Design Documentation – Free choice group

HomeDork – Interactive Smart House

Revision History

Name	Associated Letter
Fanny Söderlund	A
Malek Alabed	B
Nishat Jahan	C
Suzanne Zomer	D
Ismail Eyamba	E

Date	Version	Description	Author
22/09/2021	1.0	Initial discussion of design	A, B, C, D
06/10/2021	1.1	Design of mock lab	A, B, C, D
20/10/2021	1.2.0	Additions of figures 2 and 3	B
21/10/2021	1.2.1	Addition of D4 and complementing figure 4	A
23/10/2021	1.2.2	Rearranging text, check spelling and details	A, B, C, D, E
27/10/2021	1.3.0	Grammar revised	E
14/11/2021	1.3.1	Changes in document formatting such as versioning, tables, and titles according to group standards.	A
14/11/2021	1.3.2	Addition of D5, table of figures, and designs related to R1 and R8	A
15/11/2021	1.3.3	Addition of designs related to R2 and R3	B, C

Design item List

Design name	Requirements related	Priority
D1. Client android app environment	R1, R2, R3	Essential
D2. Client web app environment	R1, R2, R3	Essential
D3. Server/API connection	R2, R4, R5, R6, R8	Essential
D4. Home page designs	R4, R5, R7, R8, R9	Desirable
D5. Settings designs	R1, R2, R3, R6, R10	Desirable

Table of figures

Design Item Descriptions

D1.

A collection of the designs for the requirements relating to the android client side of the system.

Mocked environment
The mocked environment allows the features to be developed encapsulated without outside interaction. The mocked client app is a simple android app that contains the necessary components for the related requirements. The mocked environment is created using Android Studio using mainly Java to integrate easier with the other subgroups. As seen in *figure 1,* the mocked environment aims to be simple with a few components to make it easy for us to develop our features in it.
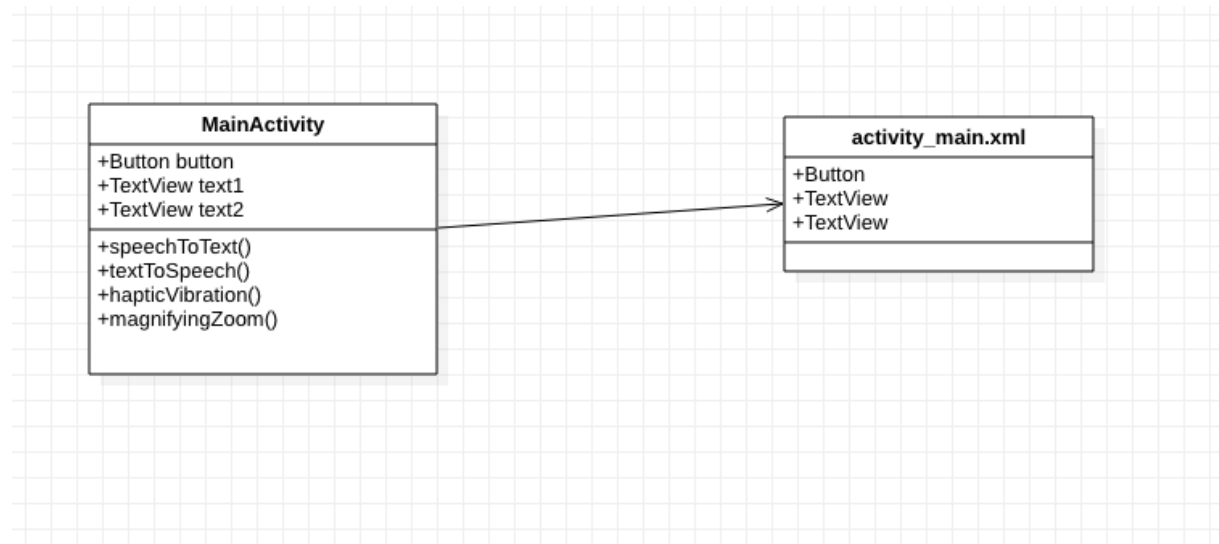

Figure 1 - Class diagram of mock client android app

D2.

A collection of the designs for the requirements relating to the web client side of the system.

