THE UNIVERSITY OF BUEA
P.O.BOX 63,
Buea, South West Region Cameroon

REPUBLIC OF CAMEROON

CEF 440

Internet Programming and Mobile Programming

Department: Computer Engineering

Faculty Of Engineering and Technology

TASK 6

Road Sign and Road State Mobile Notification Application

DATABASE DESIGN AND IMPLEMENTATION

Submitted to:

Dr. Nkemeni Valery

By:

Group 14

2023/2024

Contents

1. Introduction 1
   1.1. Database models 1
   1.2. Why data modelling is important 1
   1.3. What are the three levels of data abstraction? 2
      1.3.1. Conceptual data model 2
      1.3.2. Logical data model 2
      1.3.3. Physical data model 2
2. Choosing the right database type 3
   2.1. Why use a NoSQL database 3
   2.2. Using Firebase for the application 4
3. Modelling the database 6
   3.1. The conceptual Data Model 6
   3.2. The logical data model 7
   3.3. The physical data model 8
4. Conclusion 9
5. References 9

1. Introduction

Designing a database for a mobile application backend is a critical step in ensuring the scalability, performance and efficiency of the application. A well-designed database can improve data management, simplify queries, and enhance overall user experience.

In this report, we will explore the key considerations and best practices for designing a database for a mobile app backend, by understanding different different entities, attributes, and relationships between them.

When designing a database for a mobile app backend, several factors need to be considered to ensure its effectiveness and efficiency. These factors include the nature of the application, the volume and complexity of data, and the expected user interactions.

By following the best practices and principles of database design, we can create a database that meets the specific needs of our mobile app while ensuring scalability, performance, and data integrity.

1.1. Database models

Data modelling is the process of diagramming data flows. When creating a new or alternate database structure, the designer starts with a diagram of how data will flow into and out of the database. This flow diagram is used to define the characteristics of the data formats, structures, and database handling functions to efficiently support the data flow requirements. After the database has been built and deployed, the data model lives on to become the documentation and justification for why the database exists and how the data flows were designed.

The data model that results from this process provides a framework of relationships between data elements within a database as well as a guide for use of the data. Data models are a foundational element of software development and analytics. They provide a standardised method for defining and formatting database contents consistently across systems, enabling different applications to share the same data.

1.2. Why data modelling is important

A comprehensive and optimised data model helps create a simplified, logical database that eliminates redundancy, reduces storage requirements, and enables efficient retrieval. It also equips all systems with a ‘single source of truth’ – which is essential for effective operations and provable compliance

with regulations and regulatory requirements. Data modelling is a key step in two vital functions of a digital enterprise.

1.3. What are the three levels of data abstraction?

There are many types of data models, with different types of possible layouts. The data processing community identifies three kinds of modelling to represent levels of thought as the models are developed.

1.3.1. Conceptual data model

This is the "big picture" model that represents the overall structure and content but not the detail of the data plan. It is the typical starting point for data modelling, identifying the various data sets and data flow through the organisation. The conceptual model is the high-level blueprint for development of the logical and physical models.

1.3.2. Logical data model

The second level of detail is the logical data model. It most closely relates to the general definition of "data model" in that it describes the data flow and database content. The logical model adds detail to the overall structure in the conceptual model but does not include specifications for the database itself as the model can be applied to various database technologies and products.

1.3.3. Physical data model

The physical database model describes the specifics of how the logical model will be realised. It must contain enough detail to enable technologists to create the actual database structure in hardware and software to support the applications that will use it. Needless to say, the physical data model is specific to a designated database software system. There can be multiple physical models derived from a single logical model if different database systems will be used.

2. Choosing the right database type

The first step in designing a database for a mobile app is to choose the right database type for the app's needs. There are two main types of databases: relational and non-relational. Relational databases store data in tables with predefined schemas and use SQL to query and manipulate data. Non-relational databases store data in collections of documents, key-value pairs, graphs, or other formats, and use different query languages or APIs to access data.

Relational databases are good for structured and consistent data that requires complex queries, transactions, and integrity constraints. Non-relational databases are good for unstructured and dynamic data that requires high scalability, flexibility, and performance. Depending on your app's data model, functionality, and growth potential, the database type that best suits the app's requirements should be chosen.

2.1. Why use a NoSQL database

- **Handle large volumes of data at high speed with a scale-out architecture**

  The scale-out architecture of NoSQL systems provides a clear path to scalability when data volume or traffic grows. Achieving the same type of scalability with SQL databases can be expensive, require lots of engineering, or may not be feasible.

- **Store unstructured, semi-structured, or structured data**

  NoSQL databases have proven popular because they allow the data to be stored in ways that are easier to understand or closer to the way the data is used by applications. Fewer transformations are required when the data is stored or retrieved for use. Many different types of data, whether structured, unstructured, or semi-structured, can be stored and retrieved more easily.

