## DATABASE SELECTION REPORT

**Available Options:**

- **Firebase Firestore**
- MongoDB atlas
- Supabase

**My Selection Criteria:**

- Structure and Schema
- Fetching and storing speed
- Query complexity
- Integration with other services (in my case Firebase Auth)
- Offline data caching
- ease of integration and usage
- community and support (within Dart community)
- Overall app compatibility

**Comparison:**

| SUPA | MONGO Atlas | FIRESTORE |
|---|---|---|
| SQL based on PostgresQL | No-SQL | No-SQL |
| Non-flexible schema | flexible schema | flexible schema |
| Relativity fast based on PostgresQL - | Fast - Perfect for apps with complex queries - latency depends of the complexity of the quey | Super fast and perfect for real-time queries for simple datasets - Low latency - needed in my case because it's a task management app, so why would the user wait. |
| SQL - full-on queries including complex and nested queries | Allows complex queries - supports search and filtering | Queries level simple to moderate. Suits my App. |
| Integrating with App Auth services is not easy.<br>Has it's own Auth services | Can be integrated but needs customization and backend apis. | Best for Firebase Auth integration and other Firebase services. |
| No builtin offline caching, need to store in sharedprefs or hive or any local storage | No built-in caching | Has offline data caching, perfect for the app |
| Integration and implementation not so easy<br>Moderate Community support | Abit more complex implementation.<br>dedicated Flutter plugin<br>Moderate Community support<br>Community support in general but not tailored to flutter | It's a google service so integration is seamless.<br>configuration is easy on the firebase console.<br>Official Flutter plugins to help implement the firebase services.<br>Highest Community support-tailored to flutter and dart |

## Conclusion:

- I Needed NoSql DB for a more flexible schema and for a Faster and more Real Time data fetching and storing.
- Needed support just in case I encountered any errors or issues.
- Didn't need complex Queries - only simple data reference in each collection
- Needed the seamless integration between the auth services and my firestore services (relation between the logged in/ signed up users and other project collections (tasks and projects))
- I changed the collection schema multiple times until i was satisfied with the current schema that I have, so flexibility was important in my case.
- Built-in offline  caching comes in handy when dealing with a cloud-based task manager.
- Faster storing and fetching is a must in a task manager app, why would a user wait for a task manager App to finish loading ?

**Rewan Gameel**

**rewangameelrr2008@gmail.com**