Mocked environment
The mocked environment allows the features to be developed encapsulated without outside interaction. The mocked client app is a simple web app that contains the necessary components for the related requirements. *Figure 2* shows the shell of the mock environment of the web client. *Figure 3* shows the connection of the mocked web client to the sever, whether a mocked API call or the actual system server.
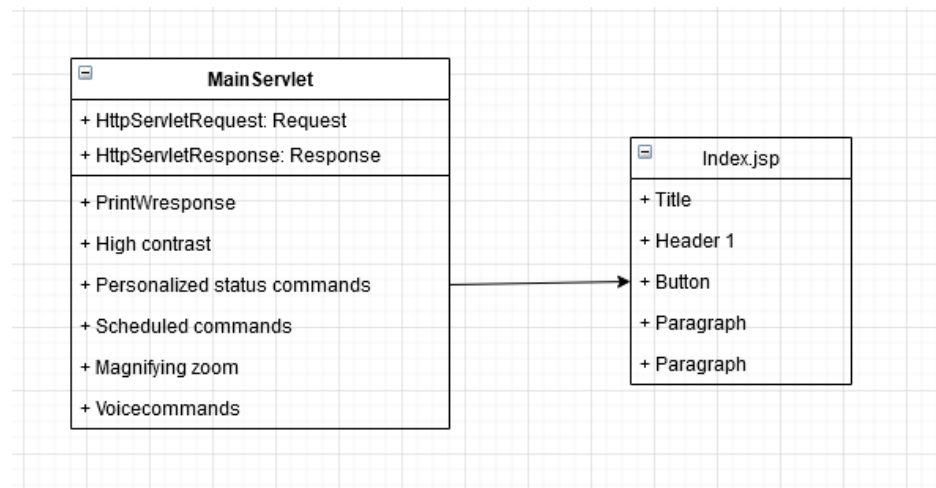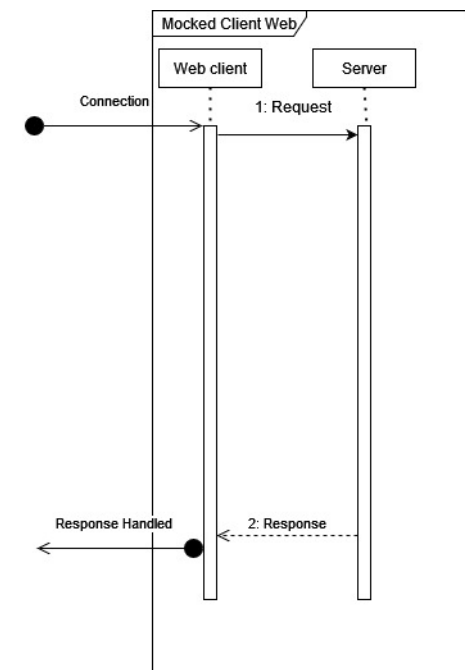

Figure 2 - Class diagram of mock client android app

Figure 3 - Sequence diagram of connection of the mocked web client

D3.

A collection of the designs for the requirements relating to the server of the system. Initially, for testing and developing, the mocked environment will be used, but eventually the implementations will migrate to the actual server/API.

