**Project Overview**

**The project aims to create a system that provides real-time flight status updates and notifications to passengers. This system retrieves data from airport databases, processes it on the backend, and displays it on the frontend. Additionally, it sends notifications to users about any changes in flight status.**

**Key Features**

**Real-time Updates: Displays the current status of flights, including delays, cancellations, and gate changes.**

**Push Notifications: Sends notifications via SMS, email, or app notifications using Kafka or RabbitMQ.**

**Integration with Airport Systems: Retrieves data from airport databases for accurate and timely information.**

**Technologies Used**

**Frontend: HTML, CSS, React.js**

**Backend: Python (Flask)**

**Database: MongoDB**

**Notifications: Firebase Cloud Messaging (FCM)**

**File Structure**

**plaintext**

**Copy code**

**flight-status-notifications/**

**│**

**├── backend/**

**│ ├── app.py**

**│ ├── config.py**

**│ ├── flight\_status.py**

**│ ├── requirements.txt**

**│ └── templates/**

**│ └── index.html**

**│**

**├── frontend/**

**│ ├── public/**

**│ │ ├── index.html**

**│ │ └── ...**

**│ ├── src/**

**│ │ ├── App.js**

**│ │ ├── index.js**

**│ │ └── ...**

**│ └── package.json**

**│**

**├── .gitignore**

**├── README.md**

**└── docker-compose.yml**

**Implementation**

**1. Backend**

**File: backend/app.py**

**Description: This file initializes the Flask app, connects to MongoDB, and sets up Firebase for notifications. It includes routes for fetching flight status and sending notifications.**

**Code:**

**python**

**from flask import Flask, jsonify, render\_template**

**from flask\_pymongo import PyMongo**

**import firebase\_admin**

**from firebase\_admin import messaging, credentials**

**from flight\_status import get\_flight\_status**

**app = Flask(\_\_name\_\_)**

**app.config["MONGO\_URI"] = "mongodb://localhost:27017/flightStatusDB"**

**mongo = PyMongo(app)**

**cred = credentials.Certificate("path/to/firebase\_credentials.json")**

**firebase\_admin.initialize\_app(cred)**

**@app.route('/')**

**def index():**

**return render\_template('index.html')**

**@app.route('/api/status/<flight\_id>')**

**def flight\_status(flight\_id):**

**status = get\_flight\_status(flight\_id)**

**if status:**

**return jsonify(status)**

**else:**

**return jsonify({"error": "Flight not found"}), 404**

**@app.route('/api/notify', methods=['POST'])**

**def notify():**

**message = messaging.Message(**

**notification=messaging.Notification(**

**title='Flight Status Update',**

**body='Your flight status has changed.'**

**),**

**topic='flight\_updates'**

**)**

**response = messaging.send(message)**

**return jsonify({"message\_id": response})**

**if \_\_name\_\_ == '\_\_main\_\_':**

**app.run(debug=True)**

**File: backend/flight\_status.py**

**Description: Contains a mock function to simulate fetching flight status from a database.**

**Code:**

**python**

**def get\_flight\_status(flight\_id):**

**# Mock data for demonstration**

**flights = {**

**"AB123": {"status": "On Time", "gate": "A1"},**

**"CD456": {"status": "Delayed", "gate": "B2"},**

**}**

**return flights.get(flight\_id, None)**

**File: backend/requirements.txt**

**Description: Lists the dependencies required for the backend.**

**Content:**

**Flask**

**Flask-PyMongo**

**firebase-admin**

**2. Frontend**

**File: frontend/src/App.js**

**Description: Main React component for the frontend. It allows users to input a flight ID and check its status.**

**Code:**

**javascript**

**import React, { useState } from 'react';**

**import './App.css';**

**function App() {**

**const [flightId, setFlightId] = useState('');**

**const [status, setStatus] = useState(null);**

**const checkStatus = async () => {**

**const response = await fetch(`/api/status/${flightId}`);**

**const data = await response.json();**

**setStatus(data);**

**};**

**return (**

**<div className="App">**

**<header className="App-header">**

**<h1>Flight Status Checker</h1>**

**<input**

**type="text"**

**value={flightId}**

**onChange={(e) => setFlightId(e.target.value)}**

**placeholder="Enter Flight ID"**

**/>**

**<button onClick={checkStatus}>Check Status</button>**

**{status && (**

**<div>**

**<h2>Status: {status.status}</h2>**

**<p>Gate: {status.gate}</p>**

**</div>**

**)}**

**</header>**

**</div>**

**);**

**}**

**export default App;**

**File: frontend/package.json**

**Description: Specifies the project’s metadata and dependencies for the frontend.**

**Content:**

**json**

**{**

**"name": "flight-status-frontend",**

**"version": "1.0.0",**

**"main": "index.js",**

**"dependencies": {**

**"react": "^17.0.2",**

**"react-dom": "^17.0.2",**

**"react-scripts": "4.0.3"**

**},**

**"scripts": {**

**"start": "react-scripts start",**

**"build": "react-scripts build",**

**"test": "react-scripts test",**

**"eject": "react-scripts eject"**

**}**

**}**

**File: frontend/public/index.html**

**Description: The HTML template for the React app.**

**Content:**

**html**

**<!DOCTYPE html>**

**<html lang="en">**

**<head>**

**<meta charset="UTF-8">**

**<meta name="viewport" content="width=device-width, initial-scale=1.0">**

**<title>Flight Status Checker</title>**

**</head>**

**<body>**

**<div id="root"></div>**

**</body>**

**</html>**

**Docker Setup**

**File: docker-compose.yml**

**Description: Defines the services for Docker Compose to set up and run the application.**

**Content:**

**yaml**

**version: '3'**

**services:**

**backend:**

**build: ./backend**

**ports:**

**- "5000:5000"**

**frontend:**

**build: ./frontend**

**ports:**

**- "3000:3000"**

**Instructions**

**Clone the Repository**

**git clone <repository\_url>**

**cd flight-status-notifications**

**Set up Backend**

**cd backend**

**python3 -m venv venv**

**source venv/bin/activate**

**pip install -r requirements.txt**

**python app.py**

**Set up Frontend**

**cd frontend**

**npm install**

**npm start**

**Run Docker**

**docker-compose up**$env:NODE\_OPTIONS = "--openssl-legacy-provider"

SCREENSCREENSHOTS :



