SafePath

Home Product Service Contact

TASK 6

Database Design and Implementation

Road Sign and Road State Mobile Notification Application (SafePath)

GROUP 1

SafePath

AGENDA

- Core Data Elements & Attributes 01
- Conceptual Design 02
- ER Diagram 03
- Database Implementation 04
- Connecting to the database to Backend 05

Core Data Elements and Attributes

The SafePath application requires the following core data elements to support its functionality:

- **User**: Name, email, location (optional), preferred settings (e.g., notification preferences).
- **RoadSign**: Type, icon, meaning, category (e.g., warning, regulatory)
- **TrafficAlert**: Location, description, severity (high, moderate, low), timestamp
- **Report**: User-generated reports including type (accident, hazard), location, description, timestamp.
- **PetrolStation**: Location, name, distance from user (optional).

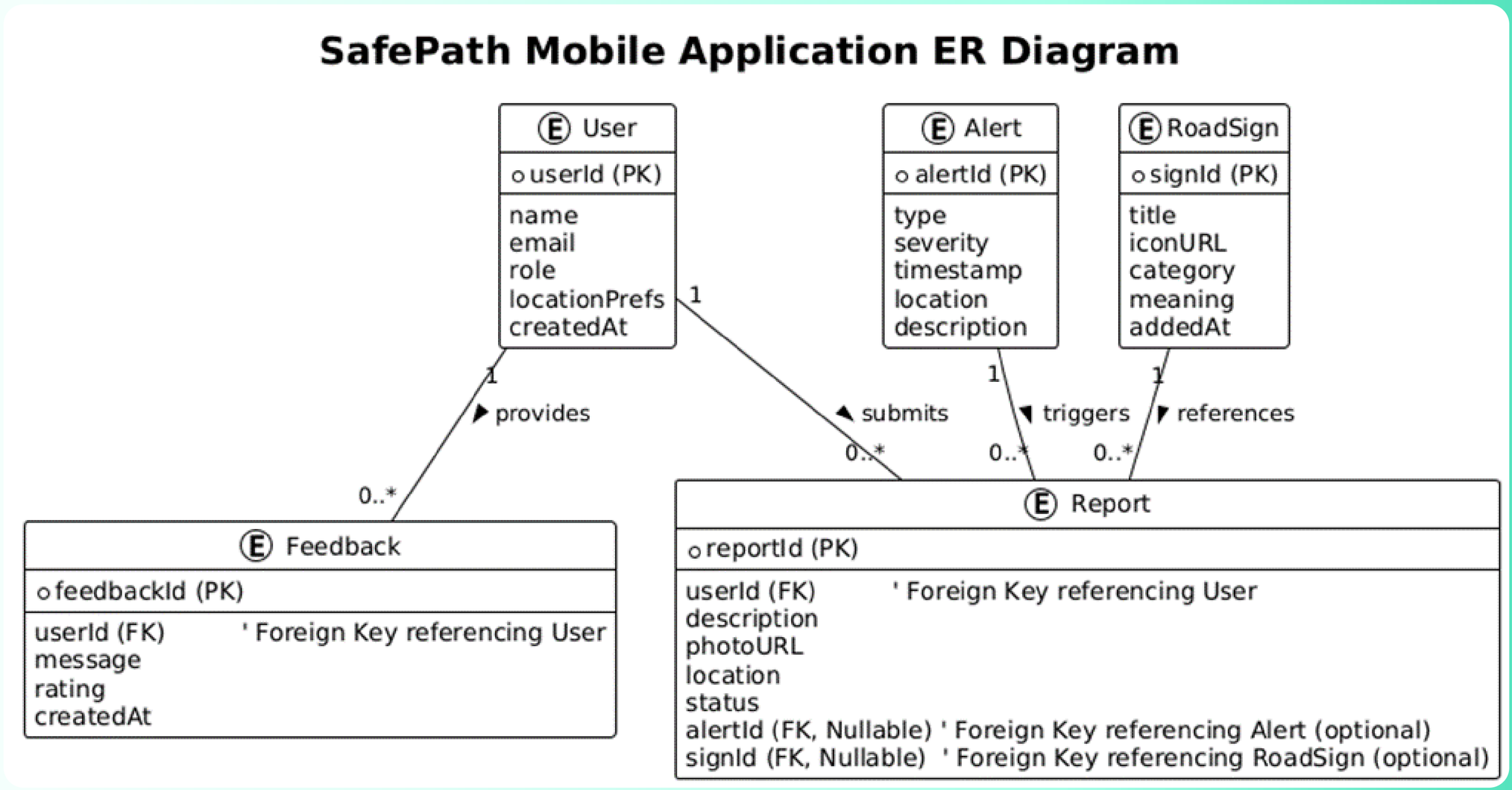
Entity	Primary Key	Attributes	Special Notes
User	userId	name, email, role, locationPrefs, createdAt	Central account entity
Alert	alertId	type, severity, timestamp, location, description	Real-time safety notifications
RoadSign	signId	title, iconURL, category, meaning, addedAt	Traffic sign repository
Report	reportId	userId (FK), photoURL, location, status, alertId (FK, nullable), signId (FK, nullable)	Incident documentation
Feedback	feedbackId	userId (FK), message, rating, createdAt	User experience records

