Universität des Saarlandes

Software Engineering
Wintersemester 2019/20

Contract
Conference Management Tool

January 27, 2020

Contents

1 Objectives

The objective of this project is to develop a "Conference Management Tool" used for managing elections/votings and formal requests during a conference.
In the following "attendee" describes a person that is registered for a conference and logged in, i.e. a person that attends the conference. Simmilarly an "admin" is a person responsable for managing the conference data (starting votes, uploading documents etc.).
The Conference Management Tool consists of three components: a graphical **attendee interface** which is used by the attendees during the conference; a graphical **admin interface** which is used by the organizers to manage users and the contents shown in the user interface; a **backend** managing the data of the conference.

1.1 Must-Have Criteria

/M1/ Secure Log-in and Communication:

Each user is assigned a unique and secure password by the administrator as a scanable QR-Code, and an alphanumerical representation of that code in case the device has no camera. The user can use this password in combination with their name exactly once to certify a device for the rest of the conference. The application does not provide functionality for the user to log themselves out, but will log out the user after the conference has ended.
The communication between the clients and the the server is secured.

/M2/ Multi-Platform-Access:

The attendee interface must work on most platforms. This means that there will be an Android application, an iOS application and a web-application, such that it can be accessed on Dektops without downloading a designated application. The admin interface must run on Windows, Linux and macOS.

/M3/ Requests:

The user must be able to create a "request for speech" or "request of change" at any time using a form provided by the application. A "request for speech" must consist of the name of the requesting person, their group and an agenda item or document the request refers to. A "request for change" consists of the same items as well as the proposed amendment. Since the name and the group can be assumed to be recorded in the database of registered persons, the attendee do not need to provide these in the attendee interface.

/M4/ Voting Process:

An attendee has the possibility to cast a vote once for any active voting. A voting will display as a set of choices. An attendee can click on the item they want to vote for. This will not instantly cast the vote until the attendee clicks on a different button to actually submit the vote.

/M5/ Voting Notification:

Once a voting is started, the user interface must switch the view to the voting. However, the attendee must be able to ignore it and switch back to any other place inside the attendee interface. There remains a small counter reminding the attendee of the remaining time.

/M6/ Attendee Interface:

Inside the application, the attendee can see the updated agenda, the documents of the conference, the current voting and their personal details. Additionally, Requests of Change and Requests of Speech can be submitted.

/M7/ Administration:

Only a person with access to the backend is allowed to create new administrative accounts. Each admin can use a web-based GUI to change the agenda, manage documents, create, change, start and end votings, see the requests of the attendees and add new attendee accounts. Each voting can be modified until the vote has started.

1.2 May-Have Criteria

/O1/ Group Eligibility:

The application may provide functionality to decide which groups of attendees are eligible for the current voting and whether the attendee is eligible to request a change or to speak.

/O2/ Multi-Device-Access:

The application may provide functionality for the attendees to use more than one device at once. This must still ensure that each user can only cast one vote.

/O3/ Customizability:

The application may provide functionality to customize parts of the design, especially the colour and logo of the conference.

1.3 Must-Not-Have Criteria

/N1/ Data Storage Policy:

The application must not delete the overall results of the votings, the information about who attended the conference and the requests.

/N2/ Single and Equal Votes:

Each attendee must be unable to cast more than one vote and each vote is counted equally and only once.

/N3/ Voting only on time:

The application needs to ensure that voting is only possible during the voting period.

/N4/ Privacy:

No user must be able to see the votes of any other user, unless the type of voting intends this. In particular this means that no user is allowed to see the results of the voting before it is closed.

/N5/ Login Period:

No user shall be logged out before the conference is over, no matter the timezones of the individual devices.

2 Product Use

2.1 Target Groups

The application can be used by organizers of conferences of any type as well as their attendees.

2.2 Operative Conditions

The users log in before the start of a conference or at its begining and get logged out automatically at the end of the conference. A conference can be at any time and can last multiple days.

3 Product Functions

In the following, names of functions and buttons are not specified to be in the english language like in this document. These might change in the application due to localisation.

3.1 User Application

3.1.1 Conference Timeline

/F1/ The administrators register the attendees before the conference starts. There is no functionality to register on the attendee interface.

/F2/ The attendee provides their username and password (which they recieve from an admin) to log in.

/F3/ The attendee can see the agenda by pressing the Agenda-Button.

/F4/ The attendee can see the documents for this conference uploaded by the administrative side by pressing the Documents-Button.

/F5/ The attendee can see the current active voting, as well as previous votes, by pressing the voting-button.

/F6/ The attendee can create "Requests of change" or "Requests for speech" by pressing the Request-Button and then picking the type of request.

3.1.2 Voting

/F7/ The attendee is notified when a new voting starts.

/F8/ When a new voting starts and the attendee has opened the user interface, it changes to the voting view, as if the Voting-Button was pressed.

/F9/ The attendee can click on any item they want to vote for.

/F10/ The attendee can click on the vote-button to cast a vote for the selected item.

3.1.3 Request of Change

/F11/ The attendee can choose the agenda item or document they want to change by using a dropdown menu.

/F12/ The attendee can describe the proposed change in a text field.

/F13/ The attendee can press the Send-Button to send the request.

3.1.4 Request for Speech

/F14/ The attendee can choose the agenda item or document they want to talk about using a dropdown menu.

/F15/ The attendee can press the Send-Button to send the request.

3.2 Administrative Application

3.2.1 Agenda

/F16/ The Admin can add a new topic to the agenda using a text field.

/F17/ The Admin can change the current topic to the next one.

/F18/ The Admin can change the order of the agenda topics by changing the ID of some topic.

/F19/ The Admin can edit an already existing topic, i.e. change its name.

/F20/ The Admin can delete an already existing topic.

3.2.2 Participants

/F21/ The Admin can create an account for a new participant using their name, location, their group and their function.

