**Course Project – VOTES!**

# 

**Team:** Yankee

**Members:** Usman Ahmad

Keliana Fazlli

Prantik Goswami

Mahmoona Shahzadi

Ahmad Alsamman

# Table of contents

Contents

[1](#_Toc21892712)

[Table of contents 2](#_Toc21892713)

[Introduction 3](#_Toc21892714)

[Project Requirements 4](#_Toc21892715)

[Functional Requirements 4](#_Toc21892716)

[Non-Functional Requirements 8](#_Toc21892717)

[Installation of the application 9](#_Toc21892718)

[Setup of the system 9](#_Toc21892719)

[Create a JDBC connection 9](#_Toc21892720)

[Installation 16](#_Toc21892721)

[Beginning 16](#_Toc21892722)

[Application 17](#_Toc21892723)

[Glossary 23](#_Toc21892724)

# Introduction

In the Merriam-Webster dictionary the word Voting is described as “an expression of opinion and preference” which draws our interest to work on this topic as we personally believe that, everyone should vote, because everyone has an opinion. In our work we mainly focus on the poll system of the university body where the students are given the right to poll on the web and cast their votes in any field of choice. It is gradually becoming difficult to conduct live meetings in order to consider an opinion of the masses, therefore web polling results as a fairly good solution to know and understand the opinion of the crowd.

Our poll system supports internalizations, clear navigation structure, logical design, and uniformity of design.

Our application hence functions as follows:

Every member of the university can be an organizer and create polls, while to be a participant and vote, it doesn’t require to be a member of it. By getting a token, the participants can choose to vote for yes/no questions, one or more options or even abstain.

Participants who haven’t voted yet, can also receive an email by the system to remind them about the poll.

The system supports anonymity and after the voting has started, the options and the list of the participants cannot be changed anymore. At the end, the results are shown to the organizer.

# Project Requirements

## Functional Requirements

1. **Polls**

|  |  |  |
| --- | --- | --- |
|  | **Requirement** | **Complete/Incomplete** |
| 1.1 | Support electronic polls with one or more items | Completed |
| 1.2 | Supports unique titles | Completed |
| 1.3 | Poll includes a description | Completed |
| 1.4 | Poll has a voting period | Completed |
| 1.5 | Poll has at least one item | Completed |
| 1.6 | System supports multiple items into one poll | Completed |

1. **Poll states**

|  |  |  |
| --- | --- | --- |
|  | **Requirement** | **Complete/Incomplete** |
| 2.1 | System implements four states: PREPARED, STARTED, VOTING, FINISHED | Completed |
| 2.2 | Submit to finish once voted | Completed |

1. **Organizers**

|  |  |  |
| --- | --- | --- |
|  | **Requirement** | **Complete/Incomplete** |
| 3.1 | Allow university members to act as organizers | Completed |
| 3.2 | Organizer identification: username & password | Completed |
| 3.3 | Supports multiple arbitrary polls | Completed |

1. **Participants**

|  |  |  |
| --- | --- | --- |
|  | **Requirement** | **Complete/Incomplete** |
| 4.1 | Minimum 3 participants required in order to assert anonymity | Completed |
| 4.2 | Identification of participants via Email Address | Completed |
| 4.3 | Supports participants outside University | Completed |
| 4.4 | The information mail includes the title, the start & end dates, the no. of participants, token | Completed |

1. **Participant lists**

|  |  |  |
| --- | --- | --- |
|  | **Requirement** | **Complete/Incomplete** |
| 5.1 | Supports modification of the participant list until a poll is started | Completed |

1. **Tokens**

|  |  |  |
| --- | --- | --- |
|  | **Requirement** | **Complete/Incomplete** |
| 6.1 | The token is chosen randomly | Completed |
| 6.2 | The token is unique | Completed |
| 6.3 | The token is too long & secure so nobody creates a valid token | Completed |

