**Software Requirements Specification**

**For**

**PILLAR**

**PILLAR Team**

**Venkata Yedidi**

**Teja Kakulavar**

**Andrew Touchet**

**Wennan Ma**

**Hugh DeJarnette**

**Louisiana Tech University**

**10/10/2016**

**Table of Contents**

**Table of Contents 2**

**Revision History 5**

**1.** **Introduction 6**

1.1 Purpose 6

1.2 Document Conventions 6

1.3 Intended Audience and Reading Suggestions 6

1.4 Product Scope 6

1.5 References 6

**2.** **Overall Description 7**

2.1 Product Perspective 7

2.2 Product Functions 7

2.3 User Classes and Characteristics 7

2.4 Operating Environment 7

2.5 Design and Implementation Constraints 7

2.6 User Documentation 7

2.7 Assumptions and Dependencies 8

**3.** **External Interface Requirements 8**

3.1 User Interfaces 8

3.2 Hardware Interfaces 8

3.3 Software Interfaces 8

3.4 Communications Interfaces 8

**4.** **System Features (Use Case Realization Analysis) 9**

4.1 Registration 9

4.1.1 Name: 9

4.1.2 Goal: 9

4.1.3 Input: 9

4.1.4 Output: 9

4.1.5 Main Scenario: 9

4.1.6 Pre-condition: 9

4.1.7 Steps: 9

4.1.8 Post-condition 9

4.1.9 Representation 9

4.2 User Login 9

4.2.1 Name: 9

4.2.2 Goal: 9

4.2.3 Input: 9

4.2.4 Output: 9

4.2.5 Main Scenario: 9

4.2.6 Pre-condition: 10

4.2.7 Steps: 10

4.2.8 Post-condition 10

4.2.9 Representation 10

4.3 Add Device 10

4.3.1 Name: 10

4.3.2 Goal: 10

4.3.3 Input: 10

4.3.4 Output: 10

4.3.5 Main Scenario: 10

4.3.6 Pre-condition: 10

4.3.7 Steps: 10

4.3.8 Post-condition 10

4.3.9 Representation 10

4.4 Remove Device 11

4.4.1 Name: 11

4.4.2 Goal: 11

4.4.3 Input: 11

4.4.4 Output: 11

4.4.5 Main Scenario: 11

4.4.6 Pre-condition: 11

4.4.7 Steps: 11

4.4.8 Post-condition 11

4.4.9 Representation 11

4.5 Group Devices based on the Location 12

4.5.1 Name: 12

4.5.2 Goal: 12

4.5.3 Input: 12

4.5.4 Output: 12

4.5.5 Main Scenario: 12

4.5.6 Pre-condition: 12

4.5.7 Steps: 12

4.5.8 Post-condition 12

4.5. 9 Representation 12

4.6 Display Device States 12

4.6.1 Name: 12

4.6.2 Goal: 13

4.6.3 Input: 13

4.6.4 Output: 13

4.6.5 Main Scenario: 13

4.6.6 Pre-condition: 13

4.6.7 Steps: 13

4.6.8 Post-condition 13

4.6.9 Representation 13

4.7 Control Device States 14

4.7.1 Name: 14

4.7.2 Goal: 14

4.7.3 Input: 14

4.7.4 Output: 14

4.7.5 Main Scenario: 14

4.7.6 Pre-condition: 14

4.7.7 Steps: 14

4.7.8 Post-condition 14

4.7.9 Representation 14

4.8 Get the Device State and Store in the Database 14

4.8.1 Name: 14

4.8.2 Goal: 14

4.8.3 Input: 14

4.8.4 Output: 14

4.8.5 Main Scenario: 14

4.8.6 Pre-condition: 14

4.8.7 Steps: 14

4.8.8 Post-condition 15

4.8.9 Representation 15

4.9 Get the Device History 15

4.9.1 Name: 15

4.9.2 Goal: 15

4.9.3 Input: 15

4.9.4 Output: 15

4.9.5 Main Scenario: 15

4.9.6 Pre-condition: 15

4.9.7 Steps: 15

4.9.8 Post-condition 15

4.9.9 Representation 15

4.10 Automated Device Control 15

4.10.1 Name: 15

4.10.2 Goal: 15

4.10.3 Input: 15

4.10.4 Output: 15

4.10.5 Main Scenario: 15

4.10.6 Pre-condition: 15

4.10.7 Steps: 16

4.10.8 Post-condition 16

4.10.9 Representation 16

4.11 Adding Intelligence to the collected data 16

4.11.1 Name: 16

4.11.2 Goal: 16

4.11.3 Input: 16

4.11.4 Output: 16

4.11.5 Main Scenario: 16

4.11.6 Pre-condition: 16

4.11.7 Steps: 17

4.11.8 Post-condition 17

4.11.9 Representation 17

4.12 Logoff 17

4.12.1 Name: 17

4.12.2 Goal: 17

4.12.3 Input: 17

4.12.4 Output: 17

4.12.5 Main Scenario: 17

4.12.6 Pre-condition: 17

4.12.7 Steps: 17

4.12.8 Post-condition 17

4.12.9 Representation 18

**5.** **Other Nonfunctional Requirements 18**

5.1 Performance Requirements 18

5.2 Safety Requirements 18

5.3 Security Requirements 18

5.4 Software Quality Attributes 18

**Appendix A: Glossary 19**

**Appendix B: Analysis Models 19**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason for Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

# **1. Introduction**

## **1.1 Purpose**

The purpose of the Software Requirement Specification is to describe the behavior and design for collecting data from smart devices in real-time. It describes the design constraints that are to be considered when the system is to be designed, and captures the software requirements for the system.

