F21AS - Software Development Plan for Flight Management System

F21AS Edinburgh CW25

Presented by: Ikedi JACOBS, Abdulmalik IBRAHIM, Etoro EKONG, Hafiz MAHMOOD, Wilson DAGAH.

Introduction:

Previously, flight management, encompassing booking reservations, check-in, and luggage handling, proved to be a herculean task riddled with errors, missed flights, and unaccounted luggage until the advent of a flight management system. Since its introduction in 1994, flight management systems have addressed the bottlenecks of the manual system and fostered a smooth and friendly check-in process for each customer.

The design, creation, and implementation of the flight management system follow a planned iterative development approach, allowing us room to identify and mitigate risks and provide continuous improvement in each iteration.

Functionality-wise, the software is expected to contain passenger information in passengers.csv, including names, unique booking reference codes, flight codes, etc., based on the assumption that passengers have already purchased tickets. It should also contain flight information, including destination airports and luggage details, among others, in flight.csv. Furthermore, it is expected to compute excess baggage for passengers during check-in and generate an excess fee charge based on either additional kilograms and/or additional volume, as determined by the flight carrier's discretion.

Class diagram and data structure:

The figure below is the class diagram showing the classes and their relationships.

(If image is unclear image, kindly click HERE to download a clearer copy).

The selected data structures primarily consist of integers and doubles for managing numerical data, strings for handling text data, and hashmaps for handling data input from both flight.csv and passengers.csv. We opted for a hashmap due to the presence of unique entries, such as booking references for passengers and flight codes for flights. Additionally, we considered the performance and efficiency benefits that hashmaps offer when conducting lookup and search iterations.

Fig. 1.: Class diagram

Weekly Development Plan

Week 1

UML Class Diagram

- Task 1: Identify classes and attributes.
- Task 2: Define relationships between classes.
- Task 3: Finalize UML diagram.

GitHub Repository Setup

- Task 1: Create repository on GitHub.
- Task 2: Add team members as collaborators and initialize repository.

Week 2

Data Structure Design

- Task 1: Research and select suitable data structures.
- Task 2: Implement data structures in code.

Week 3

Development of Core Classes and Functionalities

- Task 1: Define Passenger, Flight, and Baggage classes.
- Task 2: Implement PassengerRepository and FlightRepository classes.

Week 4

Implementation of Application Logic and GUI

- Task 1: Implement CheckinApp GUI Class.
- Task 2: Develop main CheckinApp class and integrate with GUI.

Week 5

Testing, Debugging, Documentation, and Deployment

- Task 1: Test components individually and perform integration testing.
- Task 2: Debug any issues encountered during testing.
- Task 3: Document code thoroughly, prepare for deployment, and deploy the application.

References:

https://www.nytimes.com/1997/08/10/travel/e-tickets-begin-to-catch-on.html