  In addition, the schemas of many NoSQL databases are flexible and under the control of the developers, making it easier to adapt the database to new forms of data. This removes bottlenecks in the development process associated with asking a database administrator to redesign a SQL database.

- **NoSQL databases support widely used data formats**
  - Big data of all kinds — text data as well as time-series data

- JSON files, which are nested human-readable files consisting of names and value pairs. This format can capture highly complex parent-child hierarchical structures, which can be efficiently stored in document databases
- Simple binary values, lists, maps, and strings can be handled at high speed in key-value stores
- Sparse data can be efficiently stored in columnar databases, where null values take up no room at all. They are also effective for information that does not change frequently (nonvolatile data)
- Networks of interrelated information can be stored in graph databases.

- **Developer-friendly**

  Document databases use JSON as a way to turn data into something much more like code. This allows the structure of the data to be under the control of the developer.

  In addition, NoSQL databases store data in forms that are close to the kind of data objects used in applications, so fewer transformations are required when moving data in and out of the databases.

- **Wide Community base**

  Most NoSQL databases have a strong community of developers surrounding them. This means that there is an ecosystem of tools available and a community of other developers with which to connect.

- **Take full advantage of the cloud to deliver zero downtime**

  Delivering a database using a cluster of computers also allows the database to expand and contract capacity automatically.

  In addition, many NoSQL databases can be upgraded and allow the structure of the database to change with zero downtime.

2.2. Using Firebase for the application

Firebase by Google offers a suite of features that can be well-suited for developing the road state and road sign notification application. Here are the reasons why Firebase is a good choice for this project:

1. Real-time Database and Cloud Functions:

- Firebase provides a Realtime Database that allows for efficient storage and retrieval of real-time road state data. This is crucial for keeping users updated on current traffic conditions, accidents, and weather impacts.
- Cloud Functions within Firebase can be triggered by events like new road state data updates. These functions can process the data and send targeted notifications to users in affected areas.

2. Geospatial Functionality:

- Firebase integrates with Google Maps Platform, allowing the storage and query of geospatial data for road segments and road signs. This facilitates efficient location-based services, enabling the app to display relevant road state information and sign notifications based on the user's location.

3. Scalability and Security:

- Firebase is a cloud-based solution, so it scales automatically to handle increasing data volume and user base. This is essential for an application that might experience fluctuations in usage depending on traffic conditions.
- Firebase offers built-in security features like user authentication and authorization, helping to protect sensitive user data and ensure data integrity.

4. Offline Capabilities (Firestore):

- While the core functionality might rely on real-time data, Firebase offers Firestore, a NoSQL document database with offline capabilities. This allows users to access cached road sign data and potentially limited road state information even without an internet connection.

5. User Management:

- Firebase Authentication allows for user registration and login, enabling features like personalized notification preferences based on user location or road types of interest.