## **1.2 Document Conventions**

In this document the requirements which are underlined, Italian font are having higher priority than those which are normal. This does not mean that we are neglecting normal font requirements but it means that we are emphasizing more on underlined, Italian font requirements.

## **1.3 Intended Audience and Reading Suggestions**

## This document is intended for all individuals participating in and/or supervising Pillar – Platform for Collecting Smart Device Data (Pillar). The rest of the documents contains the overall description of the software like the environment in which the software works(hardware/software).

## **1.4 Product Scope**

Pillar is a platform that connects to all Pillar-friendly smart devices on a common network with the ability to control device states through direct user interaction or user-triggered events, and automates the collection of data of device states usable by future Pillar extensions or 3rd party software. A smart device is an electronic device connected via wireless protocols that operate, to some extent, interactively and autonomously [Wikpedia: Smart Devices]. The main objective of this project is to integrate existing tools and develop new ones to connect to devices, control their states, and track their state history relative to the user’s activities. The product will be extendable to an external engine that uses state history with predictive algorithms to predict future state with regards to expected user activities.

## **1.5 References**

References are taken from the instructions file “SRS\_Team1.docx” provided by Dr. Box.

# **2. Overall Description**

## **2.1 Product Perspective**

The main objective of this project is to develop a platform which would integrate existing tools and develop new ones to connect to smart devices, control their states, and track their state history relative to the user’s activities.

## **2.2 Product Functions**

The main function of this project is to Get/Control/Display the device state which are connected over a common network using Pillar platform

Main Requirement:

* + Pillar displaying the states of the smart devices
  + Pillar changing the state of the devices when user triggered actions
  + Pillar Tracking the device state history relative to user’s activities.
  + Pillar Controlling the current or future state of devices using the user’s preferences
  + Pillar displaying analytics of each device using the data/History stored in database

## **2.3 User Classes and Characteristics**

The Users of Pillar system are Admin (Head of the Family) and Other members. Both Users have the authority to use the Pillar System to get/control the device states and admin have an additional authority to manage the members using this system

## **2.4 Operating Environment**

The software works on Windows, Mac OS platform. For the functioning of the product the user needs to install Home Assistant software.

## **2.5 Design and Implementation Constraints**

One of the important constraint would be before running the Pillar, Configuration file has to be updated with the device access token and also all the devices must be connected to a common network to control/get the device states using Pillar.

## **2.6 User Documentation**

Pillar Team Would Prepare the documentation containing details on how a user can use Pillar, devices which can be connected to Pillar, and Present it to the user.

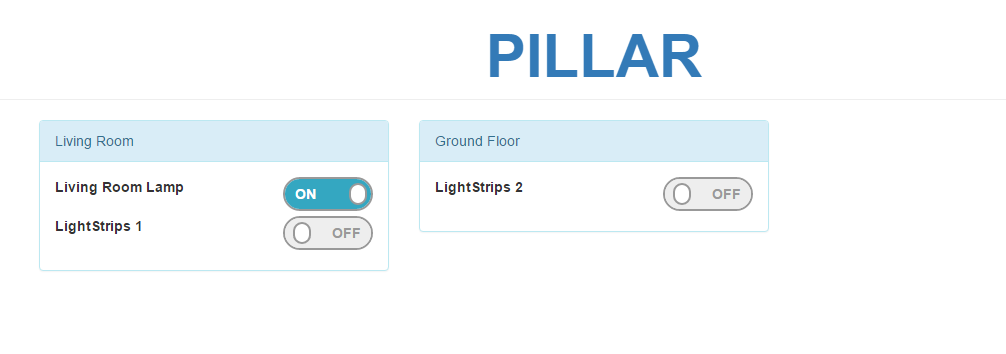
## **2.7 Assumptions and Dependencies**

Devices Displayed in the pillar are already installed and connected to Home assistant. Latest version of Python must be installed to ensure Home assistant running. Some other dependencies would be Device access token, Bootstrap.

# **3. External Interface Requirements**

## **3.1 User Interfaces**

The user interface for pillar dashboard consists of devices, which displays their states and also the user can control their states. And also it shows the hidden information from the devices history data. It can also be integrated with other applications or data including smart fridge, Fitbit, and Health data. Below is the UI of the pillar dashboard



**3.2 Hardware Interfaces**

This product runs on any device which supports a web browser such as Laptops, Mobiles, Tablets, Embedded devices (Raspberry Pi, Arduino).

## **3.3 Software Interfaces**

To run this application, we need to have a web browser. It is compatible with all kinds of browsers.

## **3.4 Communications Interfaces**

This web app communicates with the sever using HTTP protocol. We also need to have persistent connection to get real time updates from the server.

## **4. System Features (Use Case Realization Analysis):**

## We present the Use Case Realization Analysis of required functional requirements for this version 1.0 of Pillar project.

