Snow White

# Software Requirements Specification

Version 1.0

## Revision History

| Date | Version | Description | Author |
|---|---|---|---|
| 22.10.16 | 1.0 | First version of this document | Cem Philipp Freimoser<br>Mario Kunstek<br>Marc Weickenmeier |
| | | | |
| | | | |
| | | | |

## Table of content

## 1.0 Introduction

### 1.1 Purpose

*Snow White* is a software stack which enables you to build your own smartmirror by providing all necessary software and a nice user interface for the mirror.
The user interacts with the mirror by using his voice only. Snow White will be able to differentiate between multiple user and provide each user his own personal space. As a result the mirror can be used by more than one person but only one person at time.

### 1.2 Scope

- There is one default user, which will be displayed as long as no user is logged in
- Every user (except of the default user) has his own UUID, which will be stored local as well as on the Microsoft Voice Recognition server
- Each user has his own locally stored configuration, which described what should be displayed as well as account information (for example Google login data)
- The mirror comes with a configuration website which will be reachable only in you local network. Over this site the user is able to create a new user or change configuration for one user. Each user has his own password for protecting privacy

### 1.3 Definitions, Acronyms, and Abbreviations

| | |
|---|---|
| User | A user is one person which interacts with our software. He can be identified by his voice, an UUID or with his username and user password.<br>**Special case:** the default user can not be identified by voice, or UUID. Also the default user has no password. |
| Smart Mirror | A smart mirror is a mirror, which can display information on itself. It is also able to decide which information should be shown and which not |
| Voice Recognition | computer analysis of the human voice, especially for the purposes of interpreting words and phrases or identifying an individual voice[1] |

[1] https://en.wikipedia.org/wiki/Speaker_recognition

## 1.4 References

- JavaFx - http://docs.oracle.com/javase/8/javase-clienttechnologies.htm
- Play Framework - https://www.playframework.com/
- Manual for building a smart mirror - https://glancr.de/smart-mirror-selbst-bauen/
- Microsoft Speaker Recognition - https://www.microsoft.com/cognitive-services/en-us/speaker-recognition-api
- Microsoft Speech Recognition - https://www.microsoft.com/cognitive-services/en-us/speech-api
- UpdateNode - http://www.updatenode.com/

## 1.5 Overview

Snow white is all about giving you the best software for your own smart mirror. Most software implementation of smart mirrors doesn't come with multi user support. We want to end this! Snow white's core feature is multi user support, which enables you to share the mirror with the whole family.

Furthermore snow white empowers the user to get his personality on a mirror, by providing data from social media and custom data providers.

# 2.0 Overall Description

On this chapter we are going to give you closer look to the functionality of snow white.

## 2.1 Product functionality

By default the the software shows non-personalized information. This includes something like weather,  and so on. In this mode you are called default user. As soon as you use your voice and say you phrase the system will recognize that and log you in. After your are logged in you will see personalized information. This can include you facebook, google+ etc.  feed. Now you are able to interact with your voice. There will be some commands you can say to show some information you’re interested in. In order to enable a comfortable and easy way for configuration you can reach a configuration website in you local network. Here you can create new users, configure your data feed or reorder the user interface of the mirror.

## 2.2 User characteristics

There are some commercial smart mirrors in the world. But we do not expect that a user has experience with a smart mirror. Therefore we want to create a software which feels natural and don't force the user to learn something technical new. Using the voice for authentication feels natural and doesn't require the user to remember touch the mirror.
When the user use the mirror we don't expect him to want to send emails or something like that. We expect that the user just want to get an update on information of his interested. This may include post from friends on social media or his appointments. We call this "passive usage".

The design of the user interface comes with lot of transparency. By default no static information will be shown on the center. So the main functionality of the mirror still works (see yourself). The whole layout is based on grids. The design is not final yet, we will provide mockups and description as soon as we have a final specification for the design.

