**1 point is an ideal days of work (can be decimal for fraction of a days work)**

**Account Creation**

* A user can create an account, or login, on the landing page **(2 points)**
* A user can create an account with a first name, last name, username/email, password**(3 points)**

**Tests:**

* Username doesn’t already exist – should not work
* User can login successfully – should go to settings page
* Make sure user inputs a username/password – otherwise fail

**Account Management**

* Once logged in user can change settings related to the hue lights api **(1 points)**
* A users settings is saved / persisted between logins **(2 points)**

**Tests:**

* Try unsuccessful login – should not take to welcome page
* User is taken to control page after logging in - pass
* Check to see if user settings are saved (get bridges)– pass

**Scheduler**

* A user can create a schedule **(1 points)**
* A user can edit a schedule **(1 points)**
* A user can delete a schedule **(1 points)**
* A user can view a schedule **(1 points)**

**Tests:**

* Check to see if schedules happen- pass
* Try invalid input - fail
* Try a time in the past - fail

**Navigation**

* Upon successful connection a user can access buttons to change their hue settings **(1 points)**
* User can logout with the logout button **(0.25 points)**

Successfully connected users can view and edit information regarding each light and bridge with the buttons created from logging in**(1 points)**

**Tests:**

* Try changing settings with the created buttons - pass
* User settings are saved between logins after changing - pass

**Bridges**

* User can add a bridge with a name and location through control panel settings **(1 points)**
* User can edit and remove existing bridges through control panel settings **(0.5 points)**

**Tests:**

* Try blank bridge name – fail
* Try blank location – fail
* Try invalid ip or port - fail
* Try removing a bridge – pass
* Try adding a valid bridge that can connect to the emulator -pass

**Lights**

* User can add/delete lights from a bridge. **(0.2 points)**
* User can get all lights for a selected bridge. **(0.2 points)**
* User can get light attributes and state. **(0.2 points)**
* User can set light attributes. **(0.2 points)**
* User can set light state. **(0.2 points)**
* User can turn lights on/off. **(0.2 points)**
* User can adjust light brightness. **(0.2 points)**
* User can adjust colour of lights. **(0.2 points)**

**Tests:**

* Settings are changed after being changed - pass

**Groups:**

* User can group lights together and adjust them all at once. **(0.2 points)**
* User can view all groups. **(0.3 points)**
* User can add/delete groups. **(0.2 points)**
* User can view/edit group attributes. **(0.3 points)**
* User can set group state. **(0.2 points)**

**Tests:**

* Settings are changed after being changed - pass

**Additional Feature: Holiday Themed Lights:**

* User can press a button to automatically set 3 lights for the specified bridges to be themed based off the closest holiday (Christmas or Halloween) **(2 points)**

**Tests:**

* Try without having bridge specified -fail
* Try with a bridge specified – pass

**Additional Feature: Recurring Group Light Cycler:**

* User can input a time it takes for a group of lights to go through a full hue cycle. This will keep on happening in the background asynchronously while the program runs until the user presses the stop button **(2 points)**

**Tests:**

* Try inputting negative amount of times – fail (turn it into default time of 2)
* Try positive number – pass
* Changes successfully happen – pass

**Additional Feature: Recurring Hue Light Cycler:**

* User can input a time it takes for a single light to go through a full hue cycle. This will keep on happening in the background asynchronously while the program runs until the user presses the stop button **(2 points)**

**Tests:**

* Try inputting negative amount of times – fail (turn it into default time of 2)
* Try positive number – pass
* Changes successfully happen – pass