## **Registration:**

4.1.1 Name: Registration Process

4.1.2 Goal: To register user with Dashboard to provide secure authentication.

4.1.3 Input: Information provided by the user.

4.1.4 Output: User registration is successful and the information is added to the database.

4.1.5 Main Scenario: To perform registration of the user.

4.1.6 Pre- Condition: New user should have household specific token to register.

4.1.7 Steps:

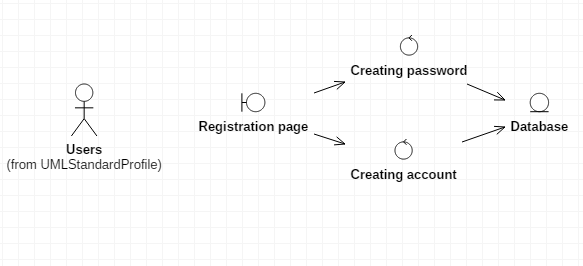
User access the signup page.

User provides registration token.

User should enter all the required details.

4.1.8 Post- Condition: User should be able to change/recover password.

4.1.9 Representation:



## **User Login:**

4.2.1 Name: Login Process

4.2.2 Goal: Enable the User to login into Pillar

4.2.3 Input: User ID and Password provided by the user.

4.2.4 Output: User Login is successful and the dashboard is visible to the user.

4.2.5 Main Scenario: To perform Login of the user.

4.2.6 Pre- Condition: User has already registered and possess User ID, Password.

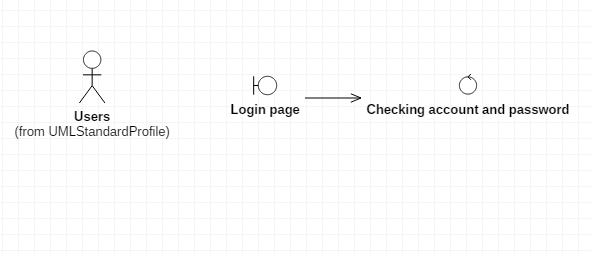
4.2.7 Steps:

User access the Login page.

User provides User ID and Password.

4.2.8 Post- Condition: User should be able to get/control device states from dashboard.

4.2.9 Representation:



## **Add device:**

4.3.1 Name: Add device Process

4.3.2 Goal: Add devices into pillar

4.3.3 Input: This device's name, icon, location and some other features.

4.3.4 Output: Add device is successful and devices information are showed on dashboard.

4.3.5 Main Scenario: Create this device's name, icon, and some other features in pillar. System can get information and state's history from device.

4.3.6 Pre- Condition: have devices name and other information.

4.3.7 Steps:

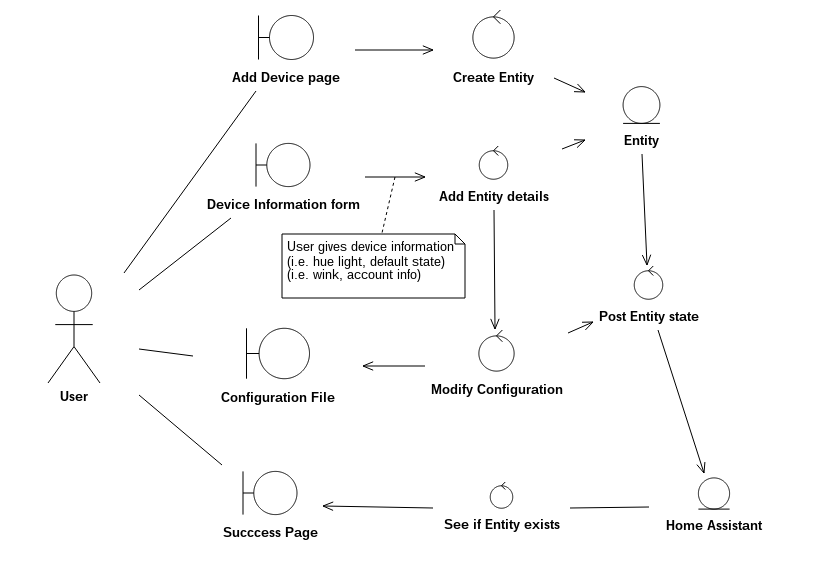
Device are connected to pillar

Create a device icon and name

Input some description and features

4.3.8 Post- Condition: Users can see and control this new device in dashboard. System can get information and state's history from device.

4.3.9Representation:



## **Remove device:**

4.4.1 Name: Remove device Process

4.4.2 Goal: Delete this device's name and features in pillar.

4.4.3 Input: device's name to search in pillar.

4.4.4 Output: eliminate information about this information.

4.4.5 Main Scenario: The device that is unconnected to the system, delete device information in pillar.

4.4.6: Pre- condition: Device's information in database.

