Requirements Document

The smart home project

Cronus

Version 7.0

Client: M.F. Lungu, A. Karountzos

C. Ausema,
S. Evanghelides,
L. Holdijk,
M. Helmus,
S. Maquelin,
F. te Nijenhuis,
F. Ritsema,
S. Zijerveld

Introduction

There are a lot of different platforms for home automation currently available. Most of them however have quite a few problems. One of the main problems is the way in which these platforms limit themselves to only work with a certain set of devices. Even when the platform is able to work with a wide variety of devices, adding new ones still requires the developers of the platform to manually update it. A good example of this is the struggle most platforms have with the new Ikea smart lighting system [1]. All, or at least all well known, platforms currently available require an update to handle the smart hub hardware used for these lights.

With the Hestia system we strive to provide a platform that can extend to new types of devices without requiring structural changes. In the end it should be possible for consumers to create their own plugins for the system. Using these plugins the Hestia platform will also be able to control privately developed peripherals.

The goal of this project is to develop an easily extendable, secure platform that allows our user to control and setup their home automation in a breeze. All devices that have a way of being remotely controlled should be able to work with the system. This includes, but is not limited to, locks, lights and refrigerators.

To prevent the system to overshoot its goals and become unusable for the average consumer, we defined some scenarios that are built around different types of customers. These can be found in the appendix of this document.

The system overview

The Hestia platform is built from three different components; the client, the server and the peripherals. A peripheral is any device which can be remotely controlled using the Hestia system. The client and the peripherals are centered around the server and depend on it for communicating with each other.

Using the client, the user can request actions that a peripheral should perform. To use a peripheral it should first be added to the platform. One of the main strengths of the Hestia platform should be that it is easy to create plugins, these plugins are specific for each kind of peripheral and define a way to interact with the peripheral. It should be possible for people outside the development team to define these plugins for their peripheral. The user of our platform should then simply be able to select a plugin and add it to the system to communicate with the peripheral.

All information regarding the plugins and peripherals is managed in the server. This is where the clients send its requests. The server is responsible for translating and forwarding the incoming requests to the peripherals.

---

[1] http://www.ikea.com/gb/en/products/lighting/smart-lighting/

Users and their stories

The requirements for the Hestia project are described in the form of user stories. Each user story follows a predefined format, first stating which types of users the story belongs to. Currently we have three different types of users, however as the system grows in the future with new user stories the amount of different user types may also increase. After the user type the user story states what the user wants the system to be able to do. This is followed by a short reason why the user wants this to be possible. Below a short introduction about the different users. Scenarios for each different user can be found in the appendix.

The user: **[scenario 1, 2 and 3]**
This is the normal user type. All stories linked to this user describe basic features of the system that would benefit every possible user. User stories from this user will therefore never be really technical, but are often about how he or she wishes to interact with the system. Most of the user stories for the Hestia platform are related to this user. This shows that the Hestia platform is a system designed to be operated by the normal user.

The client side developer: **[scenario 5]**
The Hestia platform consists of three different components, two of which are the client and the system. The user stories linked to the client side developer user are often about the connection between the client and the server.

The plugin developer: **[scenario 4]**
One of the big advantages of the Hestia platform is the possibility for users to use the system in combination with their own peripherals. To do so they should be able to write some basic code and add this to their own setup. The user stories linked to this user describe how one should be able to add their own peripherals.

Ordering of the user stories

User stories are divided into 5 sections. Firstly all user stories that were suggested by the customer but will not be included in any version of the platform are placed in the *“User stories that will not be implemented”* section. Secondly user stories that describe a non functional requirement are placed in the *“Non functional user stories”* section.

All other user stories are divided based on urgency in respectively the sections “*Critical*”, “*Important*” and “*Useful*”. User stories in the critical section are vital for the functioning of the platform. These user stories will be included in the Minimal Viable Product (MVP) of the project in combination with the user stories in the important section. User stories in the useful sections might also be in this version depending on whether or not there is time left.

