ID Building Application Documentation

# Overview

The ID Building Application is a Flask-based web application that allows users to register themselves by providing details such as username, first name, last name, email, and phone number. Upon successful registration:  
- A Universal ID is generated for each user.  
- User details are stored in a local MySQL database.  
- A confirmation email with the Universal ID is sent to the user's email address.  
- User details are forwarded to another application (App1) for Keycloak registration.

# Features

- User-friendly web form for registration.  
- Validation of all input fields.  
- Unique Universal ID generation using UUID.  
- Local storage of user data in MySQL.  
- Automated email notification upon successful registration.  
- Forwarding of user details to a second app (App1) for Keycloak user registration.  
- Comprehensive error handling and response messaging.

# Requirements

Python Libraries:  
- Flask  
- mysql-connector-python  
- Flask-Mail  
- requests  
  
Database:  
- MySQL Server (Database name: ID\_APP)  
- Table `users` with columns:  
 - id (Primary Key, Auto Increment)  
 - username (VARCHAR)  
 - first\_name (VARCHAR)  
 - last\_name (VARCHAR)  
 - email (VARCHAR, Unique)  
 - phone (VARCHAR)  
 - universal\_id (VARCHAR)  
  
Email:  
- Gmail SMTP Server  
- App Password configured for sending emails (not the regular Gmail password)

# Flask Application Structure

Database Connection:  
```  
mysql.connector.connect(  
 host="localhost",  
 user="root",  
 password="Blr@2025",  
 database="ID\_APP"  
)  
```  
  
Email Configuration (Flask-Mail):  
```  
app.config['MAIL\_SERVER'] = 'smtp.gmail.com'  
app.config['MAIL\_PORT'] = 587  
app.config['MAIL\_USE\_TLS'] = True  
app.config['MAIL\_USERNAME'] = 'tejasmgauthamapcogsys@gmail.com'  
app.config['MAIL\_PASSWORD'] = 'your-app-password'  
app.config['MAIL\_DEFAULT\_SENDER'] = 'your-email@gmail.com'  
```  
  
Universal ID Generation:  
- A 32-character unique ID is generated using `uuid.uuid4().hex`.  
  
Routes:  
- GET `/` : Serves the HTML registration form (index.html).  
- POST `/register` : Handles:  
 - Validates input data.  
 - Generates Universal ID.  
 - Inserts user details into MySQL database.  
 - Sends a confirmation email.  
 - Forwards user data to App1 (Keycloak registration).  
  
Response Scenarios:  
- 201: Registration successful (local + Keycloak).  
- 400: Missing fields or email already exists.  
- 500: Local save successful but Keycloak registration failed.  
  
Sending Confirmation Email:  
- On successful database insert, the app sends a friendly confirmation email with Universal ID to the registered email.

# Frontend (index.html)

The HTML form collects:  
- Username  
- First Name  
- Last Name  
- Email  
- Phone Number  
  
Upon submission:  
- JavaScript fetch sends a POST request with user data to the /register endpoint.  
- Displays the registration result (success or error message) dynamically.

# Forwarding to App1 (Keycloak Registration)

After storing user details locally:  
- The app makes a POST request to http://localhost:5016/register (App1).  
- Sends:  
 - username  
 - email  
 - first\_name  
 - last\_name  
 - phone  
 - externalId (Universal ID)  
  
App1 is responsible for creating the user in Keycloak with the given details.

# Error Handling

- If the email already exists in the MySQL database, registration is blocked with a clear error message.  
- If saving in MySQL succeeds but forwarding to App1 fails, the user gets an appropriate error that local registration succeeded but Keycloak registration failed.

# Changes Made During Development

- Initially, only username, first name, last name, and email were handled.  
- Added Phone Number field to both frontend form and backend database.  
- Forwarded complete user data including phone number and external ID to App1.  
- Improved error handling to distinguish between local save success and external (Keycloak) registration failure.  
- Added robust email confirmation using Flask-Mail.  
- Added structured response messaging for better frontend experience.

# Port and Running

- The app runs on localhost:5015.  
  
To run:  
```  
python app.py  
```

# Important Notes

- Ensure the MySQL database is running.  
- Ensure App1 server is running on port 5016 to receive forwarded registration.  
- Correctly configure Flask-Mail with an app-specific password if using Gmail.

----------------------------------------- END OF DOCUMENTATION ----------------------------------------