/F22/ The Admin can view and edit the info of already existing participants.

/F23/ The Admin can delete a participant, i.e. completely remove them from the conference.

/F24/ The Admin can kick a participant, i.e. invalidate their login data.

/F25/ The Admin can create new login data for an existing participant, invalidating their old login data if that didn't already happen.

3.2.3 Documents

/F26/ The Admin can upload a document and give it a file name.

/F27/ The Admin can download any uploaded documents again in case they can't access it anymore.

/F28/ The Admin can edit or delete an already uploaded document.

3.2.4 Requests

/F29/ The Admin can sort all requests e.g. using their time stamp or the topic they are related to.

/F30/ The Admin can open any request and have a look on their details.

/F31/ The Admin can approve or disapprove of a request of change.

3.2.5 Votings

/F32/ The Admin can create a new voting, which does not start the vote yet.

/F33/ The Admin can edit an existing voting until it is started.

/F34/ The Admin can start the voting, which starts the voting timer and closes it automatically as soon as the timer has run out.

/F35/ The Admin can explicitly close a voting before the timer has run out.

/F36/ The Admin can display the results in a pie chart after the voting is closed.

/F37/ The Admin can choose to delete a voting. That option won't be available during the voting period itself.

4 Product Data

The following data is stored on the devices:

4.1 User Device

/DU1/ Token to authenticate the device for backend requests.

4.2 Administrative Device

/DU1/ Token to authenticate the device for backend requests.

4.3 Server Backend

/DA1/ The overall result of the votings.

/DA2/ Requests made by participants and their results.

/DA3/ Data to authenticate the participants as well as the admins.

/DA4/ Personal Data that the administrator provides about the user.

/DA5/ The conference agenda as well as the current topic.

5 Performance

The following performance specifications assume that there are no major server or network failures which can not be avoided by the software.

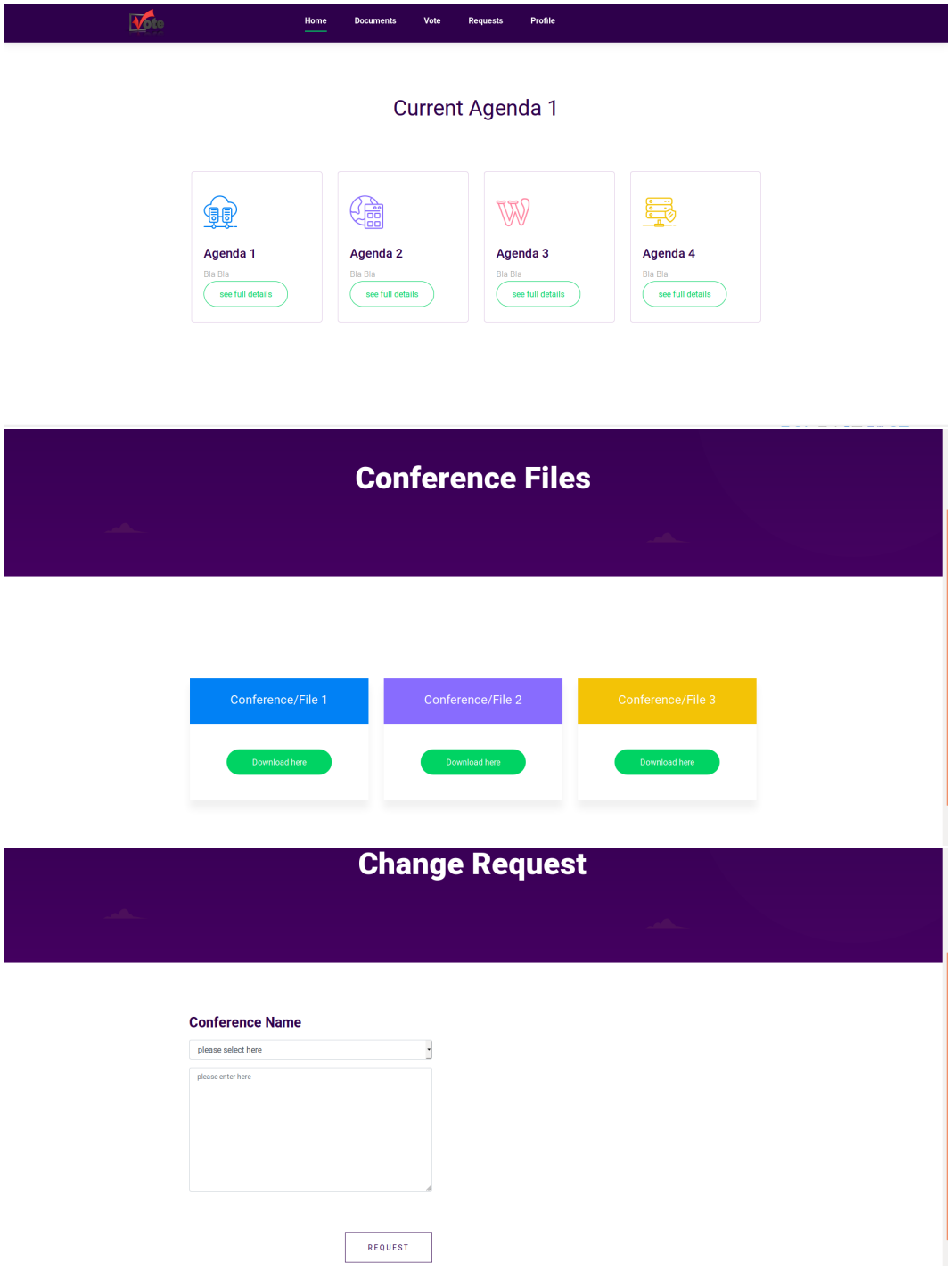
/P1/ The application should be able to handle at least 10.000 active attendees at the same time.

