Swami Keshvanand Institute of Technology,

Management & Gramothan, Jaipur

PROJECT KIT

Title of the Project

Miles Acquisition System

Abstract

The Miles Acquisition System (MAS) is a modern, user-centric platform developed to streamline the process of tracking and awarding airline miles to frequent flyers. Designed for airline staff, administrators, and passengers, the system allows users to submit flight details, which are then verified and approved by authorized personnel. Once approved, miles are credited to the user's account in real time. Built using Angular for the frontend and Spring Boot for the backend, MAS integrates a secure authentication mechanism and provides a responsive, intuitive interface. With a scalable backend and structured data management, the system ensures efficient handling of user information, flight data, and mileage transactions, making it a reliable solution for enhancing airline customer loyalty programs.

Objectives

The Miles Acquisition System aims to deliver a digital solution for tracking, verifying, and rewarding airline miles to frequent flyers. The key objectives include:

- Streamlining the submission and verification of flight details by users.
- Enabling airline staff and administrators to approve or reject mile requests efficiently.

- Automatically updating the user's miles balance upon approval to reflect earned rewards.
- Ensuring secure user authentication and robust backend data handling for reliability and scalability.

Generic Keywords -

- Miles Acquisition
- Digital Miles Repository
- Reward Tracking System
- Digital Repository

Specific Technology -

- Angular
- SpringBoot
- MySql
- REST APIs

Key Features -

Feature	**Description**
Responsive User Interface	Designed using Angular for an engaging user experience.
Secure Authentication	Implemented tokens for robust security.
Database Integration	MySQL for secure file storage and management.
Interactive Interface	Enables knowledge sharing and resolution of queries.
Admin Dashboard	Tools for managing content, users, and resources efficiently.

Functional Components -

Component	**Functionality**
User Authentication	Register, log in with role-based access (User, Staff, Admin), and manage sessions securely.
Miles Submission	Allow users to submit flight details to request mileage points.
Admin Approval System	Admin can approve or reject mileage requests based on submitted data.
Miles Balance Tracking	Track and display updated miles for each user upon request approval.
Admin Dashboard	View and manage all requests and user activities in a centralized interface.

Functionality -

The Miles Acquisition System offers the following functionalities:

- User registration, login, and role-based authentication (User, Staff, Admin).
- Flight details submission by users to request miles based on their flight activity.
- Admin dashboard for management tasks.
- Admin approval/rejection of mileage requests based on submission data.

Functional Requirements -

1. Hardware Requirements

1. Client System: Minimum 4GB RAM, Dual-core processor, 500GB storage.

2. Server System: Minimum 8GB RAM, Quad-core processor, 1TB storage.
3. Internet Connectivity: Stable connection for accessing cloud services.

2. Software Requirements

1. Operating System: Windows 10 or later, Ubuntu 20.04 or later.
2. Development Tools: Visual Studio Code, Postman.
3. Backend: SpringBoot
4. Frontend: ReactJS.
5. Database: SQL, No SQL
6. Cloud Services: AWS S3, AWS EC2.

3. Manpower Requirements

1. Frontend Developer: To design and develop the user interface using ReactJS.
2. Backend Developer: To implement the server-side logic using SpringBoot.
3. Database Administrator: To manage and optimize database collections.
4. Cloud Specialist: To configure and maintain AWS services.

Non-Functional Requirements-

1. Performance Requirements

- The system should handle up to 100 simultaneous users without performance degradation.
- Response time for any action should not exceed 2 seconds under normal load.
- The system should process file uploads/downloads within 5 seconds for files up to 100 MB.

2. Scalability

- The system should support scaling to accommodate increased user demand (e.g., 500 users in peak times).
- The database should be designed to handle growth in data size by 10% annually.

3. Reliability and Availability

- The system should maintain an uptime of 99.9% during operational hours.
- Backup systems should restore data within 30 minutes in case of system failure.

4. Usability

- The interface should be intuitive and require no more than 1 hour of training for a new user.
- Provide multilingual support if users are from diverse language backgrounds.

5. Security

- All user data should be encrypted using AES-256 encryption.
- Implement role-based access control (RBAC) to restrict user actions based on roles (e.g., admin, librarian, user).
- The system should automatically log out inactive users after 10 minutes of inactivity.

6. Maintainability

- Codebase should follow standard coding practices and include clear documentation for future developers.
- Software updates and patches should be deployed with zero downtime.

7. Portability

- The system should be accessible on various platforms, including desktops, tablets, and mobile devices.
- Ensure compatibility with major web browsers like Chrome, Firefox, Edge, and Safari.

8. Compliance

- Adhere to relevant standards, such as GDPR for data protection if dealing with user data in the EU.
- Follow accessibility standards like WCAG 2.1 to make the system usable for people with disabilities.

9. Efficiency

- Optimize resource utilization to ensure the system runs efficiently on hardware with minimal specifications.
- Avoid memory leaks and unnecessary background processes.

10. Support and Serviceability

- Provide 24/7 technical support for critical issues.
- Include detailed troubleshooting guides and FAQs within the system.

Expected Outcomes

1. Streamlined and transparent mileage acquisition process for airline users.
2. Enhanced engagement between users, staff, and administrators through role-based workflows.
3. Accurate and real-time tracking of earned miles with secure and structured data management.
4. Improved operational efficiency for airline loyalty programs through digital verification and approval mechanisms.

Guidelines

To ensure the success of the Miles Acquisition System, the following guidelines should be adhered to:

1. Ensure modular development to simplify debugging and scalability.
2. Follow RESTful API design principles for seamless backend communication.
3. Utilize database services like MySQL for secure and efficient file storage.
4. Adhere to user authentication standards using tokens.
5. Regularly test all features to ensure functionality and robustness.
6. Secure sensitive data using encryption and authentication techniques.
7. Optimize database queries to maintain performance efficiency.

References:

[1] Angular Official Documentation: https://angular.io/docs
[2] Spring Boot Official Guide: https://spring.io/projects/spring-boot
[3] MySQL Reference Manual: https://dev.mysql.com/doc/