Client Name: DVT
Project Name: KinderFinder

Functional Requirements and Application Design

Team Name: MAU Technologies

Uteshlen Nadesan 28163304
Michael Johnston 12053300
Po-Han Chiu 11063612

https://github.com/MrBean355/KinderFinder.git

Version: 0.2

22 August 2014

Contents

1 Functional requirements and application design

1.1 Introduction

This section discusses the functional requirements for the KinderFinder System, this introduces the core domain concepts and relationships between these concepts.

The requirement is to create a system, that will through the use of RFID technology in wristbands and matching readers / receivers, be able to track children throughout a restaurant or designated area. This information needs to be fed to an API on a server that will push information to the relevant devices (mobile) to inform parents of the children's whereabouts within the designated area.

1.2 Required functionality

1.2.1 Web Administration

The Web administration must:

- Be used to allow installers of the system to set-up restaurants and their layouts (using maps).
- Enable restaurants to link wristbands to a Patron's mobile application, or assigned electronic tracking device (device not in scope).
- Enable restaurants to clear and unlink wristbands from devices and patrons' mobile apps.
- Allow access to basic reporting will also be done through the Web Administration.

1.2.2 Mobile Application

The mobile implementation must:

- Allow for users to view a map (divided into zones) of the restaurant.
- Overlay all the wristbands registered on the app user's name on the map to show where they are (some patrons might have multiple children to track).
- Allow for setting up of alarms and notifications based on movement of tracked wristbands.
- Allow for a user to select a restaurant, and a restaurant branch in order to get the map for the set-up.
- Enable the user to link the wristband to their mobile application

1.2.3 Basic Reporting

- Facility usage (zone based) by children. This will enable a restaurant to see statistics on which areas are most popular for children.
- Usage statistic of mobile app users that could possibly enable restaurant chains to implement a loyalty program.
- Health reports, reports that will indicate the health of the hardware being used. This should pick up trends that are impossible with the restaurant layout.

1.2.4 Embedded Hardware and prototyping

Physical hardware devices used in this project includes:

- Receiver/Reader prototype
  - These are devices that are placed at predetermined locations within the designated area where tracking will occur in order to pick up wristbands and their relative signal strengths. These devices will use wireless communication will to communicate this information to an access point with internet access
- Wristband prototype
  - These can be active or passive RFID tags, or wireless transponders/transceivers. They will be used to communicate its existence in range of a receiver/reader and its signal strength (RS SI)
- Electronic tracking device
  - This device, when developed, will be a piece of hardware the patron can put on their table with LED's representing zones on the restaurant map. However, for this project it is not of concern, however, it should be kept in mind during implementation for future implementation and programming.

1.3 Use case prioritization

Here we are considering a simple 3 level use case prioritization, with critical functionality (a use case or function that is absolutely essential), important functionality (if a system would still be basically functioning but still less than the client specification) and, nice-to-haves (added functionality that the client would not consider core but would add to the presentation of the project).

1.3.1 Critical

- Database - A database containing all relevant information about users, restaurants/stores and the RFID bands and readers.
- Web Administration - This requirement is essential. It allows the proper operation of the program, as web administration allows installation of this system.
- Mobile Administration - This is the part of the system the end-user will be able to use. This is one of two main interfaces with the end-user.
- WEB API - The Web API is the core of the system, it provides functionality to the client browsers and the mobile applications.

1.3.2 Important

- Basic Reporting - The reporting portion of the system is useful for logging and statistics. Even though the data gathered will be useful, the core system will still function. (Not to say this will not be done.)

1.3.3 Nice-To-Have

- Embedded Hardware and Prototyping - This is the RFID wrist bands and RFID readers. Mock data will be used for testing and implementation, after which, our team will focus on the hardware prototyping and testing.

