REPUBLIQUE DU CAMEROUN

PAIX – TRAVAIL – PATRIE

MINISTERE DE L'ENSEIGNEMENT SUPERIEURE

UNIVERSITE DE BUEA

REPUBLIC OF CAMEROON

PEACE – WORK – FATHERLAND

MINISTRY OF HIGHER EDUCATION

UNIVERSITY OF BUEA

FACULTY OF ENGINEERING AND TECHNOLOGY

PO BOX 63

BUEA, SOUTH WEST REGION

DEPARTMENT OF COMPUTER ENGINEERING

COURSE TITLE/CODE: Internet Programming (J2EE) and Mobile Programming – CEF440

COURSE INSTRUCTOR: Dr. NKEMENI Valery

REPORT OF DATABASE DESIGN (ERD) AND IMPLEMENTATION (TASK 6)

Presented by **GROUP 13**:

AFAYI MUNSFU OBASE NGALA – FE22A136

ATANKEU TCHAKOUTE ANGE N. – FE22A158

BETNESSI GRACE BLESSING – FE22A446

CHESSEU NJIPDI TERTULLIEN – FE22A181

JAMES ARISTIDE MASSANGO – FE22A226

JUNE 2025

ACADEMIC YEAR: 2024/2025

Table of Content

1. Introduction

In today's digital age, mobile network service quality and user satisfaction are crucial parameters in measuring the performance of network operators. As mobile usage continues to expand across regions, especially in developing countries, understanding the needs, frustrations, and performance experiences of subscribers is more important than ever. This report provides a detailed blueprint of a backend system that collects, processes, and manages data from mobile users and operators using Firebase as the cloud platform and an Entity Relationship Diagram (ERD) as the foundational model for data design.

The system includes features for user registration, issue reporting, feedback collection, notification management, operator management, and automated metrics logging. The report walks through the objective, the modeling of the ERD using PlantUML, the implementation using Firebase (Firestore), and the challenges encountered during development.

2. Objectives

a. Main Objective

To design and implement a Firebase-based data management system for collecting, analyzing, and managing user experience reports and operator performance using an Entity Relationship Diagram (ERD) as the foundational data design model.

b. Specific Objectives

1. To design an efficient and scalable ERD that represents all entities and their relationships within the system.
2. To implement the ERD structure using Firebase Firestore as a NoSQL cloud database.
3. To facilitate real-time reporting and feedback mechanisms for users.
4. To enable operators to manage access requests, view performance reports, and analyze user-submitted data.
5. To log network metrics such as speed, latency, and location in a structured and accessible manner.
6. To ensure data integrity, accessibility, and proper referencing between related documents.

3. Implementation

a. Explanation of ERD Designed

The ERD consists of 8 main entities:

1. Users

- **userID** (PK): Unique identifier for each user.
- **userName**, **phoneNumber**, **registrationDate**: Store personal details and account creation date.

2. ReportIssue

- **issueID** (PK), **userID** (FK), **operatorID** (FK): Tracks issues raised by users.
- Includes **type**, **activity**, **description**, **timestamp**, **location**.

3. FeedbackSurvey

- **feedbackID** (PK), **userID** (FK): Surveys filled by users.
- Contains ratings and a comment for qualitative insights.

4. Notification

- **notificationID** (PK), **userID** (FK): Sends system messages to users.
- **type**, **status**, **timestamp** track nature and read/unread status.

5. Operator

- **operatorID** (PK): Unique ID for network operators.
- **email**, **password**, **username**, **operatorType**: Manage credentials and classification.

6. RequestAccess

- **requestID** (PK), **operatorID** (FK): Access control system.
- **email**, **status**, **companyID**, **requestDate**, **approvedDate**, **docFilePath**.

7. Report

- **reportID** (PK), **operatorID** (FK): Performance reports.
- **dateGenerated**, **reportName**, **reportFormat**, **URL**: Documentation.

8. Metrics

- **metricID** (PK), **userID** (FK): Tracks telemetry.
- **metricType**, **deviceID**, **metricValue**.

Relationships:

- One-to-many relationships between Users and FeedbackSurvey, ReportIssue, Notification, Metrics.
- One-to-many relationships between Operator and RequestAccess, Report, ReportIssue.

This ERD gives us a structured way of understanding how the system's data is interconnected.