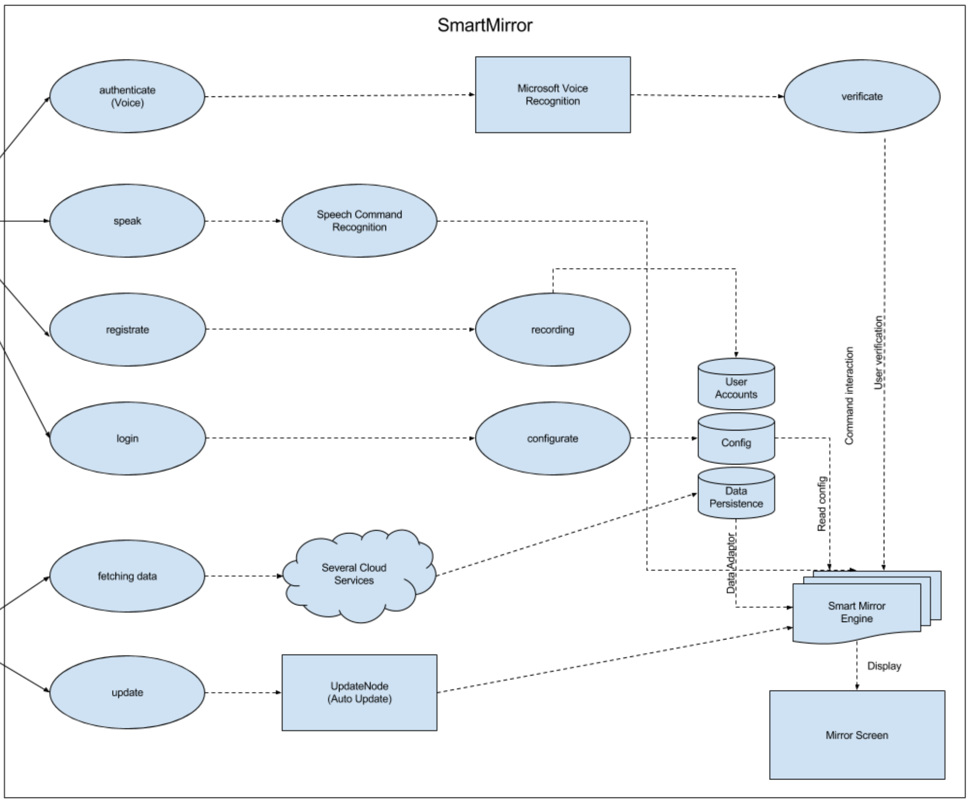
## 2.3 Constraints

The developers have limited time. By the end of the summer semester 2017 the project needs to be finished.  Five use cases will be finished by the end of the winter semester 2016. Following use cases are implemented this semester: *authenticate, registration, login, fetch data and update.*

## 2. 4 Assumptions and dependencies

- IDEA: Intellij and Eclipse
- Programming Language: Java, HTML, JS
- Microsoft Cognitive Services
- UpdateNode
- More coming...

## 2.5 Overall use cases diagram



## 3.0 Specific requirements

### 3.1 Functionality

#### 3.1.1 Authenticate

The user authentication includes anything from recording an audio sample, handling the result from microsoft's API and also load the personalized space for the user.

#### 3.1.2 Speak

Like authentication speak includes anything from recording the audio to response on the voice command. A user should be able to say something like “What's the weather like in New York?” to get the desired information.

### 3.1.2 Registrate

Registration of new user will be doable in the configuration page. A new user has to create a password for the configuration page. Furthermore the user has to train his voice model. This can be done at the mirror (Sadly we can't allow the user to record some voice sample with his smartphone or pc microphone due to the fact that microsoft recommends to use the same microphone for training and verifying)

### 3.1.3 Login

Login just means to login on the configuration page. Like your home router this can be done by a default password for all users (each user space is protected by the user password).

### 3.1.4 Fetching Data

In order to provide the user with his personal information from social media or online accounts. We use the process of data fetching. It is triggered by the system periodically to be up to date.

### 3.1.5 Update

Update is the process of software updates. Like Google Chrome, Snow White will update automatically.

## 3.2 Usability

Now we are going to explain some usability requirements

### 3.2.1 User-Interface

#### 3.2.1 User-Interface-mirror

The mirror only requires one sentence from the user to log in. After that no more interaction is required.

#### 3.2.1 User-Interface-configuration-page

Thee configuration page reminds the user on its router. We will use minimalistic design to provide the best usability.

## 3.3 Reliability

Snow white is a always on software. The user can interact with snow white at any time he wants. If no user is logged in the default user is displayed to provide basic information such as current time and weather. As a result the user can always rely on snow white.

## 3.4 Performance

### 3.4.1 Speed

In order to provide the fastest login experience as possible. Snow white pursuing a always listing strategy. So no extra step is required to get your personalized information shown. Snow white also storage your personal data locally so no fetching from online services (at the time of usage) will be required. This in mind we are able to give the user fastest “ready to start” experience we can think of.
One drawback of our dependencies will slow this up. In order to find out which user is talking we need to verify all known profiles (Microsoft is going to say which profile is verified). As a result the speed of our login depends on the response time of microsoft servers. *We will use as much concurrency as possible to verify all profiles*.

### 3.4.1 Animation

We want to provide a nice user interface with a lot of meaningful[2] animation. But we also want to build on top of low hardware requirements. So we need to optimize. Each animation should be as smooth as possible.

## 3.5 Supportability

no statement can be made at this time of development

## 3.6 Design Constraints

Not yet determined

## 3.7 On-line User Documentation and Help System Requirements

no statement can be made at this time of development

[2] Check out material design guidelines

## 3.8 Purchased Components

no statement can be made at this time of development

## 3.9 Interfaces

### 3.9.1 User Interfaces

#### 3.9.1.1 Mirror-Interface

By default the middle area of the mirror should be clear of any information, in order to enable the user to reflect himself[3].
The whole mirror will divided into a grid. Each grid cell can hold a blox. Each blox can hold information from the same kind. *We are going to update this requirement as soon as we are able to provide some mockups.*

As technology JavaFX is used.

#### 3.9.1.1 Configuration-Interface

The configuration page remind on a router configuration page (fritz box).

As technology Play Framwork is used.

### 3.9.2 Hardware Interfaces

no statement can be made at this time of development

### 3.9.3 Software Interfaces

Mostly snow white will communicate with microsoft cognitive services. But also other apis for data fetching will be used but can not be specificat yet.

### 3.9.4 Communications Interfaces

no statement can be made at this time of development

## 3.10 Licensing Requirements

no statement can be made at this time of development

[3] Actually the middle area is used for hot-information. This means short animation are allowed to be displays on this area.

### 3.11 Legal, Copyright, and Other Notices

no statement can be made at this time of development

### 3.12 Applicable Standards

no statement can be made at this time of development

## 4. Supporting Information

None so far