Use Cases

for

P.H.L.I.P

(Philip Hue Light Interface Program)

Version 0.2

Prepared by Karan

CS 411 — Group 4

November 1, 2016

https://github.com/bogjacob/cs411-group4

This file is located:
https://github.com/bogjacob/cs411-group4/tree/master/Planning

1. Guidance for Use Case Template

Document each use case using the template shown in the Appendix. This section provides a description of each section in the use case template.

2. Use Case Identification

1.1. Use Case ID

Give each use case a unique integer sequence number identifier. Alternatively, use a hierarchical form: X.Y. Related use cases can be grouped in the hierarchy.

1.2. Use Case Name

State a concise, results-oriented name for the use case. These reflect the tasks the user needs to be able to accomplish using the system. Include an action verb and a noun. Some examples:

- View part number information.
- Manually mark hypertext source and establish link to target.
- Place an order for a CD with the updated software version.

1.3. Use Case History

1.1.1. Created By

Supply the name of the person who initially documented this use case.

1.1.2. Date Created

Enter the date on which the use case was initially documented.

1.1.3. Last Updated By

Supply the name of the person who performed the most recent update to the use case description.

1.1.4. Date Last Updated

Enter the date on which the use case was most recently updated.

3. Use Case Definition

1.1. Actors

An actor is a person or other entity external to the software system being specified who interacts with the system and performs use cases to accomplish tasks. Different actors often correspond to different user classes, or roles, identified from the customer community that will use the product. Name the actor that will be initiating this use case and any other actors who will participate in completing the use case.

1.2. Trigger

Identify the event that initiates the use case. This could be an external business event or system event that causes the use case to begin, or it could be the first step in the normal flow.

1.3. Description

Provide a brief description of the reason for and outcome of this use case, or a high-level description of the sequence of actions and the outcome of executing the use case.

1.4. Preconditions

List any activities that must take place, or any conditions that must be true, before the use case can be started. Number each precondition. Examples:

1. User's identity has been authenticated.
2. User's computer has sufficient free memory available to launch task.

1.5. Postconditions

Describe the state of the system at the conclusion of the use case execution. Number each postcondition. Examples:

1. Document contains only valid SGML tags.
2. Price of item in database has been updated with new value.

1.6. Normal Flow

Provide a detailed description of the user actions and system responses that will take place during execution of the use case under normal, expected conditions. This dialog sequence will ultimately lead to accomplishing the goal stated in the use case name and description. This description may be written as an answer to the hypothetical question, "How do I <accomplish the task stated in the use case name>?" This is best done as a numbered list of actions performed by the actor, alternating with responses provided by the system. The normal flow is numbered "X.0", where "X" is the Use Case ID.

1.7. Alternative Flows

Document other, legitimate usage scenarios that can take place within this use case separately in this section. State the alternative flow, and describe any differences in the sequence of steps that take place. Number each alternative flow in the form "X.Y", where "X" is the Use Case ID and Y is a sequence number for the alternative flow. For example, "5.3" would indicate the third alternative flow for use case number 5.

1.8. Exceptions

Describe any anticipated error conditions that could occur during execution of the use case, and define how the system is to respond to those conditions. Also, describe how the system is to respond if the use case execution fails for some unanticipated reason. If the use case results in a durable state change in a database or the outside world, state whether the change is rolled back, completed correctly, partially completed with a known state, or left in an undetermined state as a result of the exception. Number each alternative flow in the form "X.Y.E.Z", where "X" is the Use Case ID, Y indicates the normal (0) or alternative (>0) flow during which this exception could take place, "E" indicates an exception, and "Z" is a sequence number for the exceptions. For example "5.0.E.2" would indicate the second exception for the normal flow for use case number 5.

1.9. Includes

List any other use cases that are included (“called”) by this use case. Common functionality that appears in multiple use cases can be split out into a separate use case that is included by the ones that need that common functionality.

1.10. Priority

Indicate the relative priority of implementing the functionality required to allow this use case to be executed. The priority scheme used must be the same as that used in the software requirements specification.

1.11. Frequency of Use

Estimate the number of times this use case will be performed by the actors per some appropriate unit of time.

1.12. Business Rules

List any business rules that influence this use case.

1.13. Special Requirements

Identify any additional requirements, such as nonfunctional requirements, for the use case that may need to be addressed during design or implementation. These may include performance requirements or other quality attributes.

1.14. Assumptions

List any assumptions that were made in the analysis that led to accepting this use case into the product description and writing the use case description.

1.15. Notes and Issues

List any additional comments about this use case or any remaining open issues or TBDs (To Be Determined) that must be resolved. Identify who will resolve each issue, the due date, and what the resolution ultimately is.

