id | INT | REFERENCES users(user\_id) ON DELETE CASCADE | User who reported the incident |
| incident\_type | ENUM | NOT NULL | Type of cyber incident |
| description | TEXT | NOT NULL | Detailed description of the incident |
| status | ENUM | DEFAULT 'pending' | Incident status (pending/reviewed/resolved) |
| reported\_at | TIMESTAMP | DEFAULT CURRENT\_TIMESTAMP | Incident report timestamp |

**3. Incident Evidence Table**

| **Column Name** | **Data Type** | **Constraints** | **Description** |
| --- | --- | --- | --- |
| evidence\_id | SERIAL | PRIMARY KEY | Unique identifier for evidence |
| incident\_id | INT | REFERENCES incidents(incident\_id) ON DELETE CASCADE | Linked incident ID |
| file\_path | TEXT | NOT NULL | File path where evidence is stored |
| file\_type | VARCHAR(50) | NULLABLE | Type of file (image, video, document) |
| uploaded\_at | TIMESTAMP | DEFAULT CURRENT\_TIMESTAMP | Timestamp when file was uploaded |

**4. Reports & Visualization Table**

| **Column Name** | **Data Type** | **Constraints** | **Description** |
| --- | --- | --- | --- |
| report\_id | SERIAL | PRIMARY KEY | Unique report identifier |
| total\_incidents | INT | DEFAULT 0 | Total number of reported incidents |
| phishing\_count | INT | DEFAULT 0 | Count of phishing cases |
| malware\_count | INT | DEFAULT 0 | Count of malware cases |
| fraud\_count | INT | DEFAULT 0 | Count of fraud cases |
| identity\_theft\_count | INT | DEFAULT 0 | Count of identity theft cases |
| last\_updated | TIMESTAMP | DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP | Last update timestamp |

