Software Design Specification

for

Luggage Tracking Application

Version 2.0

Prepared by

Name	Student ID	E-mail
Tam Minh Chau Bui	**1640110**	**tabui@uw.edu**
Vadim M Goncharuk	**1668360**	**vadimg@uw.edu**
Binh Hue Hua	**1650510**	**binhhua@uw.edu**
Leah M Ruisenor	**1569598**	**lmr81@uw.edu**
Norris Spencer	**1573221**	**nisj@uw.edu**
Group Name	**Lüg-er**	

Prepared for

Course	TCSS 360 Software Development and QA
Instructor	Universal RFID Inc. & Dr. Eyhab Al-Masri
Date	**11/20/2017**

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
Version 1.0	Tam Bui Vadim Goncharuk Binh Hue Hua Leah Ruisenor Norris Spencer	Initial Document Draft	11/20/17

1 Introduction

Universal RFID Inc. has tasked our group, Lüg-er, to create a passenger application that will allow the passenger to effortlessly track the physical location of their baggage. This application will take advantage of the existing tracking system emplaced in airports and aircrafts that utilizes RFID technology.
The goal is to provide the passenger with a peace of mind that airline passengers have historically not been able to enjoy. Lüg-er intends to make the interface as user-friendly as possible and, in the process, alleviate a major concern for airline passengers, lost baggage.

1.1 Document Purpose

This document is intended to address the plans for the design of the software components for the Lüg-er baggage tracking application, the responsibilities for each component and the interfacing/relationships between the components. This document is intended to work in tandem with the SRS for this application and must not be considered an individual component in the development of this application. Ensure that the revision dates match those with the SRS.

1.2 Intended Audience and Document Overview

This document is intended for Universal RFID Inc. Interested investment stakeholders to include software developers, hardware engineers, and other interested parties. Additionally, in the realm of the study of this subject, Dr. Eyhab Al-Masri and personnel deemed by him to be interested are targeted audience members as well.

What follows is a synopsis of the information gathered to assist the stakeholders and interested parties in designing the application and its components as efficiently as possible. The table of contents is the point of reference for the organization of this document and should be referred to for information needed by interested parties or individuals.

1.3 Definitions, Acronyms and Abbreviations

The following are terms and abbreviations contained in this document:

- API: Application Programing Interface
- IEEE: Institute of Electrical and Electronic Engineers
- ISP: Internet Service Provider
- Lüg-er: The name of the group tasked by Universal RFID Inc. to design the application
- PII: Personally Identifiable Information
- RFID: Radio Frequency Identification
- SMD: State Machine Diagram
- SRS: Software Requirement Specification

1.4 Document Conventions

- This document follows IEEE formatting requirements.

- Any reference to the application that is being directly developed by Lüg-er under the direction of Universal RFID Inc. is to be considered the Lüg-er application.

1.5 Assumptions and Dependencies

- The assumptions that can affect the requirements stated in document involve the use and access of third-party databases and APIs.
- This application will need to have access to its own database in conjunction with access to the airline/airport database and Google Maps database.
- It is assumed that all airports will have RFID hand scanners, RFID antennas and a database that stores the given position of a bag at any given time.
- It is assumed that airplanes will also have RFID antennas installed. The airplane antennas will allow for tracking of baggage away from the airport's fixed antennas.
- It is assumed that all passenger baggage is manufactured with RFID tags.
- It is assumed that the baggage handler will have an RFID reader that can scan the RFID tag of the given bag when the passenger is checking in. The RFID number will be connected to the passenger's flight information which will then be part of the airport/airline database.
- To be able to display this location of the RFID tag (passenger baggage), the application must have access to and take advantage of the Google Maps API. This information, combined with the airport database, will display the passenger's current location and the location of the RFID tag (passenger baggage).
- It is assumed that the Airport database will store and process lost baggage requests.

