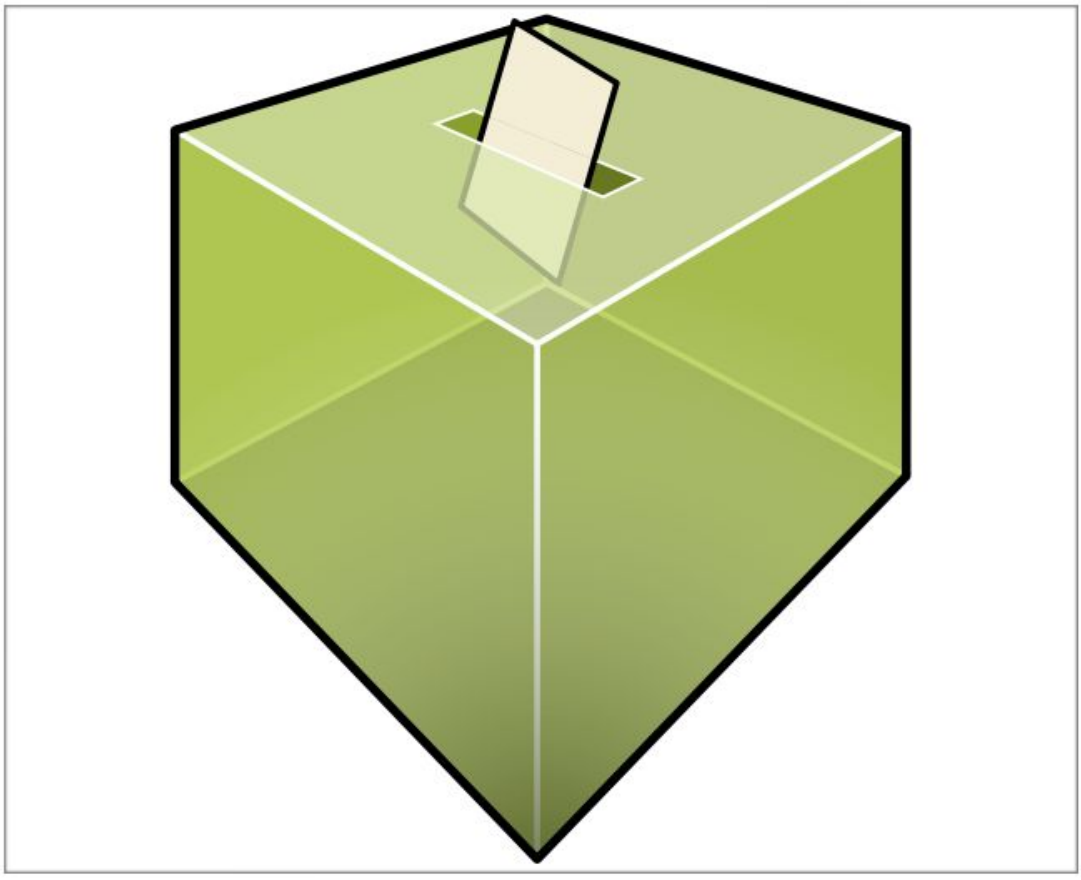
EVS Project Manual

Team Echo



Team Members:

- Jorge Gavilan
- Jennifer Gerbl
- Shivam Rajendra Nirhali

Table of Contents

Software Description (PM01)

The Electronic Voting System (EVS) supports electronic polls. The voting system is accessible via the Web and allows an Organizer to create Polls and the respective questions for the polls. The system supports 3 different types of questions: yes-no, single choice, and multiple choice. Voters are able to abstain from voting. Participants of a poll receive a token via E-Mail when a poll is started. This token represents their ballot paper. When the voting period finishes the organizer can view the results with a graphical representation.

Installation & Configuration

The system where the application is going to be installed requires the following software to be installed

- OpenJDK 11
- Payara Server v5+
- MariaDB
- Web Browser: Chrome 84 and above, Firefox 79 and above

You can follow the installation instructions in the course material if you are having difficulties.

Database

The application uses a MariaDB database which is necessary to set up in order for the application to work properly:

1. Create a user with name APP and password APP

```
create user 'APP'@'localhost' identified by 'APP';
grant all privileges on *.* to 'APP'@'localhost';
```

2. Create a database called **echoevs**

```
create database echoevs default character set = utf8mb4;
```

If the application is going to be deployed on a UNIX system it is necessary to make the database case insensitive, for this, please add the following lines in the mariadb configuration file (my.cnf) and restart the database service:

```
[mariadb]
lower_case_table_names=1
```

3. Activate the Event Scheduler on the database:

Using an user that has the SUPER privilege (e.g. root) execute the next query:

```
SET GLOBAL event_scheduler = ON;
```

Payara Server Configuration

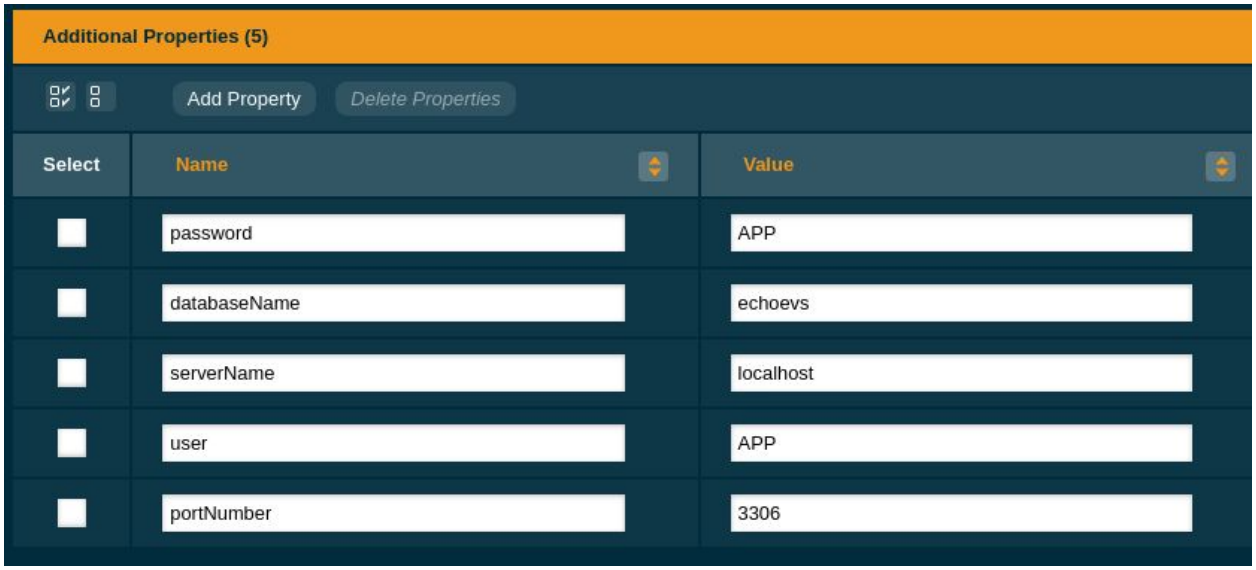
JDBC Connection Pools

Create a new JDBC Connection Pool on Payara with the following settings:

Pool Name:	echoEVS-pool
Resource Type:	javax.sql.DataSource
Datasource Classname:	org.mariadb.jdbc.MariaDbDataSource

Additional Properties:

