AIRLINE FLIGHT SCHEDULER

REQUIREMENTS ANALYSIS DOCUMENT

Team 18
Ryan Witkowski
Nicholas Lawrence
James Fox

INTRODUCTION

PURPOSE

The purpose of the database-centric application (DCA) for the Airline Crew Scheduler is to keep track of which employees are on--or are scheduled to be on--each flight.

SCOPE

The scope of the DCA is to keep track of all airline employees on—or scheduled to be on--each flight. The system will run additional checks and updates recorded in an electronic log, which can be searched by the client, flight crewmember, airport, and other users. The DCA software will provide an interface application that will act as a user-friendly and easy to use searching, viewing, and updating interface, instead of having to interact with the code directly.

OBJECTIVES AND SUCCESS CRITERIA

The objectives and success criteria of the DCA project is to identify if qualifications are met, develop a database, and develop an easy to use interface that will act as a user-friendly interactive shortcut to the database. The application will be able to provide consistent and precise accuracy by making sure all data is correct throughout the program application. If the application software detects a violation of a constraint that has been implemented, the application will alert the user of the violation(s) occurring.

DEFINITIONS, ACRONYMS, AND ABBREVIATIONS

DCA = Database-centric Application

GUI = Graphical User Interface

OOP = Object Oriented Programming

SQL = Structured Query Language

Overview

The overview of the project is to create a working database-centric application system that is able to operate on a large scale by storing data of various information, which can be used to monitor and provide info to the user in an easy to use/view interface. We would first start off by establishing the database and then designing an easy to use, user-friendly interface for traversing through the data.

Proposed System

Overview

The proposed system “database-centric application” will be an easy to use database application software. It will consist of a large database that will offer an easy to use and interactive interface to traverse through data through the use of searches of different variables, and the interface will then return specific data relevant to the input data. The database will be implemented in a closed source database, but will be able to be modified by the client after development release. The easy to use interface will ask for a variable input such as flight ID number, and it will then return the departure and destination airports and times, along with other possible variables if the client would like.

Functional Requirements

F1. The DCA will be accessed by users through the use of a user-friendly interface. Since users have no need to change information regarding flight details or scheduling, they will not be provided with options to edit or add new flight information. These features will be reserved for airline managers themselves, or other authorized personnel that need to edit information.

F2. In order to ensure reliability in the DCA, all data stored in the database will be backed-up so that any unforeseen failure in the database will not result in total loss of critical information. In the result of a system crash, system administrators will retrieve the backed-up data and restore the database again.

F3. The software will also be able to track updates made to the scheduling of crew members including: placement of aircrafts and employees, creation and cancellation of flights, adjustments to takeoff and touchdown times, and changes to on-flight crew or the aircraft itself. All updates to the scheduler will be tracked by update numbers.

F4. The software is also responsible for keeping track of the hours worked by employees. Scheduling is subject to constraints such as eight-hour days, rest periods between shifts, and in-flight delays that cause an extension of hours. All these constraints will be implemented into the DCA to ensure that no employees are over-scheduled.

F5. The DCA will allow for authorized users to add and modify flight details for all flights through the software. This will include adding a flight, changing a flight, cancelling a flight, adjusting flight times, and adjusting crew members.

NONFUNCTIONAL REQUIREMENTS

N1. USABILITY

Clients and users will be able to view the information stored by the DCA through the use of a simple and user-friendly interface that accesses the database for them. Users will be given instructions on the web page that detail how to successfully and efficiently navigate the interface. Information will be displayed in "spreadsheets" that provide readability and simplicity.

N2. RELIABILITY

The data stored by the DCA in the database will implement basic-level security as to prevent breaches such as SQL interjection attacks or access of information by unauthorized personnel. Any errors that may result from incorrect user input or unavailability of information will be handled by the DCA as to protect the performance of the system so that it will not crash. These attributes of the DCA ensure the protection of all collected information and increases system reliability.

N3. PERFORMANCE

The DCA will be designed and implemented in such a way as to maximize usage and performance efficiency. Updates to information stored by the DCA will be fast and precise and will be available almost immediately to users.

N4. IMPLEMENTATION

The DCA can be accessed via the internet if the user knows the link needed to access the website. Online access increases usability of the software by ensuring that information is not solely limited to local access.

