**Project Title:**

**COMPLAINTS RESPONSE APPLICATION WITH GEOLOCATION TRACKING**

Proponent: **MARCHEL GREGORY DECENA**

Subject: Mobile Computing

Platform used: Flutter/Android Studio

**Chapter I – Introduction**

### ****BACKGROUND OF THE STUDY****

In many communities, citizens often encounter public service issues such as water leaks, waste disposal problems, broken streetlights, or road damage. Despite the presence of local government units and service providers, these complaints frequently go unreported or unresolved due to the lack of accessible and efficient reporting mechanisms. Traditional methods like physically visiting offices, calling hotlines, or writing letters are time-consuming and can discourage community members from participating in civic engagement.

With the widespread use of smartphones and the internet, there is an opportunity to modernize the way complaints are submitted and addressed. A digital complaint reporting system integrated with geolocation technology enables users to quickly report issues from anywhere, along with precise location data. This improves the response time of concerned authorities and enhances transparency and accountability in public service.

This study aims to develop a **Complaints-Response Complaint Reporting System with Geolocation**, a mobile application that allows users to register, log in, and report various complaints with their descriptions and real-time location. By leveraging geolocation, authorities can identify problem areas more accurately and prioritize responses based on proximity or severity.

The system also benefits local government units and service providers by streamlining the intake of complaints, minimizing manual paperwork, and fostering better communication between the public and officials. In the long run, the system promotes a safer, cleaner, and more responsive community environment through accessible technology.

**PROBLEM STATEMENT**

Despite the growing need for responsive and efficient public service, many communities still rely on traditional, manual methods of reporting complaints. Residents often face difficulties in communicating local issues such as damaged infrastructure, water supply problems, or sanitation concerns due to:

* Lack of accessible complaint channels
* Time-consuming reporting processes
* Inability to provide accurate location of the issue
* Delayed response from authorities due to unclear or incomplete reports

These limitations result in unresolved community problems, reduced trust in local governance, and decreased civic participation.

To address this gap, there is a need for a **mobile-based complaint reporting system** that simplifies the process of filing community-related complaints. By integrating **geolocation features**, the system can automatically capture the exact location of the incident, making it easier for service providers to respond effectively and promptly.

This study aims to develop a **Community-Based Complaint Reporting System with Geolocation**, enabling users to submit real-time, location-tagged complaints through a mobile application. The system seeks to bridge the communication gap between residents and authorities, improve public service delivery, and promote community engagement.

**OBJECTIVES OF THE STUDY**

*General Objectives*

To develop a mobile-based complaint reporting system integrated with geolocation technology that allows community members to report public service issues efficiently, accurately, and in real time.

*Specific Objectives*

1. **To design and implement a user-friendly mobile application** that allows users to register, log in, and manage their complaints.
2. **To integrate geolocation functionality** to automatically capture and attach the user’s current location to each complaint submitted.
3. **To develop a backend API using PHP and MySQL** that processes user data, stores complaints, and manages location-based entries.
4. **To allow users to choose complaint categories and input descriptions** for more detailed and organized reporting.
5. **To enable viewing of submitted complaints and their status**, allowing users to track the progress and ensure transparency.
6. **To assist local authorities or administrators** in accessing complaint data with location points, improving response times and decision-making.
7. **To promote community involvement** by providing a convenient and accessible platform for reporting local issues.

### ****Scope and Limitations****

#### **Scope**

This study focuses on the development of a mobile-based complaint reporting system that allows users within a community to report public concerns using their mobile devices. The system includes the following features:

* **User Registration and Login**: Basic authentication for users to access the app securely.
* **Complaint Submission**: Users can select a category, write a description, and submit complaints through the app.
* **Geolocation Integration**: The system automatically captures the latitude and longitude of the user to pinpoint the location of the complaint.
* **Complaint History**: Users can view a list of complaints they have submitted along with the current status (e.g., pending, resolved).
* **Backend Management**: A PHP and MySQL-based backend handles API requests, complaint storage, and user data.
* **Admin Access (optional)**: Administrators or service providers may view complaints and their geolocations through a web-based interface for monitoring and action.