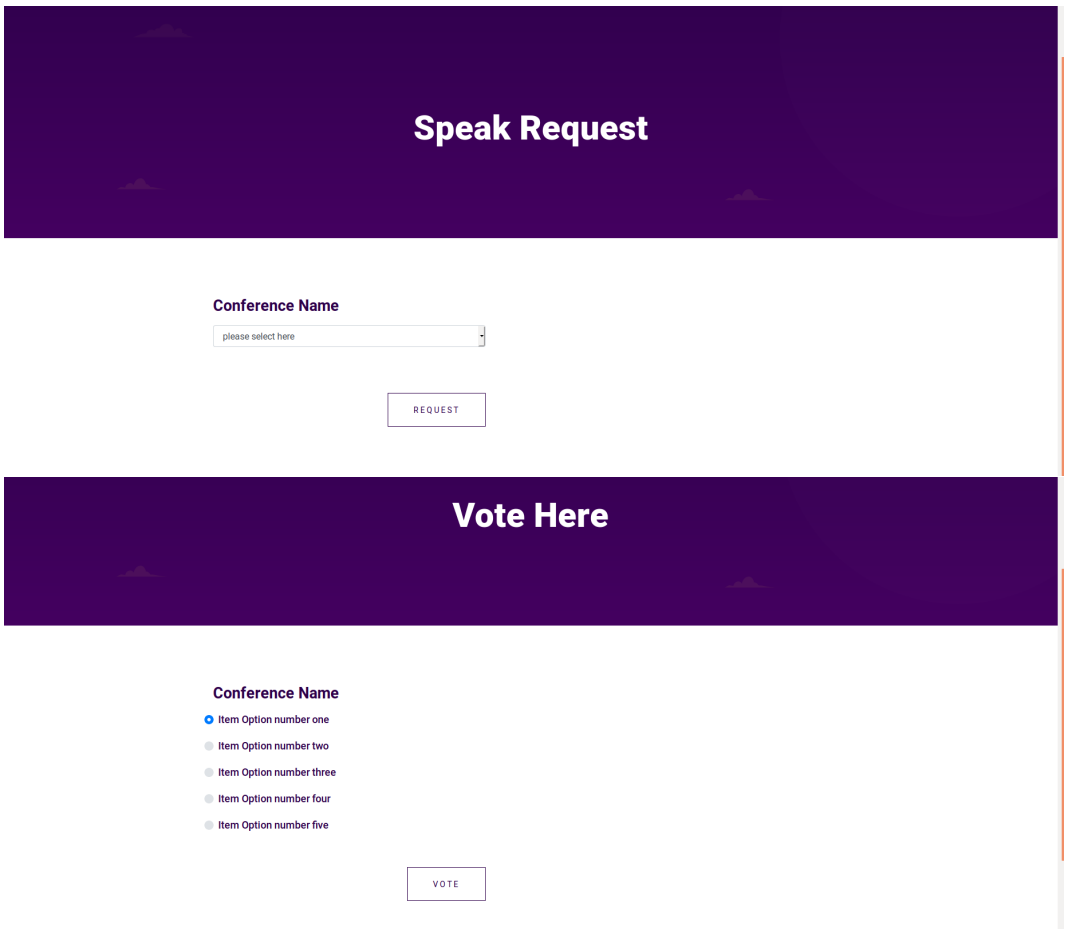
/P2/ The application should run smoothly on all reasonably modern devices.

/P3/ Processing user requests as well as notifying attendees about votings should not take longer than a few seconds.