N5. INTERFACE

As stated previously, users will be able to access data through the use of a user-friendly interface. This Graphical User Interface (GUI) will be simple and easily navigated so that the interface communicates with the database, and there is no need for the user to interact with the code or database directly. Solicited information will then be displayed in an easy to read format to maximize simplicity for the user.

SYSTEM MODELS

SCENARIOS

A flight crewmember is up to their 8-hour flight restriction when they arrive at their destination airport. The database then sends an alert to the system and the system will alert the standby crew members. The standby

crewmember is then scheduled to replace the previous crewmember on sight because the scheduler has scheduled the positions appropriately.

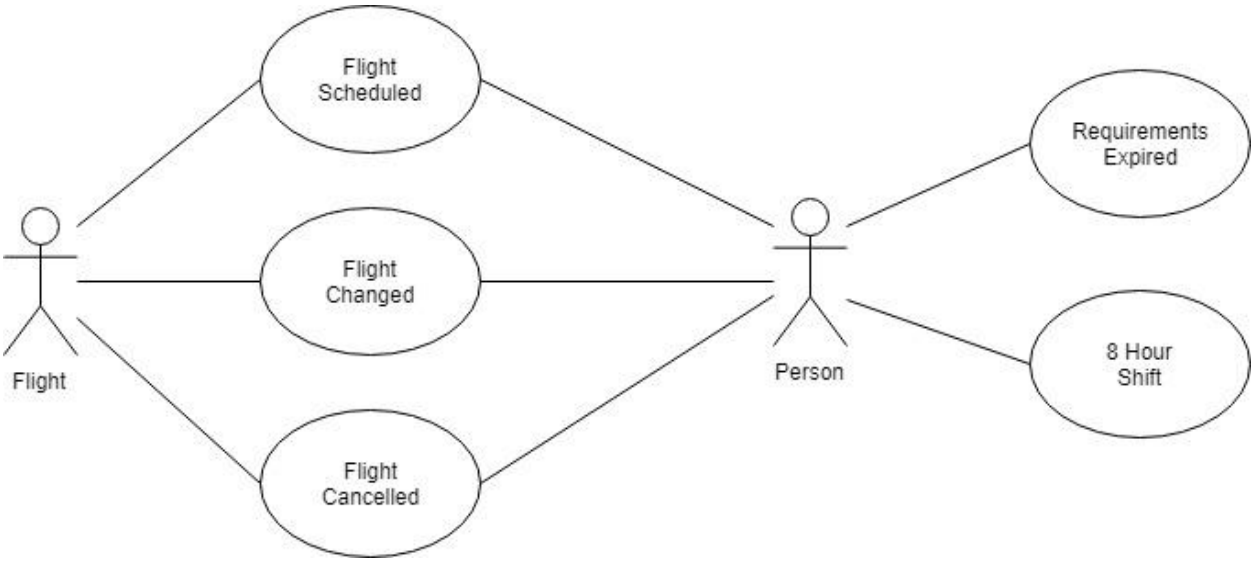
A flight is departing on November 9, 2018 from Lincoln at 2:30 pm and will arrive at Iowa City 3:30pm. Sixty-Five passengers will be on the flight, along with two flight attendants, a pilot, and a first officer. All personnel will be put onto the NU-150 plane that will depart from Lincoln, and a flight ID will automatically be generated. The system will find available crew members for the flight, and schedule them automatically and record all flight data in the database.

A flight has already been scheduled from West Lafayette to Evanston on November 10, 2018 and was originally scheduled for 1:00pm. There is a rain delay in West Lafayette causing the flight to be delayed for 2 hours. The flight crew remains on standby while the airline waits for the storm to pass. After 2 hours, the storm clears, and the delayed plane is able to depart at 3:00pm. The updated information is then stored in the database.

