



TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA ĐIỆN TỬ - VIỄN THÔNG



Tích hợp Firebase Realtime Database cho dữ liệu real-time

GVHD: TS. Nguyễn Duy Nhật Viễn

Nhóm sinh viên thực hiện (7):

Ngô Trung Chính - 106220211

Nguyễn Hữu Duy - 106220215

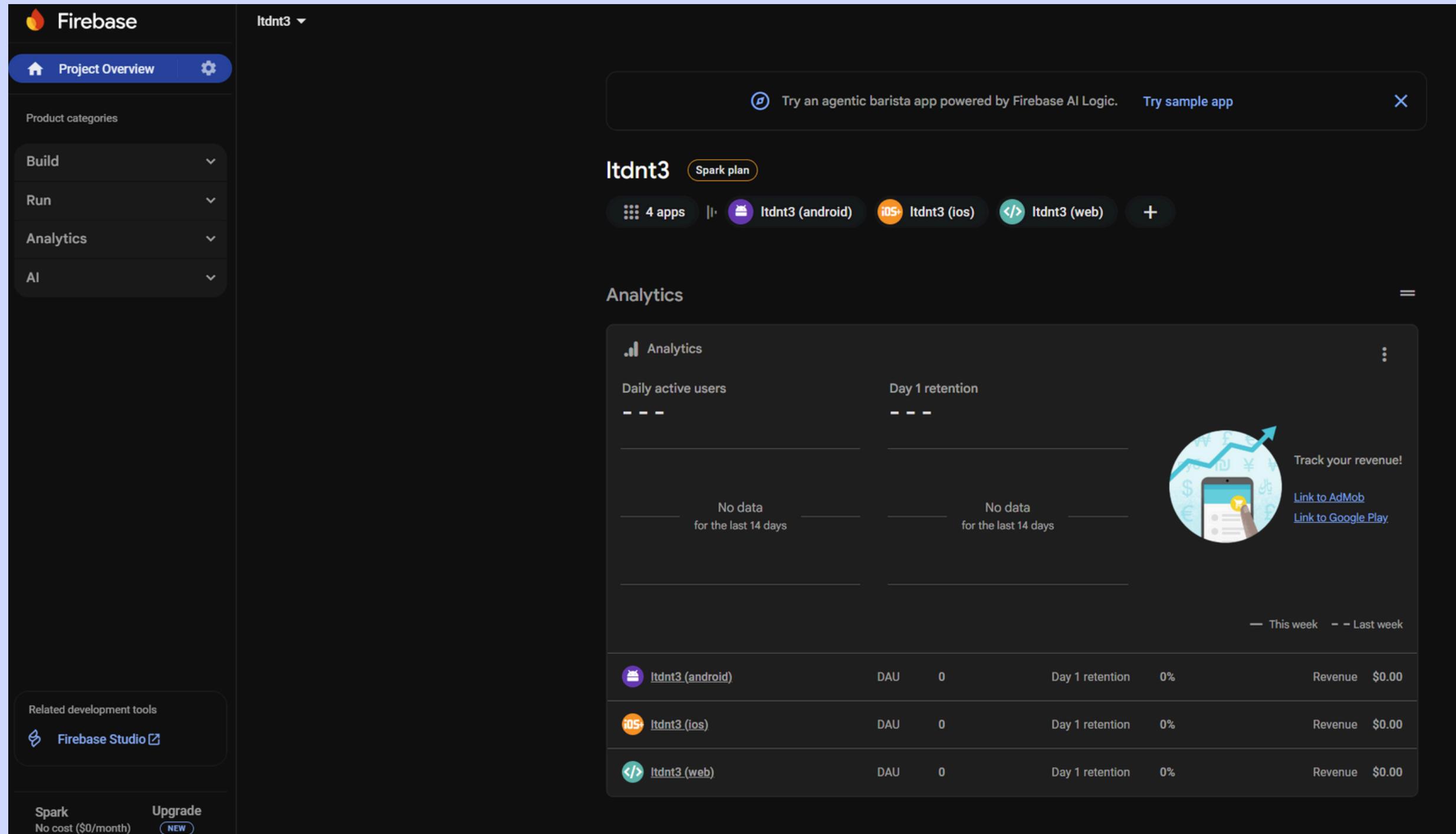
Đà Nẵng, 2025

Nội Dung

- Setup Firebase project
- CRUD operations với Realtime Database
- Offline capabilities và sync
- Security rules và data validation

1) Setup Firebase project

1. Truy cập Firebase Console → tạo Firebase Project mới.
2. Thêm ứng dụng **Web app** hoặc Android/iOS nếu muốn chạy đa nền tảng.