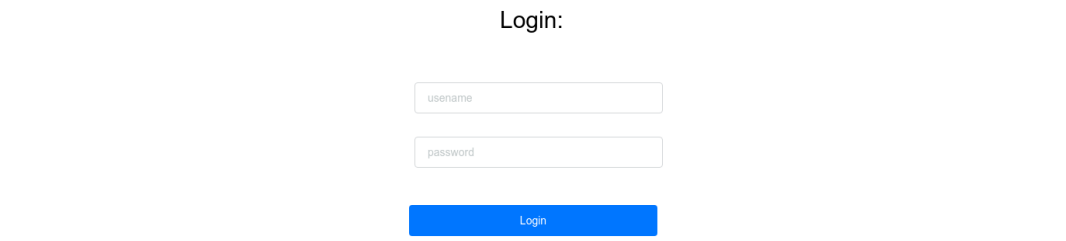
6 Interface

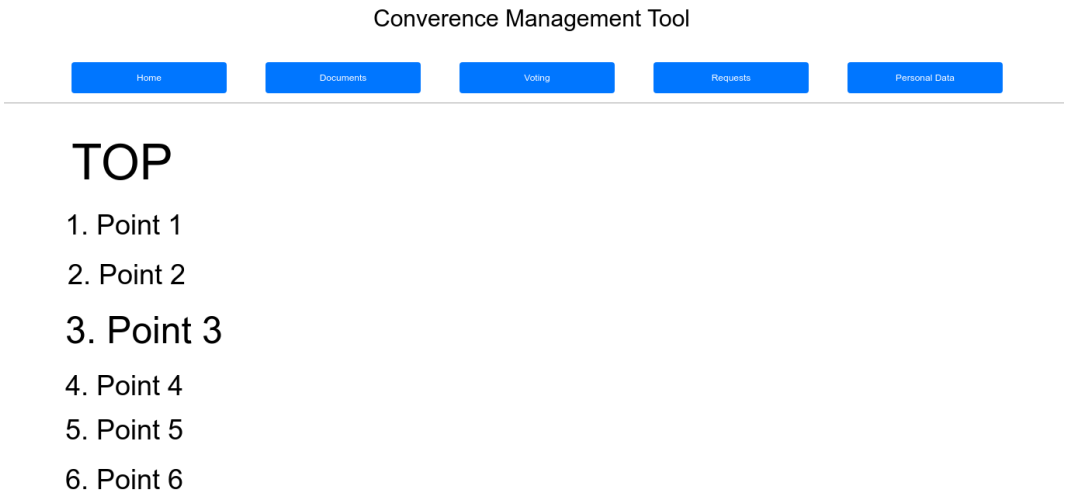
6.1 User Interface

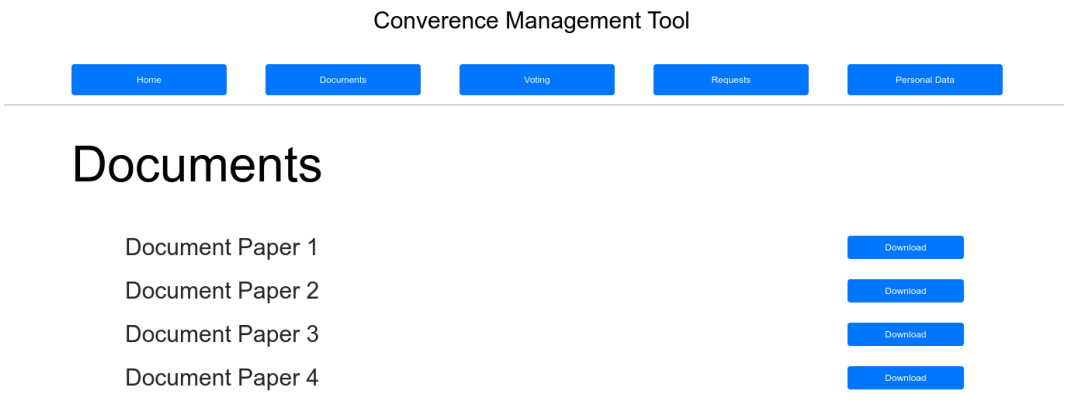


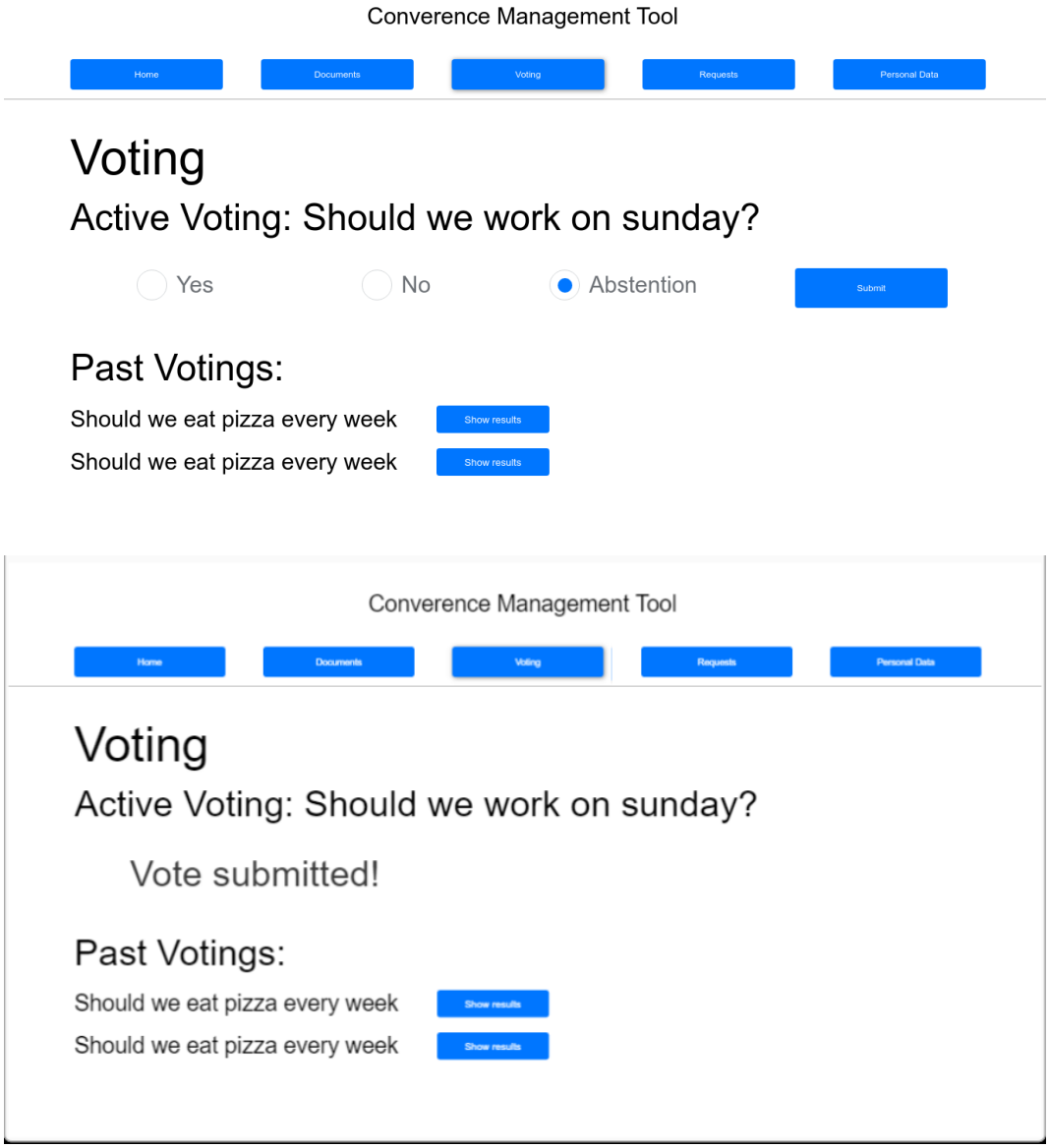

Speak Request
Conference Name
please select here
REQUEST
Vote Here
Conference Name
Item Option number one
Item Option number two
Item Option number three
Item Option number four
Item Option number five
VOTE