#### **Limitations**

* **Internet Dependency**: The app requires an active internet connection to submit complaints and access server data.
* **Geolocation Accuracy**: The precision of location data depends on the user's device and GPS signal, which may vary indoors or in areas with weak signals.
* **Limited Admin Features**: The current version may not include advanced features like automated routing, notification systems, or in-depth analytics for authorities.
* **Platform Restriction**: The application is developed and tested only for Android platforms; iOS support is not included in this version.
* **Security Considerations**: While basic authentication is implemented, the system may not yet include advanced encryption or data privacy features.
* **No Offline Submission**: Users cannot store complaints offline and sync them later; submission must be done in real time.

**SIGNIFICANCE OF THE STUDY**

The **Complaint Reporting App with Geolocation** is a valuable tool that simplifies and modernizes the process of reporting issues. It allows users to submit complaints quickly and accurately with real-time location data, reducing delays and improving service response.

For service providers or organizations, the app enhances operational efficiency by helping identify the exact location of problems, prioritize responses, and maintain a digital log of reports. It also demonstrates the practical application of mobile and geolocation technologies, making it useful for developers, IT professionals, and academic research.

**Chapter - Review of Related Systems / Literature**

## BRIEF DISCUSSION OF THE EXISTING SYSTEM

The current complaint reporting process in many organizations or service areas is often **manual and inefficient**. Users typically report issues through **walk-in visits, phone calls**, or **written letters**, which can result in delayed communication, lost records, and lack of accurate data—especially regarding the **exact location of the problem.**

Some organizations may use basic web-based forms or hotline numbers, but these **systems lack automation, geolocation support,** and **real-time tracking**. As a result, service providers often struggle with:

* Incomplete or vague information
* Difficulty locating reported issues
* Slow response times
* Low customer satisfaction

**EXISTING RELATED LITERATURE**

### 1. ****Mobile-Based Complaint Management Systems****

A study by Sharma et al. (2020) developed a mobile-based complaint system for municipal services, allowing users to report issues through a smartphone app. Their research emphasized the importance of digital reporting in reducing manual effort, improving transparency, and accelerating response times. The findings showed that mobile applications provide an efficient alternative to walk-in and phone-based complaint systems.

### 2. ****Use of Geolocation in Service Reporting****

According to Kim & Park (2018), integrating GPS/geolocation in mobile apps significantly enhances issue tracking and response prioritization. In their logistics management study, location-tagged data enabled faster field response and more accurate resource deployment—insights directly applicable to complaint handling in service-based industries.

### 3. ****E-Governance and Citizen Feedback Systems****

The paper "E-Governance and Public Grievance Redressal Mechanisms" (Rao, 2017) explored how digital systems are transforming public service delivery. It concluded that electronic complaint systems not only reduce processing time but also promote transparency and build public trust—goals that align with complaint reporting solutions in both government and private sectors.

### 4. ****Smart City Feedback Platforms****

Smart City initiatives in countries like India and Singapore have implemented mobile apps (e.g., SmartCity App, CleanIndia App) that allow citizens to report environmental or infrastructural issues. These apps use real-time GPS data to help authorities locate and resolve problems efficiently. The success of such apps demonstrates the effectiveness of integrating geolocation in complaint systems.

### 5. ****Limitations of Traditional Systems****

Several studies, such as by Tan and Li (2019), outline the inefficiencies of traditional reporting systems—lack of automation, paper-based logs, and inaccurate issue reporting. These limitations highlight the growing need for systems that use mobile and location-based technologies for better service quality.

**COMPARISON WITH YOUR PROJECT**

Based on the reviewed literature, many existing complaint systems have introduced features like mobile accessibility and user feedback. However, these systems often cater to large-scale government platforms or smart city projects and may not be customized for smaller organizations or private service providers. Below is a comparison of key points:

**WHAT MAKES YOURS SIMPLER OR UNIQUE FOR BEGINNERS?**

The **Complaint Reporting App with Geolocation** stands out from existing systems due to its simplicity, accessibility, and ease of implementation—especially for beginners in programming, IT, and systems development.

