# Use Case # [1: Add User]

|  |  |
| --- | --- |
| GENERAL CHARACTERISTICS | |
| **Author** | Yibing Zhang |
| **Last Update:** | 9/25/2017 |
| **Scope** | Home Automation System |
| **Level** | User-goal |
| **Status** | Incomplete Conceptualization |
| **Primary Actor** | Mobile Application User |
| **Secondary Actors** | Server; Mobile Application |
| **Stakeholders and Interests** | Mobile App User: Wants to register a new person to the system. |
| **Preconditions** | User has Mobile application installed and registered. |
| **Success Post Condition** | The user is successfully registered for |
| **Failed Post Condition** | The user is not registered for |

|  |  |
| --- | --- |
| MAIN SUCCESS SCENARIO (or basic flow) | |
| **Step** | **Action -** description in words of each step in success scenario |
| 1 | The administrator goes to the administrator page and navigates to the “add a new person” menu |
| 2 | The administrator system asks the administrator to fill in the information and then click register. |
| 3 | The administrator system sends the information to the server. |
| 4 | The server add a new user its database. |
| 5 | The user is now registered. |

|  |  |
| --- | --- |
| EXTENSIONS or Alternate Flows | |
| **Step** | **Branching Action** |
| *n..m* | \*a At any time the server fails:   1. Server attempt to find and fix issues   1a. Server is offline  1. App informs the user that no server is detected  2. User restarts the server  3. App reconnects to the server    1b. Server crash  1. Server auto restart  1a. Server fails to auto restart  1. Mobile app informs the user after 1 minute of no response from the server  2. User manually restarts server   1. Server requests information from the mobile application and resumes normal functionality   \*b At any time the mobile application fails:  1. Mobile app searches for issue  1a. Mobile app has no network connection  1. App attempts to connect to wifi to restore connection  1a. App fails to connect to wifi  1. The information is stored on the app and queue to send when connection is restored  2. The app informs the user that it has no network connection.  2a. Mobile application crashes  1. Mobile app sends information about the cause of the crash  2. App attempts to auto restart  2a. Fails to auto restart  1. User manually restarts the app  2. App sends information to the server  3. Mobile app sends its information to the server and reestablishes connections based on what the server sends back |
| 4. The user cannot be registered  1. The input password doesn’t match twice  2. The user is already existing |

|  |  |
| --- | --- |
| SPECIAL REQUIREMENTS | |
| **Req Num** | **Requirement** |
| *n* | 1. Speaker connections restricted to users 2. Server communications require authentication 3. Server communication allowed outside of the local network |

|  |  |
| --- | --- |
| TECHNOLOGY AND DATA VARIATIONS LIST | |
| **Var Num** | **Variation** |
| *n* | 7a. Communications are done over a wireless network so server would need internet and mobile device would need mobile data or be connected to wifi  7b. Keyboard is required to restart server as admin passwords would be needed |

***FREQUENCY OF OCCURRENCE***: Not often. It only occurs when there comes a new roommate that wants share this system

|  |  |
| --- | --- |
| OTHER ISSUES | |
| **Issue Num** | **Issue** |
| *n* | 1. How will we verify this one is the person that is harmless? 2. How do we deal with someone who knows the admin password but is someone we don’t know? |

# Use Case # [2: Changing a user’s password]

|  |  |
| --- | --- |
| GENERAL CHARACTERISTICS | |
| **Author** | Jordan Persson |
| **Last Update:** | 9/23/2017 |
| **Scope** | Home Automation System |
| **Level** | User-goal |
| **Status** | Incomplete Conceptualization |
| **Primary Actor** | Mobile Application User |
| **Secondary Actors** | Server; Mobile Application |
| **Stakeholders and Interests** | Mobile App User: Wants to change their user password |
| **Preconditions** | User has Mobile application installed and registered. |
| **Success Post Condition** | The password has successfully been reset |
| **Failed Post Condition** | The password was not reset. |

|  |  |
| --- | --- |
| MAIN SUCCESS SCENARIO (or basic flow) | |
| **Step** | **Action -** description in words of each step in success scenario |
| 1 | The user navigates to the profile settings page on the mobile application and clicks “Change Password” |
| 2 | The mobile app displays the change password screen and asks the user for their old password, the new password, and to re-enter the new password. |
| 3 | The user inputs this information and clicks submit. |
| 4 | The mobile app sends the old password and new password to the server. |
| 5 | The server confirms that the password matches and sets the new user password. |
| 6 | The server sends a confirmation success to the mobile app. |
| 7 | The mobile app informs the user that the password has been updated. |