6.2 Alternative User Interface

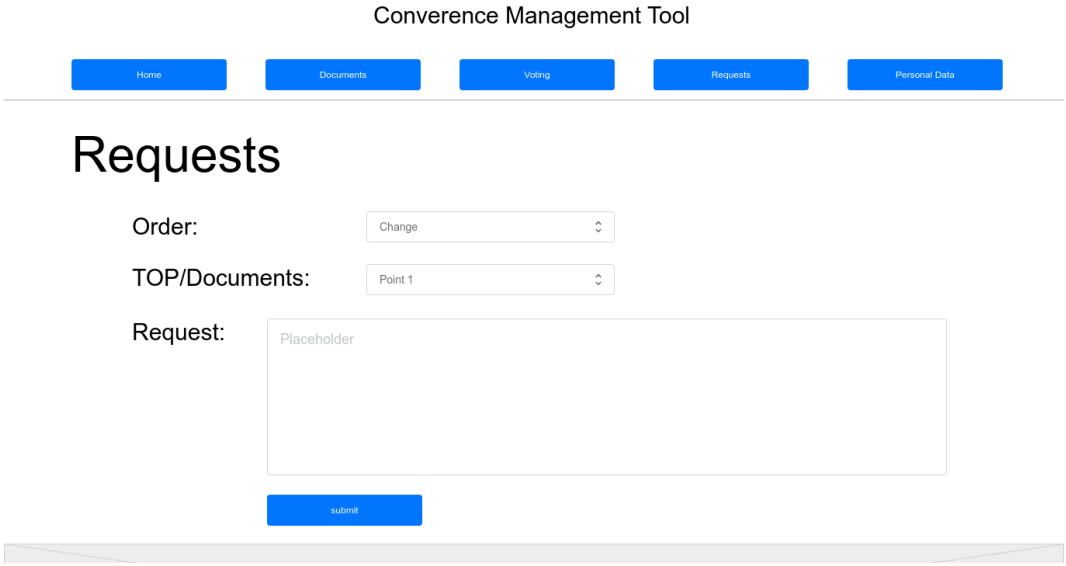


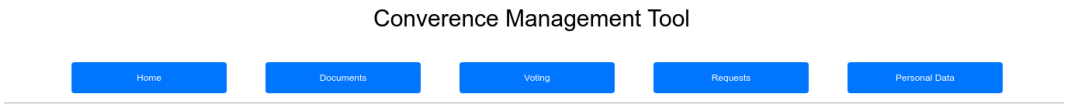





Converence Management Tool
Home
Documents
Voting
Requests
Personal Data
Voting
Active Voting: Should we work on sunday?
Yes
No
Abstention
Submit
Past Votings:
Should we eat pizza every week
Show results
Should we eat pizza every week
Show results

Converence Management Tool
Home
Documents
Voting
Requests
Personal Data
Voting
Active Voting: Should we work on sunday?
Vote submitted!
Past Votings:
Should we eat pizza every week
Show results
Should we eat pizza every week
Show results

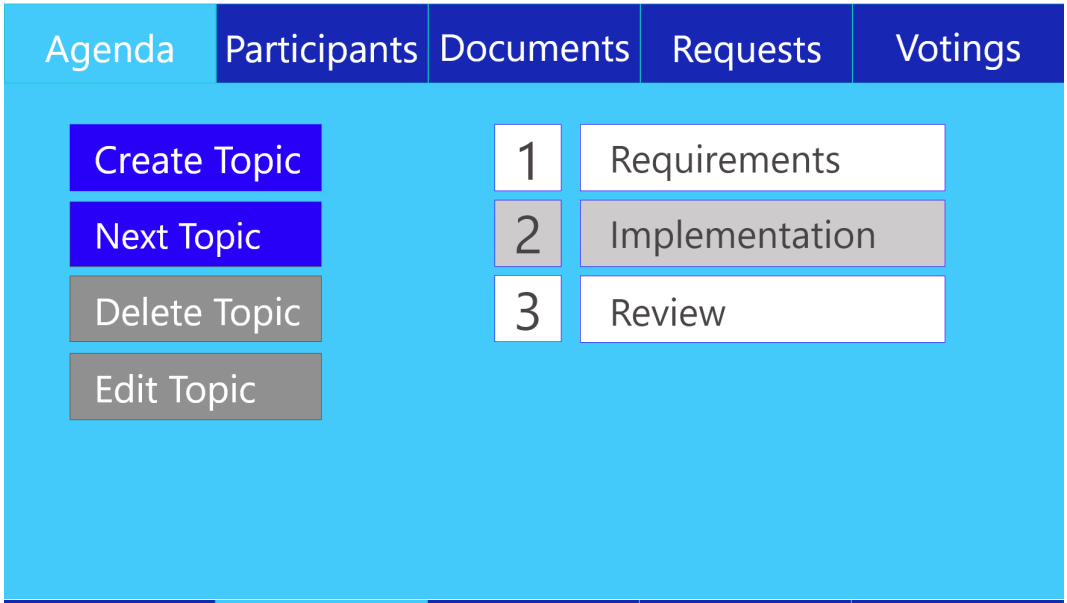

Converence Management Tool
Home
Documents
Voting
Requests
Personal Data
Requests
Order:
Change
TOP/Documents:
Point 1
Request:
Placeholder
submit