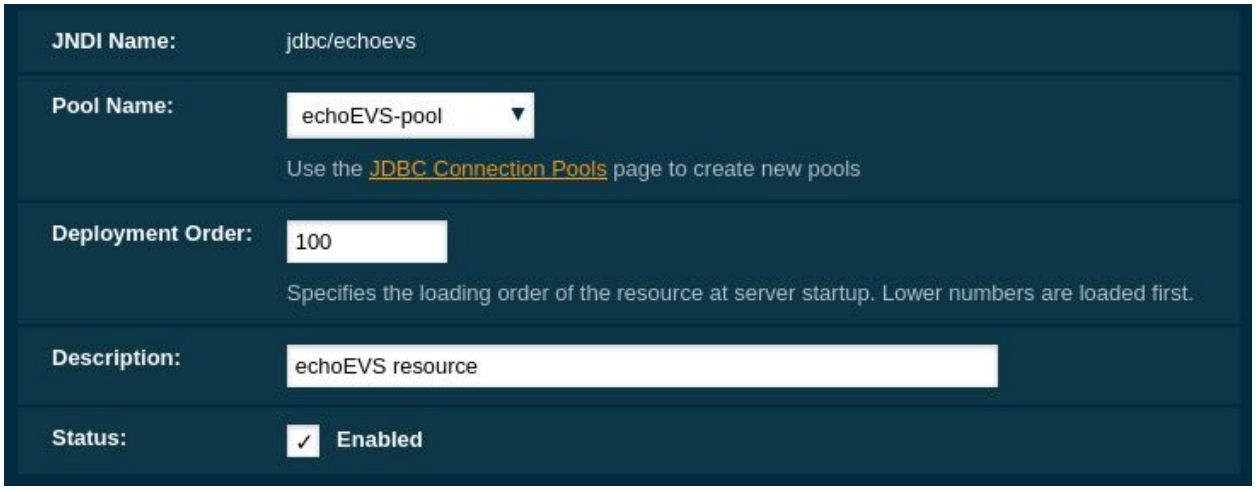
databaseName	echoevs
serverName	localhost
user	APP
password	APP
portNumber	3306



JDBC Resources

Create a JDBC Resource to provide the application with a means to connect to a database using the JDBC connection pool previously created.

JNDI Name: jdbc/echoevs
Pool Name: echoEVS-pool



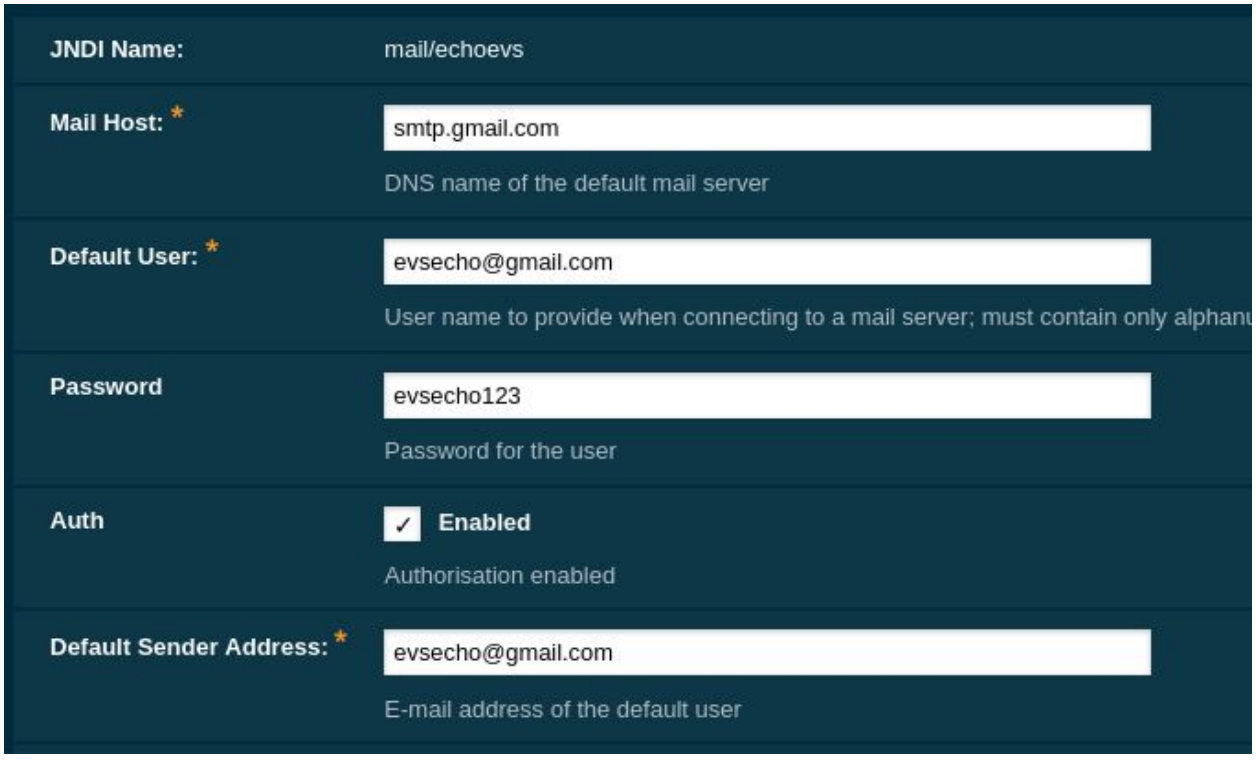
JavaMail Session

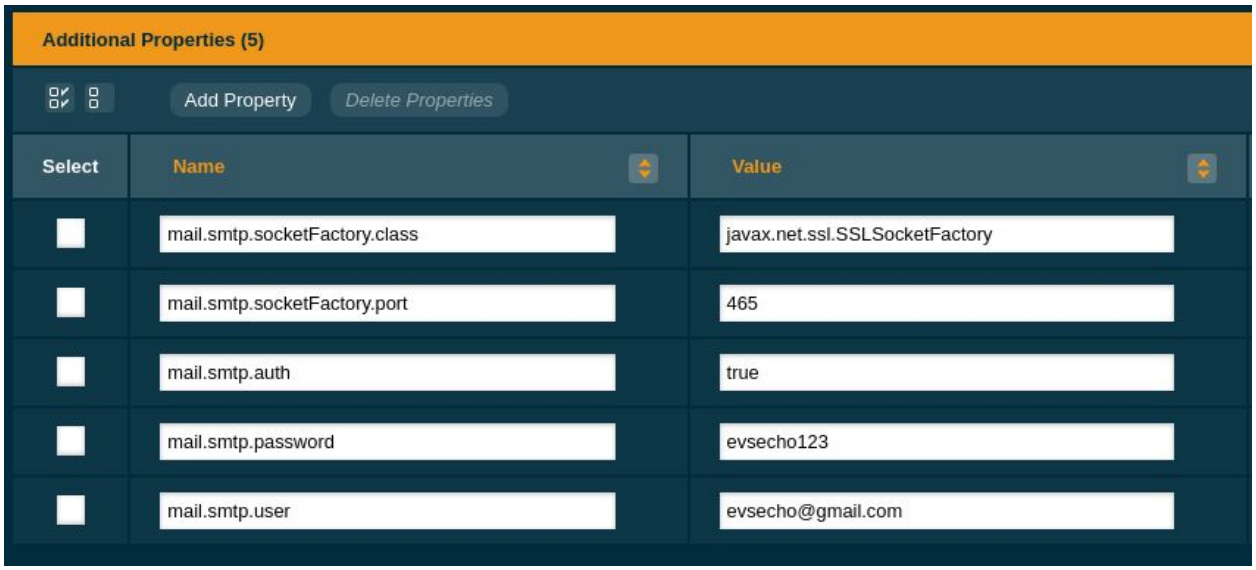
For the sending of email from the application to work properly it is necessary to configure a JavaMail session resource on Payara Server with the following settings:

JNDI Name: `mail/echoevs`
Mail Host: `smtp.gmail.com`
Default User: `evsecho@gmail.com`
Password: `evsecho123`
Default Sender Address: `evsecho@gmail.com`

Additional properties:

`mail.smtp.socketFactory.class`	`javax.net.ssl.SSLSocketFactory`
`mail.smtp.socketFactory.port`	`465`
`mail.smtp.auth`	`true`
`mail.smtp.password`	`evsecho123`
`mail.smtp.user`	`evsecho@gmail.com`