|  |  |
| --- | --- |
| EXTENSIONS or Alternate Flows | |
| **Step** | **Branching Action** |
| *n..m* | \*a At any time the server fails:   1. Server attempt to find and fix issues   1a. Server is offline  1. App informs the user that no server is detected  2. User restarts the server  3. App reconnects to the server    1b. Server crash  1. Server auto restart  1a. Server fails to auto restart  1. Mobile app informs the user after 1 minute of no response from the server  2. User manually restarts server   1. Server requests information from the mobile application and resumes normal functionality   \*b At any time the mobile application fails:  1. Mobile app searches for issue  1a. Mobile app has no network connection  1. App attempts to connect to wifi to restore connection  1a. App fails to connect to wifi  1. The information is stored on the app and queue to send when connection is restored  2. The app informs the user that it has no network connection.  2. App sends information to the server  2. Mobile application crashes  1a. Mobile app fails to auto restart  1. Mobile app sends information about the cause of the crash  2. User manually restarts the app  3. Mobile app sends its information to the server and reestablishes connections based on what the server sends back |
| 1a. The user is not logged in to the mobile app.  1. The user selects the forgot password option  2. The mobile app asks for the user’s email associated with the accout.  3. The user inputs the email.  3a. The user forgot the email.  1. The user clicks forgot email.  2. The mobile asks for the phone number or username associated with the account.  2a. The user doesn’t know this information.  1. The user must create a new account.  3. The user inputs the information and clicks submit.  4. The mobile app sends the information to the server.  5. The server confirms the information and sends a reset password link to the email for the account.  5a. Account not found.  1. The server will tell the mobile app that the information was not associated with a known account.  2. The mobile app will inform the user that the account was not found.  3. The user can repeat step 1a.  6. The user follows that link and resets their password.  3a. The new password and conformation entry do not match  1. The mobile app informs the user that the passwords do not match.  2. The user must reenter the passwords and click submit.  3b. The password doesn’t met security standards  1. The mobile app checks the password against its security standards and tells the user what criteria the password is missing.  2. The user inputs a new password and submits.  5a. The old password doesn’t match what is on the server  1. The server tells the mobile app that the passwords didn’t match  2. The mobile app tells the user that an invalid password was used.  3. The users inputs new password and submits.  5b. The new password and the old password were the same  1. The server tells the mobile app that the new and old passwords were the same.  2. The mobile app tells the user that the new password cannot match the old one.  3. The user puts in a new password and submits. |
|  |  |

|  |  |
| --- | --- |
| SPECIAL REQUIREMENTS | |
| **Req Num** | **Requirement** |
| *n* | 1. Server communications require authentication 2. Multiple failed reset attempts will cause the server to send a warning email to the email associated with the account. |

|  |  |
| --- | --- |
| TECHNOLOGY AND DATA VARIATIONS LIST | |
| **Var Num** | **Variation** |
| *n* | \*a. Communications are done over a wireless network so server would need internet and mobile device would need mobile data or be connected to wifi  \*b. Keyboard is required to restart server as admin passwords would be needed |

***FREQUENCY OF OCCURRENCE***: Often. It would occur as often as the user changes rooms

|  |  |
| --- | --- |
| OTHER ISSUES | |
| **Issue Num** | **Issue** |
| *n* | 1. How many failed attempts should we allow before locking out an account? 2. How will we encrypt the passwords? |

**Use Case # [3: Calibrate room boundaries]**

|  |  |
| --- | --- |
| GENERAL CHARACTERISTICS | |
| **Author** | Yibing Zhang |
| **Last Update:** | 9/25/2017 |
| **Scope** | Home Automation System |
| **Level** | User-goal |
| **Status** | Incomplete Conceptualization |
| **Primary Actor** | Mobile Application User |
| **Secondary Actors** | Server; Mobile Application |
| **Stakeholders and Interests** | Mobile App User: Wants to calibrate room boundaries to support multiple users have the music and lights on at the same time |
| **Preconditions** | User has Mobile application installed and registered.  One room is already occupied by someone else |
| **Success Post Condition** | Another user walking into a new room that is occupied by another user would not take control of the speaker and light (first come, first serve) |
| **Failed Post Condition** | Permission conflict occurs and the later guys takes control of everything |