1) Setup Firebase project

3. **Bật Google Sign-In:** vào Authentication → Sign-in method → Google → Enable.

The screenshot shows the Firebase console's Authentication interface. On the left, there's a sidebar with Project Overview, Authentication (which is selected and highlighted in blue), and other product categories like Build, Run, Analytics, and AI. The main content area is titled "Authentication" and has tabs for Users, Sign-in method (which is active), Templates, Usage, Settings, and Extensions. Below this, under "Sign-in providers", there's a section titled "Select a sign-in provider (step 1 of 2)". It lists "Native providers" (Email/Password, Phone, Anonymous) and "Additional providers" (Google, Facebook, Play Games, Game Center, Apple, GitHub, Microsoft, Twitter, Yahoo). The "Google" provider is checked and highlighted with a green checkmark. At the bottom right of this section, it says "Enabled". A button labeled "Add new provider" is also visible.

1) Setup Firebase project

4. **Realtime Database:** tạo database → chọn chế độ test để thử nghiệm.
5. Cập nhật **Security Rules** để kiểm soát quyền truy cập dữ liệu

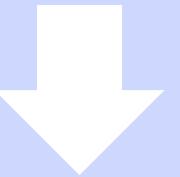
The screenshot shows the Firebase Realtime Database Rules playground. The left sidebar lists various services: Project Overview, Authentication, Realtime Database (selected), Firestore Database, Functions, Hosting, Machine Learning, Storage, Run, Analytics, and AI. The main area displays the security rules configuration. The code editor contains the following JSON-based security rules:

```
1  {
2    "rules": {
3      ".read": "auth != null",
4      ".write": "auth != null",
5
6      "messages": {
7        ".indexOn": ["time"],
8
9        "$msgId": {
10          // Chỉ chủ sở hữu mới được tạo/sửa/xoá
11          ".write": "auth != null && (
12            !data.exists() && newData.child('uid').val() === auth.uid) ||
13            (data.exists() && data.child('uid').val() === auth.uid)
14        ",
15
16          // Validate cấu trúc & kiểu dữ liệu
17          ".validate": "newData.hasChildren(['text','time','uid']) &&
18            newData.child('text').isString() &&
19            newData.child('text').val().length >= 1 &&
20            newData.child('text').val().length <= 500 &&
21            newData.child('time').isNumber() &&
22            newData.child('uid').isString()"
23        }
24      }
25    }
26  }
27 }
```

1) Setup Firebase project

6. Dùng **FlutterFire CLI** để tạo file firebase_options.dart cho Flutter project:

```
dart pub global activate flutterfire cli
```



```
firebase login --reauth
```

→ có lệnh **flutterfire** để sử dụng ở bước sau

7. Flutterfire Configure

```
flutterfire configure --project=<PROJECT_ID>
```

→ Sinh ra file: **lib/firebase_options.dart**

2) CRUD operations với Realtime Database project

Firebase Realtime Database

- **Firebase Realtime Database** là cơ sở dữ liệu **NoSQL**, lưu trữ dữ liệu dưới dạng **JSON** tree.
- Tính năng **realtime**: tất cả client đang kết nối sẽ nhận dữ liệu mới ngay lập tức khi server thay đổi.