User stories

Critical user stories

- ✓ As a user, I would like to be able to select actions of peripherals to perform so that I can interact with peripherals in my house.
    - ✓ *As a user, I would like to be able to see all the peripheral in the system so that I know which peripherals I can interact with.*
    - ✓ *As a user, I would like to be able to see what actions can be performed by each peripheral to know in which way I can interact with the peripheral.*
    - ✓ *As a user, I would like to have a graphical representation of the actions that reflect the type of action it performs to make interacting with the peripherals more intuitive.*
- ✓ As a user, I would like to be able to set the IP address at which the server can be found, to allow the client to communicate with the server.
- ✓ As a user, I would like the system to contain plugins for Philips hue lights to allow me to interact with them.
    - ✓ *As a user, I would like the system to have a peripheral defined for all Philips hue lights that can switch on/off or change the brightness so that I can interact with lights that have these functionalities.*
    - ✓ *As a user, I would like the system to have a peripheral for all Philips hue lights that can switch on/off, change the brightness or change color so that i can interact with lights that have these functionalities.*
- ✓ As a user, I would like to be able to add a new peripheral to the system so that I can interact with it.
    - ✓ *As a plugin developer, I would like to be able to give a list of information that is required to interact with the peripheral, so that I can accurately represent the peripheral as a device in the system.*
    - ✓ *As a user, I would like to be able to select which peripheral I want to add by stating the collection to which the peripheral belongs and the name of the plugin associated with the peripheral, so that I know which peripherals can be added to the system .*

Important user stories

- ✓ As a user, I would like to make sure that I am the only one who can use my phone to interact with the peripherals, so that no one with access to the phone, besides me, is able to interact with the peripherals.
  - ✓ *As a user, I would like to be able to set a username and password that can be used to log in to make sure that only I know the username and password and not anybody else.*
  - ✓ *As a user, I would like to be able to indicate that the client should remember my login credentials to allow me to use the application without the need of entering my credentials every time.*
- ✓ As a user, I would like to be able to name a peripheral to distinguish between different peripherals with the same functionality.
  - ✓ *As a user, I would like to be able to give a name to the peripheral upon adding it to the system to make it easier to see which peripheral I have just installed.*
  - ✓ *As a user, I would like to be able to change the name of a peripheral already added to the system to make it possible correct mistakes in naming.*
- ✓ As a user, I would like to be shown all possible collections and plugins whenever I want to add a new peripheral to my system, to make it easier to add new devices.
- ❏ As a user, I would like to see the current state of my peripherals so that I can know the current state of a peripheral without it being in my sight.
  - ✓ *As a user, I would like the state of a device to be retrieved from the peripheral instead of having it retrieved from the memory, to prevent the state from being altered from somewhere else without noticing.*
  - ✓ *As a user, I would like the actual state of the peripheral to be retrieved when it is added, to prevent the state from not being correctly displayed.*
  - ❏ *As a user, I would like to be notified about the result of my actions to prevent me from expecting the peripheral to be in a certain state while it isn't.*
- ❏ As a user, I would like the system to contain plugins for my IKEA smart lighting to allow me to interact with them.

Useful user stories

- ✓ As a user, I would like to be able to keep my server up to date with the latest development version to make sure all bug fixes are solved in my system as soon as possible.
- ✓ As a user, I would like to be able to restart my server without losing all my installed peripherals, preventing me from reinstalling all peripherals whenever my server requires a restart.
- ✓ As a user, I would like to able to see a description of each required info field, so that I can install the peripheral without knowing the details of every peripheral.
- ✓ As a plugin developer, I would like the code for peripherals to be centered around a configuration file, so that I can easily add new plugins without having to write a lot of code.
    - ✓ *As a user, I would like to have all the new plugins visible in the app when I add a new plugin, so that I can use my new plugin.*
    - ✓ *As a plugin developer, I would like to be able to set the properties of a new plugin using a separate configuration file for each plugin, maintaining a clear structure of the collections and plugins.*
- ✓ As a client side developer, I would like to be able to send the required information in a JSON object to make the communication consistent with REST practices.
- ✓ As a client side developer, I would like to have a distinction between the plugin information and required information, so that I can easily distinguish what field a user has to fill in.
- ✓ As a user, I would like to have a more convenient name for the grouping of the plugins, so that they do not have to be from a specific organization.
- ✓ As a user, I want to be able to automatically connected to the server if it is on the same network so that I do not need to know and type in the ip-address.
- ✓ As a user, I would like the communication between the server and the client to be secure, so that nobody else can use my devices.
- ✓ As a user, I would like to be able to install the server by retrieving a bash script using the *wget* command so that I can easily install and run the server on a headless device.
- ❏ As a user, I would like to be able to install/update the Android application from the Google PlayStore, so that it is easy to install/update.
- ❏ As a user, I would like to be able to operate the Android application using voice controls so that i don't have to take out my phone to operate my devices.
- ❏ As a user, I would like to be able to schedule my peripherals to perform certain actions at set times so that I can give the impression that I am at home and I do not have to repeat this manually every time.
- ❏ As a user, I would like to be able to see the numeric value I set with the slider so that I can accurately set the slider's position.
- ❏ As a plugin developer, I would like to have some safeguards in place while defining a configuration file for my plugin so that in the case my plugin is not working I can easily find the reason for this.