1. **Anonymity**

|  |  |  |
| --- | --- | --- |
|  | **Requirement** | **Complete/Incomplete** |
| 7.1 | Supports anonymity | Completed |
| 7.2 | Impossible to identify which participant voted. Guaranteed for polls with participation tracking | Completed |
| 7.3 | Token isn’t associated with a vote | Completed |

1. **Submitting** **a vote**

|  |  |  |
| --- | --- | --- |
|  | **Requirement** | **Complete/Incomplete** |
| 8.1 | Supports web page for vote submission | Completed |
| 8.2 | Provides input field for the token | Completed |
| 8.3 | Display items after the token is verified | Completed |
| 8.4 | Supports a button for vote submission | Completed |
| 8.5 | The token used a vote submission cannot be used again | Completed |
| 8.6 | Supports canceling a voting | Completed |
| 8.7 | The token used in a cancelled voting can be used later | Completed |

1. **Abstain from voting**

|  |  |  |
| --- | --- | --- |
|  | **Requirement** | **Complete/Incomplete** |
| 9.1 | Supports abstention in a poll | Completed |

1. **Types of items**

|  |  |  |
| --- | --- | --- |
|  | **Requirement** | **Complete/Incomplete** |
| 10.1 | Supports different types of items | Completed |
| 10.2 | Items have a title | Completed |
| 10.3 | Supports short name & description for options of the items | Completed |
| 10.4 | Support YES/NO items (description is also yes/no) | Completed |
| 10.5 | support 1 OF N item/ choosing 1 or more options | Completed |

1. **Results**

|  |  |  |
| --- | --- | --- |
|  | **Requirement** | **Complete/Incomplete** |
| 11.1 | View the results after the voting period finishes | Completed |
| 11.2 | No possibility to view the results in the “started” & “voting” states | Completed |
| 11.3 | Do not present the result if less than 3 submitted votes | Completed |
| 11.4 | Presents the number of votes for every option of each item | Completed |

## Non-Functional Requirements

1. **User interface**

|  |  |  |
| --- | --- | --- |
|  | **Requirement** | **Complete/Incomplete** |
| 12.1 | Provides a user interface for both desktop & laptop browsers | Completed |
| 12.2 | JFS is used for a user interface | Completed |

1. **Architecture**

|  |  |  |
| --- | --- | --- |
|  | **Requirement** | **Complete/Incomplete** |
| 13.1 | Contains the design and the rules shown in lecture/lab | Completed |
| 13.2 | The naming g conventions for the project were strictly followed | Completed |

1. **Security, encrypted communication**

|  |  |  |
| --- | --- | --- |
|  | **Requirement** | **Complete/Incomplete** |
| 14.1 | Saves passwords in encrypted form | Completed |

1. **Internationalization**

|  |  |  |
| --- | --- | --- |
|  | **Requirement** | **Complete/Incomplete** |
| 15.1 | Supports a user interface in German or English language | Completed |

1. **Browser support**

|  |  |  |
| --- | --- | --- |
|  | **Requirement** | **Complete/Incomplete** |
| 16.1 | Supports web browsers such as Safari, Firefox or Chrome | Completed |

# Installation of the application

## Setup of the system

Voting System is an application on the web which is able to provide functionalities to the clients on a server machine.

**CLIENT-SIDE APPLICATIONS**

To use the application, the latest JavaScript is needed in order to be accessible on the browser.

**SERVER-SIDE APPLICATIONS**

In order to host the application and later run it, we will user Payara application server.

**DATABASE SERVER**

Voting System application will be able to store all the data in the database.