1.4 Services Contracts

1.4.1 App User

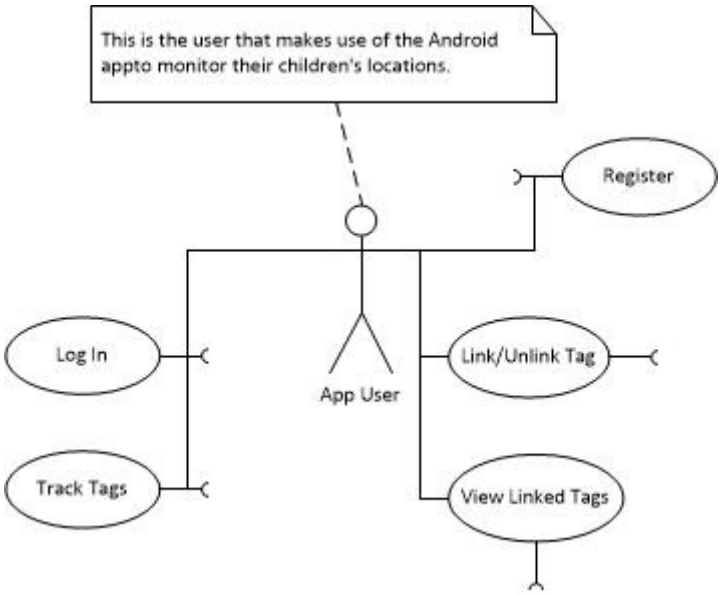

Figure 1: Use case of Android app user (first level granularity).

- Pre-Conditions - **Register**: The email address must not already be in use. **Log In**: Valid details must be provided. **View Tags**: The user must be logged in. **Track Tags**: The user must be logged in and have linked tags. **Link/Unlink Tags**: The user must be logged in. There must be available tags when linking. The tag must be linked when unlinking.
- Post-Conditions - **Register**: The account must be added to the database and the user must use it to log in. **Log In**: The user has access to all the app's functionality. **View Tags**: A list of linked tags must be displayed. **Track Tags**: The locations of linked tags must be overlayed onto the restaurant's map. **Link/Unlink Tags**: The tag must appear in the linked tags (after linking) and can be tracked. The tag must be removed from the list (after unlinking) and can't be used for tracking.