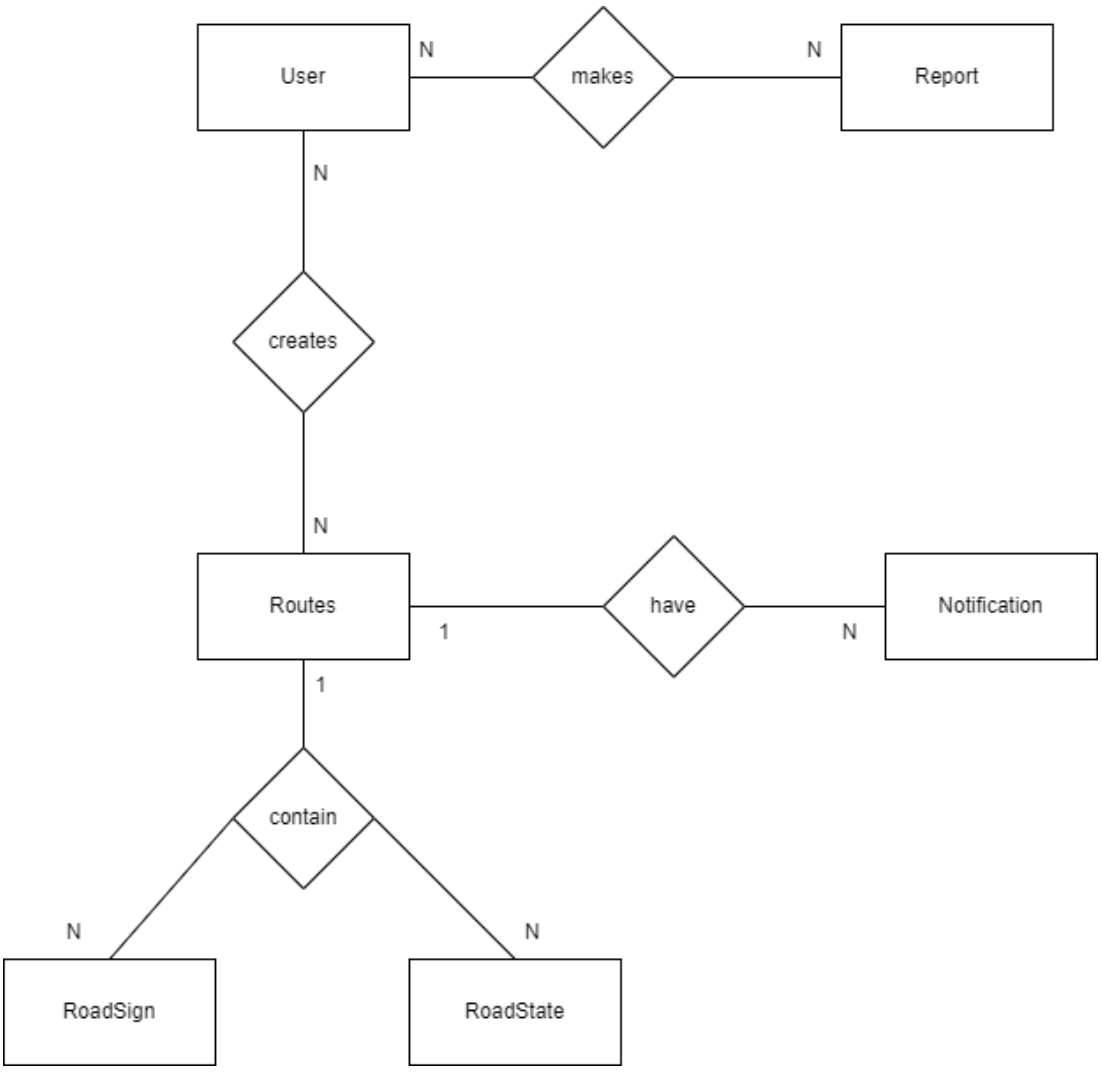
3. Modelling the database

3.1. The conceptual Data Model

The conceptual data model describes the database at a very high level and is useful to understand the needs or requirements of the database. One such popular model is the entity/relationship model (ER model). The E/R model specializes in entities, relationships, and even attributes that are used by database designers.

The conceptual model is developed independently of hardware specifications like data storage capacity, location or software specifications like DBMS vendor and technology. The focus is to represent data as a user will see it in the "real world.

Entity-relationship diagram of the application:

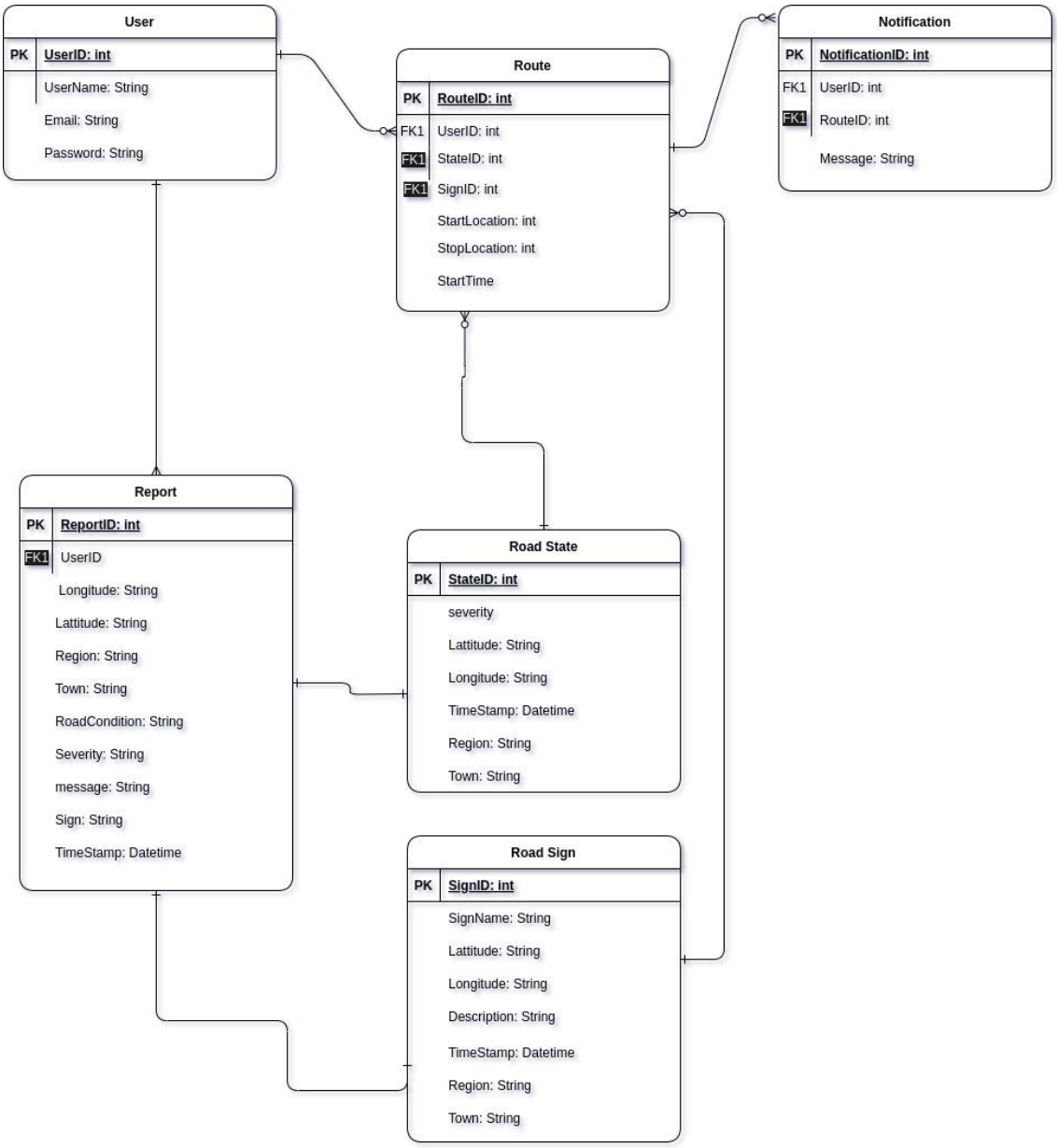


3.2. The logical data model

The logical data model contains representations that fully defines relationships in data, adding the details and structure of essential entities. It's important to note that the LDM remains data platform agnostic because it focuses on business needs, flexibility, and portability.

The LDM includes the specific attributes of each entity, the relationships between entities, and the cardinality of those relationships. It gives the team a solid framework to follow as they build the system, and it can be used to effectively and efficiently plan and implement cloud data warehouses, data marts, application databases, or data analysis datasets.

Logical model of the application:



3.3. The physical data model

A physical data model (PDM) is a data model that represents relational data objects. It describes the technology-specific and database-specific implementation of the data model and is the last step in transforming from a logical data model to a working database. A physical data model includes all the needed physical details to build a database.

Firebase is a document-oriented databases or NoSQL document store.

Physical model of the application:

```
Users{
    "UserID": string,
    "UserName": string,
    "Email": string,
    "Password": string,
},
Reports{
    "ReportID": ObjecctID,
    "longitude": number,
    "latitude": number,
    "region": string,
    "town": string,
    "roadCondition": string,
    "roadSign": string,
    "severity": string,
    "message": string,

},
Routes{
    "RouteID": ObjectID,
    "startLocation": string,
    "stopLocation": string,
},
```

```
Notifications{
    "NotificationID": ObjectID,
    "message": string,
    "timestamp": datetime
},

RoadState{
    "StateId": ObjectID,
    "severity": string,
    "latitude": number,
    "longitude": number,
    "description": string,
    "timestamp": datetime,
    "region": string,
    "town": string
},

RoadSign{
    "SignId": ObjectID,
    "signName": string,
    "latitude": number,
    "longitude": number,
    "description": string,
    "timestamp": datetime,
    "region": string,
    "town": string
}
```

The document-oriented databases or NoSQL document store is a modernised way of storing data as JSON rather than basic columns/rows — i.e. storing data in its native form. This storage system lets you retrieve, store, and manage document-oriented information

All Firebase Realtime Database data is stored as JSON objects. The database is like a cloud-hosted JSON tree. Unlike a SQL database, there are no tables or records. When data is added to the JSON tree, it becomes a node in the existing JSON structure with an associated key.

4. Conclusion

This report has explored the design and implementation of a database for a road state and road sign notification application. We've analyzed the application needs and ultimately concluded that a NoSQL database offers a more suitable solution due to its scalability, flexibility for real-time data, and efficient handling of geospatial queries.

The report further explored Firebase as a potential NoSQL database solution, highlighting its strengths in real-time data management, geospatial functionality, scalability, and security. Firebase's integration with other Google Cloud Platform services and development tools could streamline the development process.

By implementing a well-designed and secure database, the road state and road sign notification application can provide valuable information to drivers, improving road awareness and potentially enhancing safety on the roads. Additionally, future advancements like image recognition integration and predictive analytics can be explored to further enrich the application's functionalities.

5. References

https://www.altexsoft.com/blog/data-modeling/

https://www.mongodb.com/resources/basics/databases/nosql-explained/advantages

https://medium.com/@arorasagar1811/nosql-data-modelling-244ea86f6270

https://www.geeksforgeeks.org/data-models-in-dbms/

https://www.geeksforgeeks.org/how-to-design-a-database-for-mobile-app-backend/