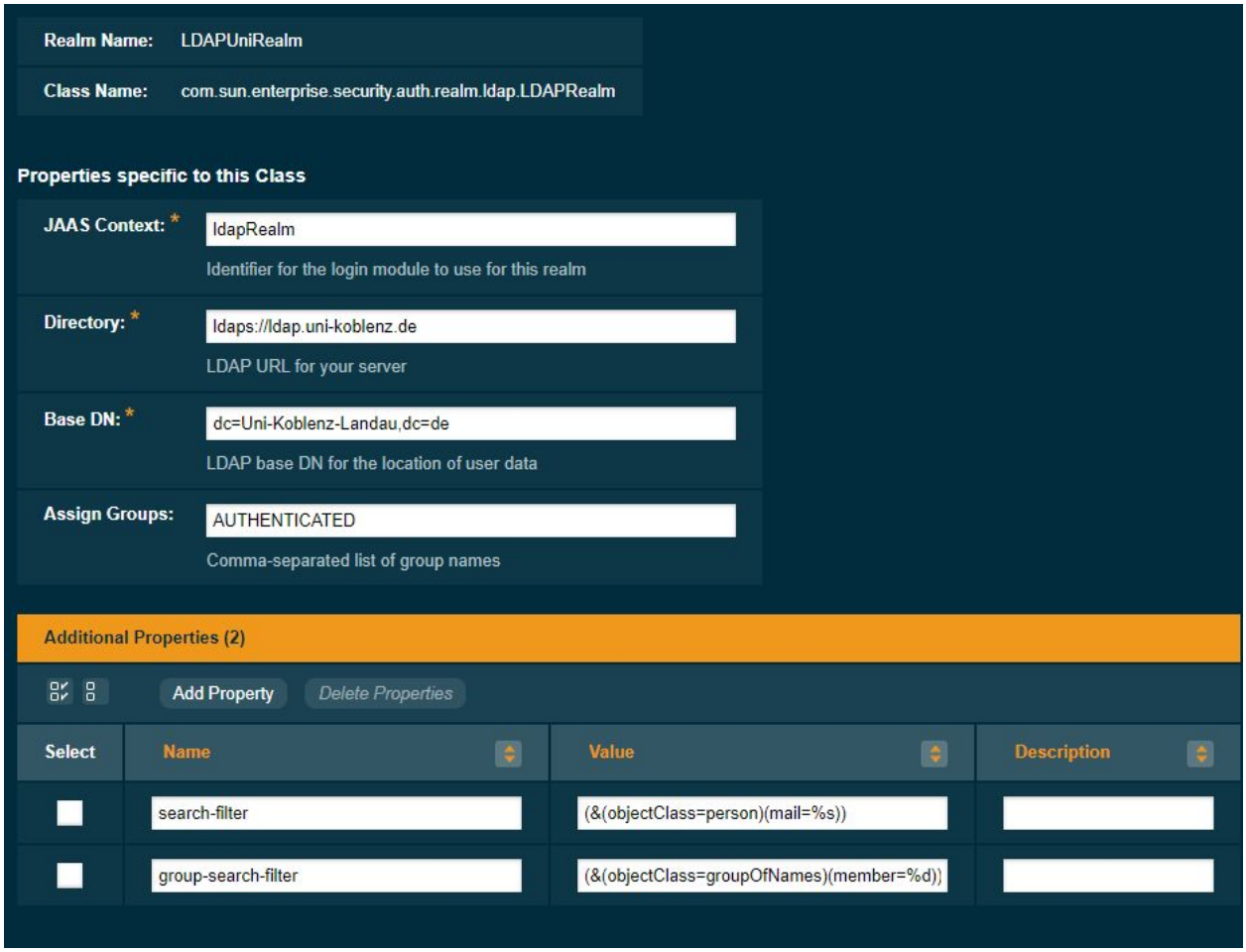
Security Realms

Create a new security authentication realms so the users can authenticate using **LDAP**:

Realm Name: `LDAPUniRealm`
Class Name: `com.sun.enterprise.security.auth.realm.ldap.LDAPRealm`
JAAS Context: `ldapRealm`
Directory: `ldaps://ldap.uni-koblenz.de`
Base DN: `dc=Uni-Koblenz-Landau,dc=de`
Assign Groups: `AUTHENTICATED`

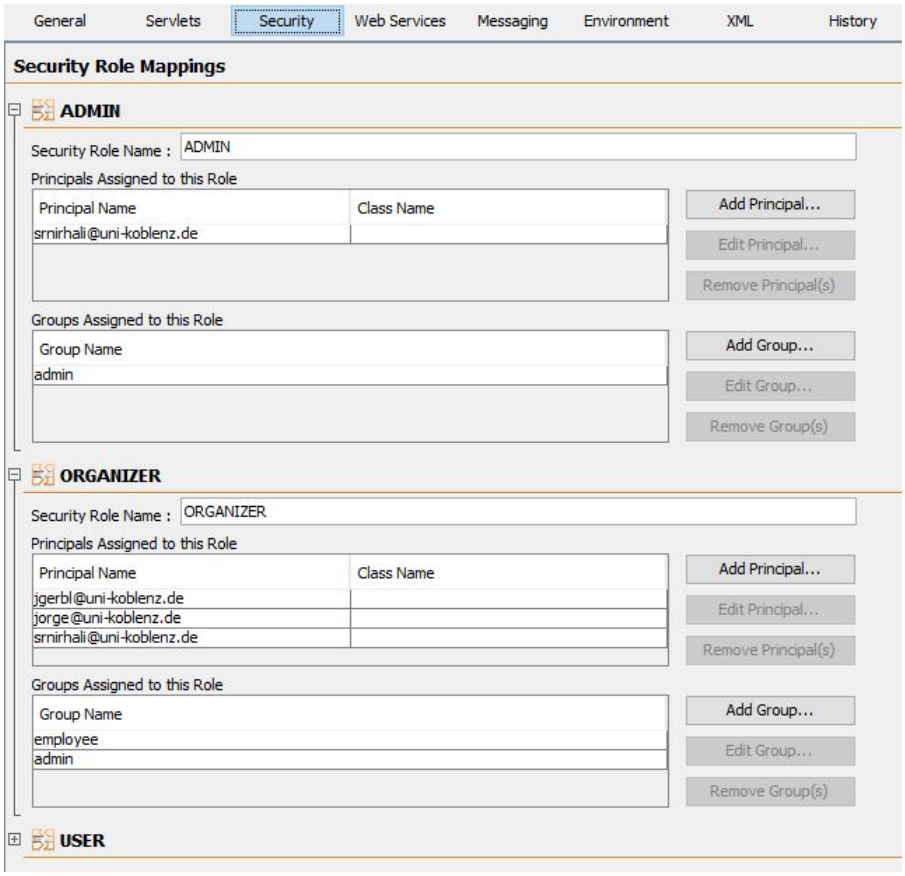
Additional Properties:

```
Search-filter          (&(objectClass=person)(mail=%s))
group-search-filter    (&(objectClass=groupOfNames)(member=%d))
```



Roles assignment for testing while using LDAP authentication:

In the glassfish-web.xml you can still edit the role assignments. This can be edited in order to test out different roles with your account without actually having the group from ldap. If you want to see the application with the admin role, you can add your email as principal there. Same goes for the organizer role.

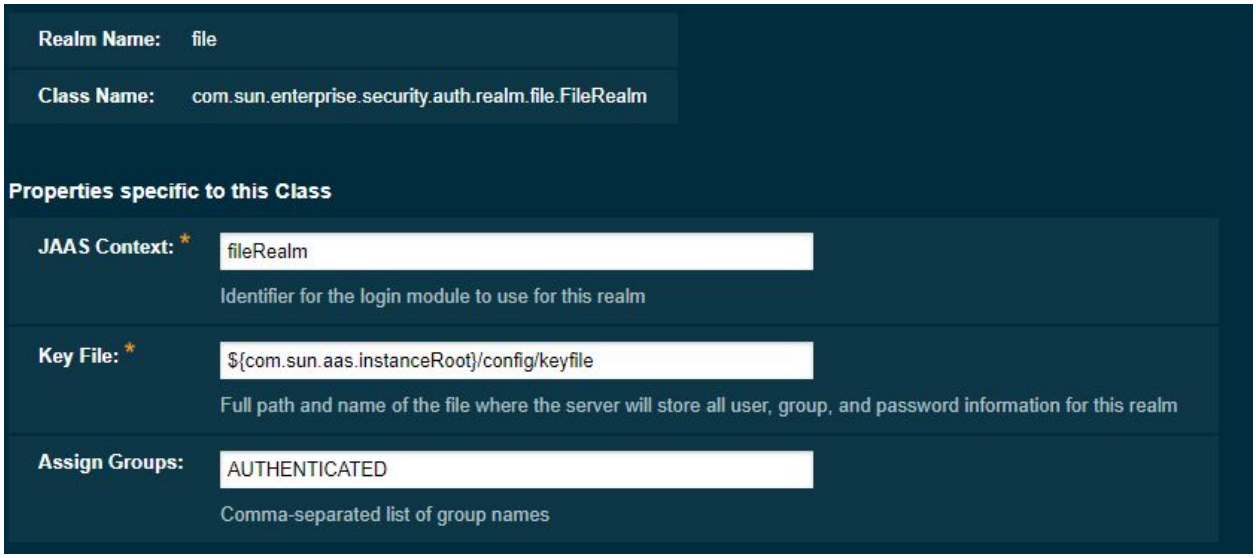


Currently only members of the employee group are considered organizer. If you add the students group to the organizer role, all students will have organizer rights as well.