|  |  |
| --- | --- |
| MAIN SUCCESS SCENARIO (or basic flow) | |
| **Step** | **Action -** description in words of each step in success scenario |
| 1 | The user walks into another room that is occupied by another user |
| 2 | The mobile app sends a message to the server telling it what room it has entered. |
| 3 | The server receives the information and check if the room is occupied by someone else. It sends that speaker’s information back to the mobile application. |
| 4 | The mobile app receives the speaker information and doesn’t connect to the speaker in this room. |
| 5 | The previous user still has access to the speaker in his room |

|  |  |
| --- | --- |
| EXTENSIONS or Alternate Flows | |
| **Step** | **Branching Action** |
| *n..m* | \*a At any time the server fails:   1. Server attempt to find and fix issues   1a. Server is offline  1. App informs the user that no server is detected  2. User restarts the server  3. App reconnects to the server    1b. Server crash  1. Server auto restart  1a. Server fails to auto restart  1. Mobile app informs the user after 1 minute of no response from the server  2. User manually restarts server  2. Server requests information from the mobile application and resumes normal functionality  \*b At any time the mobile application fails:  1. Mobile app searches for issue  1a. Mobile app has no network connection  1. App attempts to connect to wifi to restore connection  1a. App fails to connect to wifi  1. The information is stored on the app and queue to send when connection is restored  2. The app informs the user that it has no network connection.  2a. Mobile application crashes  1. Mobile app sends information about the cause of the crash  2. App attempts to auto restart  2a. Fails to auto restart  1. User manually restarts the app  2. App sends information to the server  3. Mobile app sends its information to the server and reestablishes connections based on what the server sends back |
| 4. The user cannot connect to the speaker that is occupied by another user  1. The user must ensure that the speaker is not occupied  2. The user will have no connection with any speaker since he walks into a room that is occupied by anther user (he lost connection once he leaves his own room) |

|  |  |
| --- | --- |
| SPECIAL REQUIREMENTS | |
| **Req Num** | **Requirement** |
| *n* | 1. Speaker connections restricted to users 2. Server communications require authentication 3. Server communication allowed outside of the local network |

|  |  |
| --- | --- |
| TECHNOLOGY AND DATA VARIATIONS LIST | |
| **Var Num** | **Variation** |
| *n* | 7a. Communications are done over a wireless network so server would need internet and mobile device would need mobile data or be connected to wifi  7b. Keyboard is required to restart server as admin passwords would be needed |

***FREQUENCY OF OCCURRENCE***: Not often. It only occurs when the speaker is broken or the user gets a new speaker

|  |  |
| --- | --- |
| OTHER ISSUES | |
| **Issue Num** | **Issue** |
| *n* | 1. How will we deal with case once the user lost all connections when he walks into an occupied room? Should we let the mobile app keeps looking for the nearest speaker or we force it to stop? 2. How will the occupied room’s information be stored? |

# Use Case # [4: Disable/ remove speaker]

|  |  |
| --- | --- |
| GENERAL CHARACTERISTICS | |
| **Author** | Yibing Zhang |
| **Last Update:** | 9/25/2017 |
| **Scope** | Home Automation System |
| **Level** | User-goal |
| **Status** | Incomplete Conceptualization |
| **Primary Actor** | Mobile Application User |
| **Secondary Actors** | Server; Mobile Application |
| **Stakeholders and Interests** | Mobile App User: Wants to remove or disable a speaker |
| **Preconditions** | User has Mobile application installed and registered.  Music source is in the system. |
| **Success Post Condition** | The speaker is successfully removed |
| **Failed Post Condition** | The speaker remains in the system |

|  |  |
| --- | --- |
| MAIN SUCCESS SCENARIO (or basic flow) | |
| **Step** | **Action -** description in words of each step in success scenario |
| 1 | The user uses the mobile application and navigates to the “remove a device” menu |
| 2 | The mobile app displays the options of devices to remove |
| 3 | The user selects the speaker option. |
| 3 | The mobile app instructs the user to navigate the speaker and then click remove. |
| 4 | The mobile app sends the information to the server. |
| 5 | The server removes the speaker from its database. |
| 6 | The speaker is now removed. |