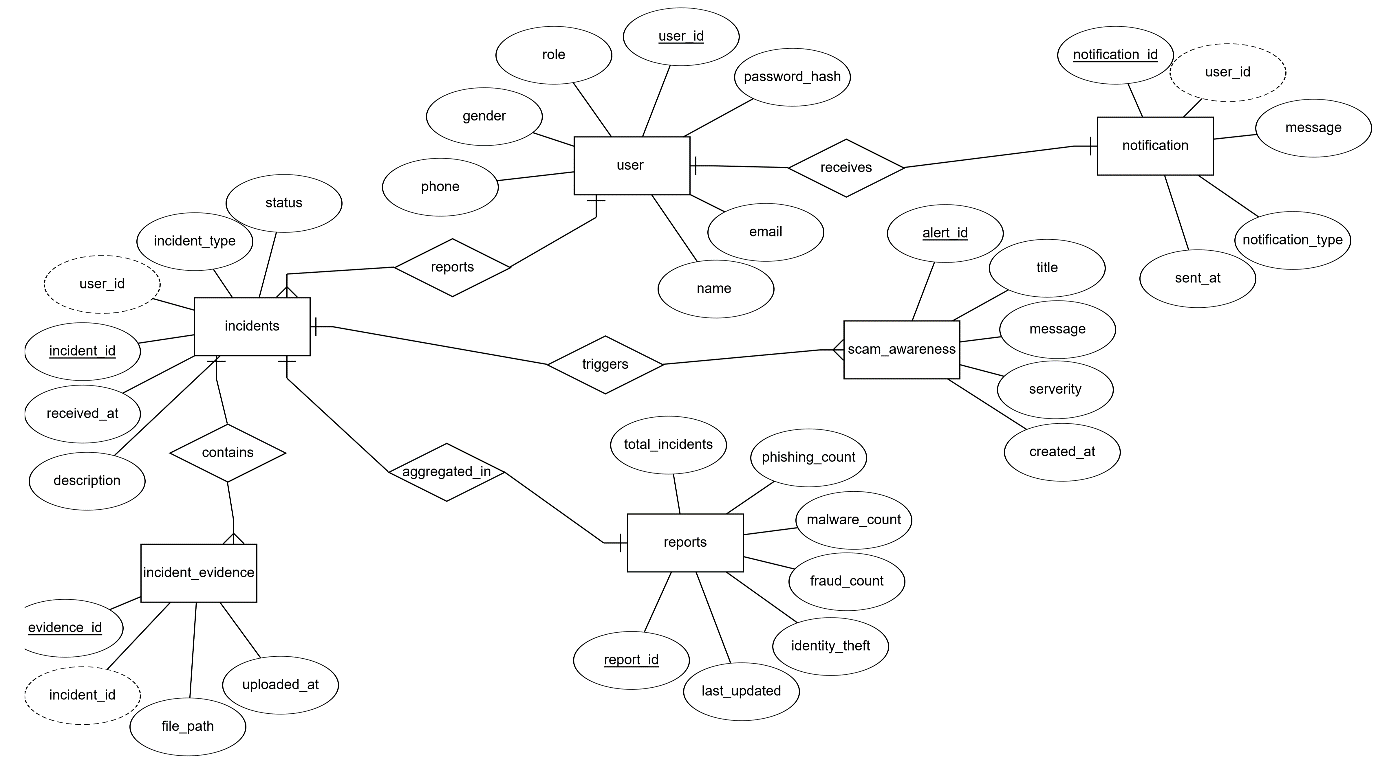
**5. Scam Awareness Alerts Table**

| **Column Name** | **Data Type** | **Constraints** | **Description** |
| --- | --- | --- | --- |
| alert\_id | SERIAL | PRIMARY KEY | Unique alert identifier |
| title | VARCHAR(255) | NOT NULL | Alert title |
| message | TEXT | NOT NULL | Scam alert message details |
| severity | ENUM | DEFAULT 'low' | Severity level (low/medium/high) |
| created\_at | TIMESTAMP | DEFAULT CURRENT\_TIMESTAMP | Timestamp when alert was created |