Other methods of authentication: File based (not recommended)

If you don't want to use LDAP, we also provided an alternative using the file based authentication.

Realm Name: `file`
Class Name: `com.sun.enterprise.security.auth.realm.ldap.FileRealm`
JAAS Context: `fileRealm`
Key File: `${com.sun.aas.instanceRoot}/config/keyfile`
Assign Groups: `AUTHENTICATED`



In the git repository we provided you with a keyfile with some emails and roles already set up for testing. You can find it in the documents/-folder in the repository. You just have to copy that to the corresponding folder in your payara installation. (Most probably %locationOfYourPayara%\glassfish\domains\domain1\config)

Clone

Clone this repo to your local machine using:

```
git clone --branch submission https://gitlab.uni-koblenz.de/echo/javaee2020-echo
```

Deploy and build the code to run the EchoEVS project.

Note:
It is assumed that the application runs on the port 8181 on localhost, therefore the base url is:
`https://localhost:8181/EchoEVS`
In case the default port or domain is different from this one please change it on the application server configuration to make it comply with this specification.

Network

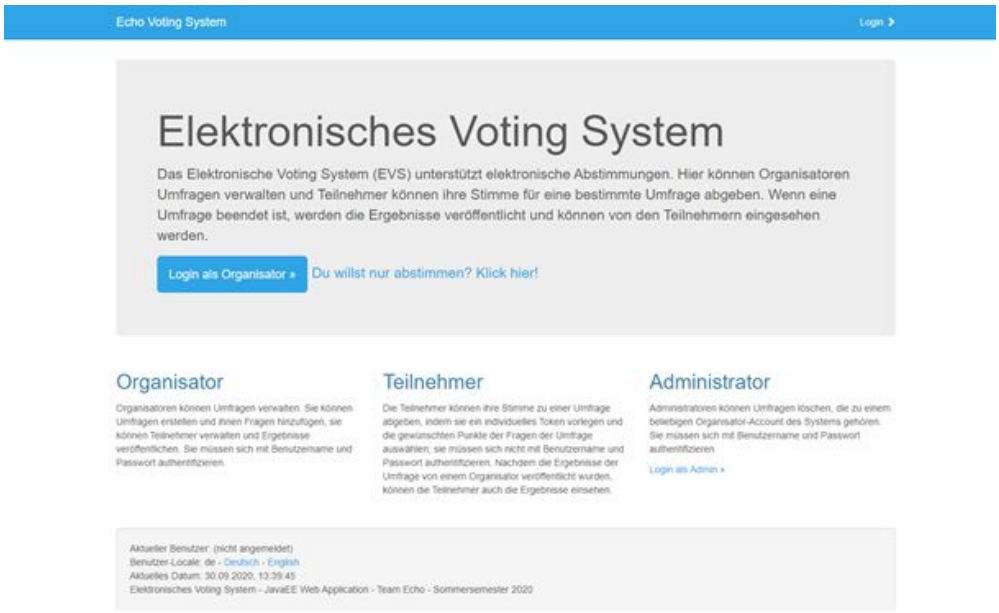
In order to be able to authenticate on the application using LDAP it is required to connect to the University VPN or to be inside the University network before executing the application.

Detailed instructions on how to connect to the VPN can be found here: L2TP-VPN

User Interface

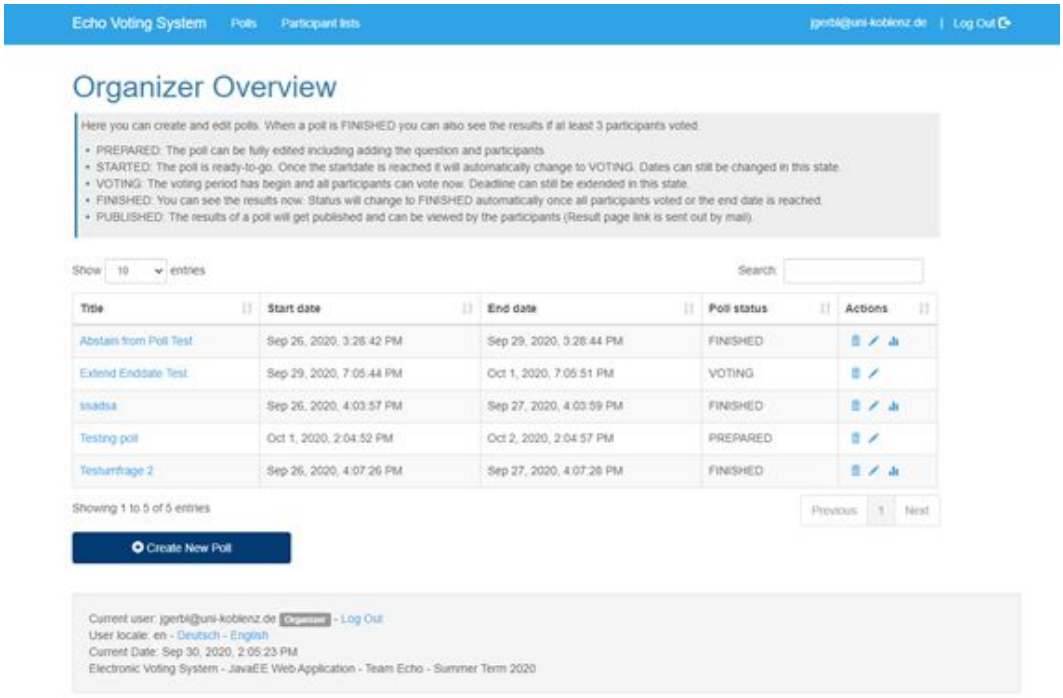
When the application is accessed it displays a welcome page, here the user can login to execute poll managing actions or only choose to vote in a poll.