|  |  |
| --- | --- |
| EXTENSIONS or Alternate Flows | |
| **Step** | **Branching Action** |
| *n..m* | \*a At any time the server fails:  1. Server attempt to find and fix issues  1a. Server is offline  1. App informs the user that no server is detected  2. User restarts the server  3. App reconnects to the server    1b. Server crash  1. Server auto restart  1a. Server fails to auto restart  1. Mobile app informs the user after 1 minute of no response from the server  2. User manually restarts server  2. Server requests information from the mobile application and resumes normal functionality  \*b At any time the mobile application fails:  1. Mobile app searches for issue  1a. Mobile app has no network connection  1. App attempts to connect to wifi to restore connection  1a. App fails to connect to wifi  1. The information is stored on the app and queue to send when connection is restored  2. The app informs the user that it has no network connection.  2a. Mobile application crashes  1. Mobile app sends information about the cause of the crash  2. App attempts to auto restart  2a. Fails to auto restart  1. User manually restarts the app  2. App sends information to the server  3. Mobile app sends its information to the server and reestablishes connections based on what the server sends back |
| 4. The user cannot find the speaker  1. The user must ensure that the speaker is in the system |

|  |  |
| --- | --- |
| SPECIAL REQUIREMENTS | |
| **Req Num** | **Requirement** |
| *n* | 1. Speaker connections restricted to users 2. Server communications require authentication 3. Server communication allowed outside of the local network |

|  |  |
| --- | --- |
| TECHNOLOGY AND DATA VARIATIONS LIST | |
| **Var Num** | **Variation** |
| *n* | 7a. Communications are done over a wireless network so server would need internet and mobile device would need mobile data or be connected to wifi  7b. Keyboard is required to restart server as admin passwords would be needed |

***FREQUENCY OF OCCURRENCE***: Not often. It only occurs when the speaker is broken or the user gets a new speaker

|  |  |
| --- | --- |
| OTHER ISSUES | |
| **Issue Num** | **Issue** |
| *n* | 1. How will device be located? 2. What will we do once we remove a device? Should we have an alternative or reset the room boundaries? 3. What if we remove a device that is currently used by another user? |

# Use Case # [5: Disable/ remove speaker]

|  |  |
| --- | --- |
| GENERAL CHARACTERISTICS | |
| **Author** | Yibing Zhang |
| **Last Update:** | 9/25/2017 |
| **Scope** | Home Automation System |
| **Level** | User-goal |
| **Status** | Incomplete Conceptualization |
| **Primary Actor** | Mobile Application User |
| **Secondary Actors** | Server; Mobile Application |
| **Stakeholders and Interests** | Mobile App User: Wants to remove or disable a speaker |
| **Preconditions** | User has Mobile application installed and registered.  Music source is in the system. |
| **Success Post Condition** | The speaker is successfully removed |
| **Failed Post Condition** | The speaker remains in the system |

|  |  |
| --- | --- |
| MAIN SUCCESS SCENARIO (or basic flow) | |
| **Step** | **Action -** description in words of each step in success scenario |
| 1 | The user uses the mobile application and navigates to the “remove a device” menu |
| 2 | The mobile app displays the options of devices to remove |
| 3 | The user selects the speaker option. |
| 3 | The mobile app instructs the user to navigate the speaker and then click remove. |
| 4 | The mobile app sends the information to the server. |
| 5 | The server removes the speaker from its database. |
| 6 | The speaker is now removed. |

|  |  |
| --- | --- |
| EXTENSIONS or Alternate Flows | |
| **Step** | **Branching Action** |
| *n..m* | \*a At any time the server fails:  1. Server attempt to find and fix issues  1a. Server is offline  1. App informs the user that no server is detected  2. User restarts the server  3. App reconnects to the server    1b. Server crash  1. Server auto restart  1a. Server fails to auto restart  1. Mobile app informs the user after 1 minute of no response from the server  2. User manually restarts server  2. Server requests information from the mobile application and resumes normal functionality  \*b At any time the mobile application fails:  1. Mobile app searches for issue  1a. Mobile app has no network connection  1. App attempts to connect to wifi to restore connection  1a. App fails to connect to wifi  1. The information is stored on the app and queue to send when connection is restored  2. The app informs the user that it has no network connection.  2a. Mobile application crashes  1. Mobile app sends information about the cause of the crash  2. App attempts to auto restart  2a. Fails to auto restart  1. User manually restarts the app  2. App sends information to the server  3. Mobile app sends its information to the server and reestablishes connections based on what the server sends back |
| 4. The user cannot find the speaker  1. The user must ensure that the speaker is in the system |