User stories that will not be implemented

- **X** As a client side developer, I would like to communicate with the server over bluetooth.

Non functional user stories

- ✓ As a client side developer, I would like the request to the server and the response from the server to be made over the internet to allow a big range of electronic devices to be used as a client.
- ✓ As a plugin developer, I would like to have the possibility of defining a new plugin without changing the framework, so that I can easily create a new plugin without being part of the development team.
- ✓ As a client side developer, I would like the server to provide an interface to communicate with that is independent of the client to allow for different electronic devices to be used as a client without changing the server.
- ✓ As a user, I would like for the failure to perform an action on a peripheral to not block further usage of the application and server to prevent the platform from breaking down when one peripheral is not properly defined.
- ✓ As a user, I would like the application to allow me to perform all actions on a peripheral in at most three presses when I am on the home screen (the screen that shows me all peripherals I can interact with), to limit the time needed to perform an action.
- ✓ As a user, I do not want to be dependent on a specific external database (such as mongoDB), so that I am not dependent on certain operating systems for running my Hestia server on the raspberry pi[2].

[2] Most Raspberry Pi's require a 32-bit Operating System, whereas MongoDB requires a 64-bit one. While it is possible to use MongoDB on such Raspberry Pi, it is not easily setup. Same holds for other external databases.

Appendix

Scenario 1: Mr. Smit
Mr. Smit is a busy executive. He is often working late at the office and not coming home before dark. He has an apartment in the city of Amsterdam where he lives on his own. During the last month, several neighbors homes have been broken into. Mr. Smit does not want this to happen to him and likes to turn on the lights at six. Therefore, he goes to the Hestia app on his phone and logs into the app. He chooses his light in the living room and presses on at six. He can now stay at the office until all the work is done.

Scenario 2: Mrs. Bakker
Mrs. Bakker is an old lady of the age 82. She still lives on her own, but has a hard time walking. Mrs. Bakker got a smartphone from her children for her birthday and has since used the Hestia app. When her grandchildren ring the door, she does not have to get up, but simply uses the app to presses on door and then presses unlock. This way, she does not have to go to a retirement home, for which she is very grateful.

Scenario 3: Mr. Techy
Mr. Techy is a man of the future. He has all the new gadgets out there. All his lights, the oven, fridge, even his shower is an IOT device which can be controlled with an app. Before he had the Hestia app he had a different app for every device. This meant that most of the time he was searching for the right app instead of enjoying all the benefits of his smart home. With the Hestia app he has one app that neatly organizes all his devices. And living in a smart home full of IOT devices finally gave him the benefits and joy he hoped he would get.

Scenario 4: Mr. Techy Junior
Mr. Techy Junior has, just like his father, a house filled with IOT devices. However, he did not buy those devices, but created them all himself. He can control all these devices from his command line on his laptop, but he would also like to interact with these devices using his android phone. Since all current home automation systems only work with out of the box IOT devices, he hasn't been able to do this yet. After learning about the Hestia platform from his dad, he is finally able to solve his problem without learning about android development himself.

Scenario 5: Ms. iCarly
Ms. iCarly is a big apple fan girl. In order to create an apple shrine in the original colors of the apple logo, she decided to buy some Philips hue lights. Half way through the process of buying the Philips hue lights, apple releases the new iPhone, almost emptying her whole bank account. To finish her shrine she decides to switch to the cheaper IKEA smart lighting system. She could not find any way of connecting the two different lighting systems in a single iOS app. Luckily she knows someone that just used the Hestia platform to link all their own IOT devices. Ms. iCarly decides to learn some iOS programming and she creates her own client for the Hestia system.