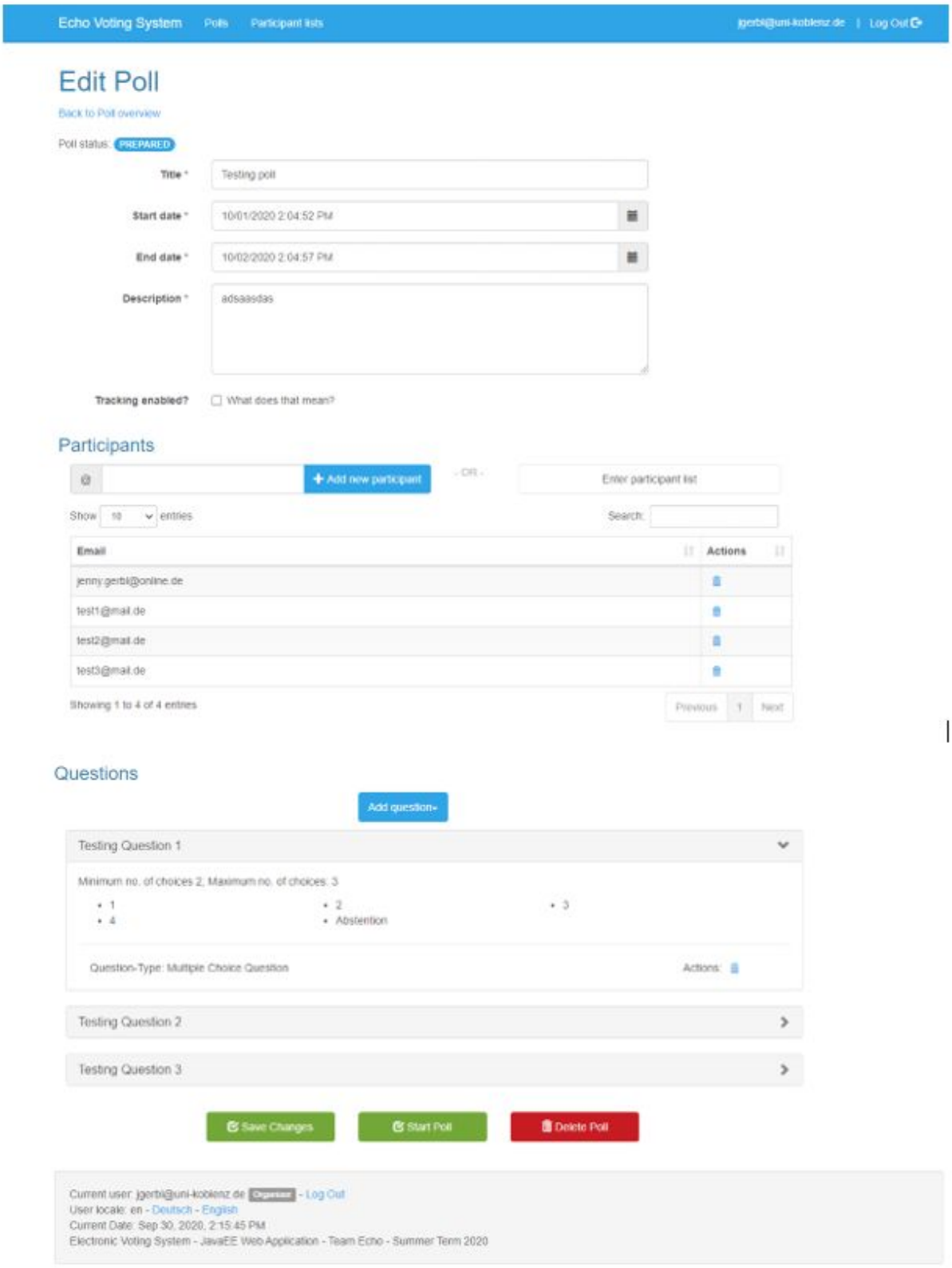
You can also switch languages (english and german are available languages) by clicking on the links in the footer.