1.6 References and Acknowledgments

- The Visual paradigm 14.2 program was used to create the State Machine Diagrams http://www.visual-paradigm.com/ Last accessed 17 Nov 2017
- Google API terms of service, https://developers.google.com/maps/terms for details Date Last Accessed 30 Oct 2017
- MySQL Workbench Community (GPL) for Mac OS X version 6.3.9 CE build 10690321 (64 bit) program was used to create the database design

2 Class Diagrams

2.1 Class Diagram Description/Figure

The class diagram in figure 2.1.1 shows the major classes of the Lüg-er application. The Login Screen class will allow the user to log in their account, reset their password, and create a new user account. For the user to log in to their account, the user must enter their login information. The class will then call the isPass(String, String) method to validate if the information entered is correct. To reset the password, the user must select the “forgot password” which the forgetPass(String) method will verify the user email and send a reset password link to the entered email. Options() method allows the user to set language, and get help. The User Interface class allows the user to enter their flight number and search the Airline Database to find the associated list of RFID tags. When the application has the list of RFID tags, the class will be able to display the map with the user’s location and their baggage(s) location. The Map class must keep track of the user location, their baggage(s) location, and the list of RFID tags. In addition, the Map class should allow the user to choose which baggage they want to display on the map and the chooseBag(String) method will retrieve the exact location of the baggage. The User class must keep track of the user’s information (name, addresses, phone number, email, password, userID, and selected language). The Options class should allow the user to update their account information, change the application language, report lost baggage(s), and get help.