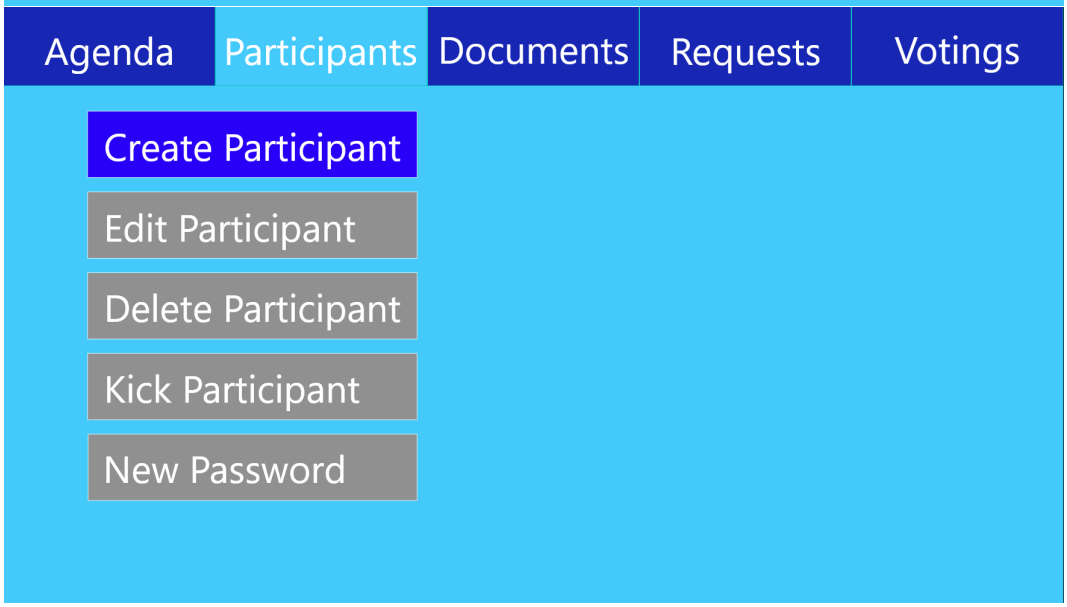


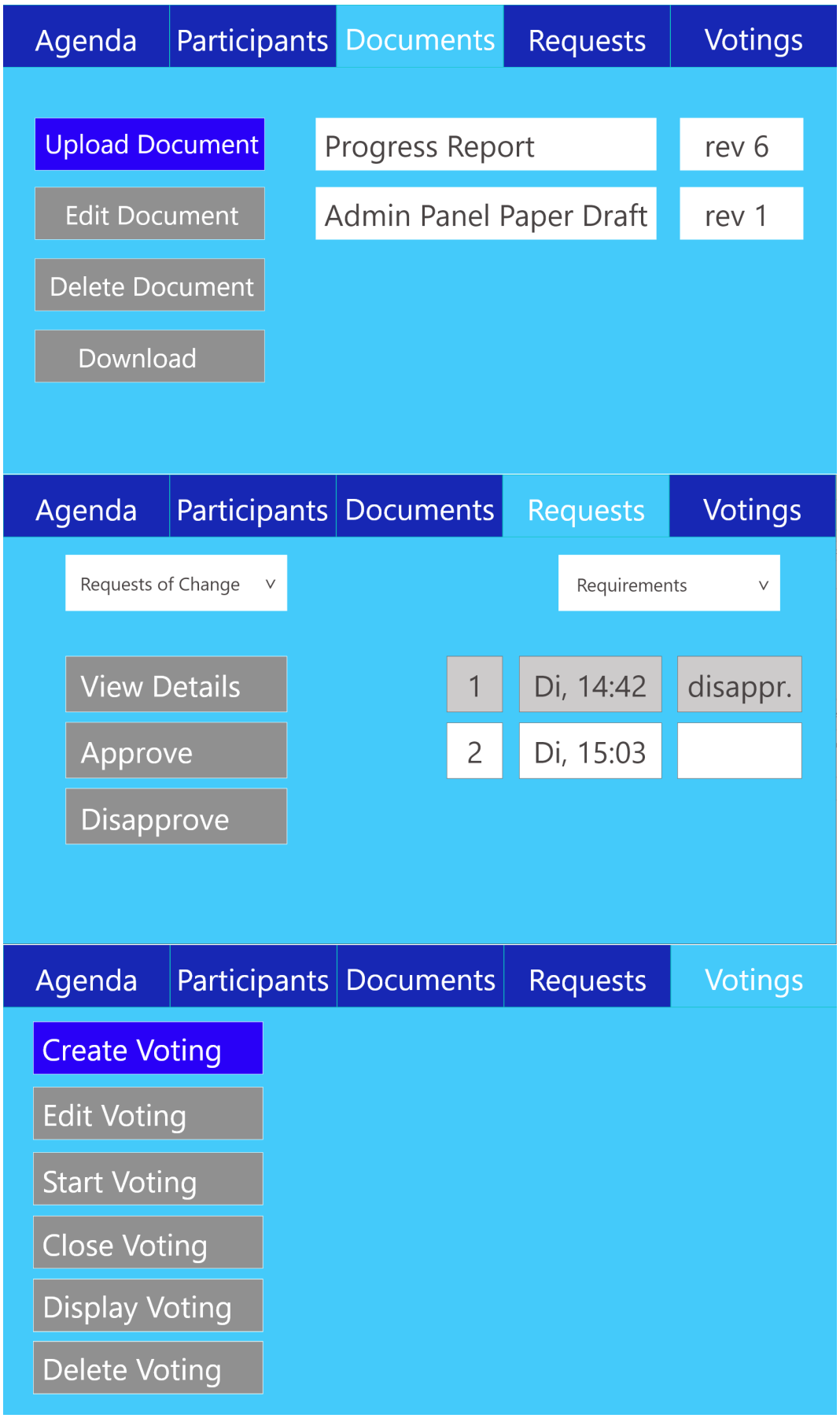
Personal Data

name:	Max Mustermann
Wohnort:	Musterstraße 23, 66666 Musterort
function:	Admin
group:	RCDS