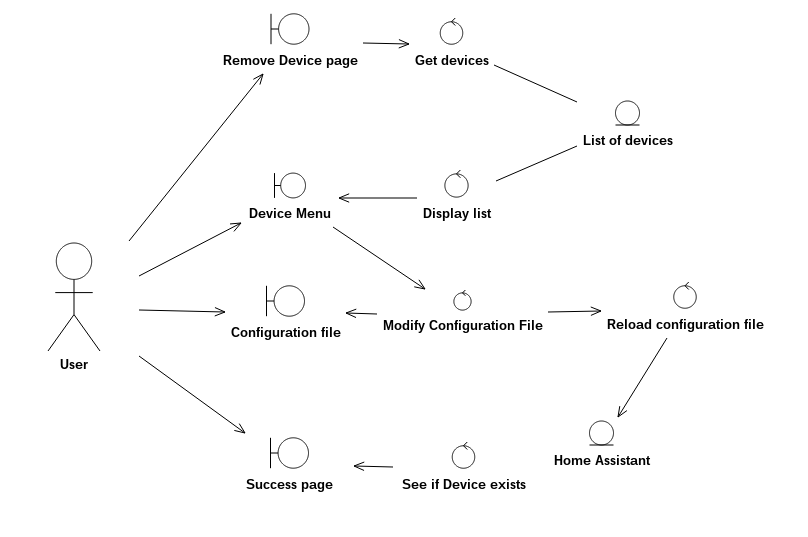
4.4.7: Steps:

Search the device's name in pillar.

Remove device's icon and features in pillar.

4.4.8 Post-condition: no this device's information in pillar.

4.4.9: Representation:



* 1. **Group devices based on the location:**

4.5.1 Name: Grouping devices based on the location.

4.5.2 Goal: To divide the devices into groups based on where they placed for easy control.

4.5.3 Input: Devices location information.

4.5.4 Output: Configuration file should be updated and devices should be displayed in groups on UI.

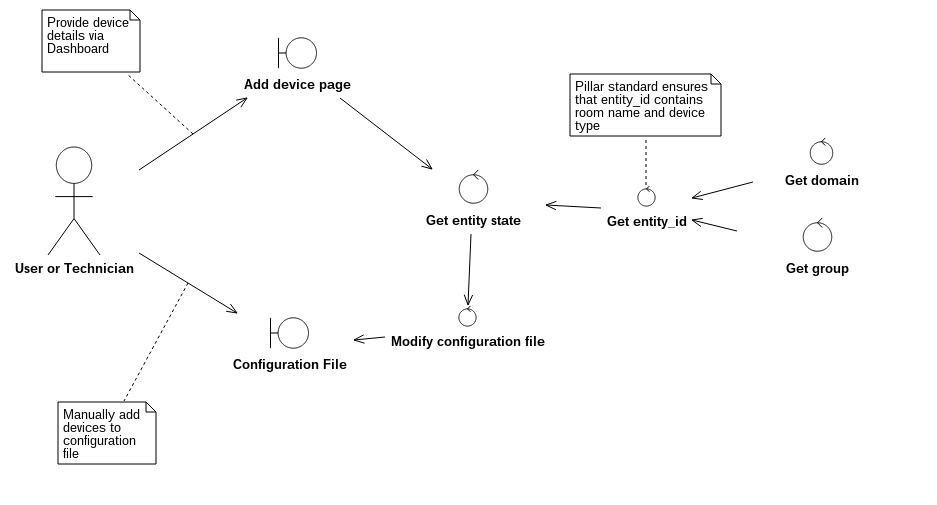
4.5.5 Main Scenario: To create tabs based on location and placing the devices in them.

4.5.6 Pre- Condition: Admin should have knowledge on YAML configuration file.

4.5.7 Steps:

Find the locations where devices placed in the house.

Change the configuration file to display tabs on UI.

* + 1. Post- Condition:
    2. Representation:
    3. 
  1. **Display device states:**