|  |  |
| --- | --- |
| SPECIAL REQUIREMENTS | |
| **Req Num** | **Requirement** |
| *n* | 1. Speaker connections restricted to users 2. Server communications require authentication 3. Server communication allowed outside of the local network |

|  |  |
| --- | --- |
| TECHNOLOGY AND DATA VARIATIONS LIST | |
| **Var Num** | **Variation** |
| *n* | 7a. Communications are done over a wireless network so server would need internet and mobile device would need mobile data or be connected to wifi  7b. Keyboard is required to restart server as admin passwords would be needed |

***FREQUENCY OF OCCURRENCE***: Not often. It only occurs when the speaker is broken or the user gets a new speaker

|  |  |
| --- | --- |
| OTHER ISSUES | |
| **Issue Num** | **Issue** |
| *n* | 1. How will device be located? 2. What will we do once we remove a device? Should we have an alternative or reset the room boundaries? 3. What if we remove a device that is currently used by another user? |

**Use Case # [6: Manually lights on and off via Amazon Dash buttons]**

|  |  |
| --- | --- |
| ***GENERAL CHARACTERISTICS*** | |
| **Author** | John Clarke |
| **Last Update:** | 9/25/2017 |
| **Scope** | Home Automation System |
| **Level** | User-goal |
| **Status** | Incomplete Conceptualization |
| **Primary Actor** | Mobile Application User |
| **Secondary Actors** | Server; Mobile Application |
| **Stakeholders and Interests** | Mobile App User: Wants to manually turn on and off lights in a given room via specific Amazon Dash buttons |
| **Preconditions** | User has Mobile application installed and registered.  User has at least one Amazon Dash button and one connected light source that the button is mapped to. |
| **Success Post Condition** | After pushing the Amazon Dash button, the light is now off it was on prior to pressing the button or the light is now on if it was previously off. |
| **Failed Post Condition** | The light source has the same state as it did prior to pressing the Amazon Dash button |

|  |  |
| --- | --- |
| ***MAIN SUCCESS SCENARIO (or basic flow)*** | |
| **Step** | **Action -** description in words of each step in success scenario |
| 1  2  3 | The User has entered a room equipped with an Amazon Dash button and a controllable light source  The User presses the Amazon Dash button.  The light source is now on if it was off prior to pressing the Amazon Dash button or is now off it was on prior to pressing the Amazon Dash button |

|  |  |
| --- | --- |
| ***EXTENSIONS or Alternate Flows*** | |
| **Step** | **Branching Action** |
| *n..m* | \*a At any time the server fails:  1. Server attempt to find and fix issues  1a. Server is offline  1. App informs the user that no server is detected  2. User restarts the server  3. App reconnects to the server    1b. Server crash  1. Server auto restart  1a. Server fails to auto restart  1. Mobile app informs the user after 1 minute of no response from the server  2. User manually restarts server   1. Server requests information from the mobile application and resumes normal functionality   \*b At any time the mobile application fails:  1. Mobile app searches for issue  1a. Mobile app has no network connection  1. App attempts to connect to wifi to restore connection  1a. App fails to connect to wifi  1. The information is stored on the app and queue to send when connection is restored  2. The app informs the user that it has no network connection.  2a. Mobile application crashes  1. Mobile app sends information about the cause of the crash  2. App attempts to auto restart  2a. Fails to auto restart  1. User manually restarts the app  2. App sends information to the server  3. Mobile app sends its information to the server and reestablishes connections based on what the server sends back |
|  | 3A. The controllable light source is not mapped to the desired light source   1. The User notifies the administrator of the central server 2. The administrator readjusts the mapping of the specific Amazon Dash button to the specific light source   3B. The light source does not switch states after pressing the Amazon Dash button   1. The User notifies the administrator who readjusts the mapping   3C. Amazon Dash button is not connected to the local wifi network.   1. The User uses Amazon app to connect Amazon Dash button to wifi |
|  |  |

|  |  |
| --- | --- |
| ***SPECIAL REQUIREMENTS*** | |
| **Req Num** | **Requirement** |
| *n* | 1. Light connections restricted to users 2. Server communications require authentication 3. The Amazon Dash button is connected to the local wi-fi network 4. Each Amazon Dash Button is mapped to a specific light source by the administrator of the central server |

|  |  |
| --- | --- |
| ***TECHNOLOGY AND DATA VARIATIONS LIST*** | |
| **Var Num** | **Variation** |
| *n* | 2a. Communications are done over a wireless network so server would need internet and mobile device would need mobile data or be connected to wifi  2b. Keyboard is required to restart server as admin passwords would be needed  3a. The Amazon Dash button must be connected to the local wifi network.  3b. The Amazon Dash button must be modified to respond to systems not provided and serviced by Amazon. |