### ****1. Beginner-Friendly Code Structure****

* Built using **Flutter (for frontend)** and **PHP PDO + MySQL (for backend)**, technologies that are widely used, easy to learn, and have large support communities.
* Follows **clean and modular code practices**, allowing students and beginners to understand how frontend connects with backend through RESTful APIs.

### ****2. Mobile-First Design****

* Unlike complex web portals, this system focuses on a simple mobile interface where users can report complaints in just a few taps.
* Interfaces are intuitive with minimal navigation, making it easy to use even for non-tech-savvy users.

### ****3. Integrated Geolocation Without Complexity****

* Location is captured automatically using the device’s GPS—no need for manual map selection or complicated APIs.
* Flutter’s geolocator plugin and Google Maps support simplify this functionality.

### ****4. Lightweight and Deployable Locally****

* The backend uses XAMPP or any basic PHP server—no need for cloud services or enterprise-level setups.
* Can run even on local networks or offline environments with basic connectivity, ideal for small-scale deployments or educational settings.

### ****5. No Complex Authentication or Permissions****

* Uses **basic username-password login**, removing the need for OAuth or token-based systems that are harder to implement for beginners.
* Security is kept simple yet effective to demonstrate core authentication concepts.

### ****6. Easy to Customize and Expand****

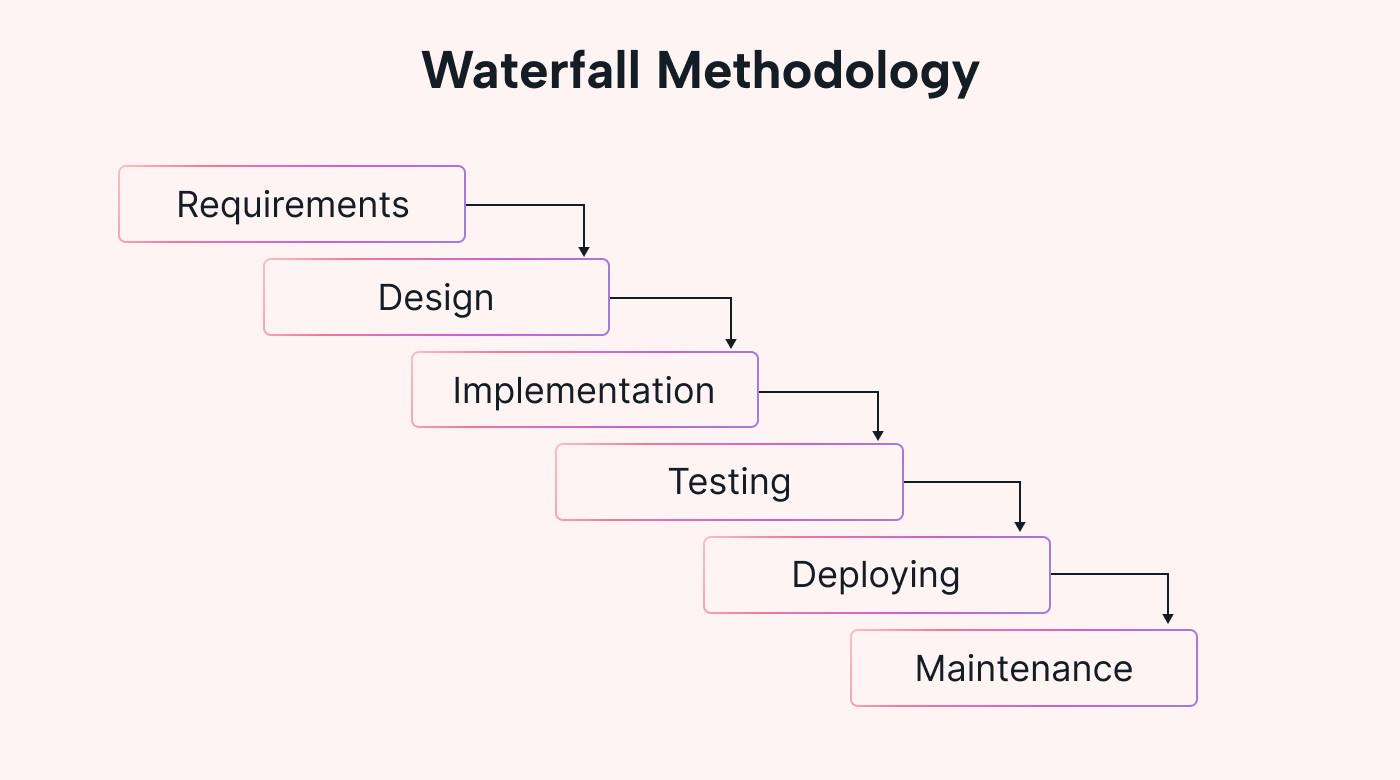
* Developers can easily add features like photo attachments, complaint filtering, or SMS/email alerts without overhauling the system.
* Ideal for capstone, thesis, or personal projects that need room for growth.