**JAVA**

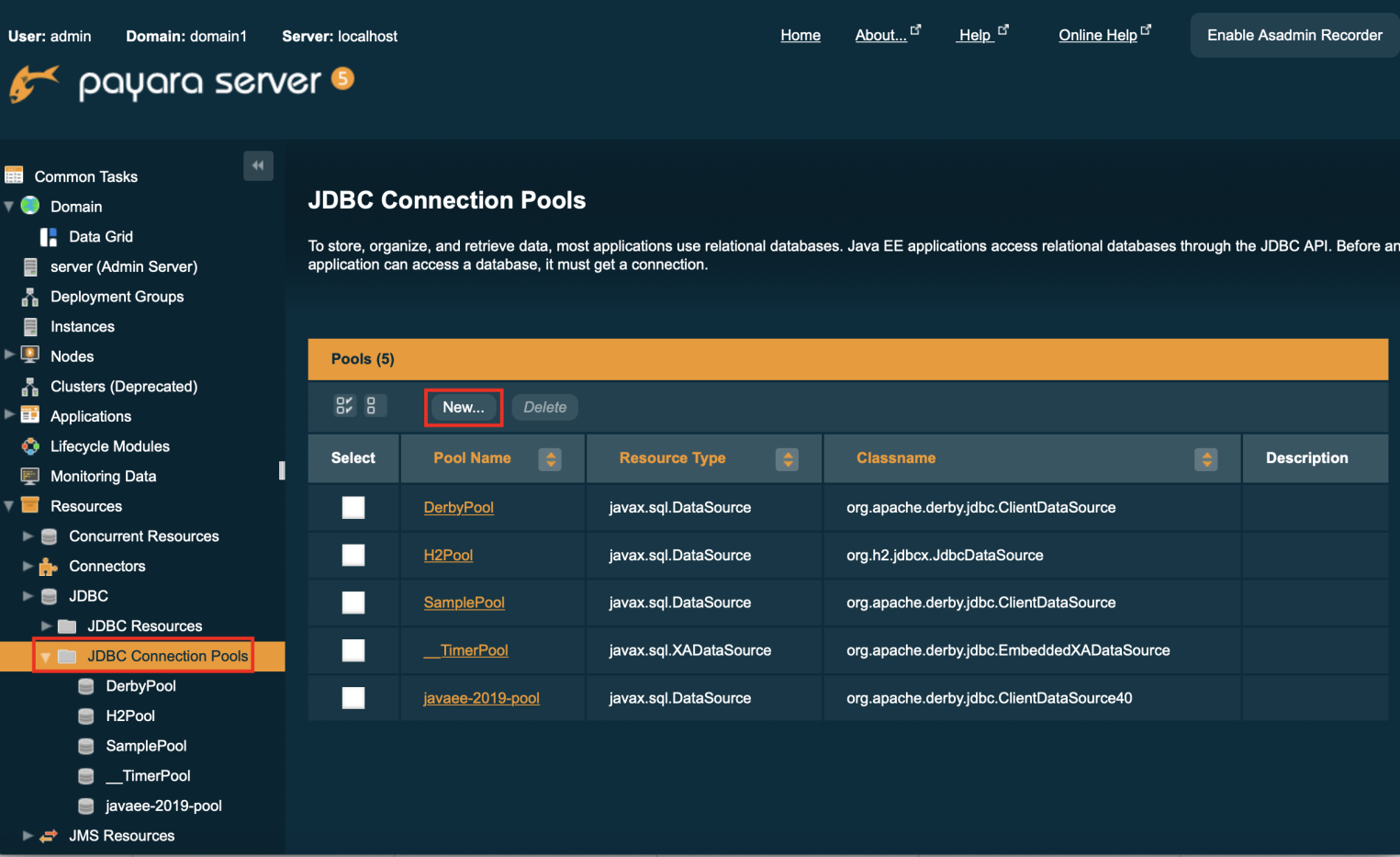
In the system server needs to be installed Java Developer Kit (JDK) or Java Runtime Environment (JRE) platform.