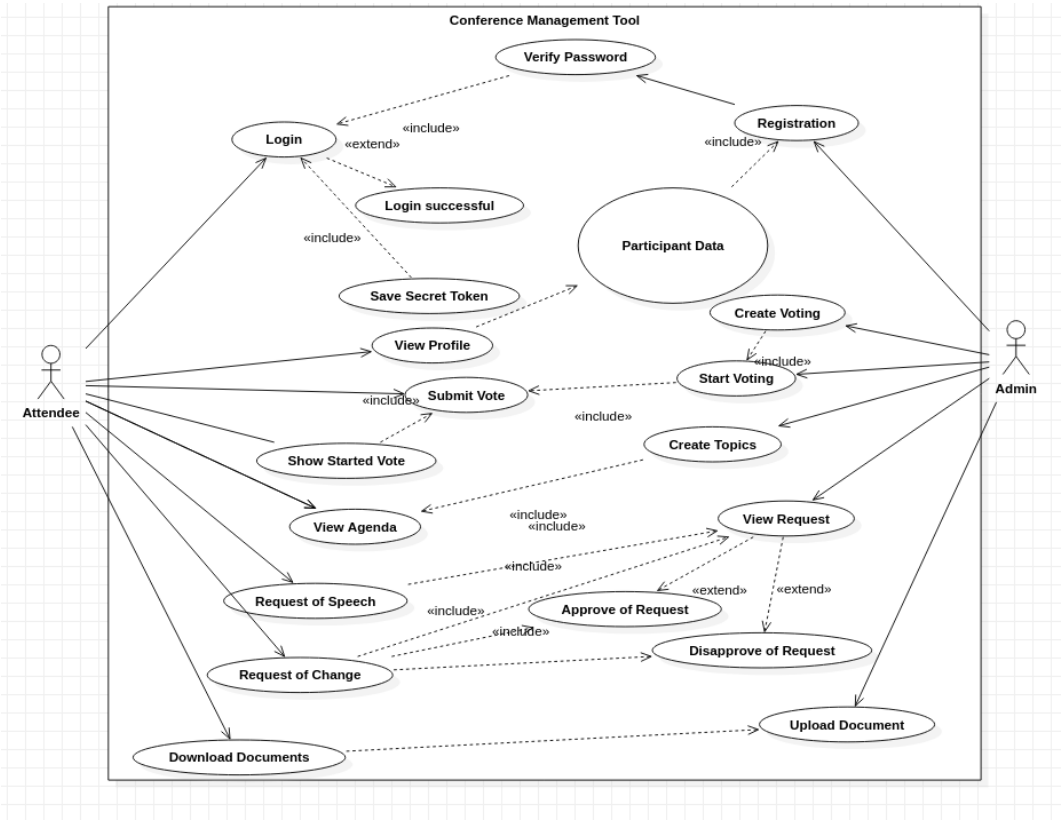
6.3 Administrative Interface






Agenda
Participants
Documents
Requests
Votings
Upload Document
Edit Document
Delete Document
Download
Progress Report
rev 6
Admin Panel Paper Draft
rev 1
Agenda
Participants
Documents
Requests
Votings
Requests of Change
Requirements
View Details
Approve
Disapprove
1
Di, 14:42
disappr.
2
Di, 15:03
Agenda
Participants
Documents
Requests
Votings
Create Voting
Edit Voting
Start Voting
Close Voting
Display Voting
Delete Voting

7 Use-Case-Diagram



For further information, please take a look at the appendix.

8 Technical Product Environment

8.1 Hardware and Software

The attendee interface should be able to run on all types of devices of reasonable ages (Notebooks, Smartphones, Tablets and similar). The administrative interface should be able to run on any common workstation with Java support. The backend should be able to run on a moderatly powerful server which provides good support for concurrency and memory access which has an webserver installation with php support.

9 Quality Requirements

Quality Requirements	very good	good	normal	irrelevant
Security	✓			
Reliability	✓			
Portability	✓			
Usability		✓		
Efficiency			✓	
Modifiability			✓	

Security & Reliability

The application should be secure & reliable since it is supposed to be used for votings.

Portability

Since the users are supposed to run the application (attendee / admin interface) on their own device it should run on all devices mentioned in section 8.1. It is recommended to run the backend on a dedicated server.

Usability

The target group, as described in section 2.1, covers a wide range of users. Since no special training or introduction is given regarding how to use the application, intuitive usage is important. As such the application will make use of the common UI conventions.

Efficiency

Since the application is to be used by multiple users at the same time it is important that it responds quickly. However, since the tasks are not time-critical and the time frames offered for voting and simmilar actions are generous, optimal efficiency is not important.

Modifiability

Customizing the attendees UI such that it reflects the conference type (e.g.: a green color pallet for conferences on climate change) is a desiarable feature. However, due to time constraints it only falls in the "may have" cathegory.
Note that the conference data itself has to be modifiable.

10 Contract Adjustment

10.1 Introduction

This document specifies changes to the **Conference Management Tool** contract, 13.12.2019, Saarbrücken. The folliwing change mentioned in **Must-Have Criteria** replaces **/M2/** in **1.1 Must-Have Criteria** of the original document. The following additions mentioned in **May-Have Criteria** are additions to the original may-have specifications mentioned in **1.2 May-Have Criteria** of the original document. Since these specifications are considered more important, the customer and developers agree on prioritizing the implementation of the specifications mentioned in this document over the original may-have specifications.

10.2 Must-Have Criteria

/M2/ Multi-Platform-Access:
The attendee interface must work on most platforms. This means that there will be a responsive web-application, such that it can be accessed on Android, IOS and Dektops without downloading a designated application. The admin interface must run on Windows, Linux and macOS.

10.3 May-Have Criteria

/O4/ Attendee CSV Parser:
The application may provide functionality for admins to submit a csv file containing multiple attendee's data to be added to the conference at once.

/O5/ Agenda TXT Parser:
In case there is currently no agenda, the application may provide functionality for admins to submit a TXT file containing the agenda points to be set as new agenda.

11 Appendix

11.1 Use cases

Login failure - password in use

Id: login_failure_1

Remarks:
The underlined text presents actions which happen outside the application.
The **bold** text represents parts which reveal the internal working of the app and that are not visible to the users / admins.