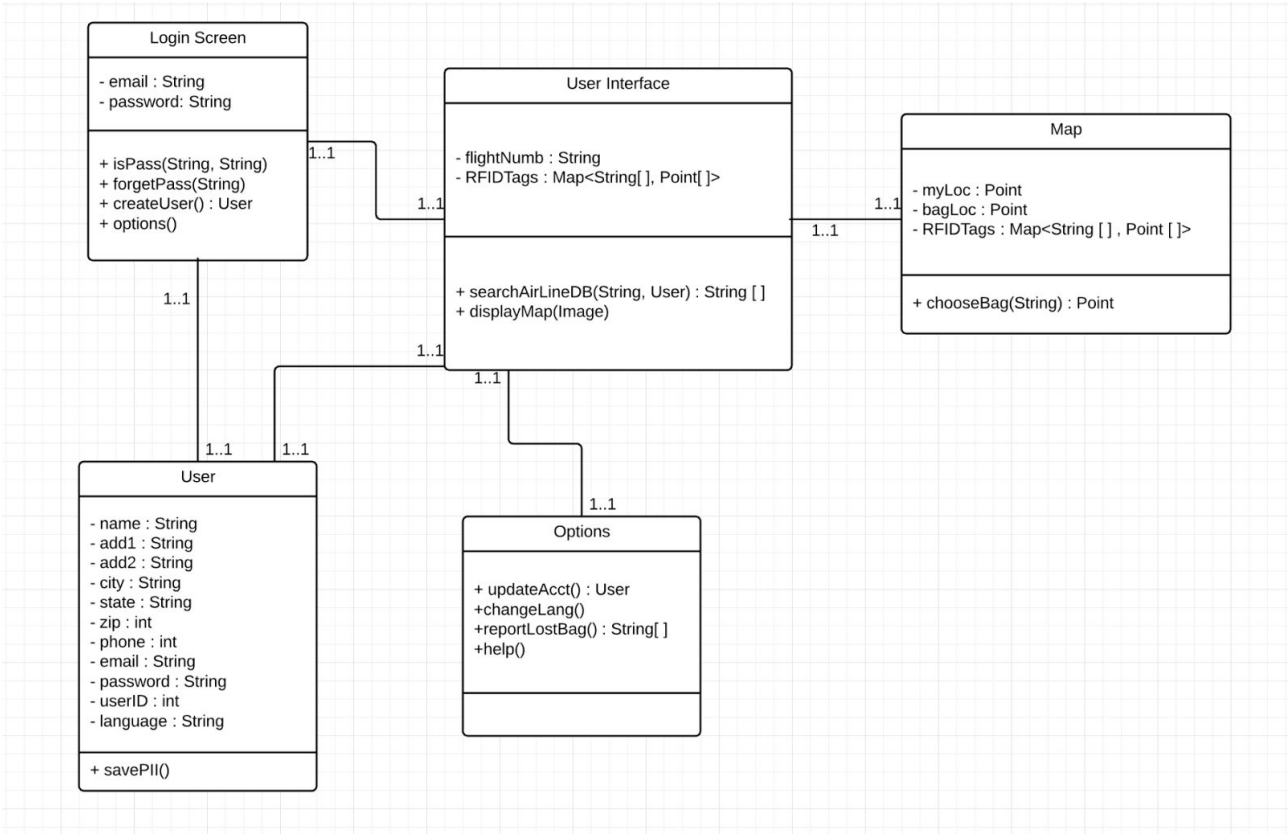

figure 2.1.1 Class Diagram

3 Sequence Diagrams

3.1 Sequence Diagram Description/Figure

The sequence diagram in figure 3.1.1 shows the process for how a user creates a new account for Lüg-er. First, the user enters an email for their account. The Lüg-er application will check the Lüg-er database if an account already exists with the given email. If an account does not exist then a confirmation email will be sent to the user to reply to. Otherwise, Lüg-er will notify the user that the email already has an account. After the user replies to the email, Lüg-er will notify them that they have created an account.

The sequence diagram in figure 3.1.2 shows the process for how baggage is tracked and displayed to the user. The user must first enter their flight number in order for Lüg-er to track their baggage. The Lüg-er database will contact the associated Airline's database to retrieve a list of RFIDs associated with that user along with the RFIDs locations. The list of RFIDs will be stored on the user's device. Then the device will contact the Google database for the locations associated with the RFIDs and the users every 5 seconds. The baggage locations and user location will then be displayed on the Lüg-er application for the user to few.

