**FIREBASE**

**WHAT IS DATABASE?**

A database is organized collection of data stored and accessed electronically. Small databases can be stored on a file system, while large databases are hosted on computer clusters or cloud storage.

**WHAT IS DMBS?**

A database management system or (DBMS) is a software which interacts with end-user. A DBMS can be used to administer (i.e Create, Read, Update or Delete) the database.

Example: MS SQL, MYSQL, Oracle

**WHAT IS FIREBASE?**

Firebase is a platform created by Google in 2011 for web and mobile applications.

Firebase is categorized as a NoSQL database program, which stores data in JSON-like documents. Firebase is a full package that can help in developing your mobile or web applications faster with fewer resources and more efficiency.

Firebase provides tools and services for the backend of application. Some of its services are listed here;

* Cloud Firestore
* Cloud Functions
* Authentication
* Hosting
* Cloud Storage
* Google Analytics
* Predictions
* Cloud Messaging
* Dynamic Links
* Remote Config

**ADVANTAGES OF FIREBASE**

1. Ease of integration
2. Fast and Safe Hosting
3. Firebase testing services to improve app quality
4. Offers no cost tier

**KEY POINTS**

* All firebase services need to be activated from console
* To use firebase services their instance needs to be created

**TO INSTALL:**

* First install npm
* npm install -g firebase-tools

**TO INSTALL IN PROJECT:**

* firebase login
* dart pub global activate flutterfire\_cli
* Go to C:\Users\Username\AppData\Local\Pub\Cache\bin
* Copy path
* Add as path
* Open cmd in project folder
* flutterfire configure
* Enter project name
* Select project to configure
* Select device to configure
* Goto android folder in project
* Open build.gradle
* Add in dependencies > classpath 'com.google.gms:google-services:4.3.10'
* Goto app folder and open build.gradle
* Add following line > apply plugin: 'com.google.gms.google-services' with other apply plugin lines
* Add following plugins in main.dart

import 'package:firebase\_core/firebase\_core.dart';

import 'firebase\_options.dart';

**FIREBASE AUTHENTICATION:**

Firebase authentication provides service to authenticate users to application. i.e. to register or to login. Every time as user is created, it will be stored in Firebase. Although password is not shown in firebase (i.e. not shown to administrator)

It provides authentication using multiple providers. Few are discussed:

1. **Email and Password:**

This service provides authentication using email and password.

To use this service:

* Go to firebase console
* Go to authentication tab
* Click add new provider and enable email and password

This service enables application to verify using email link (If enabled while adding this service in firebase console).

Good practice is to enable verification so that user can reset password. Although service can be used without verification.

1. **Social Authentication:**

This service provides authentication using social media account such as Google, Facebook, Twitter and many more.

To use this service SHA 1 or SHA 256 keys are required

* Check in environment variables if JAVA\_HOME is available
* Go to project folder
* Go to android folder
* Open PowerShell
* Run command “gradlew signingReport”
* If above command does not works us “./gradlew signingReport”
* Copy any SHA1 Key
* Go to firebase console
* Click Add Provider
* Enable any social service
* Set email same as your firebase email
* Go to firebase project settings and scroll to bottom
* Click add fingerprint
* Paste copied SHA1 Key.

To use this service in emulator, your emulator needs to be playstore enabled. i.e., Pixel 4

1. **Anonymous Login:**

This provider enables user to login as guest. This type of account is called temporary account but it is not deleted from firebase and user data can also be saved in firebase. i.e., User state/data will be retained. This provider can authenticate without email or social account.

To use this provider, enable anonymous login in firebase authentication in firebase console.

**SQL VS NOSQL DATABASE**

|  |  |
| --- | --- |
| **SQL** | **NOSQL** |
| SQL databases are relational. | NoSQL databases are non-relational. |
| SQL databases use structured query language and have a predefined schema. | NoSQL databases have dynamic schemas for unstructured data. |
| SQL databases are vertically scalable. | NoSQL databases are horizontally scalable. |
| SQL databases are table-based. | NoSQL databases are document, key-value, graph, or wide-column stores. |
| SQL databases are better for multi-row transactions | NoSQL is better for unstructured data like documents or JSON. |
| Oracle, MySQL, Microsoft SQL Server, and PostgreSQL | Document: MongoDB and CouchDB, Key-value: Redis and DynamoDB, Wide-column: Cassandra and HBase, Graph: Neo4j and Amazon Neptune |

**FIREBASE RLTDB VS FIRESTORE:**

|  |  |
| --- | --- |
| **RLTDB** | **FIRESTORE** |
| Low latency | More feature rich |
| Simple data is very easy to store. | Simple data is easy to store in documents, which are very similar to JSON. |
| Complex, hierarchical data is harder to organize at scale. | Complex, hierarchical data is easier to organize at scale. |
| Offline support for Apple Android clients. | Offline support for Apple, Android, and web clients. |
| **Queries with limited**sorting and filtering. | Queries with compound sorting and filtering. |
| Queries can sort or filter on a property, but not both. | You can chain filters and combine filtering and sorting on a property in a single query. |
| Basic write and transaction operations. | Advanced write and transaction operations. |
| Scale to around 200,000 concurrent connections and 1,000 writes/second in a single database. Scaling beyond that requires sharing data across multiple databases. | Scales completely automatically. Currently, scaling limits are around 1 million concurrent connections and 10,000 writes/second. |
| Rules that separate authorization and validation. | Rules that combine authorization and validation. |
| Reads and writes from mobile SDKs secured by Realtime Database Rules. | Reads and writes from mobile SDKs secured by Cloud Firestore Security Rules. |

**CLOUD FIRESTORE:**

Firestore stores data within "documents", which are contained within "collections". Documents can also contain nested collections.

To install: flutter pub add cloud\_firestore

Like other services firestore also needs to be activated from firebase console

* A user can have multiple documents in collection
* A collection or document can be created with any name.