***FREQUENCY OF OCCURRENCE***: Common but not necessarily often

|  |  |
| --- | --- |
| ***OTHER ISSUES*** | |
| **Issue Num**  **1.** | **Issue**  How is the Amazon Dash button modified to respond to systems other than that provided by Amazon? |

# Use Case # [7: Turning off the light as you leave a room]

|  |  |
| --- | --- |
| GENERAL CHARACTERISTICS | |
| **Author** | Jordan Persson |
| **Last Update:** | 9/23/2017 |
| **Scope** | Home Automation System |
| **Level** | User-goal |
| **Status** | Incomplete Conceptualization |
| **Primary Actor** | Mobile Application User |
| **Secondary Actors** | Server; Mobile Application |
| **Stakeholders and Interests** | Mobile App User: Wants for their lights to turn off as they leave a room. |
| **Preconditions** | User has Mobile application installed and registered. |
| **Success Post Condition** | The light registered for the room the user just left has turned off. |
| **Failed Post Condition** | The light registered for the room the user just left remains on. |

|  |  |
| --- | --- |
| MAIN SUCCESS SCENARIO (or basic flow) | |
| **Step** | **Action -** description in words of each step in success scenario |
| 1 | The user walks out of a room. |
| 2 | The mobile app sends a message to the server telling it what room it has just left. |
| 3 | The server receives the information and finds the light/lights registered for that room. |
| 4 | The server sends a signal to those lights to turn off. |
| 5 | The lights are now off in the old room. |

|  |  |
| --- | --- |
| EXTENSIONS or Alternate Flows | |
| **Step** | **Branching Action** |
| *n..m* | \*a At any time the server fails:  1. Server attempt to find and fix issues  1a. Server is offline  1. App informs the user that no server is detected  2. User restarts the server  3. App reconnects to the server    1b. Server crash  1. Server auto restart  1a. Server fails to auto restart  1. Mobile app informs the user after 1 minute of no response from the server  2. User manually restarts server   1. Server requests information from the mobile application and resumes normal functionality   \*b At any time the mobile application fails:  1. Mobile app searches for issue  1a. Mobile app has no network connection  1. App attempts to connect to wifi to restore connection  1a. App fails to connect to wifi  1. The information is stored on the app and queue to send when connection is restored  2. The app informs the user that it has no network connection.  2a. Mobile application crashes  1. Mobile app sends information about the cause of the crash  2. App attempts to auto restart  2a. Fails to auto restart  1. User manually restarts the app  2. App sends information to the server  3. Mobile app sends its information to the server and reestablishes connections based on what the server sends back |
| 3a. No light is found for that room  1. The system makes a note of the missing light and sends it to the mobile app  2. The Mobile app stores this information in order to offer suggestions to the user at a later  time.  4a. Lights fail to turn off  1. Server tries to find error  1a. Light is not connected to the server  1. The server notes the communication error and sends it to the mobile app  2. The mobile app informs the user that the lights are not connected to the server  3. The user connects the light to the system  1b. Light is connected but fails to turn off  1. The server informs the mobile app of the error  2. The mobile app tells the user that the light isn’t functioning  2. The user disconnects and reconnects the light to the system  2. The light is recalibrated and attempts to turn off again.  2a. The light still won’t work with the system  1. The system tell the mobile app the lights may be faulty.  2. The mobile app tells the user that the lights may need to be replaced  3. User replaces the lights, disconnects the old lights and connects the new ones to the system |
|  |  |

|  |  |
| --- | --- |
| SPECIAL REQUIREMENTS | |
| **Req Num** | **Requirement** |
| *n* | 1. Light connections restricted to users 2. Server communications require authentication 3. Server communication allowed outside of the local network |

|  |  |
| --- | --- |
| TECHNOLOGY AND DATA VARIATIONS LIST | |
| **Var Num** | **Variation** |
| *n* | 2a. Communications are done over a wireless network so server would need internet and mobile device would need mobile data or be connected to wifi  2b. Keyboard is required to restart server as admin passwords would be needed  4a. Lights must be connected to the server or have a switch connected to the server |