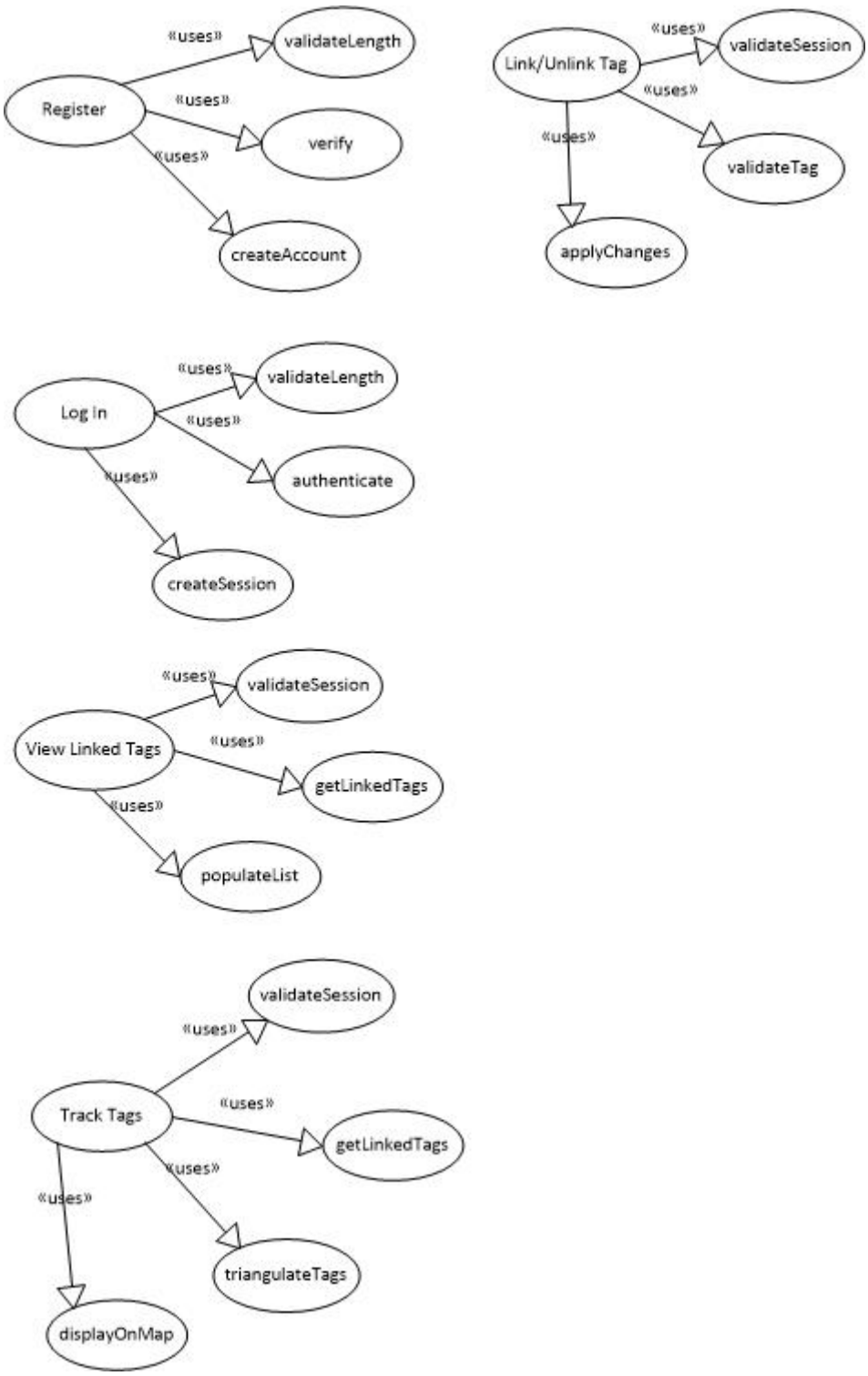

Figure 2: Use case of Android app user (second level granularity).

1.4.2 System Administrator

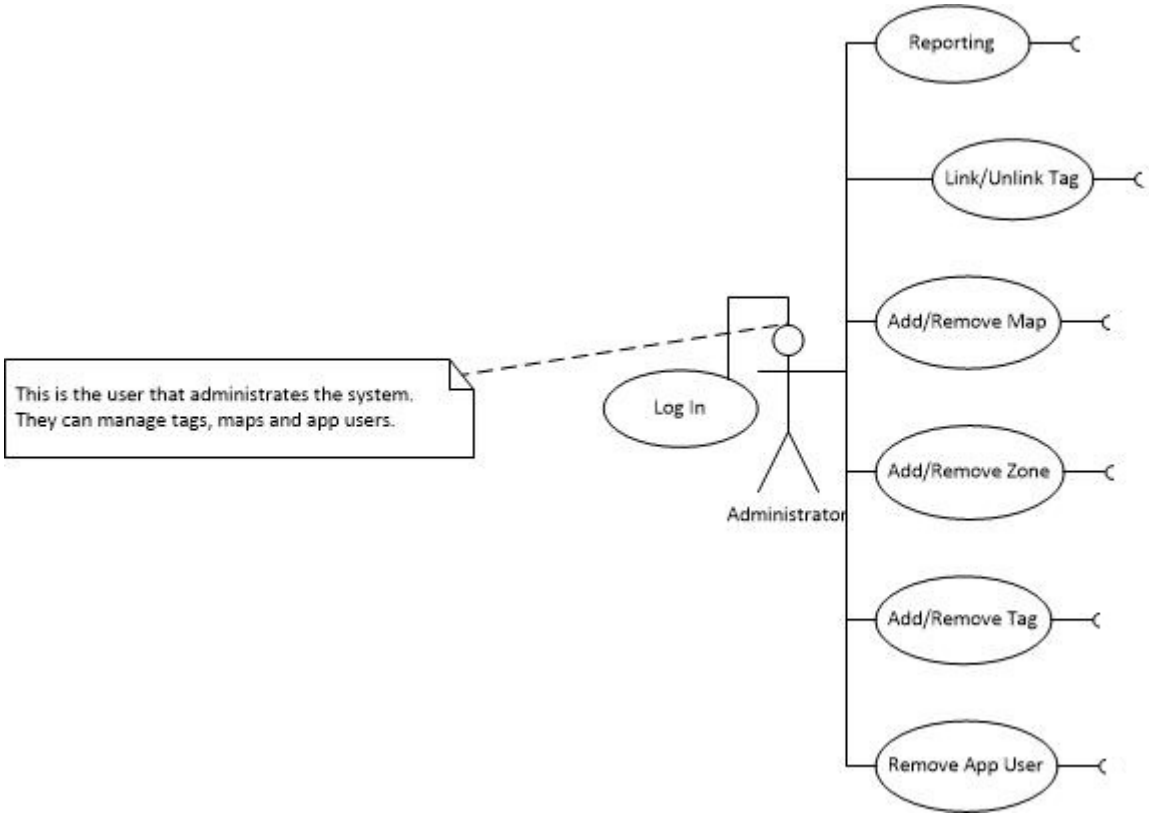

Figure 3: Use case of Android app user (first level granularity).