Mocked environment
The mocked environment allows the features to be developed and encapsulated without outside interaction. The mocked server allows for testing and implementing of features not yet handled by the current system.
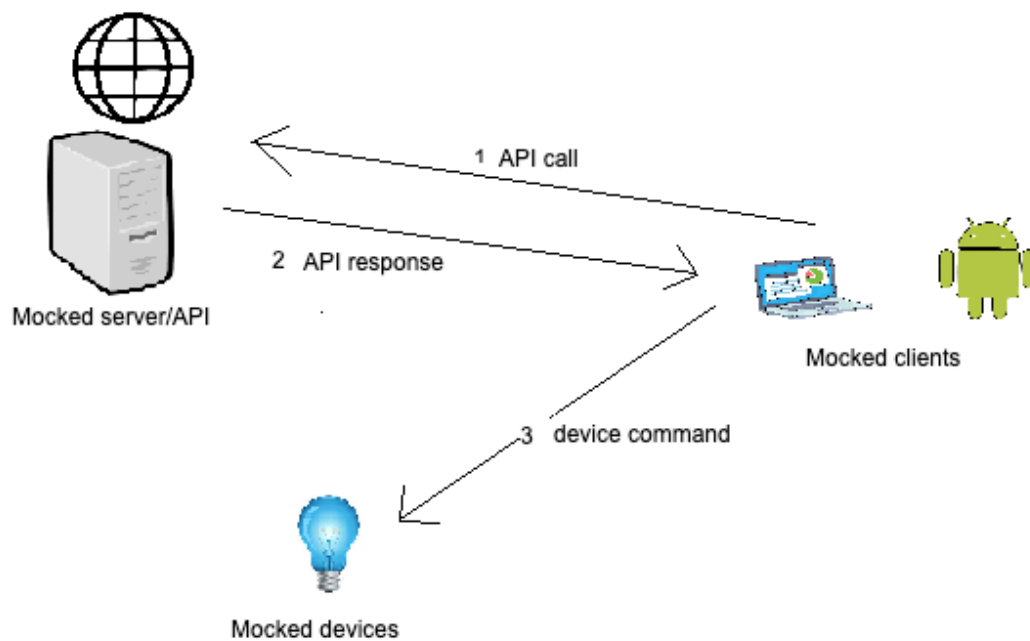

Figure 4 - General layout of mocked sever/client interaction

D4.

A collection of the designs for the requirements related to the home page, meaning all requirements that are features that the user does not toggle on and off. These features are constant to the system much like the feature to turn on and off different devices.

R4 – Personalized status commands

This feature is a home page where the user can set different command buttons that perform different actions set by the user. More is discussed in our requirements document under [R4]. For designing and implementing this feature, following figures (Figure 3) have been made. This is not a definite look of the system, but it gives an understanding of the requirement itself and allows us as a group to explain it better to other subgroups and be clear within the group how the feature should function.
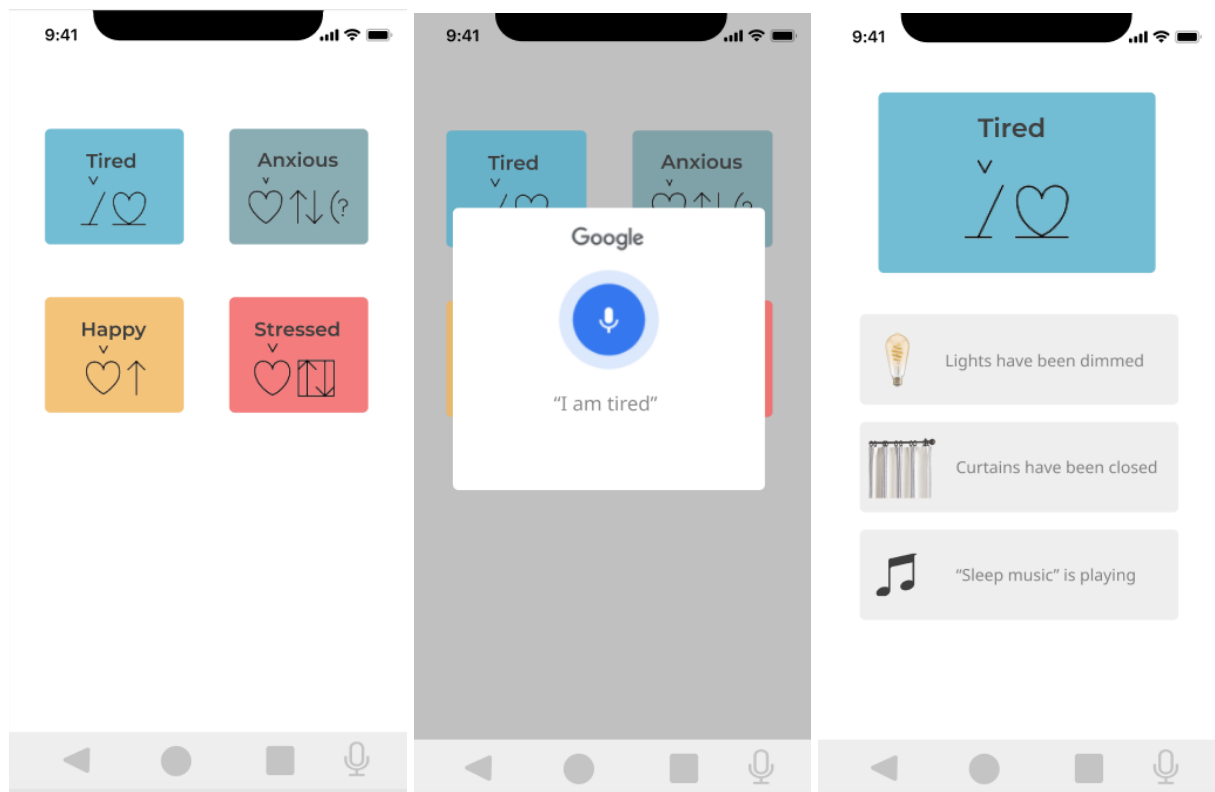