1. The user registers for a conference and recieves a password as usual

2. The user tries to log in.
3. **The app notices that the password has already been used**
4. The admins get feedback telling the user that there were multiple login attempts with the same password for the same user name. By default the account gets blocked.
5. The app automatically protocols every attempt of double password usage.
6. The user gets notified to visit a service desk immediately
7. The user goes to the service desk
8. If the user says that he accidentally used the password twice, then the admin can just unblock the account.
9. If the user says that he never used the password before the admin can delete the data and re-register the user (There should be a button that allows for automatic re-registration, which leaves everything except the login data unchanged).

Regular user login

Id: login_reg_user

Remarks:
The underlined text presents actions which happen outside the application.
The **bold** text represents parts which reveal the internal working of the app and that are not visible to the users / admins.

1. The user gets invited by an admin. The admin registers the user with some data which he collects
2. **The app registers the user. It does not yet generate the secret token**
3. The admin gets feedback about the successful registration
4. The user joins the conference
5. The admin uses a function to display the user data
6. The user is prompted to check his data, as only admins can edit it
7. The user tells the admin that there is a typo somewhere
8. The admin uses a function to edit the incorrect data.
9. The admin uses a function to generate a password. **The app generates a secret token and a password and stores them. It also stores that the password is unused and that the user is currently logged in on 0 devices**
10. The user gets the password (either as text or an qr code) from the admin. The user uses it to login
11. **The app checks the password, sends out the token, marks the password as used in order to prevent multiple logins and increments the device count of the user**
12. The user decides to log in on a second device
13. The user presses a button in order to receive a new password
14. **The app generates a new password and sends it to the user**
15. The password gets displayed on the first device and can be used to log in on the second device
16. **Step 6 gets repeated. The same token gets sent**

Registration - user not present 1

Id: reg_user_not_present_1

Remarks:
The underlined text presents actions which happen outside the application.
The **bold** text represents parts which reveal the internal working of the app and that are not visible to the users / admins.

1. The user gets invited by an admin. The admin registers the user with some data which he collects
2. **The app registers the user as usual**
3. The user notifies the admin that he will not be able to attend the conference
4. The admin can chose whether to delete the user data, or just invalidate it (making logins and voting impossible, but remembering the fact that the user wanted to attend)
5. If the second option is chosen, then the vote of that user will always count as "absent" rather than "neutral" or "abstained from voting"

Registration - user not present 2

Id: reg_user_not_present_2

Remarks:
The underlined text presents actions which happen outside the application.
The **bold** text represents parts which reveal the internal working of the app and that are not visible to the users / admins.

1. The attendees join the conference as usual
2. At any point the admin can use a function to check which of the registered users have not logged in yet (despite not having announced that they would be unable to attend)
3. The admin can now decide how to threat those users (delete their data, block further logins and voting, etc.)

Top - Changes

Id: top_change

Remarks:
The underlined text presents actions which happen outside the application.
The **bold** text represents parts which reveal the internal working of the app and that are not visible to the users / admins.

1. The attendees discuss some TOP
2. The admins decide to add a new point and change the order of the last two points and remove an other TOP
3. The admins configure this with ease
4. All user devices display the changed TOP

Voting - Accidental Multiple vote submission

Id: voting_acc_multi_vote.tex

Remarks:
The underlined text presents actions which happen outside the application.
The **bold** text represents parts which reveal the internal working of the app and that are not visible to the users / admins.

1. The admin starts a vote.
2. A user clicks the submit vote button multiple times (possible due to using a slow device)
3. The client side prevents sending multiple votes and just sends the initial vote
4. The vote of the user is counted as usual

Voting - document access

Id: voting_doc_acc

Remarks:
The underlined text presents actions which happen outside the application.
The **bold** text represents parts which reveal the internal working of the app and that are not visible to the users / admins.

1. The admin starts a vote.
2. The voting panel gets opened on each device
3. The user decides that he is not interested in the vote and wishes to browse the documents instead
4. The user uses a button to close the vote
5. The app can now be used the way it was used before the vote started
6. The user uses the app to browse the files
7. The vote gets closed
8. The backend counts the vote of the user as "present, but did not vote", which is different from "voted neutral" and "was absent"

Voting - Multiple vote submission

Id: voting_mult_votes

Remarks:
The underlined text presents actions which happen outside the application.
The **bold** text represents parts which reveal the internal working of the app and that are not visible to the users / admins.

1. The admin starts a vote.
2. The backend notices that there are two incoming votes from the same user.
3. These votes are classified as malicious, since the attendees interface doesent allow multiple votes for the same vote. The system drops both votes and notifies the admins (but not the users). The admins do see who sent multiple votes but do not see what the user voted for.
4. The admins can chose how to react to this (ex. redo the vote, block the account of that user etc.)

Voting - Option change

Id: voting_opt_change

Remarks:
The underlined text presents actions which happen outside the application.
The **bold** text represents parts which reveal the internal working of the app and that are not visible to the users / admins.

1. The admin discuss a topic as part of some TOP
2. The attendees discuss whether option A or B is better. The admins decide to hold a vote
3. The admins configure the vote
4. Before the vote starts someone proposes a third option, which is simmilar to B but not quite the same thing
5. The admins edit the voting configuration, renaming B to B' and adding option C as a possible choice
6. The vote starts and unfolds as usual

Voting - document access

Id: voting_regular

Remarks:
The underlined text presents actions which happen outside the application.
The **bold** text represents parts which reveal the internal working of the app and that are not visible to the users / admins.

1. The admin starts a vote.
2. A notification is send to the users device
3. The user gets redirected to the voting page
4. The application requests details on the voting
5. The user is displayed all information regarding the voting
6. The user submits a vote
7. A vote request containing the vote option and the users token is send to the server
8. The vote is counted as usual
9. **The server sends a notification to the client that the vote was counted, the user can still see the vote but cant submit anymore**
10. The user reloads the page
11. The application requests information (title, description, vote status) regarding the voting
12. The user still sees the voting but is unable to submit another vote