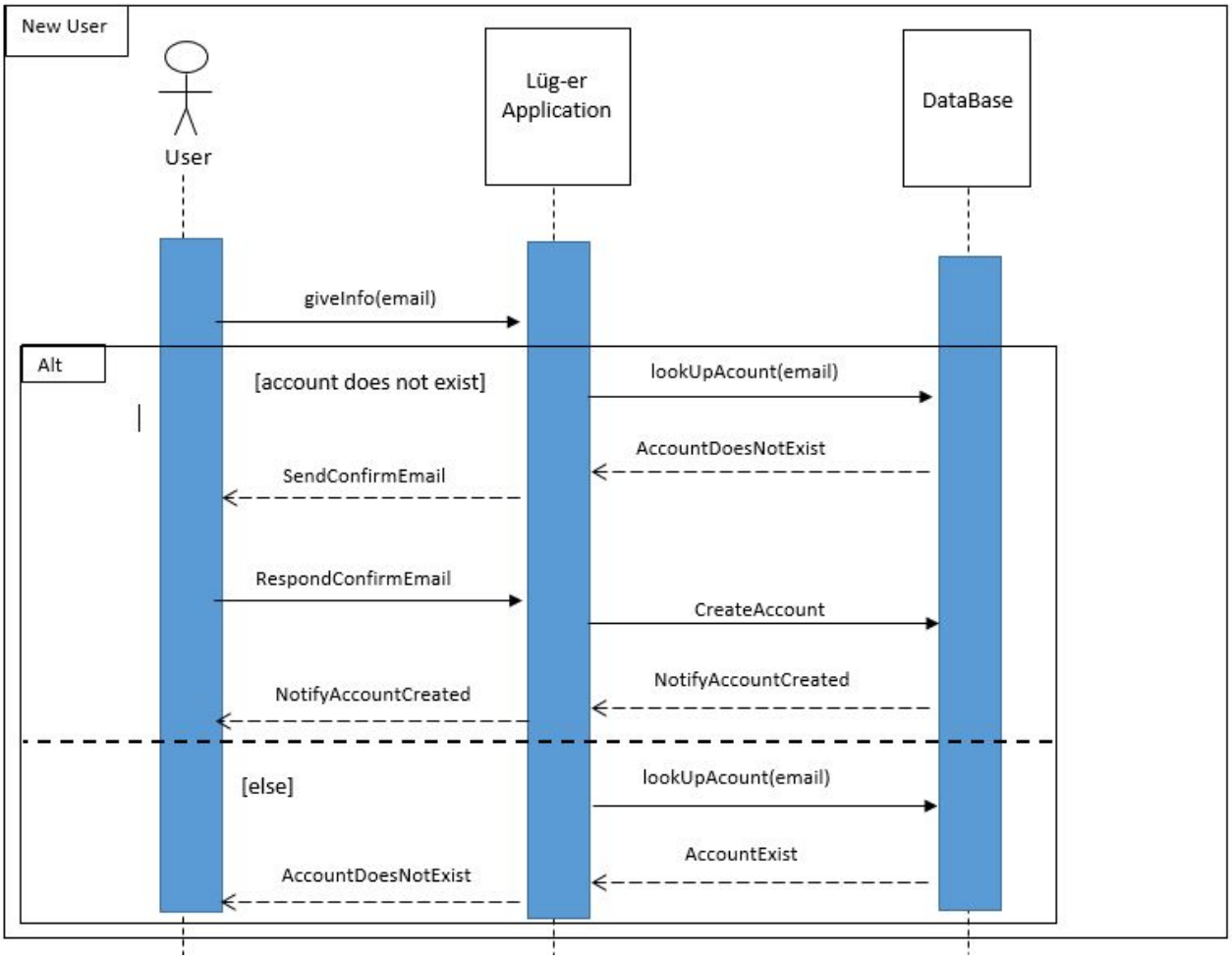

figure 3.1.1 New User Sequence Diagram

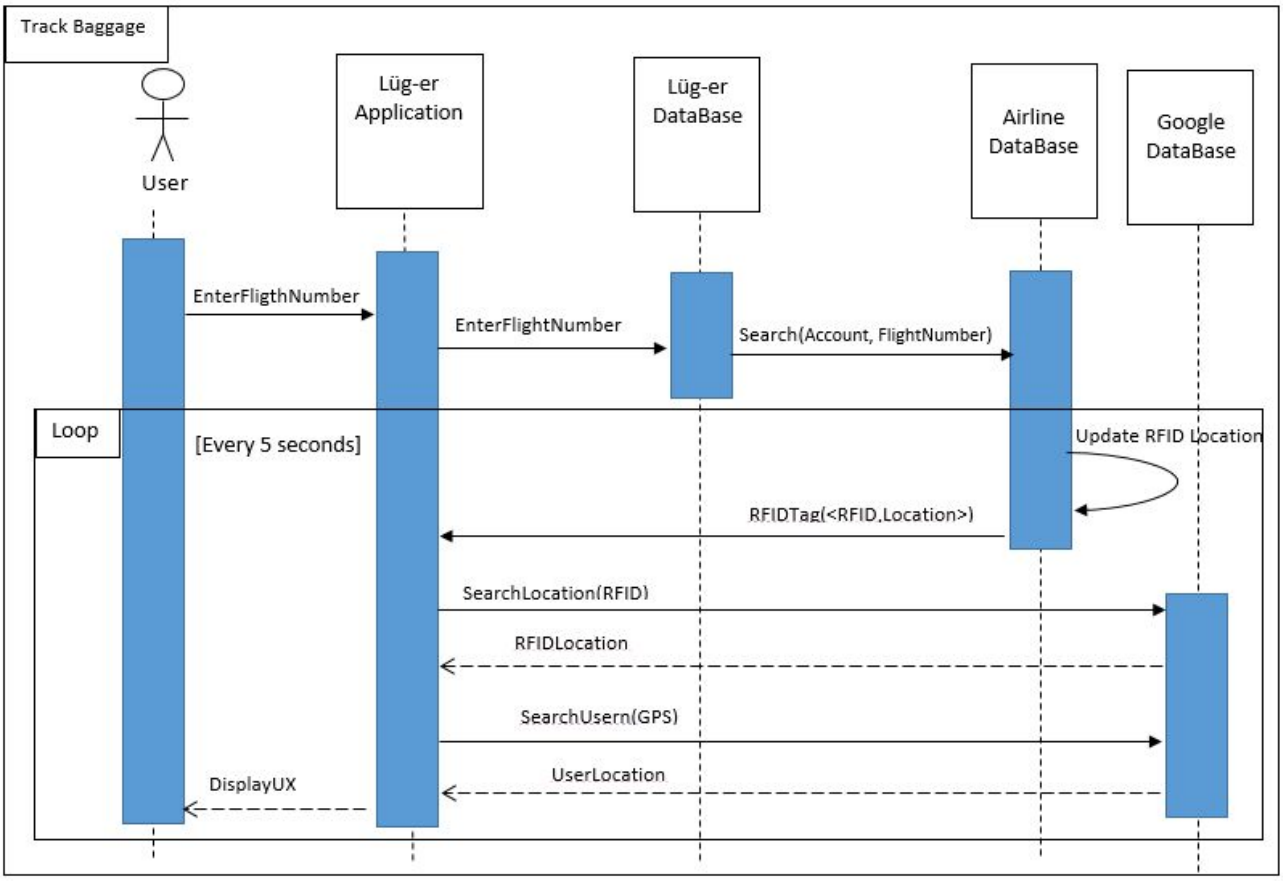

figure 3.1.2 Track Baggage Sequence Diagram

4 State Machine Diagram (SMD)

4.1 SMD Description/Figure

The SMD in *figure 4.1.*1 shows the given states and guard conditions where applicable to transition from one state to the next:
After the application is turned on and the Initialization state is complete, the application will automatically move to the Default Login state. From this state, the user may either login or create a new account.

If the user elects to create a new account, the Create Account action will commence with the implementation of the New User state followed by the Process Account state after the user has entered in all their PII. The subsequent state, Activate Account, will be processed only after the user has verified their email address. Upon completion of this state, the application will return to the Default Idle state.

Alternatively, provided the user already has an active account, the user can login which, upon verification of the email and password, will transition the state to Log-in Idle. The state will change to the Enter Flight Number when the user does so and will subsequently enter the track baggage action. Once the flight number has been validated against the airport database and the user saved account information, the application will begin two concurrent states. The first is to display the location of the user which is updated every five seconds. The second is to display the baggage location, also updated every five seconds. From this reoccurring state, the user can log out or return to the main display screen which will transition the states to the Default Idle or Log-in Idle states respectively.

The user can either log out manually or by simply closing the application. If the user closes the application without logging out, the application will automatically log him/her out before the application shuts down completely.