Figure 5 - Explainational figures of R4 (Figma, n.d.)

D5.

A collection of the designs for the requirements related to the settings panel, meaning all requirements that are features that the user can toggle on and off.

R1 - Haptic vibration

Figure 6 shows the class diagram of how the settings pane is related to its xml files for the android client. By using the special "root_preferences.xml" in the xml folder with a PreferencesPane, the changes made in that view will be global to the whole application. Figure 7 shows a use case diagram of a user turning on and off the setting for haptic vibration. Figure 8 shows a state diagram of the states the application takes when the haptic vibration setting is turned on or off.
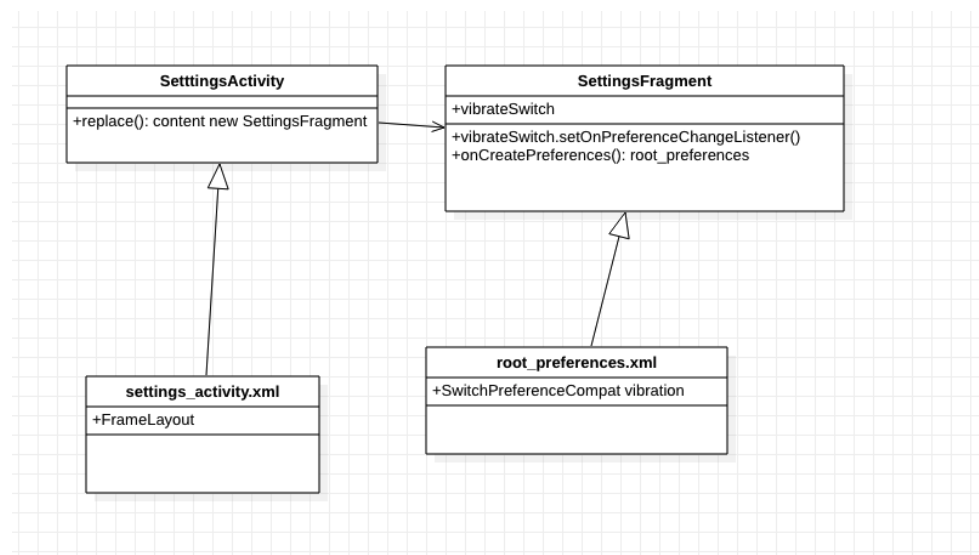
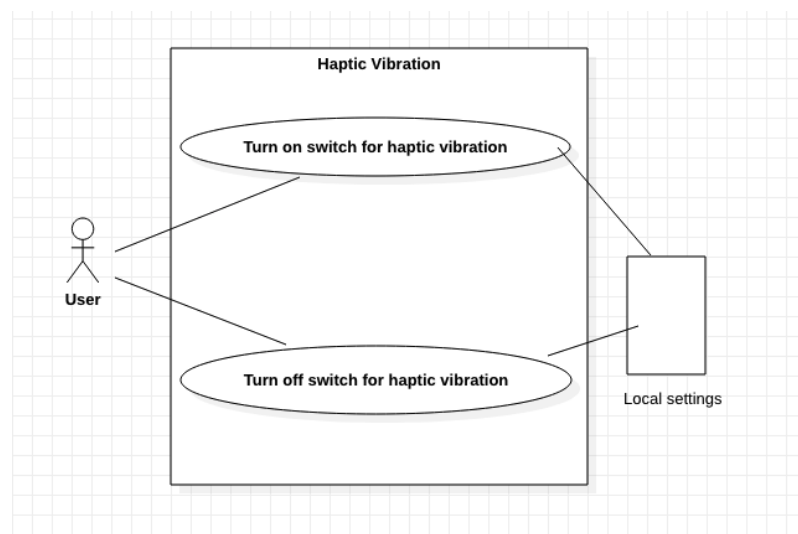*Figure 6 - Class diagram of settings pane with preferences*
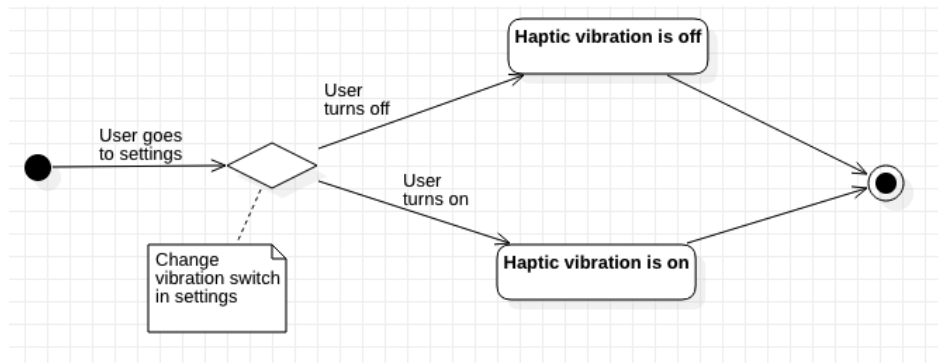