**Chapter 3 - Methodology**

**TOOLS AND TECHNOLOGIES USED (FLUTTER OR ANDROID STUDIO, PHP, MYSQL, VSCODE)**

|  |  |  |
| --- | --- | --- |
| **Tools and Technology Used** | | |
| **Development Tools** | **Back-End Technologies** | **Service** |
| Flutter | PHP (OOP + PDO) | Geolocation |
| Android Studio and VS Code | MySQL | http |
| Postman | Apache / XAMPP |  |
| Canva / Figma | WT (JSON Web Tokens |  |
|  |  |  |

**SYSTEM ARCHITECTURE DIAGRAM**

**Development process (Agile, Waterfall, etc.)**

### 1. ****Requirements Gathering****

* Conducted interviews or analysis to understand the problems in existing complaint systems.
* Identified the core features needed: user registration, login, complaint submission, geolocation tagging, and status tracking.

### 2. ****System Design****

* **Database Design**: Created an Entity-Relationship Diagram (ERD) and structured the database using MySQL. Tables include: users, complaints, and optionally admins.
* **Architecture Design**: Defined a 3-tier architecture: presentation (Flutter app), logic (PHP backend), and data (MySQL).
* **UI/UX Design**: Designed simple and intuitive screens for complaint submission and history viewing.

### 3. ****Implementation****

* **Frontend**: Developed using **Flutter**, the app allows users to:
  + Register and log in
  + Submit complaints with auto-captured GPS coordinates
  + View status of submitted complaints
* **Backend**: Built using **PHP (OOP + PDO)** with REST APIs to:
  + Handle user authentication
  + Process and store complaints
  + Fetch data for display in the app
* **Database**: Implemented in **MySQL** to store structured data securely.

### 4. ****Testing****

* **Unit Testing**: Individual modules (login, form submission, API connection) were tested.
* **Integration Testing**: Ensured smooth data flow between app, API, and database.
* **Device Testing**: Tested on Android emulator and real device to verify GPS and API communication.