## Create a JDBC connection

**1)** Start the server from NetBeans

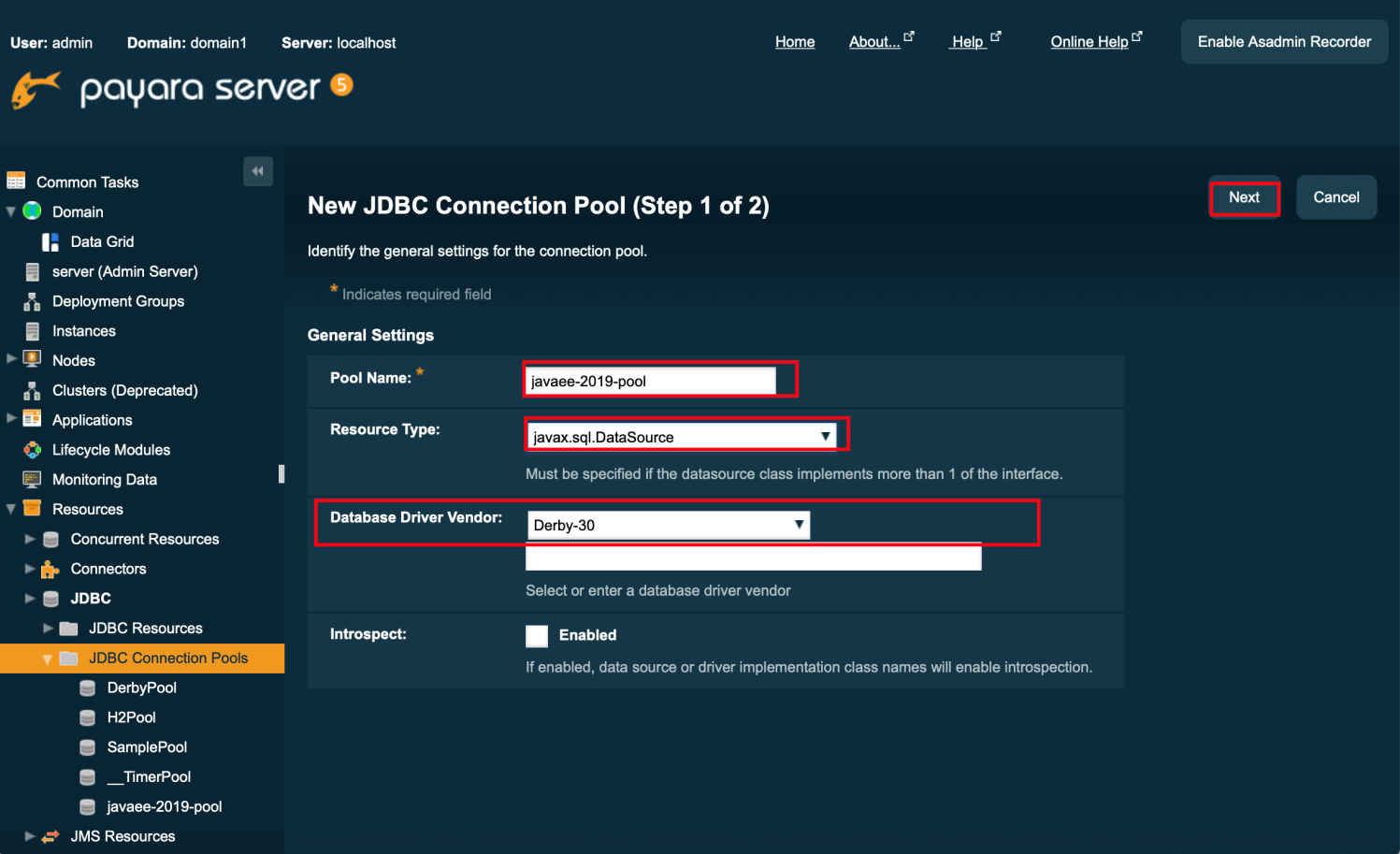
**2)** Start Payara Server from Terminal (How to start the server: <https://docs.payara.fish/getting-started/getting-started.html>)

**3)** Write in your default browser <http://localhost:4848> to access the Administration Console, and Create a new JDBC Connection pool as following:



|  |  |
| --- | --- |
| Pool Name | javaee-2019-pool |
| Resource Type | javax.sql.DataSource |
| Database Driver Vendor | Derby, Derby-30, JavaDB, or JavaDB-30 (select one of these options) |