CRUD operations

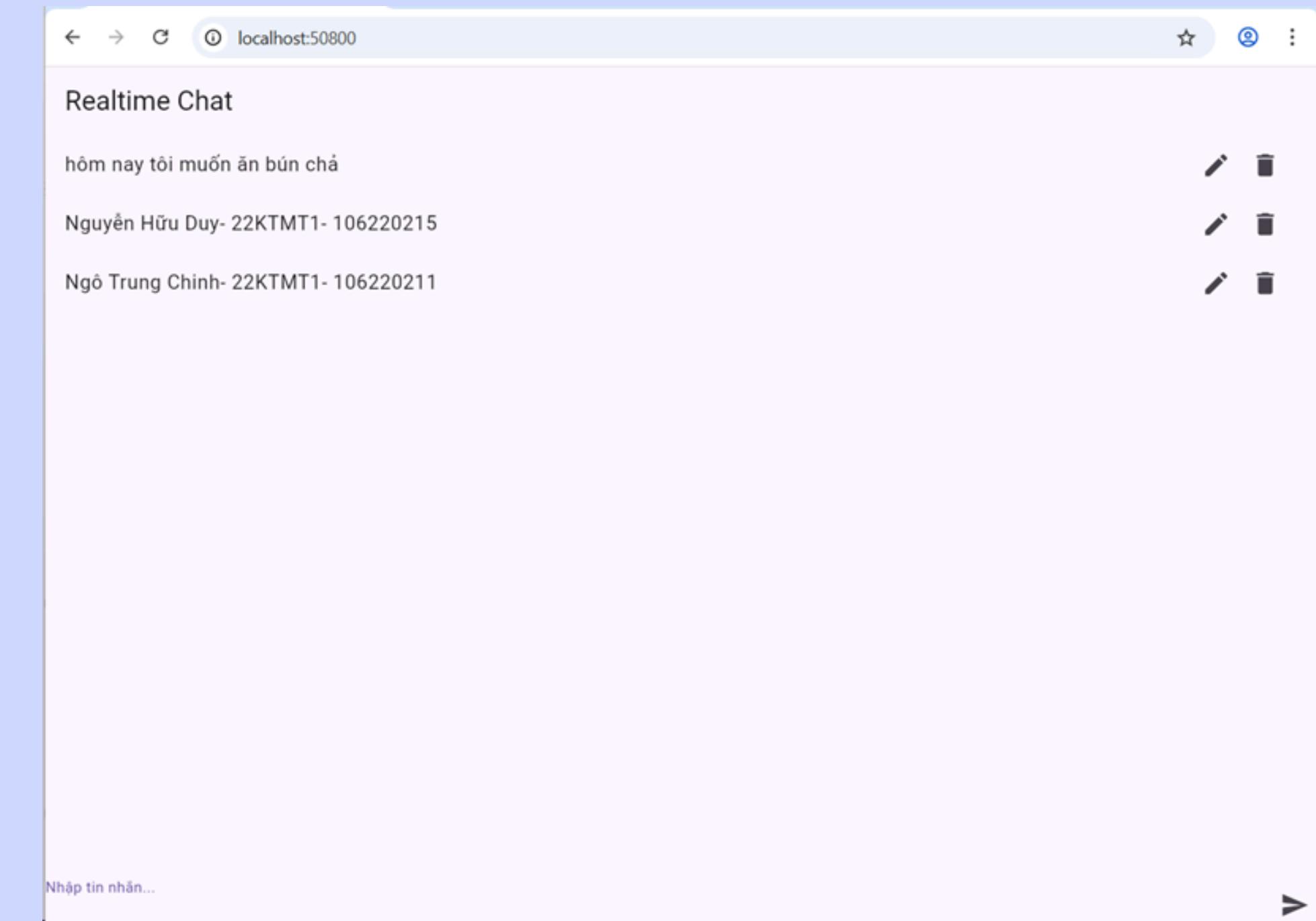
- **1. Create (Thêm tin nhắn)**
 - Dữ liệu tin nhắn gồm: text , time, uid.
 - Sử dụng db.push().set() để thêm tin nhắn mới vào Realtime Database.
- **2. Read (Đọc tin nhắn)**
 - Dùng StreamBuilder để lắng nghe dữ liệu realtime.
- **3. Update (Cập nhật tin nhắn)**
 - Chỉ owner (người gửi) mới được sửa tin nhắn.
 - Lấy snapshot theo key → kiểm tra uid → update nếu hợp lệ.
- **4. Delete (Xóa tin nhắn)**
 - Tương tự update, chỉ owner mới xóa được tin nhắn.

2) CRUD operations với Realtime Database project

```
void addMsg(String text) {
    final uid = FirebaseAuth.instance.currentUser!.uid;
    db.push().set({
        "text": text,
        "time": ServerValue.timestamp,
        "uid": uid,
    });
    c.clear();
}

void updateMsg(String key, String newText) async {
    final uid = FirebaseAuth.instance.currentUser!.uid;
    final snap = await db.child(key).get();
    if (snap.exists) {
        final data = Map<String, dynamic>.from(snap.value as Map);
        if (data['uid'] == uid) {
            db.child(key).update({"text": newText});
        }
    }
}

void delMsg(String key) async {
    final uid = FirebaseAuth.instance.currentUser!.uid;
    final snap = await db.child(key).get();
    if (snap.exists) {
        final data = Map<String, dynamic>.from(snap.value as Map);
        if (data['uid'] == uid) {
            db.child(key).remove();
        }
    }
}
```



3) Offline capabilities và sync

Offline capabilities

- **Firebase Realtime Database** hỗ trợ cơ chế lưu trữ tạm thời dữ liệu ở bộ nhớ cục bộ (local cache) → Điều này cho phép ứng dụng vẫn hoạt động ngay cả khi không có kết nối Internet.
- Khi người dùng thực hiện thay đổi dữ liệu trong trạng thái **offline**, các thao tác đó không bị bỏ qua.
Thay vào đó, Firebase sẽ:
 - Ghi dữ liệu vào **queue pending writes** (hàng đợi thay đổi tạm)
 - Hiển thị dữ liệu mới ngay lập tức lên UI thông qua cơ chế **local state update**
→ Người dùng thấy kết quả ngay mà không cần mạng (đây gọi là latency compensation).