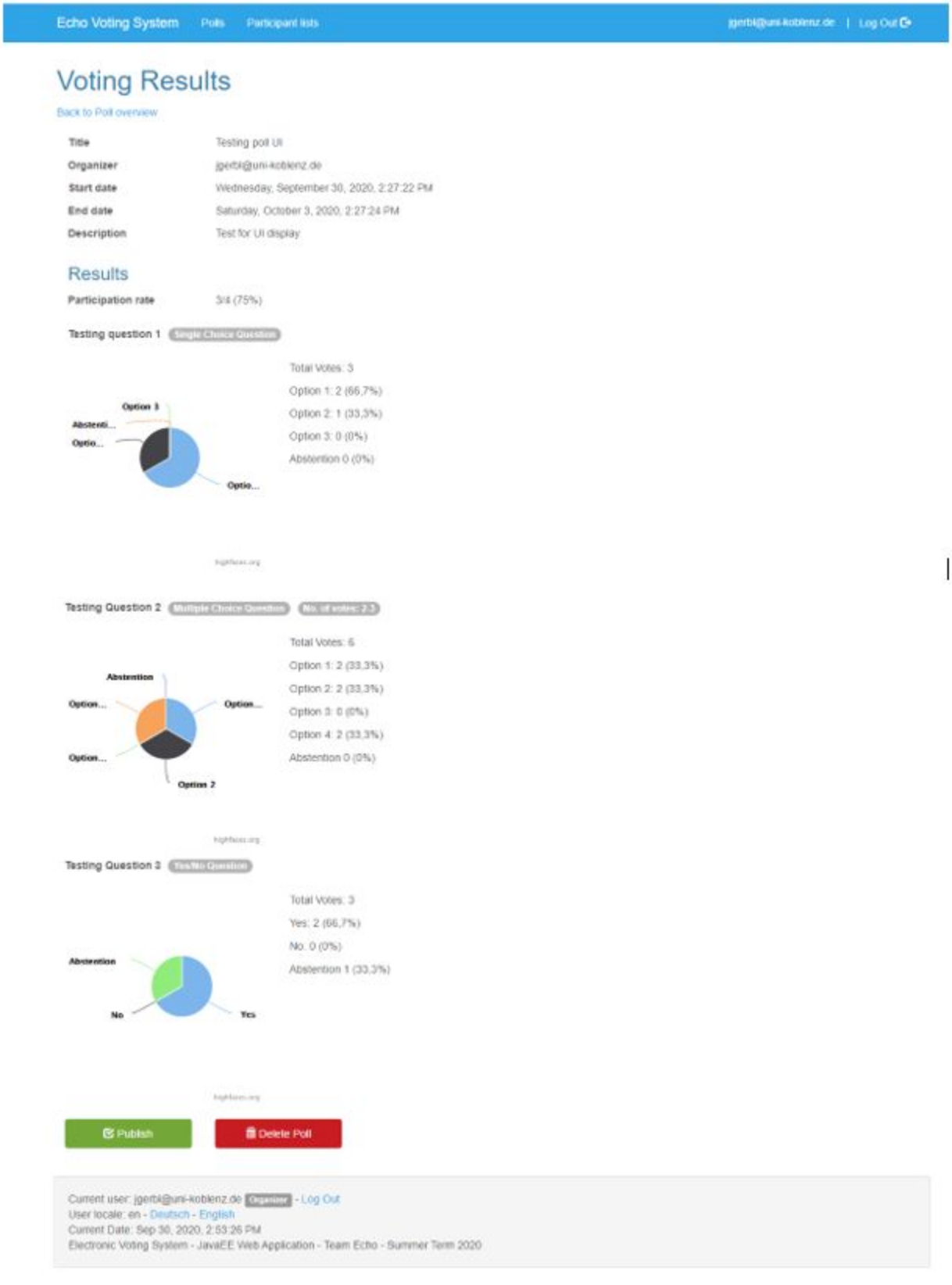


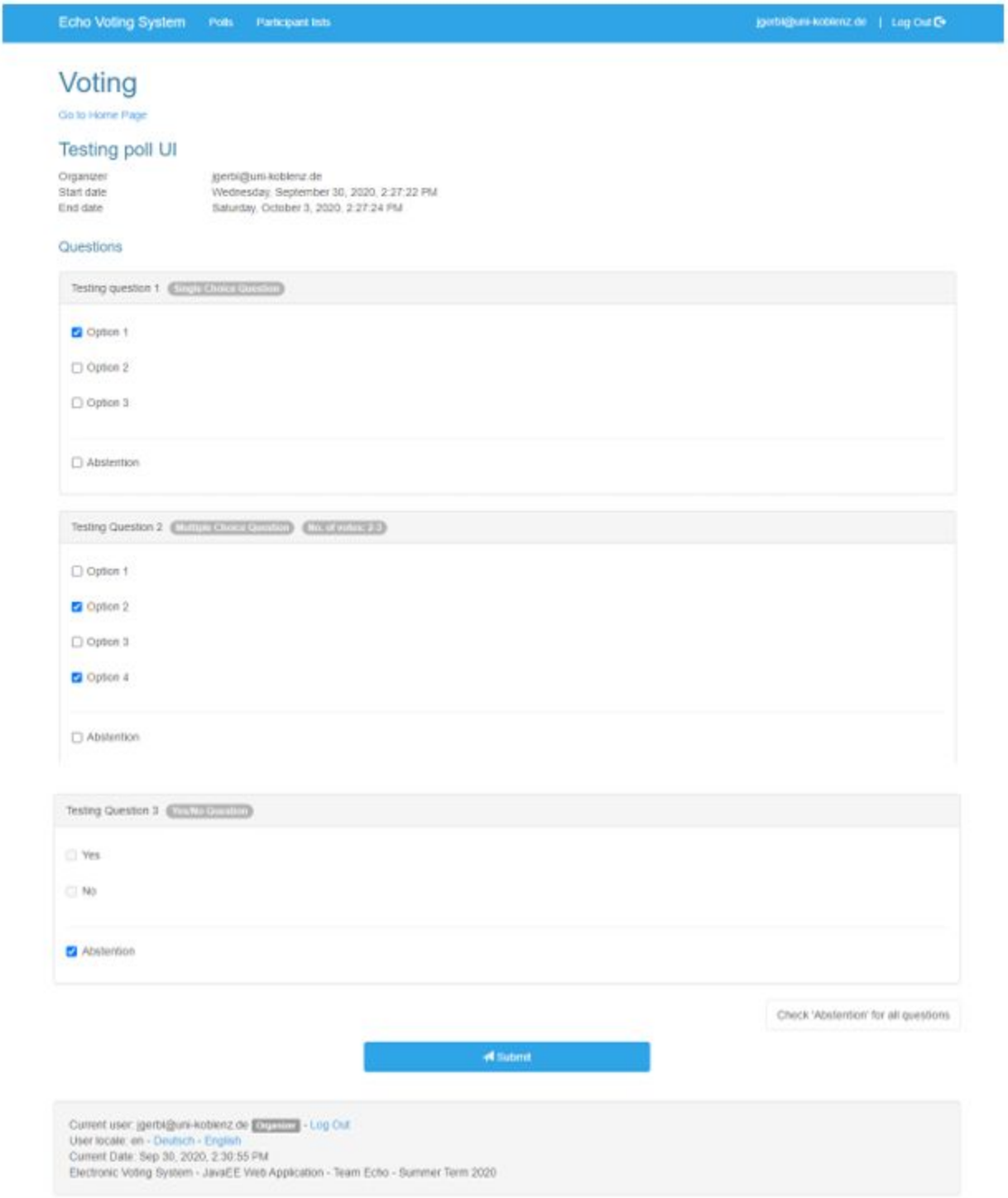
When logged in as organizer, you will have some links in the navigation bar leading to the pages of your poll overview and the overview of your participant lists. The following picture shows overview of polls of an organizer. With the admin role you will also find an additional link at the top leading to the admin overview page.

In the following you will find some screenshots of the applications user interface of different views. (This is not a complete list of all pages)




Echo Voting System
Polls
Participant lists
jgerbi@uni-koblenz.de | Log Out
Edit Poll
Back to Poll overview
Poll status: PREPARED
Title *
Testing poll
Start date *
10/01/2020 2:04:52 PM
End date *
10/02/2020 2:04:57 PM
Description *
adsaasdas
Tracking enabled?
What does that mean?
Participants
Add new participant
- OR -
Enter participant list
Show 10 entries
Search:
Email
Actions
jenny.gerbi@online.de
test1@mail.de
test2@mail.de
test3@mail.de
Showing 1 to 4 of 4 entries
Previous
1
Next
Questions
Add question
Testing Question 1
Minimum no. of choices 2, Maximum no. of choices: 3
1
2
3
4
Abstention
Question-Type: Multiple Choice Question
Actions:
Testing Question 2
Testing Question 3
Save Changes
Start Poll
Delete Poll
Current user: jgerbi@uni-koblenz.de Organizer - Log Out
User locale: en - Deutsch - English
Current Date: Sep 30, 2020, 2:15:45 PM
Electronic Voting System - JavaEE Web Application - Team Echo - Summer Term 2020


Echo Voting System
Polls
Participant lists
jgerbl@uni-koblenz.de | Log Out
Voting Results
Back to Poll overview
Title
Testing poll UI
Organizer
jgerbl@uni-koblenz.de
Start date
Wednesday, September 30, 2020, 2:27:22 PM
End date
Saturday, October 3, 2020, 2:27:24 PM
Description
Test for UI display
Results
Participation rate
3/4 (75%)
Testing question 1
Single Choice Question
Total Votes: 3
Option 1: 2 (66,7%)
Option 2: 1 (33,3%)
Option 3: 0 (0%)
Abstention 0 (0%)
Option 3
Abstenti...
Optio...
Optio...
Testing Question 2
Multiple Choice Question
No. of votes: 2-3
Total Votes: 6
Option 1: 2 (33,3%)
Option 2: 2 (33,3%)
Option 3: 0 (0%)
Option 4: 2 (33,3%)
Abstention 0 (0%)
Abstention
Option...
Option...
Option...
Option 2
Testing Question 3
Yes/No Question
Total Votes: 3
Yes: 2 (66,7%)
No: 0 (0%)
Abstention 1 (33,3%)
Abstention
No
Yes
Publish
Delete Poll
Current user: jgerbl@uni-koblenz.de Organizer - Log Out
User locale: en - Deutsch - English
Current Date: Sep 30, 2020, 2:53:26 PM
Electronic Voting System - JavaEE Web Application - Team Echo - Summer Term 2020

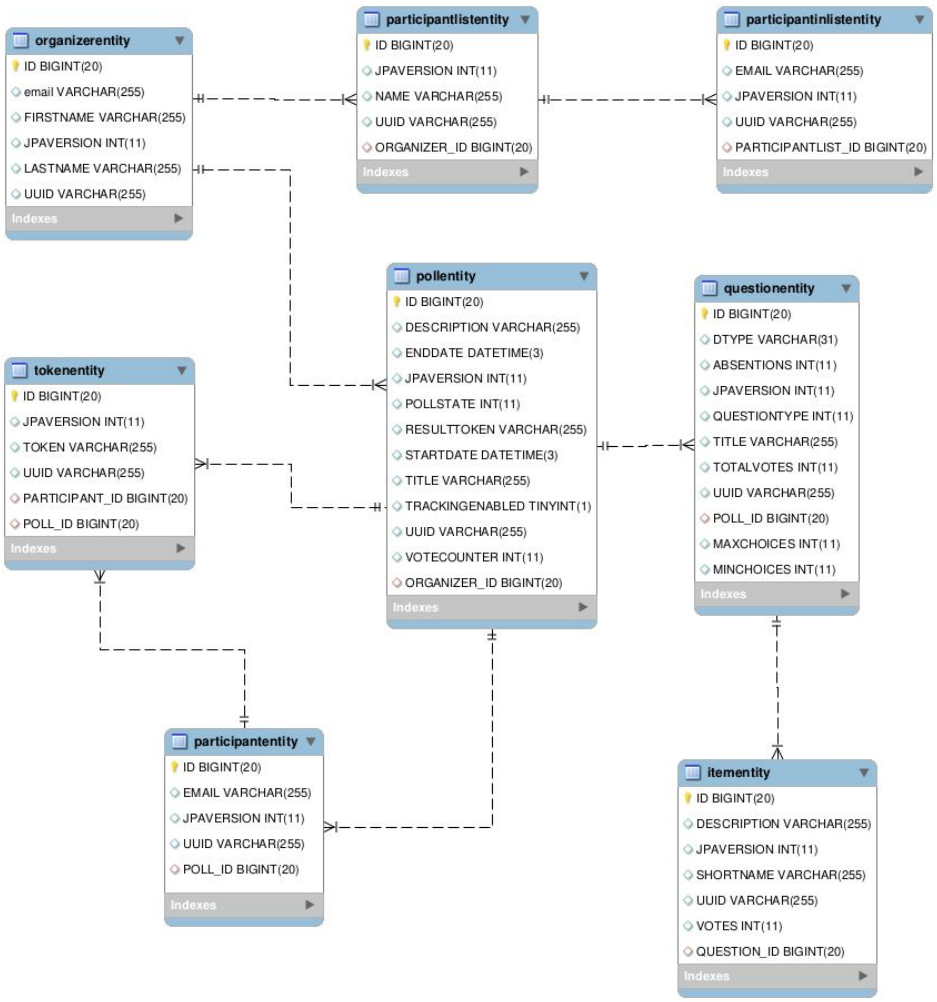

Echo Voting System
Polls
Participant lists
jgerbi@uni-koblenz.de | Log Out
Voting
Go to Home Page
Testing poll UI
Organizer jgerbi@uni-koblenz.de
Start date Wednesday, September 30, 2020, 2:27:22 PM
End date Saturday, October 3, 2020, 2:27:24 PM
Questions
Testing question 1
Option 1
Option 2
Option 3
Abstention
Testing Question 2
Option 1
Option 2
Option 3
Option 4
Abstention
Testing Question 3
Yes
No
Abstention
Check 'Abstention' for all questions
Submit
Current user: jgerbi@uni-koblenz.de - Log Out
User locale: en - Deutsch - English
Current Date: Sep 30, 2020, 2:30:55 PM
Electronic Voting System - JavaEE Web Application - Team Echo - Summer Term 2020

Models

The models used for the design and development of the EVS application can be found in the repository on the **models/** folder.