***FREQUENCY OF OCCURRENCE***: Often. It would occur as often as the user changes rooms

|  |  |
| --- | --- |
| OTHER ISSUES | |
| **Issue Num** | **Issue** |
| *n* | 1. How will be approach device authentication? 2. How will the light’s information be stored? 3. Should there be dead times to prevent accidentally turning on the lights when they wouldn’t be needed? |

# Use Case # [8: Turning on light bulb by entering a room]

|  |  |
| --- | --- |
| GENERAL CHARACTERISTICS | |
| **Author** | John Clarke |
| **Last Update:** | 9/25/2017 |
| **Scope** | Home Automation System |
| **Level** | User-goal |
| **Status** | Incomplete Conceptualization |
| **Primary Actor** | Mobile Application User |
| **Secondary Actors** | Server; Mobile Application |
| **Stakeholders and Interests** | Mobile App User: Wants a light source in room to be turned on upon entering the room |
| **Preconditions** | User has Mobile application installed and registered.  Light source in room to be entered is not currently on |
| **Success Post Condition** | The light source is turned on upon entering the room. |
| **Failed Post Condition** | The light source in room that user has entered is not turned on. |

|  |  |
| --- | --- |
| MAIN SUCCESS SCENARIO (or basic flow) | |
| **Step** | **Action -** description in words of each step in success scenario |
| 1 | The user walks into a new room that has no connected light source currently on |
| 2 | The mobile app sends a message to the server telling it what room it has entered. |
| 3  4 | The server receives the information and determines which light source to turn on.  The server turns on the specific light source for that room. |

|  |  |
| --- | --- |
| EXTENSIONS or Alternate Flows | |
| **Step** | **Branching Action** |
| *n..m* | \*a At any time the server fails:  1. Server attempt to find and fix issues  1a. Server is offline  1. App informs the user that no server is detected  2. User restarts the server  3. App reconnects to the server    1b. Server crash  1. Server auto restart  1a. Server fails to auto restart  1. Mobile app informs the user after 1 minute of no response from the server  2. User manually restarts server  2. Server requests information from the mobile application and resumes normal functionality  \*b At any time the mobile application fails:  1. Mobile app searches for issue  1a. Mobile app has no network connection  1. App attempts to connect to wifi to restore connection  1a. App fails to connect to wifi  1. The information is stored on the app and queue to send when connection is restored  2. The app informs the user that it has no network connection.  2a. Mobile application crashes  1. Mobile app sends information about the cause of the crash  2. App attempts to auto restart  2a. Fails to auto restart  1. User manually restarts the app  2. App sends information to the server  3. Mobile app sends its information to the server and reestablishes connections based on what the server sends back |
|  | 1A. Light source is manually turned on through app before entering room   1. Server determines that light source is already on by app manual override 2. No changes are made to light source’s status   1B. Light source is manually turned on through Amazon Dash button   1. Server determines that light source was manually turned on by Amazon Dash button override 2. No changes are made to light source’s status   1C. Multiple light sources are to be turned on when user enters room   1. User enters a specific room that contains multiple light sources 2. Upon entering, the app communicates with the central server 3. Server determines which light sources are in the room 4. Server turns on all light sources that are contained in the room |
|  |  |

|  |  |
| --- | --- |
| SPECIAL REQUIREMENTS | |
| **Req Num** | **Requirement** |
| *n* | 1. Light connections restricted to users 2. Server communications require authentication 3. Server communication allowed outside of the local network |

|  |  |
| --- | --- |
| TECHNOLOGY AND DATA VARIATIONS LIST | |
| **Var Num** | **Variation** |
| *n* | 2a. Communications are done over a wireless network so server would need internet and mobile device would need mobile data or be connected to wifi  2b. Keyboard is required to restart server as admin passwords would be needed |

***FREQUENCY OF OCCURRENCE***: Often. It would occur as often as the user changes rooms

|  |  |
| --- | --- |
| OTHER ISSUES | |
| **Issue Num**  **1.**  **2.**  **3.** | **Issue**  If light is turned on by one process, should it be turned off by the same process?  Are Users limited to only one connected light source per room?  If Users can have multiple light sources per room, can only one light source be turned on by entering the room? |