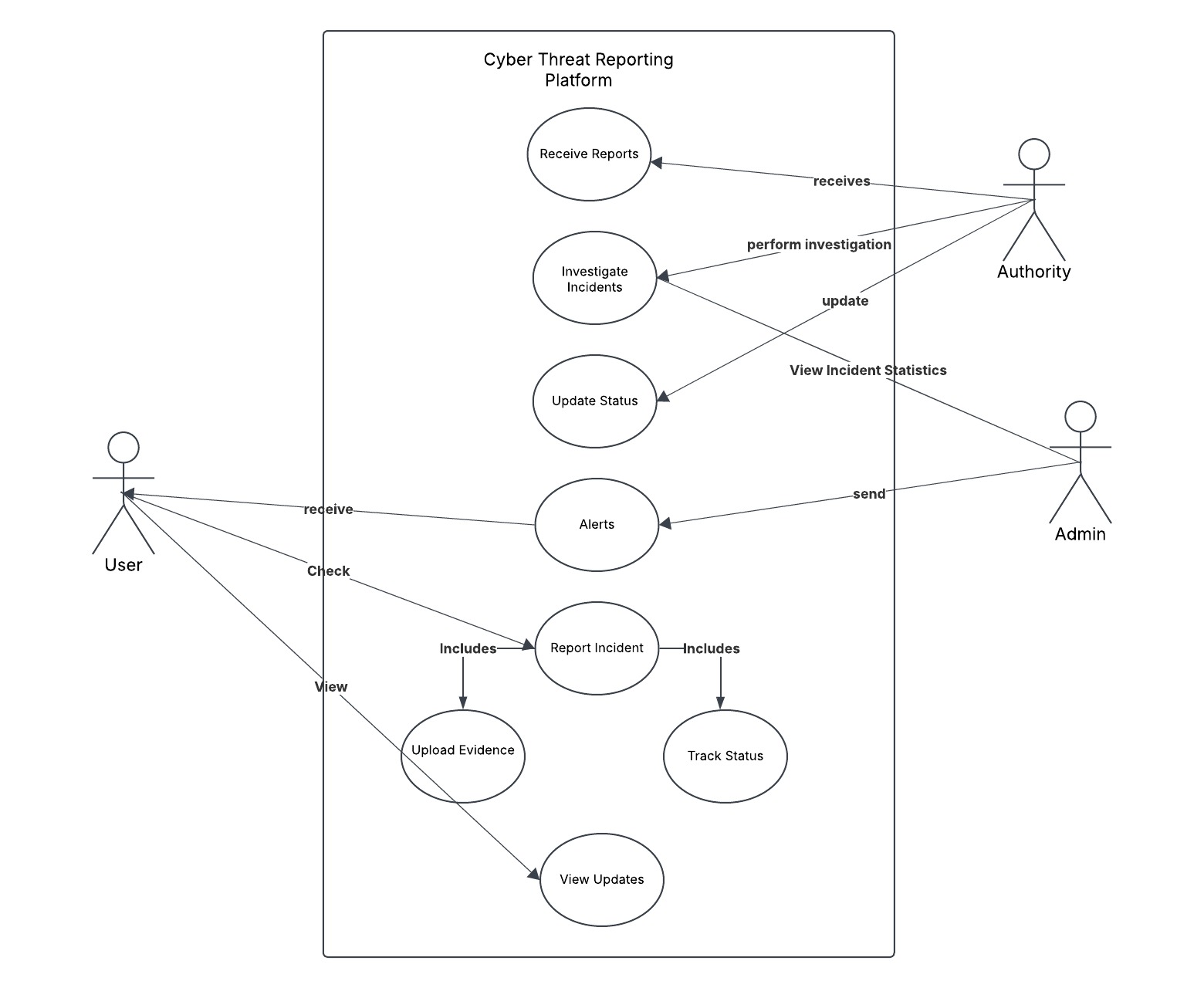
**6. Notifications Table**

| **Column Name** | **Data Type** | **Constraints** | **Description** |
| --- | --- | --- | --- |
| notification\_id | SERIAL | PRIMARY KEY | Unique notification ID |
| user\_id | INT | REFERENCES users(user\_id) ON DELETE CASCADE | User receiving the notification |
| message | TEXT | NOT NULL | Notification message |
| notification\_type | ENUM | NOT NULL | Type (incident update/security alert/general) |
| sent\_at | TIMESTAMP | DEFAULT CURRENT\_TIMESTAMP | Timestamp when notification was sent |

ER DIAGRAM:

****

USE CASE:



## **Actors in the Diagram**

1. **User**
   * A general user who reports cyber threats, uploads evidence, and tracks the status of incidents.
2. **Admin**
   * Manages cyber threat reports, sends alerts, and provides incident statistics to authorities.
3. **Authority**
   * Investigates reported cybercrime incidents and updates their status.

## **Use Cases & Functionalities**

### ****1 Report Incident****

* **Actor:** User
* **Description:** A user can report a cybercrime incident, providing necessary details like **incident type, description, date, and location**.
* **Includes:**
  + **Upload Evidence:** Users can submit **screenshots, documents, or logs** as proof.
  + **Track Status:** Users can track whether their report is being investigated, reviewed, or resolved.

### ****2️ Alerts****

* **Actor:** User
* **Description:** The system sends **real-time alerts** to users regarding **ongoing cyber threats and scams**.

### ****3️ Receive Reports****

* **Actor:** Authority
* **Description:** The authority receives cybercrime reports submitted by users for **further investigation**.

### ****4️ Investigate Incidents****

* **Actor:** Authority
* **Description:** Authorities **analyze reported threats**, validate evidence, and take necessary legal action.

### ****5️ Update Status****

* **Actor:** Authority
* **Description:** The investigating authority **updates the status** of cases, which is then visible to the user.

### ****6️ View Updates****

* **Actor:** User
* **Description:** Users can **check updates** on their submitted reports and see if any action has been taken.

### ****7️ View Incident Statistics****

* **Actor:** Admin
* **Description:** The admin can analyze **cybercrime trends** and provide statistical insights to authorities for better crime prevention.

### ****8 Send Alerts****

* **Actor:** Admin
* **Description:** The admin sends alerts about **new cyber fraud tactics** to users.