Conceptual design

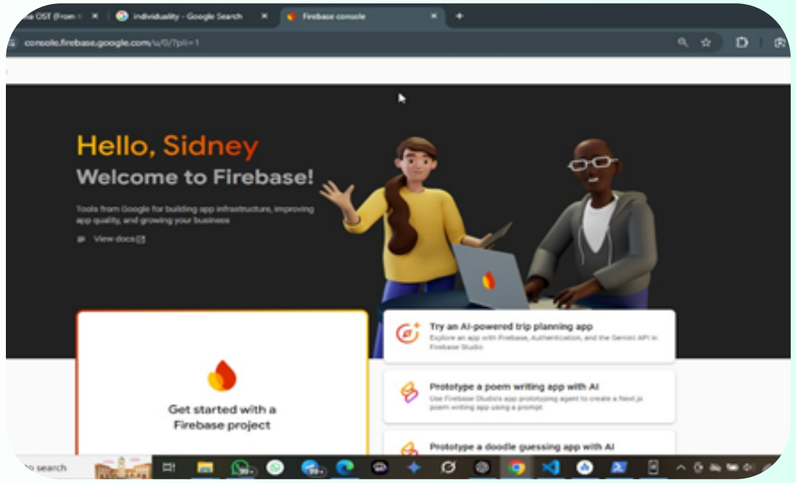
Key Relationships

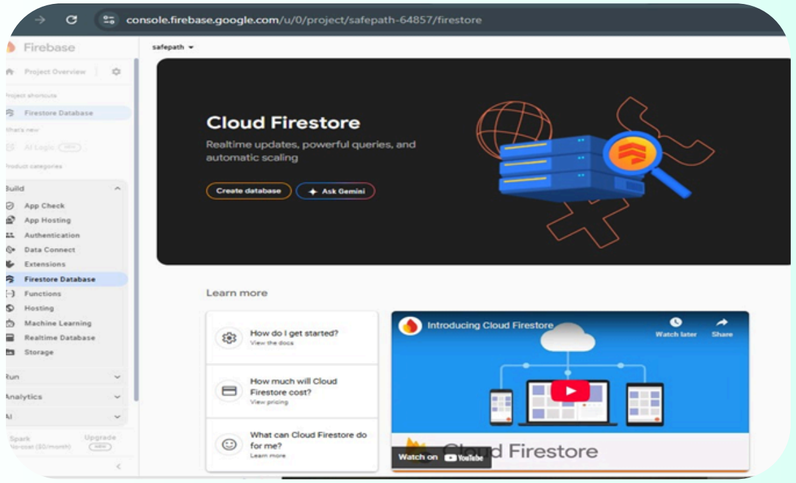
1. **User → Report** (1-to-many)
    a. A user submits multiple reports
    b. *Constraint*: Reports require a user (NOT NULL FK)
2. **User → Feedback** (1-to-many)
    a. A user provides multiple feedback entries
    b. *Constraint*: Feedback persists if user is deleted (nullable FK)
3. **Alert → Report** (0-to-many)
    a. Alerts may trigger multiple reports
    b. *Optional*: Reports can exist without alerts
4. **RoadSign → Report** (0-to-many)
    a. Signs may be referenced in multiple reports
    b. *Optional*: Reports can be sign-agnostic

ER DIAGRAM


SafePath Mobile Application ER Diagram
User
userId (PK)
name
email
role
locationPrefs
createdAt
Alert
alertId (PK)
type
severity
timestamp
location
description
RoadSign
signId (PK)
title
iconURL
category
meaning
addedAt
1
provides
0..*
submits
0..*
triggers
0..*
references
0..*
Feedback
feedbackId (PK)
userId (FK) ' Foreign Key referencing User
message
rating
createdAt
Report
reportId (PK)
userId (FK) ' Foreign Key referencing User
description
photoURL
location
status
alertId (FK, Nullable) ' Foreign Key referencing Alert (optional)
signId (FK, Nullable) ' Foreign Key referencing RoadSign (optional)

Database Implementation





FireBase

The database was set up in the **Firebase Console**

1. We Created a new Firebase project named **SafePath.**
2. We Initialized **Cloud Firestore** with security rules set to allow authenticated writes.
3. We Created the following collections with sample documents:

- **users:** { "email": "user@gmail.com", "name": "John Doe" }
- **alerts:** { "type": "Accident", "location": "Buea Checkpoint", "severity": "High", "timestamp": "2025-06-09T18:00:00Z" }
- **reports:** { "type": "Hazard", "location": "Main Street", "description": "Pothole reported" }
- **road_signs:** { "type": "Stop", "meaning": "Mandatory stop", "category": "Regulatory" }

Connecting the database to Backend

pubsec2.yaml

```dart
import 'package:flutter/material.dart';
import 'package:firebase_core/firebase_core.dart';
import 'package:cloud_firestore/cloud_firestore.dart';

void main() async {
  WidgetsFlutterBinding.ensureInitialized();
  await Firebase.initializeApp(
    options: DefaultFirebaseOptions.currentPlatform,
  );
  runApp(const SafePathApp());
}

class SafePathApp extends StatelessWidget {
  const SafePathApp({super.key});

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(title: const Text('SafePath')),
        body: Center(
          child: ElevatedButton(
            onPressed: () {
              _uploadReport();
            },
            child: const Text('Upload Sample Report'),
          ),
        ),
      ),
    );
  }
```

The backend leverages Flutter with Firebase integration for real-time data management. The following dependencies were added to pubspec.yaml

- Ran dart pub global activate flutterfire_cli and flutterfire configure to generate firebase_options.dart.
- Integrated the generated options into main.dart.

SafePath

Home Product Service Contact

Conclusion

The database design and implementation for SafePath using Firebase Cloud Firestore provide a robust foundation for real-time data management. The integration with Flutter ensures seamless connectivity, enabling features like report submission and alert delivery

See you in the Final Presentation

SafePath

Home Product Service Contact

TASK 6

THANK YOU

Road Sign and Road State Mobile Notification Application (SafePath)

GROUP 1