There you will one Astah-file, containing the domain model, the use cases and a state machine diagram for poll state changes, as well as several pictures of auto-generated class models (based on our code) and a picture of our database model.

Database Model



Code Documentation

There is a complete JavaDoc documentation of the code available in our git repository. It can be found in the `documents/JavaDoc/`-folder. There you'll find the Javadoc for both projects there. Open the `index.html` in the specific folder as a starting point to take a look at the documentation.

This version of the JavaDoc was created on the 29.09.20, so it shows the most recent version of our code. You can also generate the JavaDoc on your own via Netbeans. Just open the project in Netbeans and click on Run > Generate Javadoc.

Other notes & assumptions we made

We assume that the application is available under the url https://localhost:8181/EchoEVS/ as this url is used when sending emails. If your application uses a different url you will need to change the SERVER_URL constant in the mailservice and edit that to your needs.

Besides the submission branch in our git repository you will also find another branch called TimerServiceBranch. On this branch you will find an alternative way on how to manage timed events. (On submission branch we are using database events) These changes did not make it to our submission branch as it got finished close to the submission deadline and therefore we were not able to test it properly with these new changes.

Project Requirements (PM02)

Completed Requirements

List of completed requirements

We implemented **all** of the mandatory requirements:

1. **Polls**: PO01 (PO01.1, PO01.2, PO01.3), PO02, PO03
2. **Types of questions**: QU01, QU02, QU03, QU04
3. **Pollstates**: PS01, PS02, PS03, PS04, PS05, PS06
4. **Organizers**: OR01 (OR01.1), OR02, OR05 (OR05.1), OR06

5. **Administrators**: AD01, (AD01.1), AD02
6. **Participant Lists**: PL01 (PL01.1, PL01.2), PL03 (PL03.1), PL04, (PL04.1)
7. **Tokens**: TO01, TO02, TO03, TO04
8. **Anonymity**: AN01, AN02
9. **Participation tracking**: all optional ones
10. **Submitting a vote**: SV01, SV02 (SV02.1), SV03, SV04, SV05, SV06, SV07, SV08, SV10
11. **Results**: RS01, RS02, RS03, RS04, RS05, RS06, RS07, RS08
12. **User Interface of the EVS**: UI01, UI05
13. **Internationalization**: IN01 (IN01.1, IN01.2), IN03, IN04, IN05
14. **Architecture**: AR01, AR02, AR03, AR04, AR05, AR06, AR07
15. **Security**: SE02, SE04
16. **Project Manual**: PM01, PM02, PM03, PM04, PM05

Optional requirements we did as well are:

- **(PS07)** The EVS shall provide organizers with the ability to extend (but not shorten) the voting period when a poll is in state STARTED or VOTING.
- **(OR01.2)** The EVS shall be able to identify university employees using the LDAP service provided by the GHRKO computing center.
- **(PL02)** The EVS should provide organizers with the ability to add participants from outside the university.
- **(PL04.2)** The notification email should contain a hyperlink that already contains a pre-filled token of the participant so that the participant can immediately begin to vote.
- **(PL06)** The EVS shall provide organizers with the ability to store participant lists for easy reuse in subsequent polls.
    - **PL06.1** The EVS shall be designed in a way that the stored participant lists are private to each organizer.
    - **PL06.2** Each stored participant list shall have a unique name (scope is per organizer).
- **(PT01)** The EVS shall provide organizers with the ability to track the participation for a poll.
- **(PT02)** If participation tracking is enabled for a poll, the EVS should provide organizers with the ability to activate automatic reminders to participants who didn't vote.
- **(SV11)** The system should provide participants with the ability to abstain from voting for a complete poll (this is equivalent to abstention from all questions).
- **(RS09)** The EVS should display the results in graphical form (e.g., as pie chart).
- **(UI03)** The EVS should use third-party JSF components (e.g. bootsFaces, myFaces, richFaces).
- **(UI04)** The EVS should use advanced technologies (e.g. AJAX) supported by JSF components) to enhance user experience.
- **(UI06)** The EVS should be designed in a way that it supports mobile device browsers (limited display area, portrait page display instead of landscape).

Missing Requirements

Which part(s) of the task have not been accomplished (if there are any) and why have they not been accomplished?

We did not implement some of the following **optional** tasks:

- **(PO04)** The EVS should provide organizers with the ability to clone a poll (period, description, questions, participant list).
- **(OR03)** The EVS should provide organizers with the ability to preview their polls in the participant view.
- **(OR04)** The EVS should provide organizers with the ability to add further organizers to a poll. The poll is then owned by each of the organizers.
- **(SV09)** The EVS shall ensure that subsequent participants using the same browser window can not restore the previous choices (e.g. by the „go back“ function or auto fill capabilities of browsers).
- **(UI02)** The EVS should use third-party CSS libraries (such as Bootstrap).
  Reason: No need for it, as BootsFaces includes the Bootstrap Css.
- **(IN06)** The EVS should support multilingual text also in the data items (e.g., title, short name, description of polls, questions, and items) so that the participants can view their options in their preferred language.
- **(IN07)** The EVS should send notification and reminder mails to participants in their preferred language.
  Reason: As everyone, not only university members, can participate in a vote we do not have an option to store the language of an participant. There is also no login for participants where they could set up their language.
- **(SE01)** If LDAP is not used, the EVS shall be designed in a way that all passwords are stored in hashed/encrypted form.
  Reason: We are using LDAP.
- **(SE03)** The EVS should be designed in a way that network communication with the database server is encrypted

If no specific reason is stated, the optional task was not done due to time reasons.

Decisions on Requirements (PM04)

Decisions made to change requirements shall be documented in the project manual. E.g., if you feel that the requirements are not detailed enough, you may add more details.

PO02 The number of questions of a poll shall be at least 1.

- This will be checked if you want to start a poll. In PREPARED state this can still be empty, as you are still editing the poll.

QU04 The items of a question shall have a short name and an optional description.

- As the short name is not limited in length in our application, there is no need to set a description. Later on we decided to still include it but not very prominent in the UI. The optional description is shown as a tooltip if you hover over the item name.

PS01 The state of a poll shall be one of PREPARED, STARTED, VOTING, FINISHED.

- We added a state called PUBLISHED, which indicates that this poll got published and can be viewed by the public if the correct resultToken was given.

(PS07) The EVS shall provide organizers with the ability to extend (but not shorten) the voting period when a poll is in state STARTED or VOTING.

- We decided that if the poll is still in STARTED, you can still edit the start date, as you might want it to start a little bit earlier or entered a wrong start date. But once VOTING is reached, you can not edit the start date anymore.

OR06 As soon as a poll has at least 3 participants and is in state PREPARED, the EVS shall provide organizers with the ability to start the poll.

- We included some more restrictions e.g. end date has to be in the future or that the poll has to have at least 1 question.

AD03 If LDAP is not used, the EVS shall provide administrators with the ability to manage organizer accounts

- We did not do that, as we use LDAP.

(PT02) If participation tracking is enabled for a poll, the EVS should provide organizers with the ability to activate automatic reminders to participants who didn't vote.

- We decided against automatic reminders, as we wanted to reduce the amount of emails sent, especially if the enddate is quite close to the startdate. (So no automatic reminder is sent when the startdate has just been reached but the end date is e.g. just one day away)

The organizer can view the participation rate and decide based on that if he wants to send reminder mails to those who haven't voted. He/She has to click on the "Send Reminders" Button in the Edit Poll-View.