3) Offline capabilities và sync

Offline capabilities

```
// Bật offline persistence cho mobile
if (!kIsWeb) FirebaseDatabase.instance.setPersistenceEnabled(true);
// Luôn đồng bộ dữ liệu "messages" với Local cache
db.keepSynced(true);
}
```

Ưu điểm

- Ứng dụng không bị “đơ” khi mất mạng → Trải nghiệm người dùng tốt hơn
- Dữ liệu vẫn đọc được từ local cache → Không cần phải reload mạng
- Người dùng có thể tiếp tục nhập dữ liệu → Không bị gián đoạn thao tác

3) Offline capabilities và sync

sync (synchronize)

Khi kết nối Internet được khôi phục:

- **Firebase** sẽ tự động gửi toàn bộ các thay đổi tạm (**pending writes**) lên **server**
- Server xử lý dữ liệu và cập nhật trạng thái
- Các **client** khác đang online sẽ nhận dữ liệu mới theo thời gian thực
- **Local cache** được **đồng bộ** lại với giá trị đúng nhất từ **server**

3) Offline capabilities và sync

sync (synchronize)

Ứng dụng thực tế

- Chat realtime → Gửi tin nhắn khi “mất mạng”, sẽ gửi đi khi online lại
- Ứng dụng ghi chú → Người dùng ghi nội dung mọi lúc
- App bán hàng / kho hàng → Dữ liệu không mất khi chưa có Internet
- App du lịch / vùng hải đảo → Dùng ngay cả khi không có sóng

4) Security rules và data validation

Security rules

- **Security Rules** nhằm kiểm soát quyền đọc/ghi và xác thực dữ liệu trước khi lưu.
 - Chỉ người dùng đã đăng nhập (auth != null) mới đọc/ghi.

Data validation

- **Data Validation** giúp đảm bảo dữ liệu lưu vào database là hợp lệ
 - Mỗi tin nhắn cần có đủ trường: text, time, uid.
 - text là chuỗi, từ 1–500 ký tự.
 - uid phải trùng với auth.uid → chỉ owner được sửa/xóa.

The screenshot shows a code editor with two panes. The left pane displays line numbers from 1 to 18. The right pane shows the following code:

```
1  {
2   "rules": {
3     "messages": {
4       ".read": "auth != null",
5       ".write": "auth != null",
6       ".indexOn": ["time"],
7       "$msg": {
8         ".validate": "newData.hasChildren(['text','time','uid']) &&
9                     newData.child('text').isString() &&
10                    newData.child('text').val().length > 0 &&
11                    newData.child('text').val().length <= 500 &&
12                    newData.child('time').isNumber() &&
13                    newData.child('uid').val() == auth.uid"
14       }
15     }
16   }
17 }
18 }
```

The code defines security rules for a 'messages' node. It requires authentication for both reading and writing. It also defines a validation rule for each message object, ensuring it has children for 'text', 'time', and 'uid', that 'text' is a string, its length is between 1 and 500, 'time' is a number, and 'uid' matches the user's ID.

THANK YOU

TÀI LIỆU THAM KHẢO

- [1] Google Developers, “Firebase Realtime Database Overview,” Google, 2025. [Online]. Available: <https://firebase.google.com/docs/database>
- [2] Google Developers, “Read and Write Data on Flutter,” Google, 2025. Available: <https://firebase.google.com/docs/database/flutter/read-and-write>
- [3] Google Developers, “Offline Capabilities,” Google, 2025. Available: <https://firebase.google.com/docs/database/android/offline-capabilities>
- [4] Google Developers, “Security Rules Basics,” Google, 2025. Available: <https://firebase.google.com/docs/rules/basics>

BẢNG PHÂN CÔNG CÔNG VIỆC TRONG NHÓM

STT	HỌ VÀ TÊN	NHIỆM VỤ	KHỐI LƯỢNG
11	Ngô Trung Chinh	<p>Tìm hiểu lý thuyết về Firebase Realtime Database và Flutter.</p> <ul style="list-style-type: none"> - Thiết lập Firebase Project, cấu hình Authentication và Realtime Database. - Viết phản báo cáo lý thuyết và setup môi trường. - Soạn Security Rules & Data Validation cho Firebase. 	50%
15	Nguyễn Hữu Duy	<ul style="list-style-type: none"> - Xây dựng giao diện chat realtime bằng Flutter. - Cài đặt Firebase CRUD (Create, Read, Update, Delete). - Thêm chức năng Google Sign-In và Offline Sync. - Hoàn thiện code demo và kiểm thử ứng dụng trên web. - Tổng hợp kết quả và viết phản kết luận báo cáo. 	50%