### 5. ****Deployment****

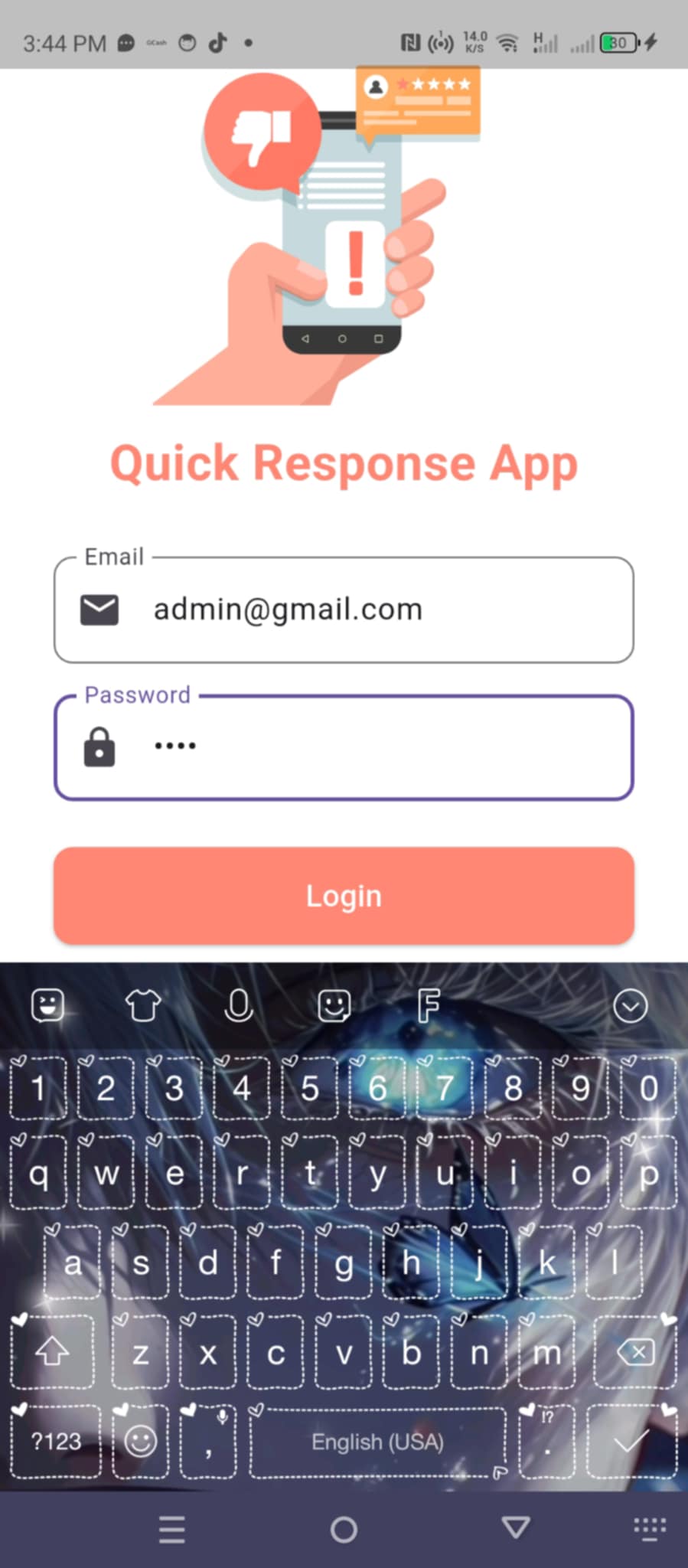
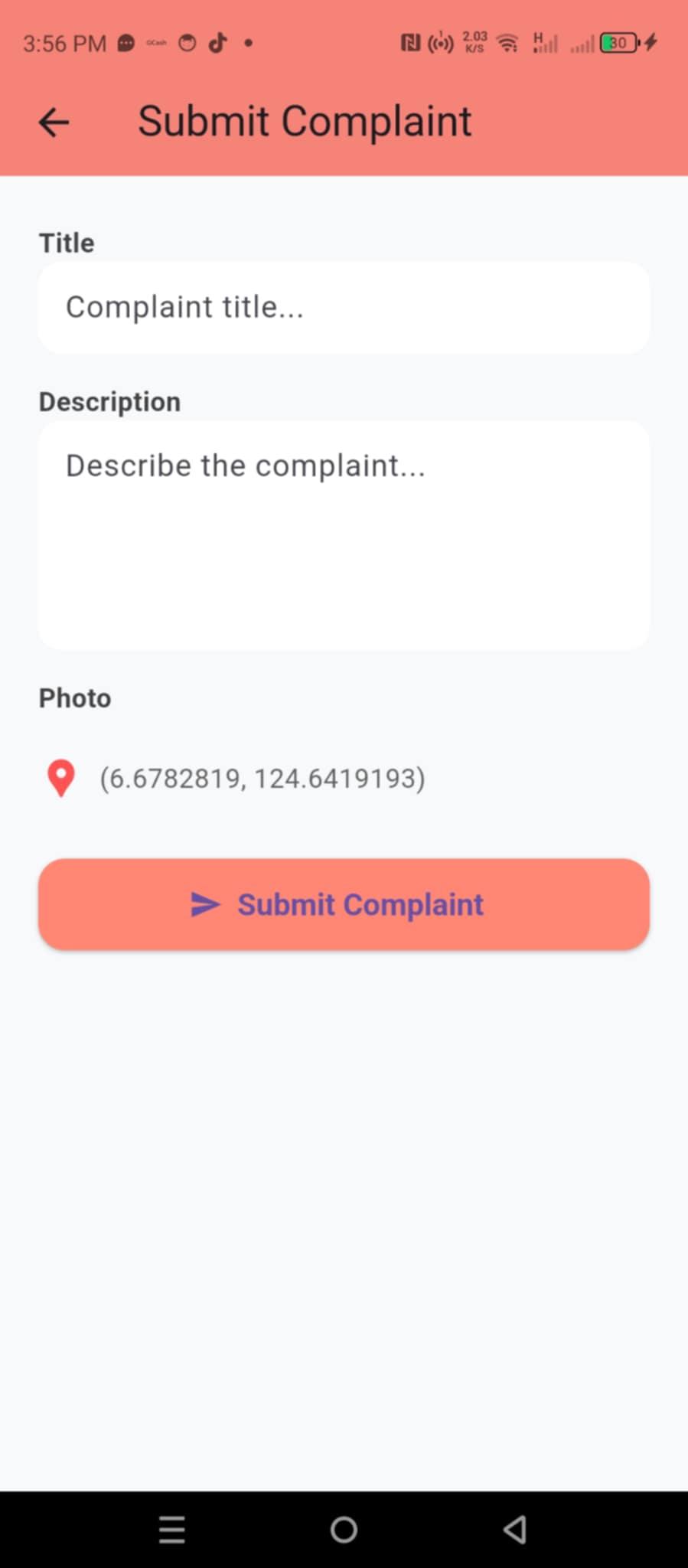
* The app is designed to run on Android devices.
* The backend and database are hosted using **XAMPP** for local testing or can be moved to an online server for public deployment.