- Pre-Conditions - **Log In**: Valid details must be provided. **Add/Remove Map**: The user must be logged in. A map must exist when attempting to remove the map. **Add/Remove Zone**: The user must be logged in. A zone must exist when trying to remove one. **Add/Remove Tag**: The user must be logged in. The tag name must be given a name (when adding) and must exist (when removing). **Link/Unlink Tag**: The user must be logged in and a valid tag selected. The target app user must be valid. When linking, the tag cannot be already linked. **Remove App User**: The user must be logged in and the app user must be valid. **Reporting**: The user must be logged in.
- Post-Conditions - **Log In**: The user must gain access to all admin features. **Add/Remove Map**: The new map must be pushed to Android devices. **Add/Remove Zone**: The tracking zone must become available when adding, and unavailable when removing.. **Add/Remove Tag**: The tag must be available for linking when adding, and must be unlinked when removing. **Link/Unlink Tag**: The app user must be able to track the tag when linking, and no longer able to track it when unlinking. **Remove App User**: The app user must be logged out (if logged in) and no longer log in. **Reporting**: The desired report must be generated and displayed.

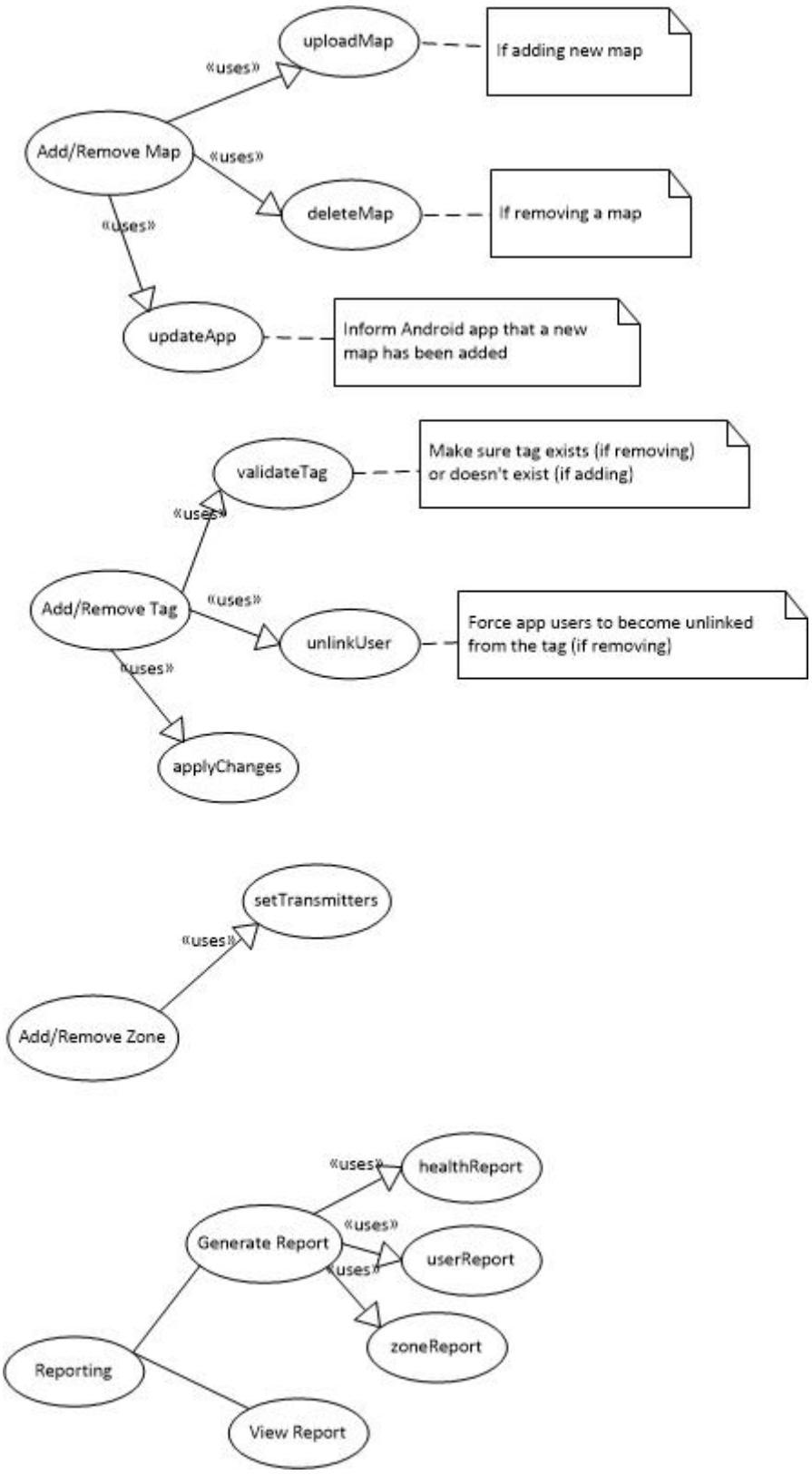

Figure 4: Use case of Android app user (second level granularity).

1.5 Activity Diagrams

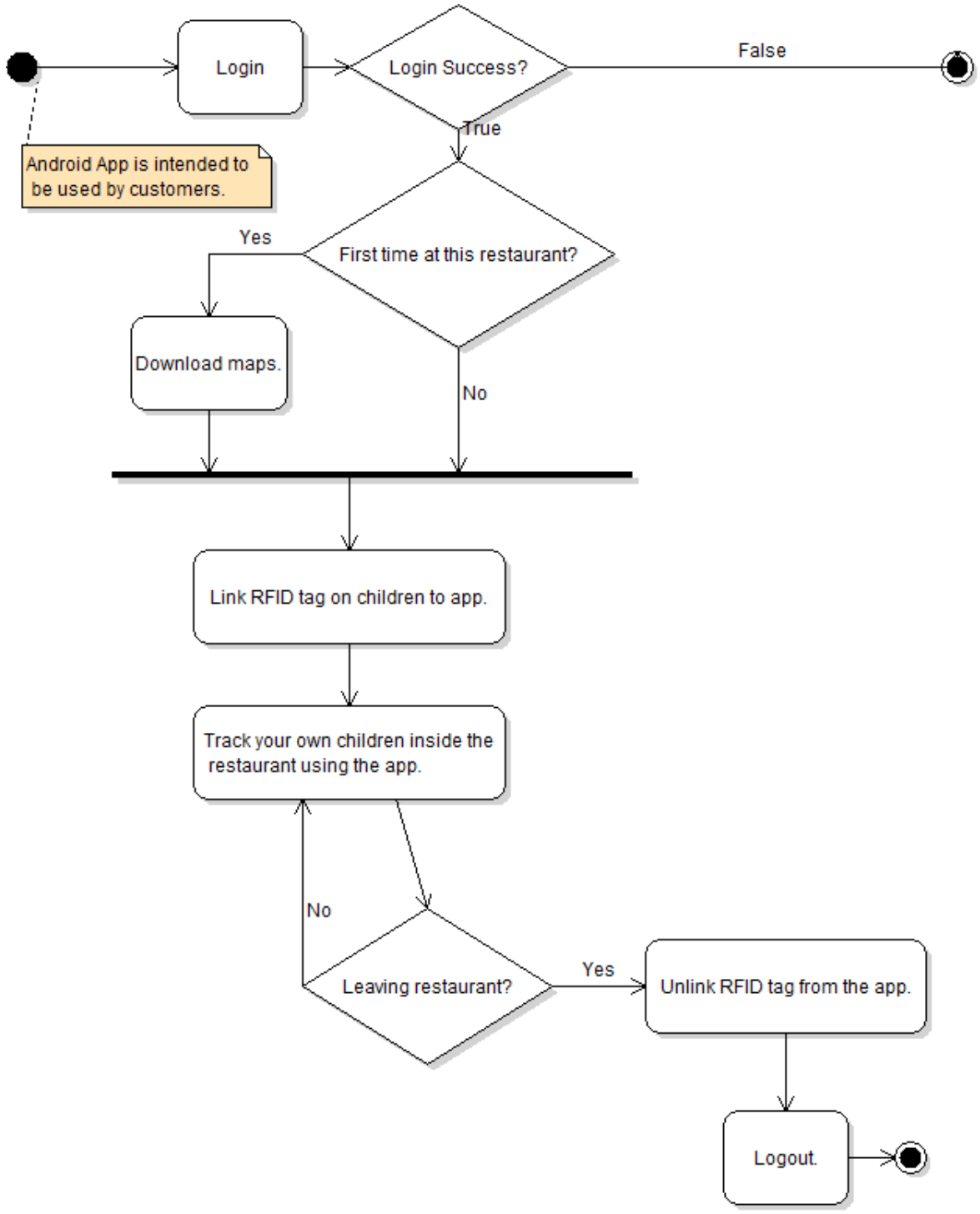

Figure 5: Activity diagram for the Android app.