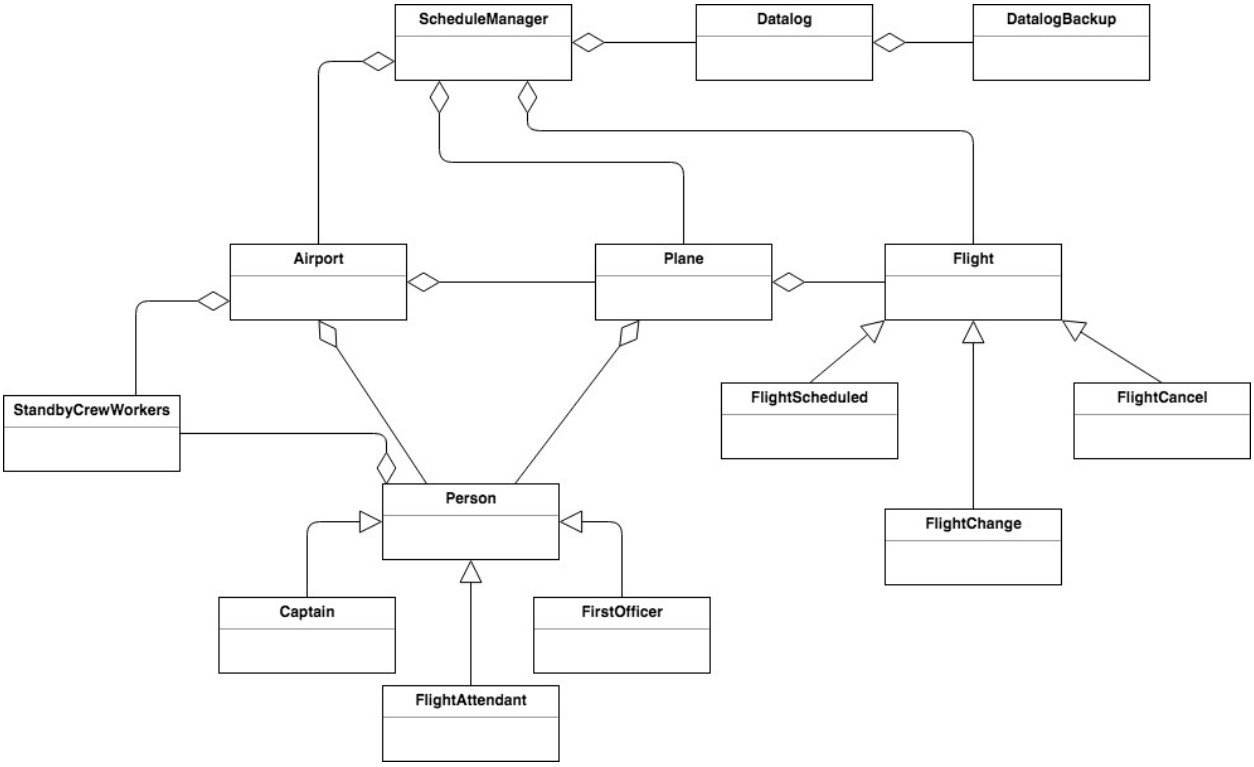
A late-night flight on December 4, 2018 departing at 9:30pm from Lincoln and arriving at 10:30pm in Iowa City has been cancelled due to an ice storm that is supposed to last all night. The status of the flight is changed to cancelled, and the flight crew is moved to standby. Meanwhile, the 40 passengers are re-assigned to a different flight. The updated information is then stored in the database.

A full flight crew consisting of a pilot, first officer, and two flight attendants was already scheduled for a flight two months ago. Since being scheduled, the pilot's qualifications have expired, and he is unable to fly the plane. The scheduler then checks to see if the first officer is a qualified pilot. The first officer is, and the scheduler replaces the pilot with the first officer, and the pilot moves to standby. The scheduler then finds a replacement standby first officer and schedules them as well. The updated information is then stored into the database.

Use Case Model



Object Model



Use Cases

Best case Scenario

The Best-case scenario

A flight crew member is up to their 8-hour flight time restriction.

They arrive at their destination airport.

The database sends an alert to the system.

The system alerts the standby crew member.

The standby crew member to replace them are on sight because they have been scheduled appropriately.

Worst case Scenario

The worst-case scenario

The user searches the database for the destination for their flight

The database retrieves a random destination because no id structure is in place.

The database sends the random destination data to the user's search results.