### 6. ****Maintenance and Updates****

* After deployment, the system will be monitored for bugs and feedback.
* Future updates may include:
  + Multimedia complaint attachments
  + Notifications
  + Admin dashboard improvements

**Chapter 4 - System Design and Implementation**

**Screenshots of your app UI**

****

**Code snippets (Flutter widgets or Java layout)**

Main.dart

import 'package:flutter/material.dart';

import 'screens/login\_screen.dart';

void main() => runApp(const MyApp());

class MyApp extends StatelessWidget {

  const MyApp({super.key});

  @override

  Widget build(BuildContext context) {

    return MaterialApp(

      title: 'Complaint Response Application with Geolocator Tracking',

      debugShowCheckedModeBanner: false,

      theme: ThemeData(

        primarySwatch: Colors.blue,

        brightness: Brightness.light,

      ),

      darkTheme: ThemeData.dark(),

      home: const LoginScreen(),

    );

  }

}

**Backend PHP script sample**

Config.php

<?php

class Database {

    private $host = 'localhost';

    private $db\_name = 'cra\_db';

    private $username = 'root';

    private $password = '';

    public $conn;

    public function connect() {

        $this->conn = null;

        try {

            $this->conn = new PDO("mysql:host={$this->host};dbname={$this->db\_name}", $this->username, $this->password);

            $this->conn->setAttribute(PDO::ATTR\_ERRMODE, PDO::ERRMODE\_EXCEPTION);

        } catch (PDOException $e) {

            echo "Connection error: " . $e->getMessage();

        }

        return $this->conn;

    }

}

**Database schema / ERD**

**COMPLAINTS**

Id (PK)

User\_id (FK)

Title (VARCHAR)

Description (VARCHAR)

Photo (VARCHAR)

Latitude (VARCHAR)

Longitude (VARCHAR)

Status (Enum)

Created\_at (Timestamp)

**USERS**

Id (PK)

name (VARCHAR)

email (VARCHAR)

password VARCHAR)

