# SSIS NextGen Flutter Mobile

# Mesina, Charly March L.

## 1. Introduction

Develop the Flutter mobile application for SSIS NextGen. The project involves implementing secure login, biometric authentication, role-based navigation, Firebase push notifications, and local notification handling etc.

## 2. Dependencies

The following Flutter packages are used in the application:

* **cupertino\_icons** (^1.0.8) — iOS-styled icons for UI consistency.
* **google\_fonts** (^6.0.0) — for consistent typography across the app.
* **http** (^1.4.0) — for REST API communication with backend services.
* **shared\_preferences** (^2.2.2) — for storing lightweight persistent user and app settings.
* **flutter\_secure\_storage** (^9.2.4) — for securely storing sensitive data (e.g., tokens).
* **intl** (^0.20.0) — for internationalization and date/number formatting.
* **firebase\_core** (^4.0.0) — for initializing Firebase services.
* **firebase\_messaging** (^16.0.0) — for handling push notifications.
* **awesome\_notifications** (^0.10.1) — for displaying and managing local notifications.
* **local\_auth** (^2.1.7) — for biometric authentication (fingerprint/Face ID).

## 3. Backend Database

The backend uses **SQL Server** as the primary data store for user accounts, roles, and preferences. In addition, the backend **API** is built with **ASP.NET Core** (Visual Studio 2022) to handle user authentication, biometric verification, and device token management for personalized push notifications.

**API Endpoints**

* POST /api/Authenticate/register = signup
* POST /api/Authenticate /login = login
* POST /api/Authenticate /biometriclogin = biometric login
* POST /api/Authenticate /verify-email = confirm email
* POST /api/Authenticate /request-password-reset = request reset
* POST /api/Authenticate /reset-password = reset password
* POST /api/Authenticate /save-fcm-token = store device token
* GET /api/Authenticate /get-fcm-token {username} = get fcm token
* POST /api/Authenticate /assign-role = assign roles(user and admin)
* GET /api/Authenticate /all-user-roles = fetch all users

* GET /api/Category = fetch categories
* GET /api/Category/{id} = get category by Id
* POST /api/Category/create = create a new category
* PUT /api/Category/update/{id} = update a existing category by Id
* DELETE /api/Category/delete/{id} = delete a existing category by Id
* POST /api/Category/seed/{count} = add more categories (2 or more)
* DELETE /api/Category/remove/{count} = remove/delete more categories
* GET /api/Product = fetch products
* GET /api/Product /{id} = get product by Id
* POST /api/Product /create = create a new product
* PUT /api/Product /update/{id} = update a existing product by Id
* DELETE /api/Product /delete/{id} = delete a existing product by Id
* POST /api/Product /seed/{count} = add more products (2 or more)
* DELETE /api/Product /remove/{count} = remove/delete more products

## 4. Data Flow

1. **Login / Registration**
   * User enters credentials in the mobile app.
   * App sends request to backend API.
   * Backend verifies credentials.
   * On success, role-based redirection occurs (Admin > MachineScreen, User > MachineScreenusers).

**Biometric Authentication**

* + User authenticates with fingerprint/Face ID(coming soon).
  + Backend verifies biometric token.
  + On success, user is redirected to the respective dashboard.

**Notifications**

* + Firebase Cloud Messaging delivers push notifications.
  + **Awesome Notifications** as local alerts.

**Preferences & Version Control**

* + App version and biometric settings stored in **SharedPreferences**.
  + Version displayed in login screen footer.

1. **Machine Screen (User).**
   * Machine Types = A-VIM, B-scale, C-scale, D-lock.
   * Machine Types = only Take and Return button.
   * About, Settings and Log-out access only for the user.
   * Version and Date/Time displayed footer.
2. **Machine Screen (Admin).**
   * Machine Types = A-VIM, B-scale, C-scale, D-lock.
   * Machine Types = Access all buttons.
   * Users, Force Sync, Initialize DB, About, Check Update, Settings and Log-out access for the Admin.
   * Version and Date/Time displayed footer.
3. **Auth Services**

**Authentication & Biometric Login**

* + Login = Username/Password Authentication and saves username for stored in SharedPreferences to able to show in the header right side.
  + Signup = Create an Account (Username, Password, Email, Biometric (Fingerprint/FaceID (coming soon)).
  + verifyEmail = Confirms user email with a verification code.
  + loginWithBiometric = Uses Fingerprint to login authentication
  + requestPasswordReset / resetPassword = Reset Password
  + logout = clears JWT and Role

**Token Handling**

* + saveUserToken = Saves FCM token in backend.
  + getToken = Read JWT from storage
  + saveUsername / getLoggedInUsername = last logged-in username
  + saveUserRole / getUserRole = last user role

**Users and Role**

* + getAllUsers = Fetches all users with roles and biometric type from database
  + assignRole = Admin and User assign role as an Admin only

**Category Management (CRUD)**

* + getCategories = Fetches all categories from database
  + createCategory = Create a new Category
  + updateCategory = Update Category by Id
  + deleteCategory = Delete Category by Id

**Product Management (CRUD)**

* + getProducts = Fetches all Product from database
  + createProduct = Create new Product with details by Id
  + updateProduct = Update Product by Id
  + deleteProduct = Delete Product by Id

## 5. Conclusion

This project provides a secure and user-friendly Flutter mobile application for SSIS NextGen, which is supported by an effective ASP.NET Core + SQL Server Database backend. It enables efficient authentication, role-based access, and effective management of users, machines, categories, products, and notifications.