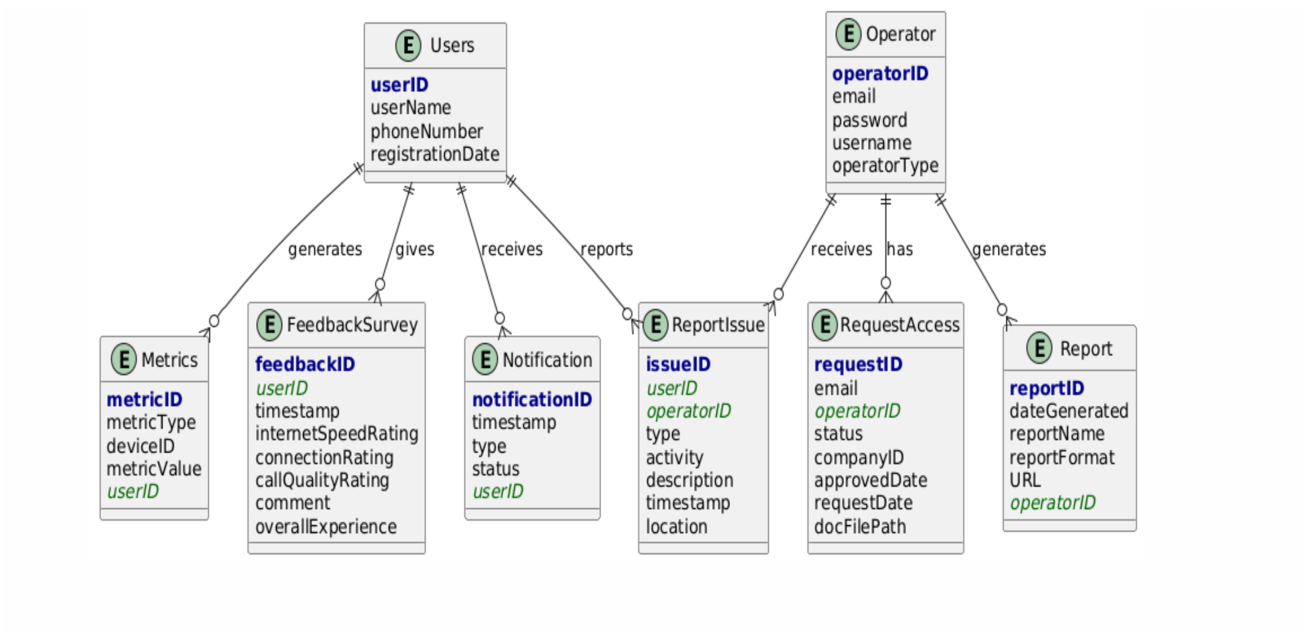

Fig 1: Entity Relation Diagram Design

b. Implementation Steps (Firestore in Firebase)

Step 1: Set Up Firebase Project

1. Go to Firebase Console (https://console.firebase.google.com/).
2. Click on "Add Project" and follow the setup wizard.
3. Enable Firestore by navigating to Build > Firestore Database and selecting "Start in test mode".

Step 2: Structure Firestore Collections

Each entity in the ERD becomes a **collection** in Firestore.

Collection: Users

- Document ID: userID
- Fields: userName, phoneNumber, registrationDate

Collection: ReportIssue

- Document ID: issueID
- Fields: userID (reference), operatorID (reference), type, activity, description, timestamp, location

Collection: FeedbackSurvey

- Document ID: feedbackID

- Fields: userID (reference), timestamp, internetSpeedRating, connectionRating, callQualityRating, comment, overallExperience

Collection: Notification

- Document ID: notificationID
- Fields: timestamp, type, status, userID (reference)

Collection: Operator

- Document ID: operatorID
- Fields: email, password, username, operatorType

Collection: RequestAccess

- Document ID: requestID
- Fields: email, operatorID (reference), status, companyID, approvedDate, requestDate, docFilePath

Collection: Report

- Document ID: reportID
- Fields: dateGenerated, reportName, reportFormat, URL, operatorID (reference)

Collection: Metrics

- Document ID: metricID
- Fields: metricType, deviceID, metricValue, userID (reference)

Step 3: Integrate Firestore with Mobile or Web App

- Use Firebase SDKs (Web, Android, Flutter, etc.) to connect frontend with Firestore.
- Use .collection("Users").doc(userID).set({...}) to insert users.
- Use .add() and .get() for query and insert operations.

Step 4: Implement Data Validation & Security Rules

- Define Firestore security rules to ensure:
  - Only authenticated users can write their feedback.
  - Operators can only access their own reports.
  - Prevent unauthorized access to request approvals.

Step 5: Test the Firestore Structure

- Populate Firestore with sample data.
- Query relationships using Firestore’s referencing system.

c. Relevance of the ERD in Firebase Implementation

The ERD serves as a **blueprint** for the Firestore data structure. Even though Firestore is non-relational, referencing documents based on the ERD ensures:

1. **Consistency**: The ERD guides the naming and layout of Firestore collections and documents.
2. **Scalability**: Well-planned references ensure data can be queried efficiently.
3. **Maintainability**: Developers can understand the relationships more clearly.
4. **Validation**: The ERD makes it easy to verify data integrity.
5. **Reference Mapping**: Although Firestore doesn't support joins, the ERD helps manage manual references between documents.

---

4. Challenges Encountered

1. **Modeling Relational Data in NoSQL**: Firestore is document-based, not relational. Creating references required thoughtful design to avoid performance bottlenecks.
2. **Security Rule Complexity**: Designing access control policies per user/operator type was time-consuming.
3. **Data Redundancy Avoidance**: Ensuring data isn't duplicated, while still being query-friendly, was challenging.
4. **Query Limitations**: Firestore doesn't support complex joins; data must be denormalized or manually joined on the client side.
5. **Real-Time Sync Management**: Handling multiple listeners and avoiding sync conflicts required careful planning.

---

5. Results

1. **Fully Functional Database Schema**: Based on the ERD, all collections and fields were successfully implemented.
2. **Integrated User Feedback System**: Users can submit feedback and report issues directly from the frontend.
3. **Operator Panel Access**: Operators can log in and view user reports and requests.
4. **Automated Metric Logging**: Background services log network performance metrics.
5. **Real-Time Notifications**: Notifications are sent and tracked with read/unread status.

---

6. Conclusion

Designing and implementing a cloud-based user feedback system with Firebase backed by an ERD structure ensures clarity, scalability, and maintainability. While Firestore is a NoSQL database, planning the data model through an ERD helps simulate relational logic, resulting in a better-organized and more efficient database system.

This report has shown how to structure a real-world system for network experience data collection and operator analysis in a way that is accessible to first-year students. It bridges the gap between theoretical ERD modeling and practical NoSQL database design, using Firebase Firestore.

Through clear modeling, detailed implementation, and awareness of potential challenges, this project serves as a strong foundation for future developments in mobile user experience systems and backend data management.