**has**

**Chapter 5: Results and**

**Testing**

Test cases (with sample inputs and outputs)

**1. Login Functionality**

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case | Input | Expected Output | Result |
| Valid user login | Email: user@example.com Password: password123 | Redirect to userComplaintScreen | Passed |
| Invalid credentials | Email: wrong@example.com Password: wrongpass | Snackbar: “Invalid credentials” | Passed |
| Empty fields | Email: <br> Password: | Snackbar: “Invalid credentials” or input error shown | Passed |

**2. Complaint Form Submission**

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case | Input | Expected Output | Result |
| Complete form input | Title: Leakage Description: Pipe broken Location: GPS enabled | Complaint saved, return to Dashboard | Passed |
| Incomplete form (empty title) | Title: `` Description: Water issue | Show validation error | Passed |
| GPS disabled | Title: No water Location: unavailable | Prompt user to enable GPS or show error | Passed |

**3. Complaint List Display**

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case | Input | Expected Output | Result |
| User with complaints | User ID: 2 | List of complaints shown in cards | Passed |
| User with no complaints | User ID: 5 | Message: “You have no complaints yet.” | Passed |
| Error fetching complaints | Simulate server error / bad user ID | Message: “Failed to load complaints” | Passed |

**4. Complaint Location Viewer**

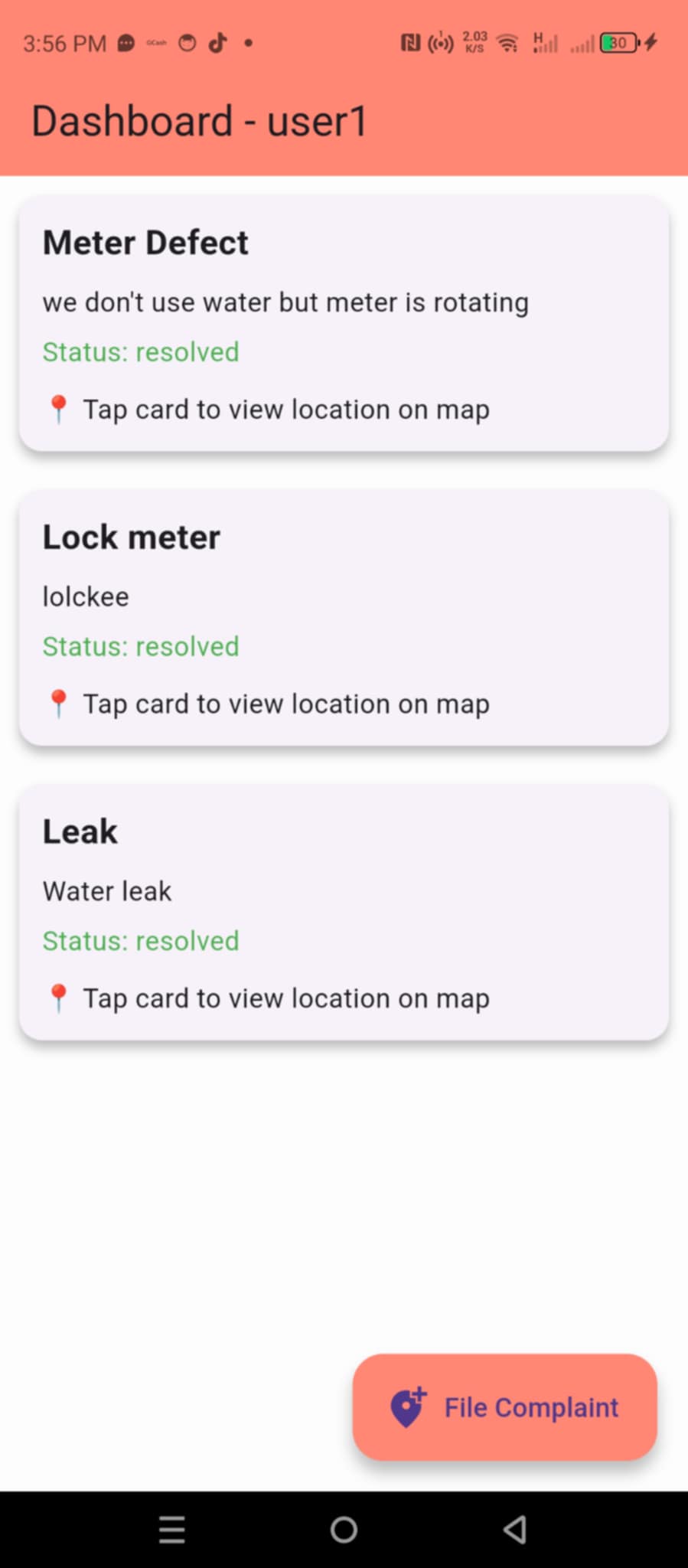
|  |  |  |  |
| --- | --- | --- | --- |
| Test Case | Input | Expected Output | Result |
| Tap on complaint card | Complaint with latitude: 10.34, longitude: 122.56 | Opens map screen with Google Maps link | Passed |
| Invalid location data | Complaint with empty or 0 coordinates | Map screen handles gracefully or shows “No location” | Passed |