Customer meetings

When	What
May 3, 2017	- Discussed the priority of the backlog with the customer. - Focussing more on the GUI and interaction with the Client, all related user stories were move up on the backlog. - Persistency of the database and workflow of the app are indicated by the customer as most important. Related user stories are selected for the next sprint. - Also, the state of a device in the app should be consistent with the state in the real world. - Security and plugin management (adding new types of peripherals) were indicated as less important.
May 24, 2017	- The customer indicated that deployment of the app should be researched. This was already in the requirements document but we moved it to a higher priority spot in the backlog. - For the numeric activators the customer wanted to see the exact state. This didn't have too much priority for them, so we added a user story but have not selected it for the next sprint. - The customer wanted safeguards for the configuration files in the server, this will cost a lot of time that is currently not available. Together with the customer it was decided to put this at a low priority.
June 7, 2017	- The customer showed interest in the having a secure connection between the Android App and the server. Eventually agreed that the current security level is good enough for now. - We brought up the point of automatically discovering the servers IP on the local Network. The customer was interested and wanted to see it implement asap, therefore the feature was added in the current sprint. - Agreed on a open Beta for publication of the android application. - Customer expressed the desire to have the option to install the server using a bash script that he could retrieve from the command line using wget. This requirement was added to the current sprint. - Urgency of user management is lowered on request of the customer - Voice control for the app was given a higher priority but will not be implemented by us.

Changelog

Who	When	Which section	What
L. Holdijk, S. Maquelin, F. Ritsema	February 22, 2017	The document	Created the document, added features.
S. Maquelin	February 27, 2017	All sections	Split up features into 3 lists. Move some functional features into non-functional features
S. Maquelin	March 5, 2017	Peripheral device, critical	Feature ‘notify server when the door is locked/unlocked manually’ moved from critical to useful section
S. Maquelin	March 5, 2017	Phone, useful Server, useful	Feature ‘set-up bluetooth connection between phone and server’ moved from useful to won’t do section
S. Maquelin	March 5, 2017	Server, important Peripheral, important	Feature ‘send encrypted messages between server and peripherals’ moved from important to won’t do section
S. Maquelin	March 5, 2017	Introduction	Improve introduction
S. Maquelin	March 5, 2017		Add logo
S. Maquelin	March 5, 2017		Add images GUI
S. Maquelin	March 6, 2017	Phone, useful	Feature ‘perform actions through app on smartwatch’ moved to non-functional

S. Maquelin	March 6, 2017	All sections	Divide non-functional requirements between 3 sections.
S. Maquelin F. te Nijenhuis	March 9, 2017	All sections	Updated document according to comments.
S. Zijerveld	March 21, 2017	All sections	Updated document according to comments and changed situation.
L. Holdijk	April 10, 2017	All sections	Updated list of requirements. Cleared up ambiguity in introduction and system structure.
L. Holdijk	April 11, 2017	Requirements	Changed the requirements to better reflect the customer's wishes.
L. Holdijk	April 11, 2017	Introduction	Removed section that covered the architecture of the system.
Stefan Evanghelides	April 13, 2017	All sections	Minor updates; Removed ambiguity.
F. te Nijenhuis	April 13, 2017	All sections	Minor modifications, changed some unnatural sentences.
S. Zijervled	April 13, 2017	Server, Important	Added sensors
Lars Holdijk	May 1, 2017	User stories	Translated all requirements to user stories.
Lars Holdijk	May 1, 2017	User stories	Added user stories based on the latest customer meeting.
Lars Holdijk	May 1, 2017	Users and their stories	ection as replacement of old Requirements section.
Lars Holdijk	May 1, 2017	Scenario 4 and 5	Created them
Stefan Evanghelides	May 1, 2017	All parts	Refactored some parts and provided feedback for others.

Stefan Evanghelides	May 29, 2017	All parts	Ticked and updated stories, based on the status of our project and feedback received
Lars Holdijk	May 29, 2017	Useful user stories	Added two user stories.
Suzanne Maquelin	June 11, 2017	Useful user stories	Added two user stories
Stefan Evanghelides	June 11, 2017	Overall	Checked 2 useful user stories and add the last customer meeting (June 7, 2017)
Lars Holdijk	June 12, 2017	User stories	Added two user stories based on last customer meeting
Lars Holdijk	June 12, 2017	Customer meeting	Added additional information