4.6.1 Name: Displaying the state information of the devices on Dashboard.

4.6.2 Goal: To get the states from the devices and display it on the Dashboard for the users.

4.6.3 Input: Device states.

4.6.4 Output: Device states should be displayed.

4.6.5 Main Scenario: To display device information on the dashboard.

4.6.6 Pre- Condition: Devices should be connected to the network and Home assistant.

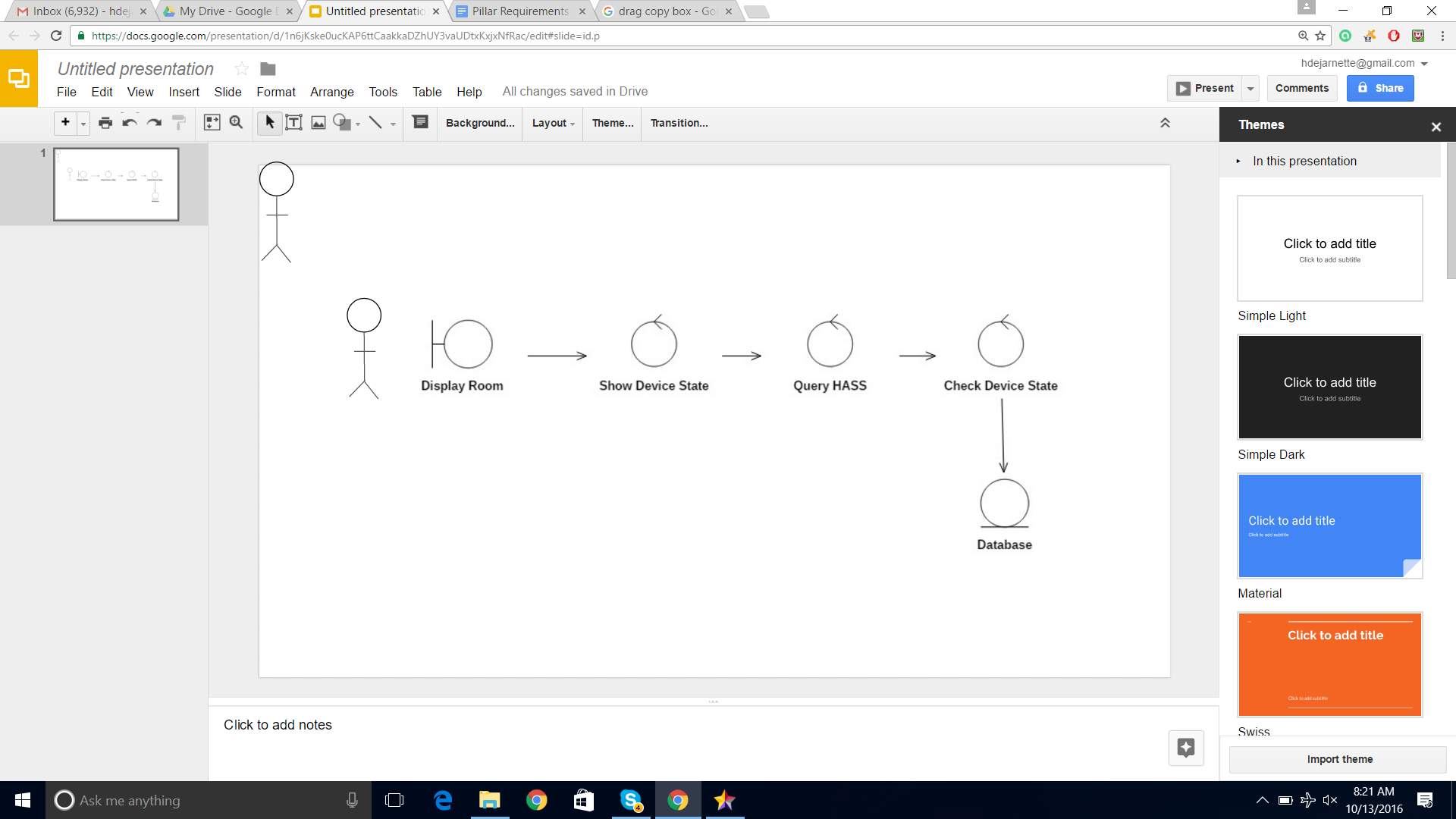
4.6.7 steps:

Send API GET request to home assistant for device states.

Display the state information that we get on the dashboard.

4.6.8 Post- Condition:

4.6.9 Representation:



* 1. **Control device states:**

4.7.1 Name: Control device states

4.7.2 Goal: Users or system can change device states by pillar.

4.7.3 Input: Users change states on dashboard when environment parameters changed.

4.7.4 Output: devices' states have been changed

4.7.5 Main Scenario: Change the devices' states. Make indoor environment more comfortable for users.

4.7.6 Per-Condition: the devices' states not change.

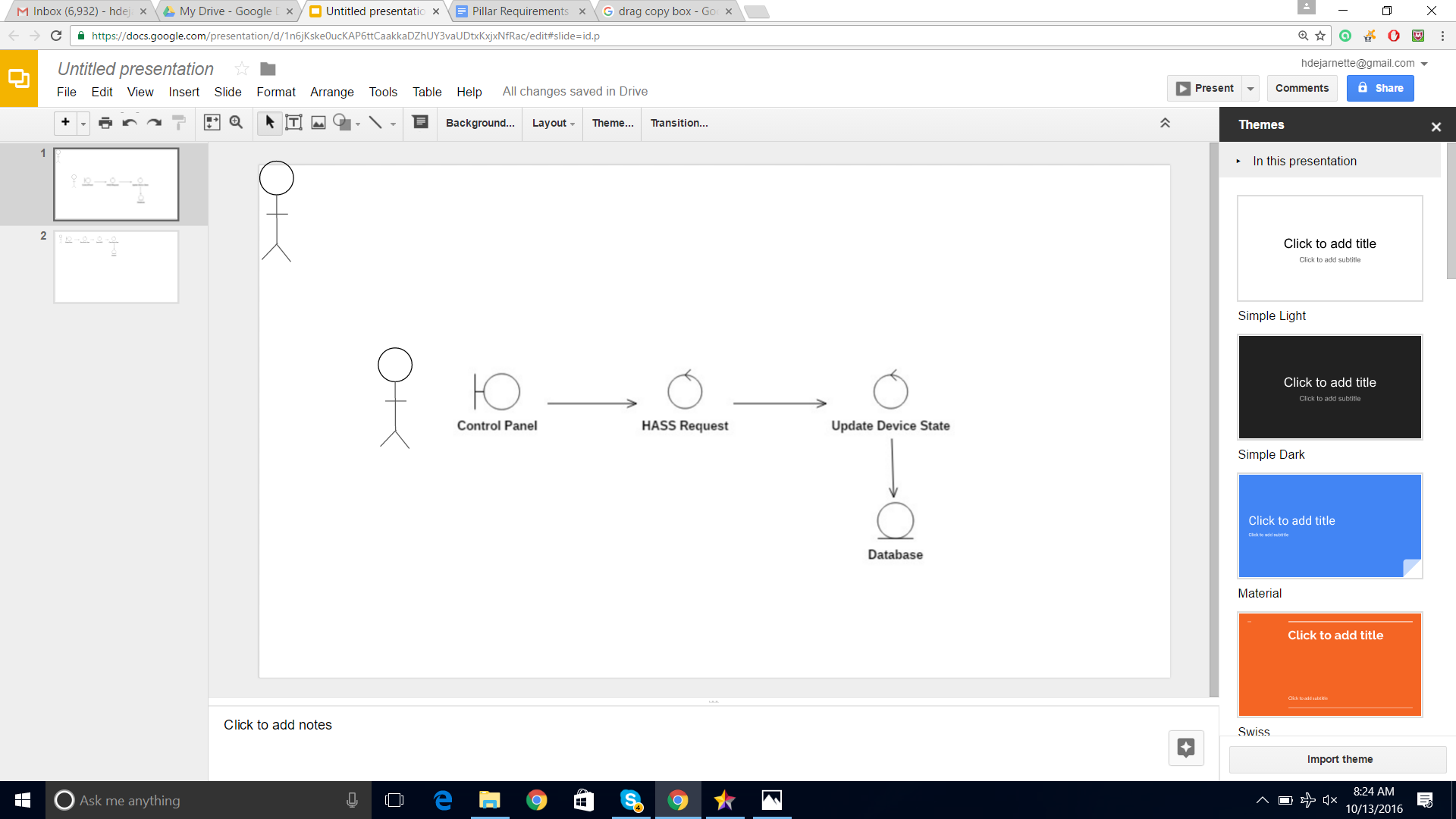
4.7.7 Steps:

User change devices' states on dashboard or system change device states when environment parameters changed.

Devices states changed.

4.7.8 Post- condition: Devices states are changed

4.7.9 Representation:



* 1. **Get the device states and store it in the Database:**

4.8.1 Name: Device State Acquisition & Storage.

4.8.2 Goal: To retrieve device states and store them within the database.

4.8.3 Input:  States of devices.

4.8.4 Output: Device states are stored in database

4.8.5 Main Scenario: To provide state data for storage or for user

4.8.6 Pre- Condition: Device should be registered to list

4.8.7 Steps:

Home Assistant accesses device list.

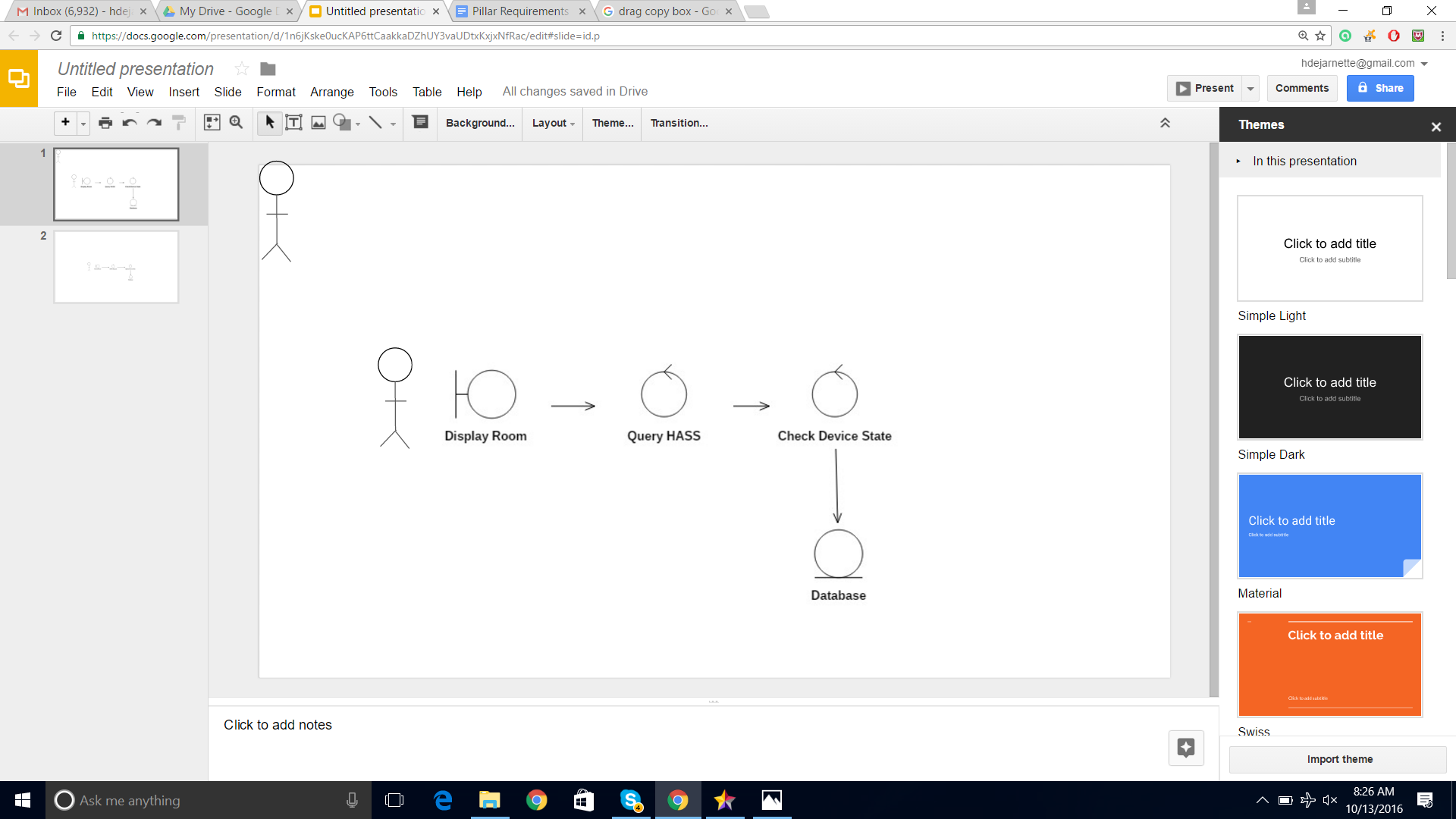
Home Assistant accesses device state.