Figure 7 - Use case diagram of haptic vibration

Figure 8 - State diagram of haptic vibration
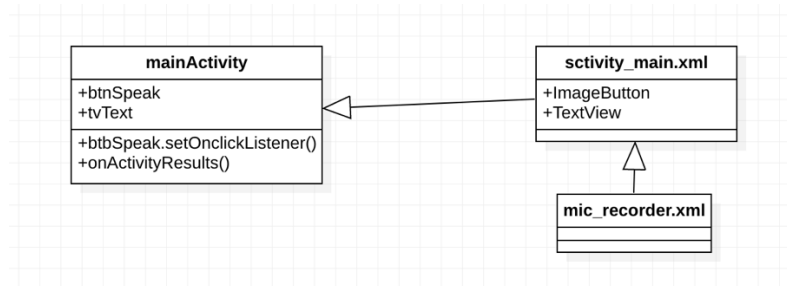
R2 – Voice commands
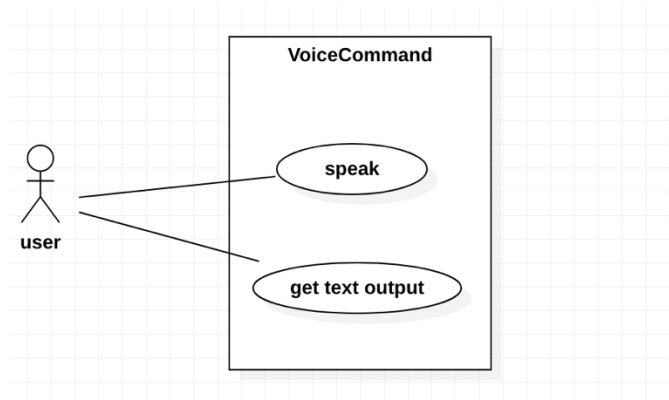

Figure 9 - Class diagram of voice command on Android

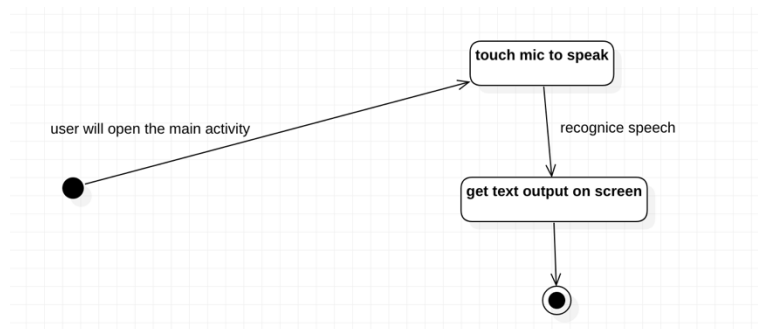*Figure 10 - Use case diagram of voice command on Android*