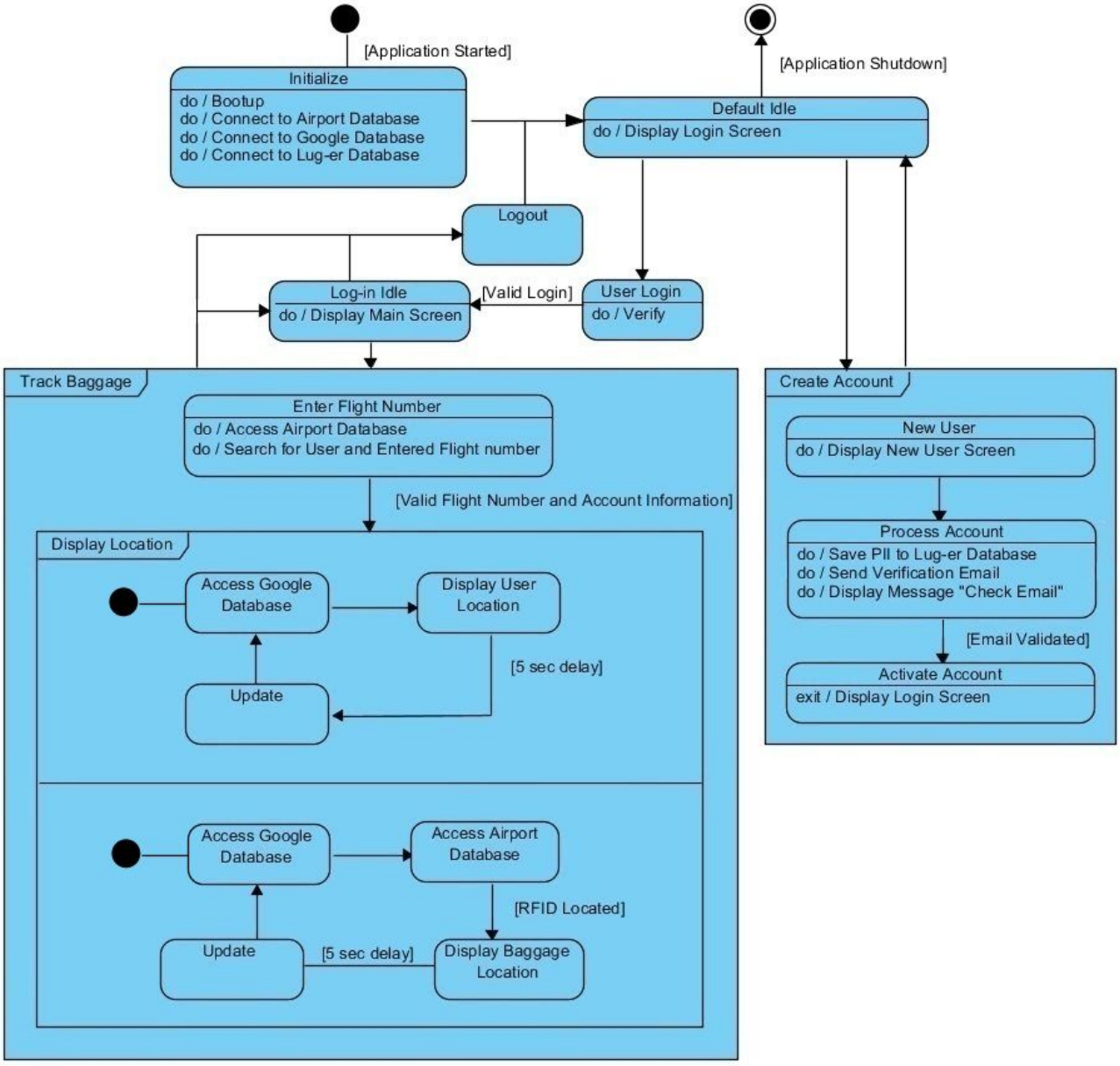

figure 4.1.1 State Machine Diagram

5 Database Model

5.1 Database Model Description

The data used for our system will be stored in a database. Our Lüg-er database will store the user id which is the key. We will also store the user's email, password and their address. We will also need to pull information from the Google maps database, which we assume hold the latitude and longitude. We will be pulling information from the airline database, which we assume will hold the following information shown in the database schema below.