Pillar grabs device state from Home assistant.

Pillar stores device state in database.

4.8.8 Post- Condition: States have been added to database

4.8.9 Representation:



## **Get the device history:**

4.9.1 Name: Device state history retrieval.

4.9.2 Goal: To retrieve prior device states from within the database.

4.9.3 Input:  Device state history request.

4.9.4 Output: Prior device states

4.9.5 Main Scenario: To provide prior state data for analysis or behavior.

4.9.6 Pre- Condition: Prior states should be acquired and stored

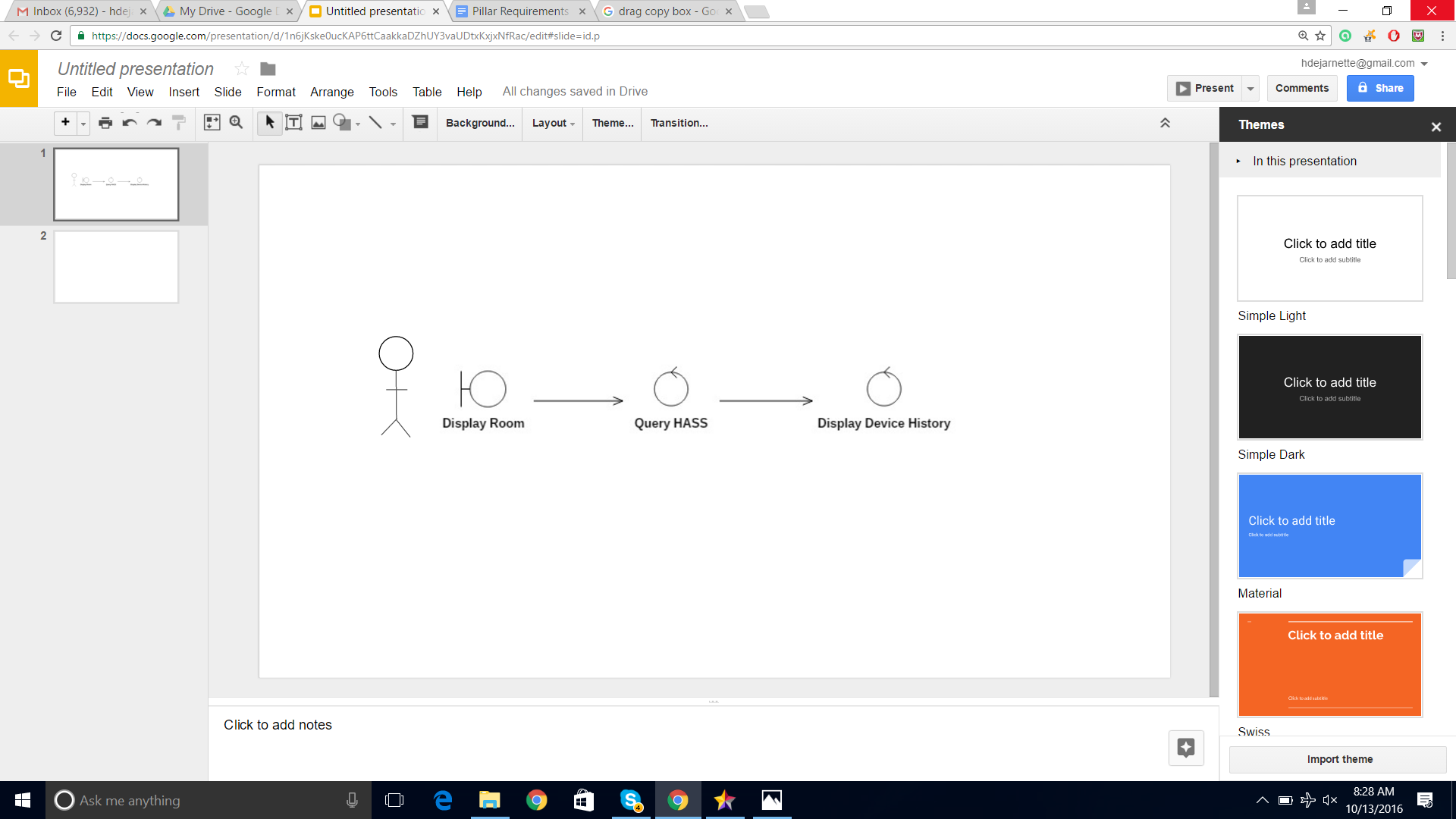
Steps:

Pillar accesses device database.