Flight Change/Flight Cancel

Best Case Scenario

Plane sends an alert to the scheduler

Scheduler sends an update to the database

Database records the scheduling change

Worst Case Scenario

Plane alerts crew

Crew fails to send change to database to update

No one then knows about the change

Requirements Expired

Best Case Scenario

Admin assigns person

Person data sends data to database

Database returns data

Data shows qualifications fulfilled

Worst Case Scenario

Admin assigns person

Pilot data sent to database

Database doesn’t receive qualifications

Plane goes down in the ocean

Flight scheduled

Best Case Scenario

Admin assigns flight

Flight data entered to scheduler

Scheduler updates database

Worst Case Scenario

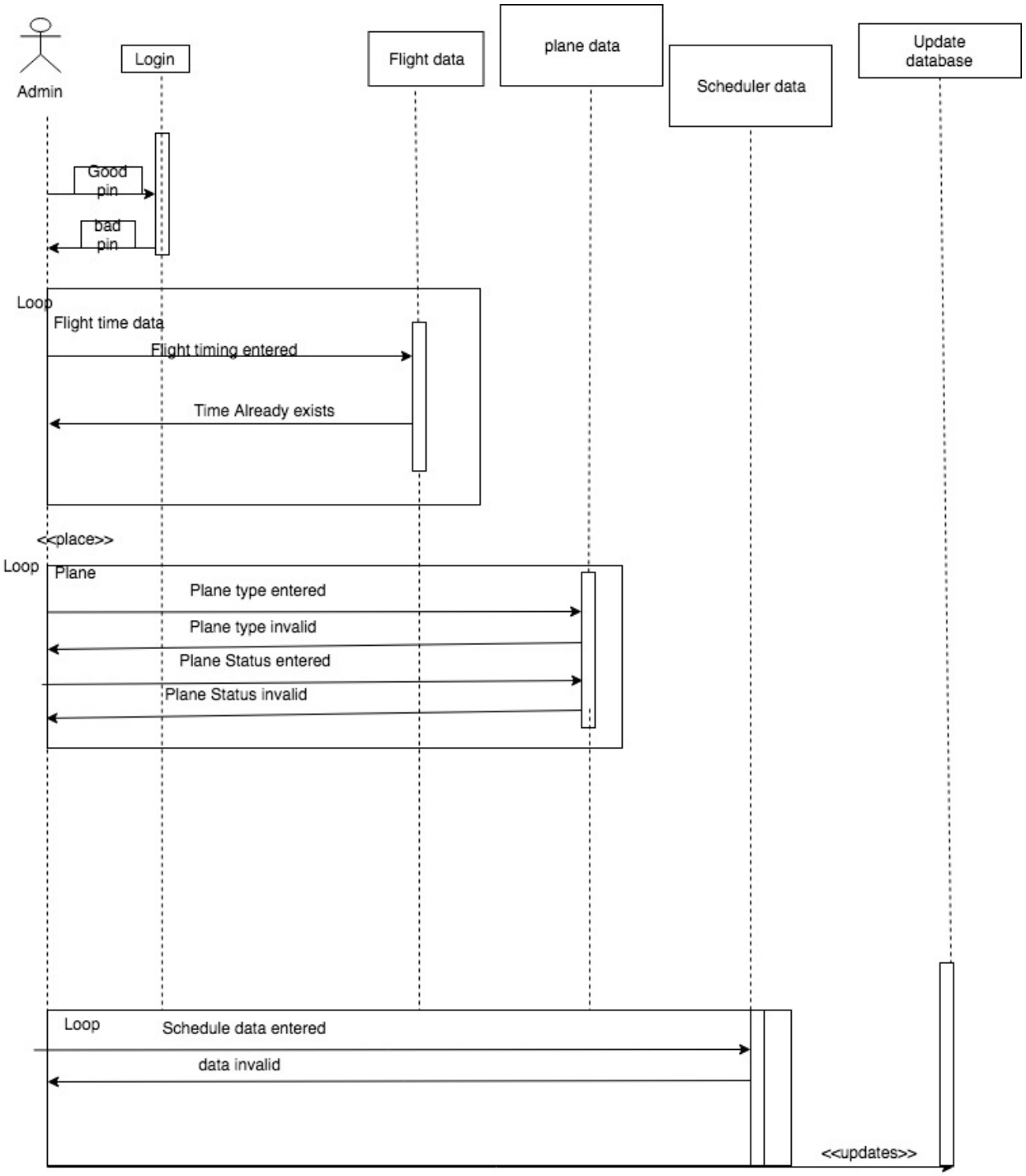