SV07 The EVS shall provide participants with the ability to cancel a voting, e.g., by closing the browser, or by clicking a cancel button. The token used in cancelled voting shall be reusable later.

- There is no cancel button as you can just close the browser/tab or just click on any other link (e.g. startpage)

(SV11) The system should provide participants with the ability to abstain from voting for a complete poll (this is equivalent to abstention from all questions).

- There is a button that marks all 'abstention' checkboxes. You still have to submit the vote by clicking on the button. (So it's not done by mistake or maybe the person still wants to answer one other question but abstain from all of the other questions)

(UI03) The EVS should use third-party JSF components (e.g. bootsFaces, myFaces, richFaces).

- We are using BootsFaces and HighFaces

(UI04) The EVS should use advanced technologies (e.g. AJAX) supported by JSF components) to enhance user experience.

- Ajax is heavily used in the "Edit poll"-View and partly on other views as well.

UI05 The EVS shall be designed in a way that it supports at least one of the following desktop/laptop web browsers: FireFox, Safari, Chrome in the most recent stable version available at delivery of the project.

- We fully tested in Chrome (and Firefox). Other browsers possibly work as well but we haven't tested them, so we can not guarantee that.

(UI06) The EVS should be designed in a way that it supports mobile device browsers (limited display area, portrait page display instead of landscape).

- As we are using BootsFaces grid system in all views, these views are all viewable and usable on a mobile device. (only hover up-tooltips are not shown)

Issues Occurred on Implementation (PM03)

Which technical, organizational or team related issues and problems occurred during the development?

Technical issues

One team member faced some issues with the IDE. This was shown as follows: When closing Netbeans, the application sometimes enters a corrupted state. When Netbeans was opened again, it was no longer possible to deploy applications with Payara. However, the error messages seemed to point to completely different "errors". To fix the problem, you had to clear Netbeans' cache and then reconfigure everything. This had to be done by the team member almost daily, sometimes several times a day.

Other issues were related to the architecture design, while trying to follow all of these architectural requirements we sometimes faced some doubts regarding where to put specific functionality.

We also faced some difficulties with the editing of questions. As we wanted to edit a poll in one view without being redirected to another view we implemented the edit poll view with a lot of ajax calls. The editing of questions was especially difficult. We wanted to make each panel editable and once you click on the edit button of the question, it should show up all the items, the title and other settings as inputs. If you are finished editing, you should save it. But as we have every input in the same form, when clicking on the “Save poll” button, it would eventually save the not saved edited questions leading to some unwanted changes. In order to change that we would have to implement some functionality with javascript or change the structure of our edit poll view. Even though we had some good ideas to solve the issue, we thought there were more urgent tasks to do and decided to do the editing of questions later. (as you can still remove and add another question) Due to time reasons this wasn’t implemented in the final result.

Another difficult task was to come up with a solution for the generation of unique tokens that should never be generated again. We tried several methods but we had doubts about the uniqueness over the lifetime of the application (meaning the same token can only be generated once) We came up with the idea to include the timestamp in an hashed way that was added to a randomly generated string. With a timestamp we made sure that a token can not be generated twice, with the random string part and the hashing of the complete token we wanted to make sure it can not be guessed.

One smaller problem we faced was how to configure the LDAP authentication so only employees could have the organizer role. As we did not know the group names we weren’t

sure how to test if it's working correctly. After the configuration got shared in mattermost, we could finally be sure that it is working.

Also we tried making the hyperlink that is sent out by the mailserive not being hard coded. We tried a couple of things but in the end did not find a feasible solution. Therefore we decided to include the server url at least as a constant string that will be used in the messages. When the server url changes at least you only have to change it at one place instead of changing every mail message.

Organizational issues

We were confronted with a few organizational problems. Due to the current situation of online teaching all team members had time management problems. There were a lot of weekly assignment submissions in other lectures and the overall time was very limited.

Due to the fact that there were no tutorial slots in which to physically meet, the team members met relatively late in the groups (via Skype).
When the weekly submissions for other lectures decreased, there was already time for exam preparation. It was generally very difficult at this time to have meetings where all team members had time. So most of the work was done after most exams were over.

This led to some other timing problems. Although we finished the project on time, we ran short of time. So we had to reduce the amount of optional tasks to make it work. Shortly before delivery we found a few errors within our application. These had to be fixed at very short notice. One of them was, for example, that we noticed very late that only the "employee" role should act as organizer. At that time we already had a fully tested application, but the role assignment was different, because everyone who could log in via LDAP got the "organizer" role and we did not have a role without rights. So this was changed last minute.

But the reason for this time-related issues was not only because we started relatively late but also because of some team-related issues mentioned in the next part.

Team-related issues

Right at the beginning we already had an inactive group member. This member did not participate in the first team meetings, but also said that he could not make it to the meetings due to personal reasons. Later, however, the team member did not get in touch with us, so he did not tell us that he would not participate in the project. All communication was ignored from that point on.

In September, in the middle of the development process, another team member surprisingly left us. The tasks previously assigned to this person were postponed over time by that person (before leaving the group) and then returned to the team unprocessed at the last moment. As a result, we had to reallocate the person's tasks to three remaining people at short notice leading to some time issues.

Description of our methods & operating principles

How did we manage the project? How did the team worked?

During the lecture period we had only irregular meetings. The reason for this was the lack of time of all team members. This was already mentioned in the previous chapter under "Organizational problems". At that time, development work was only sporadically done.

In the middle of August (after the exams of all team members were over) the work was continued on a regular basis. Until the submission at the end of September, a meeting was held every 4 days. The goal of the meetings was to discuss the distribution of work and to review the status of the current tasks.

For us to connect the application's architecture and requirements to its visual design we decide to create some Wireframes, they helped us to maintain an unified interface structure and a general idea of how the application should work at first. EVS Wireframes

We used Trello, a lightweight project management tool, to keep track of tasks and bugs. This was a very helpful tool and used a lot during meetings to have an agenda but also to know who is working on what.
Our trello board can be viewed under https://trello.com/b/sftJZV10/javaee-web-application

In the backlog tab all pending tasks were listed, sorted by importance. For each task there was, depending on its complexity, a description or checklist, so that it is clear what has to be done. Optional tasks were marked by a blue label, and bugs with a yellow one. There are still some optional tasks on the backlog as we haven't managed to do all of them.
When a team member started a task, the task tile was moved to "In Progress". At completion, it was moved to "Code Review & Testing" so that the other team members could review the work. If any bugs were found, they were recorded in the "Bugs" tab.
If further documentation was needed, such as configurations, there was a tab for documentation. When this was extensively tested, the task tile could be moved to "Done".

Team Time Track (PM05)

Record of the time spent on the project.
How was the workload distributed in your team? (a record of topics and working hours per team member)

Activity/Task	**Shivam Nirhali**	**Jennifer Gerbl**	**Jorge Gavilan**
Creation of Entities/ DOA/ DTOs / Setup	5 hours	3 hours	1 hour
Login functionality	2 hours	1 hour	--
Creation of Polls (+ Delete)	2 hours	1 hour	1 hour
Editing of Polls	2 hours	10 hours	5 hours
Creation/ Editing of Participant Lists (+ Delete)	2 hours	3 hours	--
Adding of participant to polls / lists	--	7 hours	1 hour
Adding of questions to polls	--	9 hours	--
PollState changes / Events	5 hours	1 hour	2 hours
Administrator functionality	1 hour	--	--
MailService	3 hours	2 hours	4 hours
Token Generation	1 hour	--	4 hours
Participant Voting	2 hour	8 hours	1 hour
Voting Results & Publish Results	--	6 hours	--
More general tasks			
Testing	3 hours	3 hours	5 hours

Bug Fixing	3 hours	7 hours	2 hours
Localisation & Improving Frontend	1 hour	8 hours	1 hour
Code Review	2 hours	4 hours	2 hours
Documentation (incl. JavaDoc)	--	6 hours	7 hours
Modeling	1 hour	2 hours	7 hours
Meetings	8 hours	8 hours	8 hours

There are still some tasks not mentioned here, so the real spent time is higher than depicted times in this time tracker. Also the entered times are an approximation based on the commits done as the recording of time was done not right after the task was finished.