Pillar retrieves device state history.

4.9.7 Post- Condition: Prior device states available for analysis or behavior.

4.9.8 Representation:



## **Automated device control:**

4.10.1 Name: The Great Pillar Intelligence

4.10.2 Goal: Control the current or future state of devices based on previous user activity/input and current device environment.

4.10.3 Input: Device history or results of analysis performed on device history

4.10.4 Output: A JSON encoded message of the expected, current state of each device

4.10.5 Main Scenario: Change state of device to meet user’s expectations without immediate user input.

4.10.6 Pre-Condition: A sufficient amount of history has been collected to use on existing, predictive algorithms.

4.10.7 Steps:

Collect device history

Run predictive algorithms on device history and environment

Compare device’s current JSON contents to predicted JSON contents.

If the two JSON contents don’t match, update JSON contents

Change state accordingly based on JSON contents

4.10.8 Post- Condition: Watch to see if user changes state to overrule prediction and update algorithms accordingly

4.10.9 Representation:

## Model3__ClassDiagram1_6.png**4.11 Adding intelligence to the collected data:**

4.11.1 Name: Analytics

4.11.2 Goal: Show user analytics of device history

4.11.3 Input: User request to view analytics of device history

4.11.4 Output: A display of analytics of each device in a format appropriate to the units and analysis

4.11.5 Main Scenario: To show user analytics from analyses performed on device history

4.11.6 Pre-Condition: Sufficient history exists to produce analytics

4.11.7 Steps:

User selects Analytics from a menu

The Dashboard finds data previously flagged to be displayed in Analytics

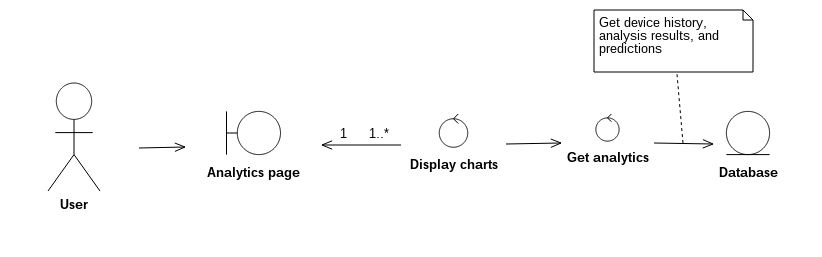
The Dashboard displays selected data

User can select whether or not they found a particular dataset useful

User exists Analytics

4.11.8 Post- Condition: Pillar updates flagged Analytics based on user’s updated selections

* + 1. Representation:



## **4.12 Logoff:**

4.12.1 Name: Logoff Process

4.12.2 Goal: Enable the User to logoff from Pillar

4.12.3 Input: User Clicks the logoff button.

4.12.4 Output: User Logoff is successful.

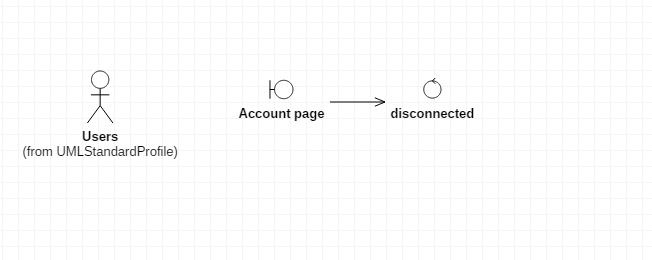
4.12.5 Main Scenario: To perform Logoff of the user.

4.12.6 Pre- Condition: User has already Login into the Pillar

4.12.7 Steps:

User Clicks the Logoff button.

4.12.8 Post- Condition: User should be able to see the Login Page



**5. Other Nonfunctional Requirements**

## **5.1 Performance Requirements**

Pillar should have low latency with regards to user actions and provide an interface that is responsive to user interactions.

**5.2 Safety Requirements**

All devices should be installed as stated in the documentation specific to each device. No documentation by pillar is given on how to physically install devices.

**5.3 Security Requirements**

All communications between Pillar components and external actors cannot reveal information that invades on user privacy. Any data collected by Pillar must be encrypted using information known only the user and distribution of data to 3rd party is done only with explicit consent from the user with minimal invasion of privacy.

**5.4 Software Quality Attributes**

The software should be adaptable to any well supported, 3rd party accessible devices and should not require any technical understand of pillar to use all features. It should be adaptable to any network environment, available to all household members, and provide an intuitive UI for the convenience of the users. The life expectancy of this software should not be shorter than any of the devices it is compatible with.

**Appendix A: Glossary**

HTTP – Hyper Text Transfer Protocol

WAMP – Cross Platform Software

Eclipse – Integrated Development Environment

**Appendix B: USE-CASE Model**