**- click "Next"**

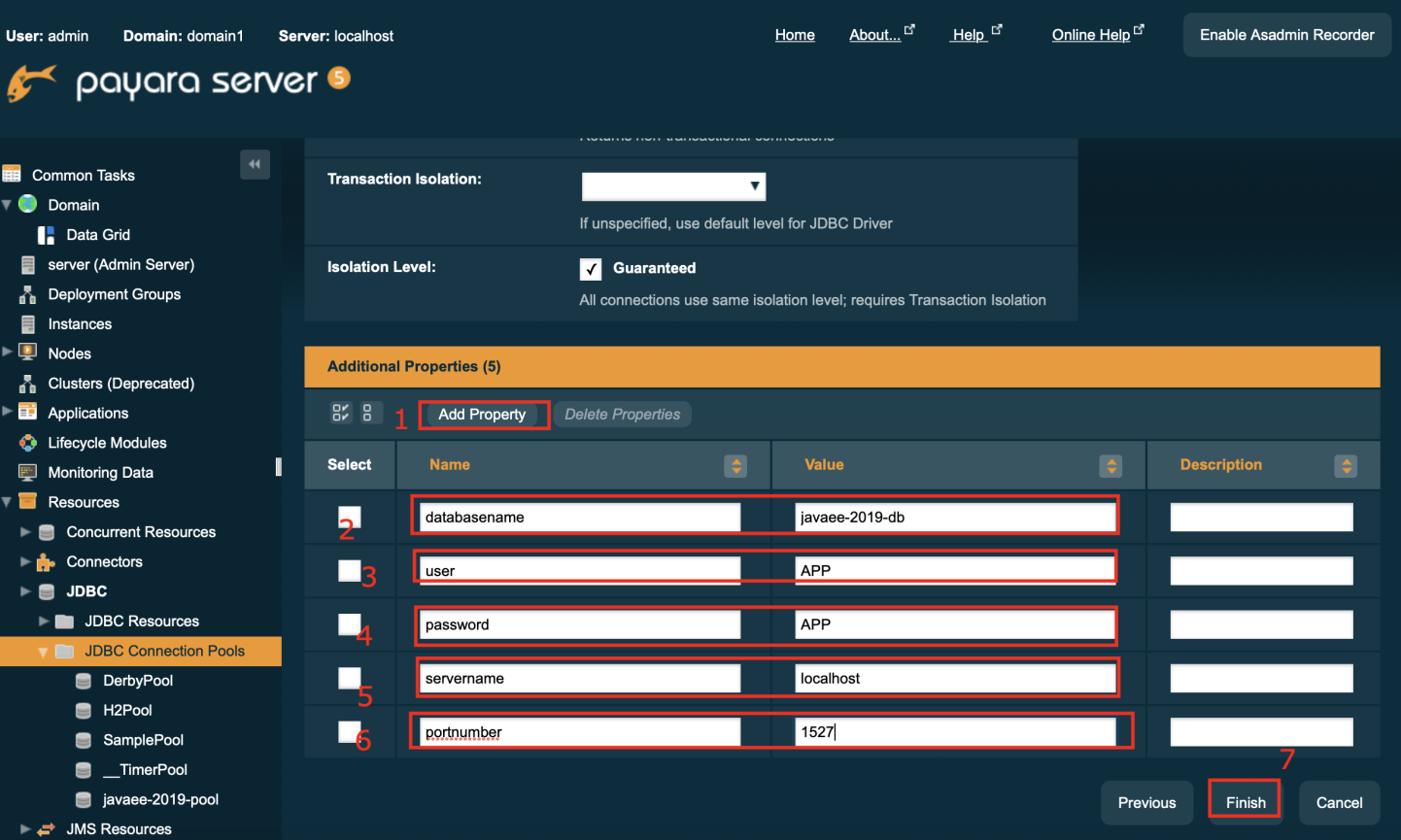


**4)**

- add the following five properties (at the bottom of the page):

