Data Distribution Document

S6 Software Engineering

4207734

Matei-Cristian Mitran

Fontys Eindhoven

21.05.2023

1. Introduction

YouSound, an enterprise-grade music application, utilizes standard data handling practices to ensure a seamless and secure user experience. This document outlines my approach related to data access patterns, implications of the CAP theorem, GDPR implications and backup and recovery protocols.

2. Data Access Patterns

Data access patterns play a critical role in the functioning of YouSound's microservices architecture. I make use of the database-per-service pattern, where each service User, Social, and Music manages its own database. This division ensures that the databases can be independently developed, deployed, and scaled.

MongoDB is chosen for the User and Social services due to its flexible, document-oriented model that makes it an excellent fit for handling rapidly changing data. With MongoDB's inherent support for horizontal scaling through sharding, it efficiently accommodates the high-volume nature of social interaction data, and the evolving attributes of user profiles.

The Music service, on the other hand, uses MariaDB, a highly performant relational database. The structured, consistent nature of music data makes it well-suited for a relational database. MariaDB's powerful querying capabilities, transactional integrity, and robustness make it ideal for managing and querying our music metadata, ensuring the reliable delivery of music streaming services.

Microservices communicate with their respective databases, preventing conflicts, and enhancing overall performance. Furthermore, caching is employed to speed up common data access operations, further improving system performance and user experience.

Through these strategic choices and tailored data access patterns, we ensure that YouSound delivers a robust, responsive, and reliable user experience.

3. CAP Theorem

The CAP Theorem is a fundamental principle that details the behavior of distributed systems. Standing for Consistency, Availability, and Partition Tolerance, the theorem states that a distributed system can only guarantee two out of these three properties at any given time.

In YouSound's system design, MongoDB and MariaDB are the main databases used for storing data. To stay true to the CAP theorem, the system was designed and built in a way that finds a careful balance between these three key aspects.

Consistency in my application is about ensuring that all users see the same data, no matter which node they interact with. This is particularly critical for social features, where posts, reels, and comments must be up to date across all users.

Availability refers to the system's ability to serve requests, irrespective of the state of some nodes. The music service, for instance, must continue to operate, delivering uninterrupted music streaming, even in the event of a partition or node failure.

Partition Tolerance means the system continues to operate despite network failures that result in some nodes being cut-off from others.

However, the CAP theorem states we can't achieve all three at once. Depending on the specific requirements of a service - be it user, social, or music - we then choose between consistency and availability, ensuring the right balance for the best user experience.

In conclusion, the CAP theorem forms a fundamental guide in the design and implementation decisions for YouSound, providing a robust and efficient data handling in this application.

4. Data Privacy (GDPR)

In my project, YouSound, I prioritize adherence to data privacy regulations, especially the General Data Protection Regulation (GDPR) from the European Union. I have designed my data handling practices to respect user privacy and give users control over their personal information.

A key aspect of GDPR compliance is respecting the "Right to be Forgotten". I implemented a feature that allows users to request the deletion of all their stored data across our databases. This process is facilitated using the RabbitMQ Message Broker, which ensures the request is communicated to all relevant microservices and actioned promptly.

The "Right to Access" is another crucial GDPR principle. YouSound users can request access to their personal data at any time. A comprehensive report that includes all the information YouSound has gathered about them is generated, helping them understand what data is being processed, and why.

In accordance with the "Right to Rectification", users can correct any inaccurate personal data. This ensures accuracy and up-to-dateness of the data stored.

Finally, the "Right to Data Portability" allows users to receive their data in a standard, machine-readable format. In this project, this means allowing users to export their liked songs and user data as CSV files. This gives users control over their data, which can be used as they see fit.

5. Data Backup & Disaster Recovery

5.1 MariaDB GCP VM

In the event of a disaster that results in data loss, follow the steps outlined below. This guide works because a backup of the Virtual Machine is done every 72 hours. The MariaDB server is deployed on Google Cloud Platform on an Ubuntu 20.4 LTS Virtual Machine.

1. **Examine Disaster:** Examine logs and error messages, perform queries and test database connection to find out extent of data loss.
2. **Stop VM:** Log into GCP and stop the Virtual Machine.
3. **Create new VM From Backup:** Create new VM from the most recent backup.
4. **Start new VM:** Start the new Virtual Machine created from the most recent backup.
5. **Verify Data:** Verify the data to examine the integrity of the data keeping in mind the data stored after the last backup will not be available
6. **Update Connection:** Update database connection in the application
7. **Testing:** Perform all the unit tests, integration tests, and the system tests.

5.2 MongoDB Atlas Cluster

For the MongoDB Cloud Atlas Cluster, the steps are basically the same. The only difference is the data is backed up every 7 days due to the music data in MariaDB changing more frequently compared to the user and social data in MongoDB. For instance, new songs might be added to the platform more regularly, or the metadata associated with the music might be updated frequently.

1. **Examine Disaster:** Examine logs and error messages, perform queries and test database connection to find out extent of data loss.
2. **Access Dashboard:** Log into MongoDB Atlas Account and go to Clusters.
3. **Create new Cluster from Snapshot:** Create new cluster from the most recent snapshot.
4. **Verify Data:** Verify the data to examine the integrity of the data keeping in mind the data stored after the last backup will not be available
5. **Update Connection:** Update database connection in the application
6. **Testing:** Perform all the unit tests, integration tests, and the system tests.