Admin assigns flight

Flight data gets sent to plane data

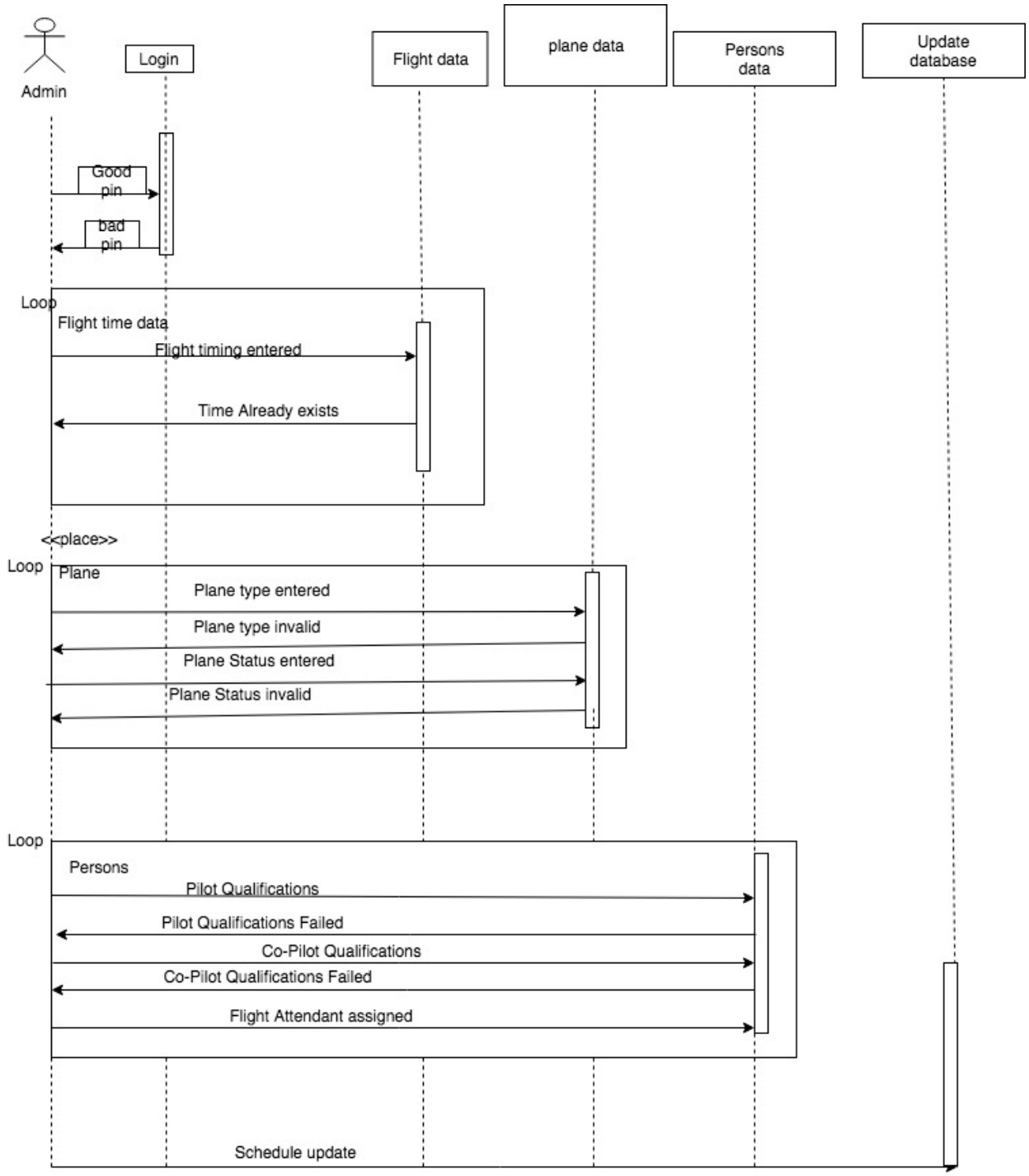
Plane data sends back flight data

Flight data never gets sent to database

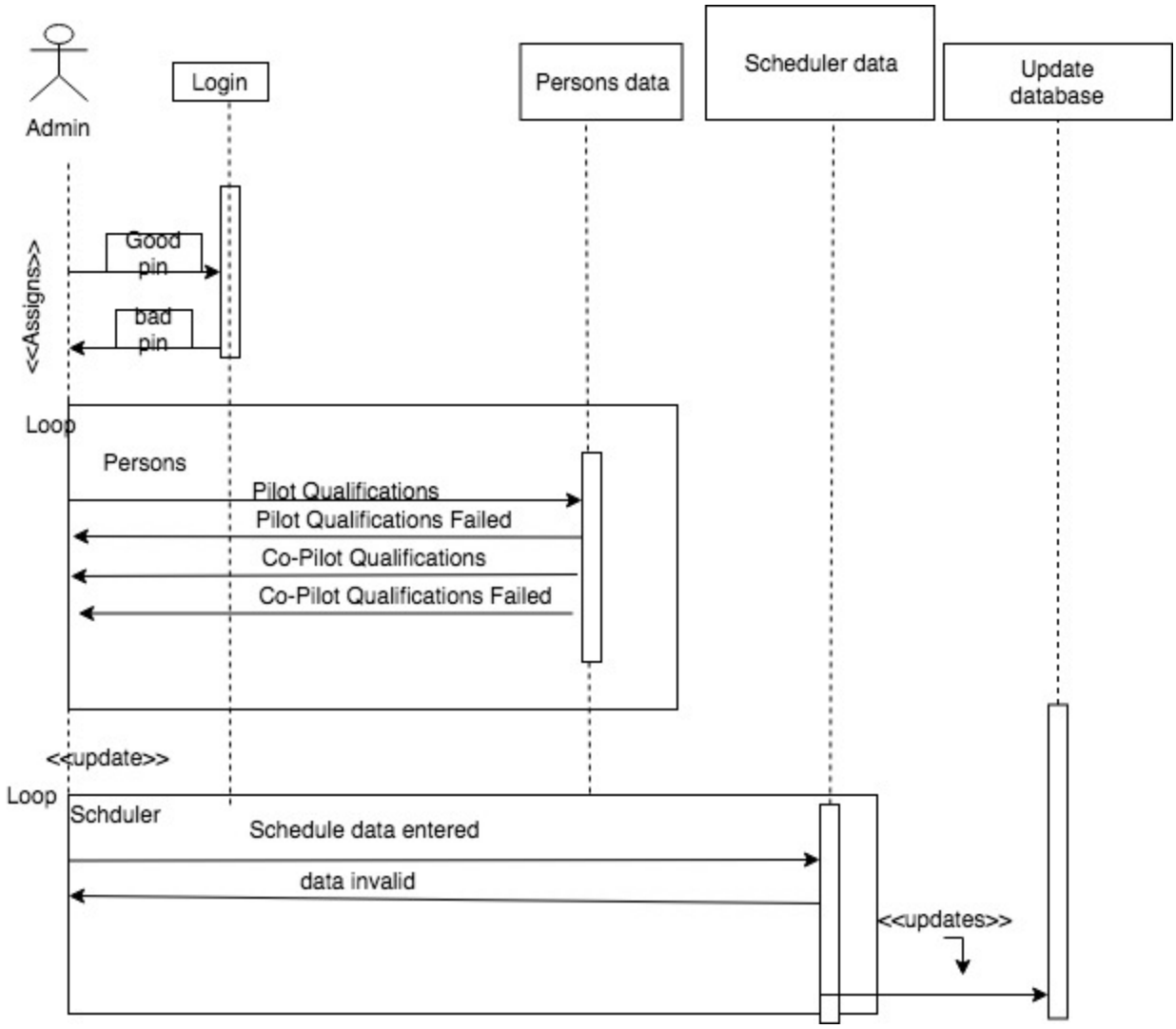
SEQUENCE DIAGRAMS



Flight Changed and Flight Cancelled Diagram



Flight Scheduled Diagram



Requirements Expired Diagram

USER INTERFACE: NAVIGATIONAL PATHS AND SCREEN MOCKUPS