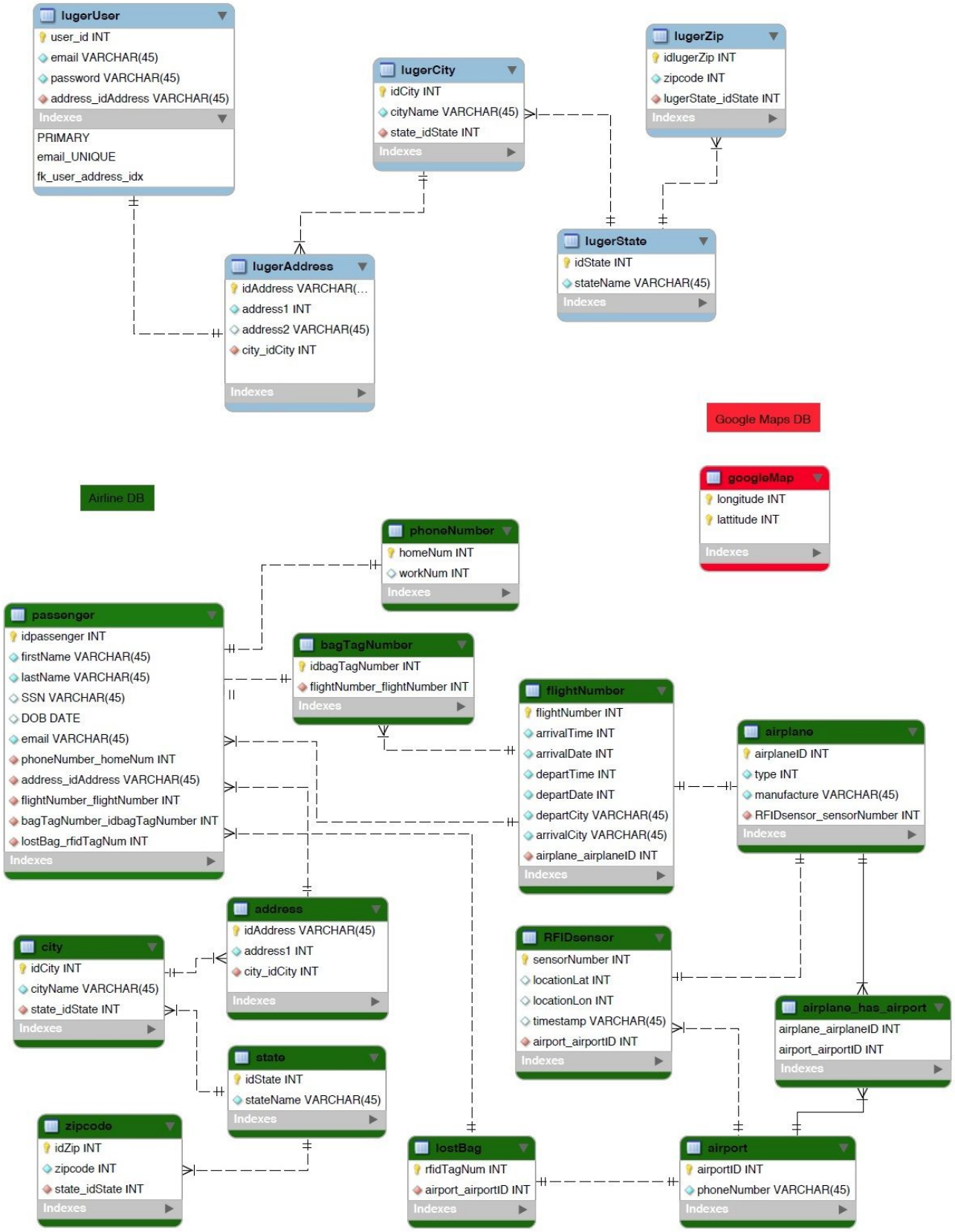

figure 5.1.1 Database Model Diagram

Appendix A – Glossary

- Figure 2.1.1 Class Diagram: Displays the major classes of the Lüg-er application.
- Figure 3.1.1 New User Sequence Diagram: Displays the sequences involved in executing the creation of a new account.
- Figure 3.1.2 Track Baggage Sequence Diagram: Displays the sequences involved in executing the track baggage process.
- Figure 4.1.1 State Machine Diagram: a Simple diagram that displays the given state of the application from the moment it is turned on to the eventual closing of the application.
- Figure 5.1.1 Database Model Diagram: Representation of the assumed database model for Lüg-er database, Google map database, and the Airline database.
- (API) Application Programming Interface: A set of subroutine definitions, protocols, and tools for building application software.
- (IEEE) Institute of Electrical and Electronic Engineers: A professional association whose objectives are the educational and technical advancement of electrical and electronic engineering, telecommunications, computer engineering and allied disciplines.
- (PII) Personally Identifiable Information: this includes username, addresses, phone number, email, password, and the number of baggage(s).
- (RFID) Radio Frequency Identification Device: it is a small electronic device that consists of a small chip and an antenna.