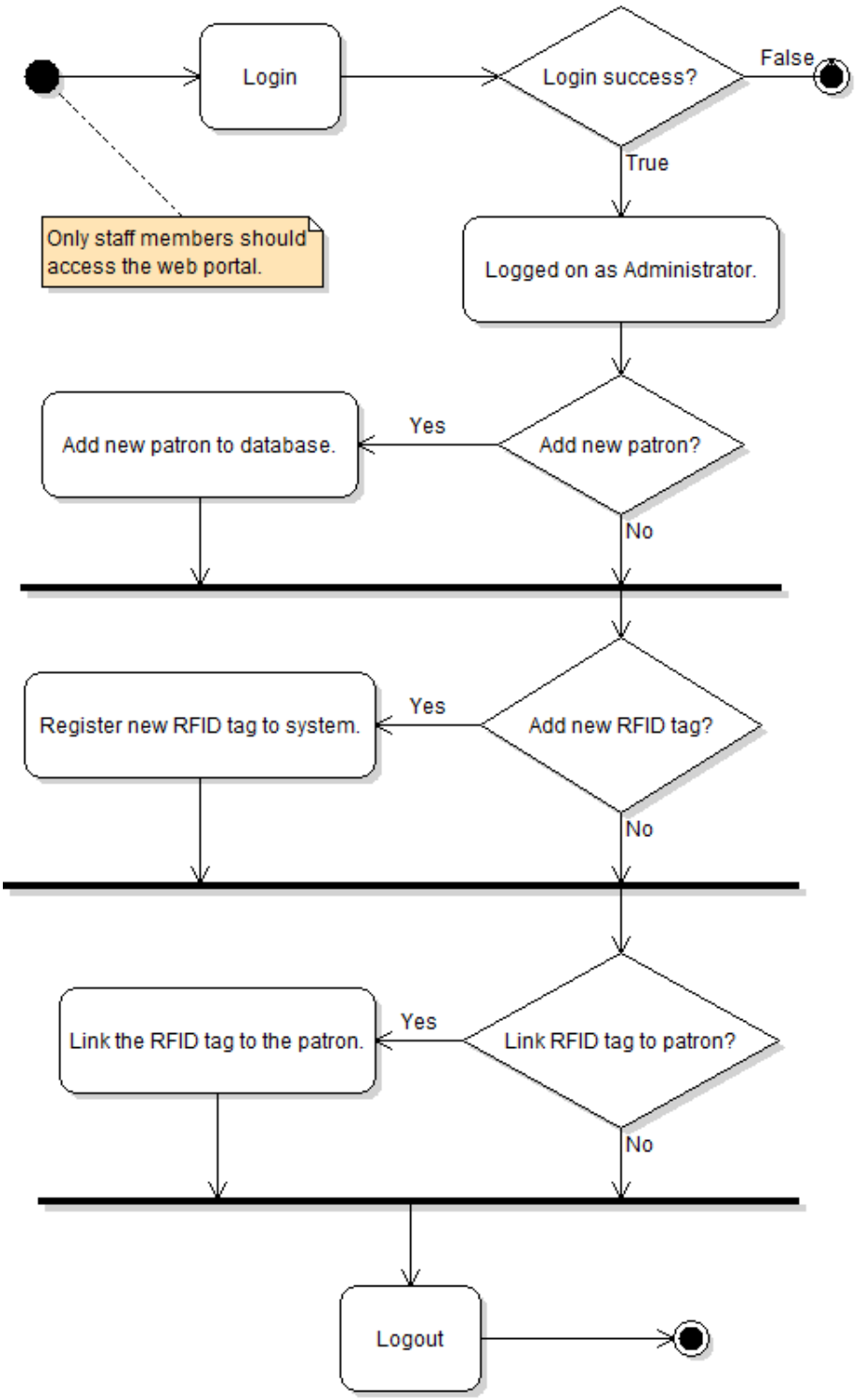

Figure 6: Activity diagram for the web application (administration portal).