Use Case List

ID	*Primary Actor*	*Use Case Title*
0	User	Audio Visualizer
1	User	Weather Alerts

Use Case 0

Use Case ID:	0		
Use Case Name:	Audio Visualization		
Created By:	Jacob	Last Updated By:	Karan
Date Created:	10/3/16	Date Last Updated:	12/1/16

Actors:	User
Description:	The audio visualizer use case analyzes a song or audio chosen by the user and utilizes the Philips Hue Lights in tandem with set visualization parameters to provide an accompanying audio visualization.
Trigger:	User selects “Audio Visualization” mode.
Preconditions:	1. User must be logged in. 2. User must have completed initial setup of connecting Hue lights 3. User confirms the visualization mode. (Audio Visualization is the default mode.) 4. User must have input URL of desired audio. 5. Audio track must be downloaded to the web server.
Postconditions:	1. Music stops playing. 2. Lights return to their default mode. 3. Light visualization saved to database for case in which user listens to the same song again.
Normal Flow:	0.0.0. User searches for songs by artist. 0.0.1. User downloads the mp3 version of the song. 0.0.2. System stores the mp3 audio file. 0.1.0. User presses play on the audio page. 0.1.1. System plays the audio. 0.2.0. System sends the audio data to the Philips Hue API. 0.2.1. Philip Hue API stores and processes the audio data. 0.3.0. Philip Hue lights start flashing according to the audio. 0.4.0. Following completion of song system asks user for evaluation of visualization performance. 0.4.1. System stores responses in database. 0.5.0. If no song is queued, returns user to audio selection menu.

	0.5.1. If song is queued, normal flow resumes from 0.1.0.
Alternative Flows:	--
Exceptions:	1. Hue internal error - tell the user to fix Hue and then re-setup. 2. Hue connection error - prompt the user to reconnect Hue controller to continue. 3. Jamendo internal error - tell user to come back later. 4. Jamendo connection error - prompt user to try another song. 5. Setup error - send the user back to the setup page and have them redo that step. 6. Web server error (database error, etc.) - prompt user to restart web services.
Includes:	1. Includes waveform analyzer which detects and determines audio length, beats, volume fluctuation, etc.
Priority:	1. A continuous connection to the internet. 2. A continuous connection to the Philips Hue lights. 3. Access to Jamendo API
Frequency of Use:	1 usage / 5-10 minutes/session
Business Rules:	--
Special Requirements:	1. If necessary, while the current song is running, the visualization needs to be precomputed for next song.
Assumptions:	1. User owns and has the capability to connect all necessary hardware (Philips Hue lights, bridge, etc.). 2. User has access to a computer with the required dependencies (Node.js / Flask / MySQL) to use as a local server. 3. User has access to Jamendo API 4. User has completed initial setup. 5. User is logged into their account.
Notes and Issues:	--

Use Case 1

Use Case ID:	1		
Use Case Name:	Weather Alerts		
Created By:	Dillon	Last Updated By:	Tony
Date Created:	11/1/16	Date Last Updated:	12/1/16

Actors:	User
Description:	The weather alerts use case utilizes the Philips Hue lights in tandem with alert parameters and preferences to provide visual weather alerts.
Trigger:	User selects “Weather Alerts” mode.
Preconditions:	1. User must be logged in. 2. User must have completed initial setup of connecting Hue lights 3. User must select the “Weather Alerts” mode, since it is not the default mode. 4. User must have input weather alert preferences.
Postconditions:	1. Weather alerts have stopped. 2. Return user to “Weather Alerts” default mode.
Normal Flow:	1.0.0. User inputs which weather condition they want alerted for. 1.0.1. User inputs their color preferences for weather alerts. 1.0.2. System stores these preference details. 1.1.0. User presses start on the weather alert page. 1.2.0. System requests and sends the source data to the Philips Hue API. 1.2.1. Philips Hue API processes the notification data. 1.3.0. Philips Hue lights flash and change according to the provided data. 1.4.0 At desired time, User stops notification “session.” 1.5.0. Returns user to “Weather Alert” menu. 1.5.1. If new session desired, normal flow resumes from 1.1.0.
Alternative Flows:	--
Exceptions:	1. Hue internal error - tell the user to fix Hue and then re-setup.

	2. Hue connection error - prompt the user to reconnect Hue controller to continue. 3. Setup error - send the user back to the setup page and have them redo that step. 4. Web server error (database error, etc.) - prompt user to restart web services.
Includes:	1. Weather notifier
Priority:	1. A required connection to the internet. 2. A successful, maintained connection to the Philips Hue lights. 3. A configuration of the Philips Hue lights layout within software.
Frequency of Use:	1 usage / 5-10 minutes/session
Business Rules:	--
Special Requirements:	--
Assumptions:	1. User owns and has the capability to connect all necessary hardware (Philips Hue lights, bridge, etc.). 2. User has access to a computer with the required dependencies (Node.js / Flask / MySQL) to use as a local server. 3. User has completed initial setup. 4. User is logged into their account.
Notes and Issues:	--

Revision History

Name	**Date**	**Reason For Changes**	**Version**
Jacob	10/4	Initial	Use Case 0
Dillon	11/1	Update to Use Case 1	Use Case 1
Karan	12/1	Update to Use Case 0	Use Case 0
Tony	12/1	Update to Use Case 1	Use Case 1