Figure 11 - State diagram of voice command on Android
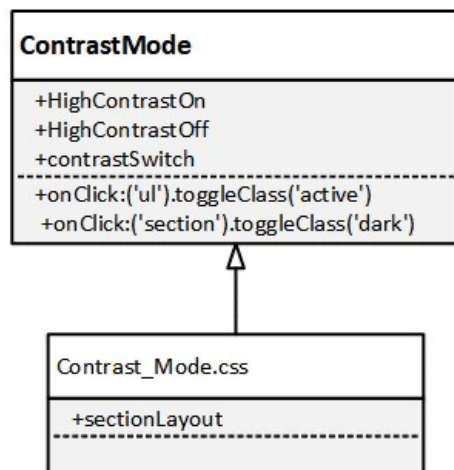
R3 – High contrast

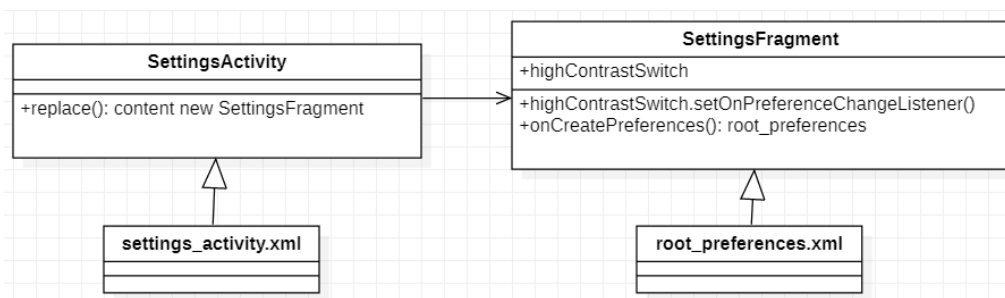*Figure 12 - Class diagram of high contrast on web*

Figure 13 - Class diagram of high contrast on Android
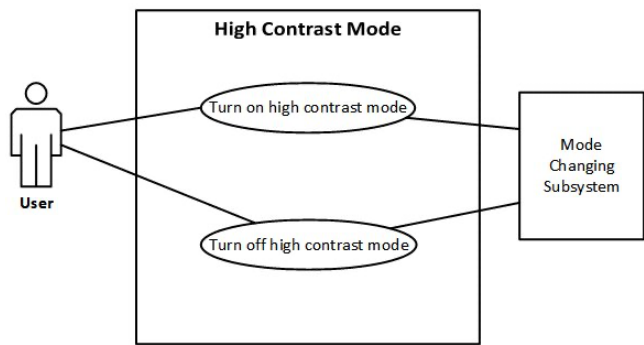
Use case diagram:

Figure 14 - Use case diagram of high contrast on web
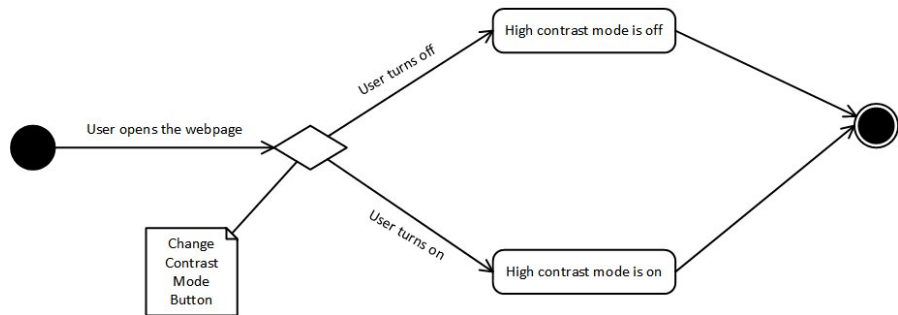
State diagram:

Figure 15 - State diagram of high contrast on web

Contents

Figma. (u.d.). *Figma* . Hämtat från Figma presentations:
https://www.figma.com/proto/qpTRnpeAb3j8zieV8muJEV/Mood-commands?node-id=3%3A360&scaling=min-zoom&page-id=0%3A1