**5. Dashboard Refresh Button**

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case | Action | Expected Output | Result |
| Tap refresh icon | Reload complaints | Latest complaints fetched & UI updated | Passed |
| Pull to refresh | Swipe down on list | Same: List refreshes | Passed |

**6. Admin Complaint List Access**

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case | Input | Expected Output | Result |
| Login as admin | Email: admin@example.com, role: admin | Redirect to AdminComplaintList | Passed |
| Login as user | Regular user login | Redirect to DashboardScreen | Passed |

**Screenshot of submitted entries**

**COMMON BUGS AND HOW YOU FIXED THEM**

1. **NDK Compatibility Fix**
   * Installed the **exact NDK version** required by dependencies (e.g., 27.0.12077973).
   * Set the correct path in local.properties:

properties

CopyEdit

ndk.dir=D:\\Android\\Sdk\\ndk\\27.0.12077973

1. **Correct Gradle and AGP Versions**
   * Updated gradle-wrapper.properties to a compatible Gradle version (e.g., 8.5).
   * Matched with Android Gradle Plugin (build.gradle) version (e.g., 8.1.1):

gradle

CopyEdit

classpath 'com.android.tools.build:gradle:8.1.1'

1. **Kotlin Version Alignment**
   * Set ext.kotlin\_version = '1.9.10' to match plugin compatibility.
2. **JDK Compatibility**
   * Switched to **JDK 17**, which is compatible with Gradle 8+.
3. **Flutter Clean & Rebuild**
   * Used:

bash

CopyEdit

flutter clean

flutter pub get

flutter build apk

* + This solved build inconsistencies and plugin errors.

1. **Device & Emulator Setup**
   * Ensured real devices were recognized via flutter devices.
   * For emulator, used the correct localhost IP 10.0.2.2 for API testing.
2. **Improved UI & Functionality**
   * Added:
     + Card headers and icons
     + Google Maps integration
     + Drive links with clickable icons
     + Refresh button in DashboardScreen

**Chapter 6: Conclusion and Recommendations**

**SUMMARY OF LEARNINGS**

In this project, I’ve gained valuable experience in various areas such as debugging and error fixing, understanding Flutter syntax, working with both stateful and stateless widgets, designing user interfaces in Flutter, connecting the frontend to a PHP backend, and organizing the widget tree effectively to create a clean and well-structured UI.

What can be improved?

Future developers should focus on improving geolocation accuracy, enhancing the user interface, and enabling media uploads for better incident reporting. Key features to add include report generation, real-time notifications, and interactive maps to track complaint locations. The backend should be secure and well-structured, with proper validation and scalable API design. Maintaining clean code, documentation, and using version control are essential for smooth development and future expansion.

**REFERENCES**

* Flutter - https://docs.flutter.dev
* PHP - https://www.php.net/manual/en/
* ChatGPT - https://chat.openai.com

**APPENDICES**

* Full source code (open source code docs inside repository)
* SQL Dump file (included in source code document)