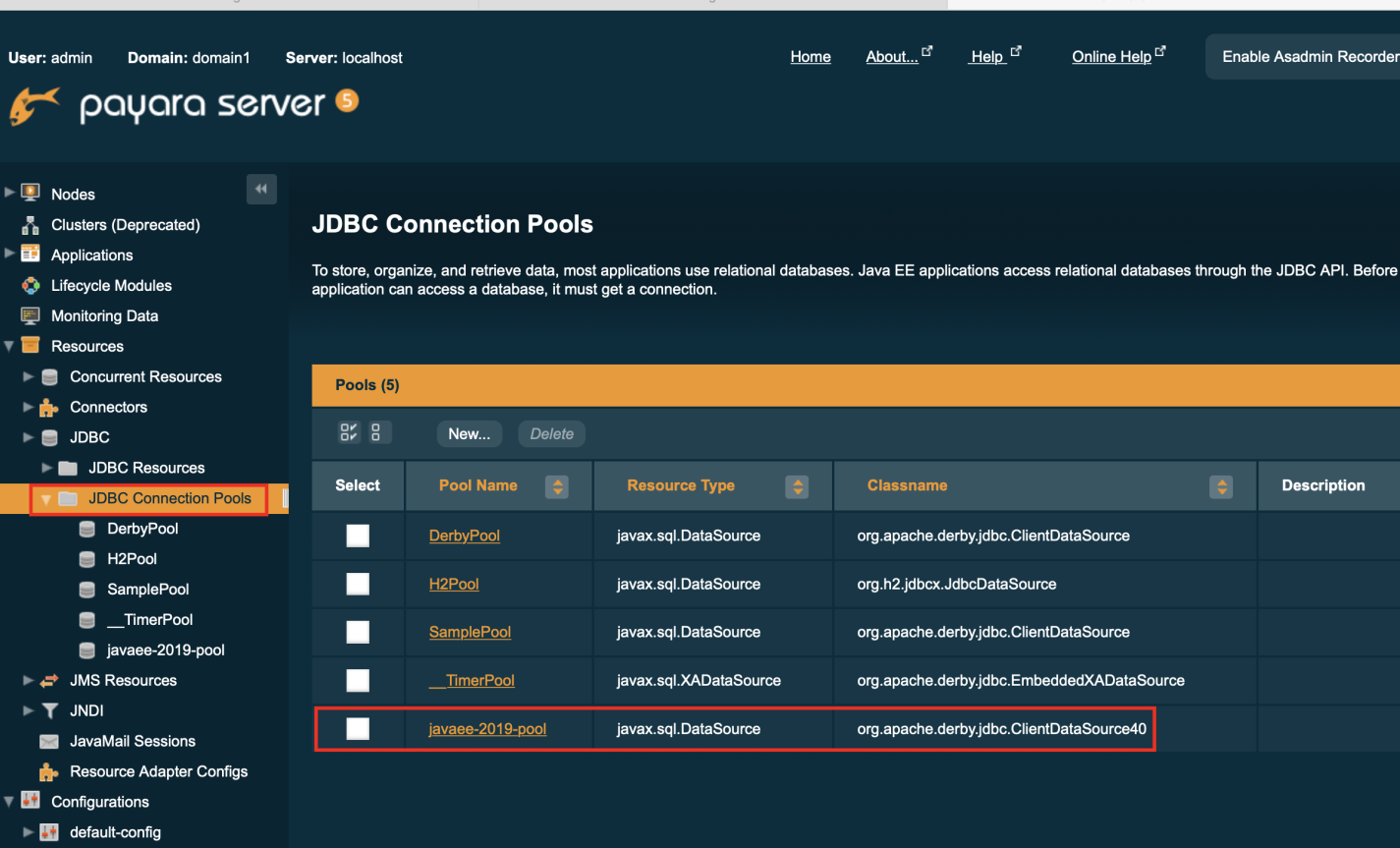
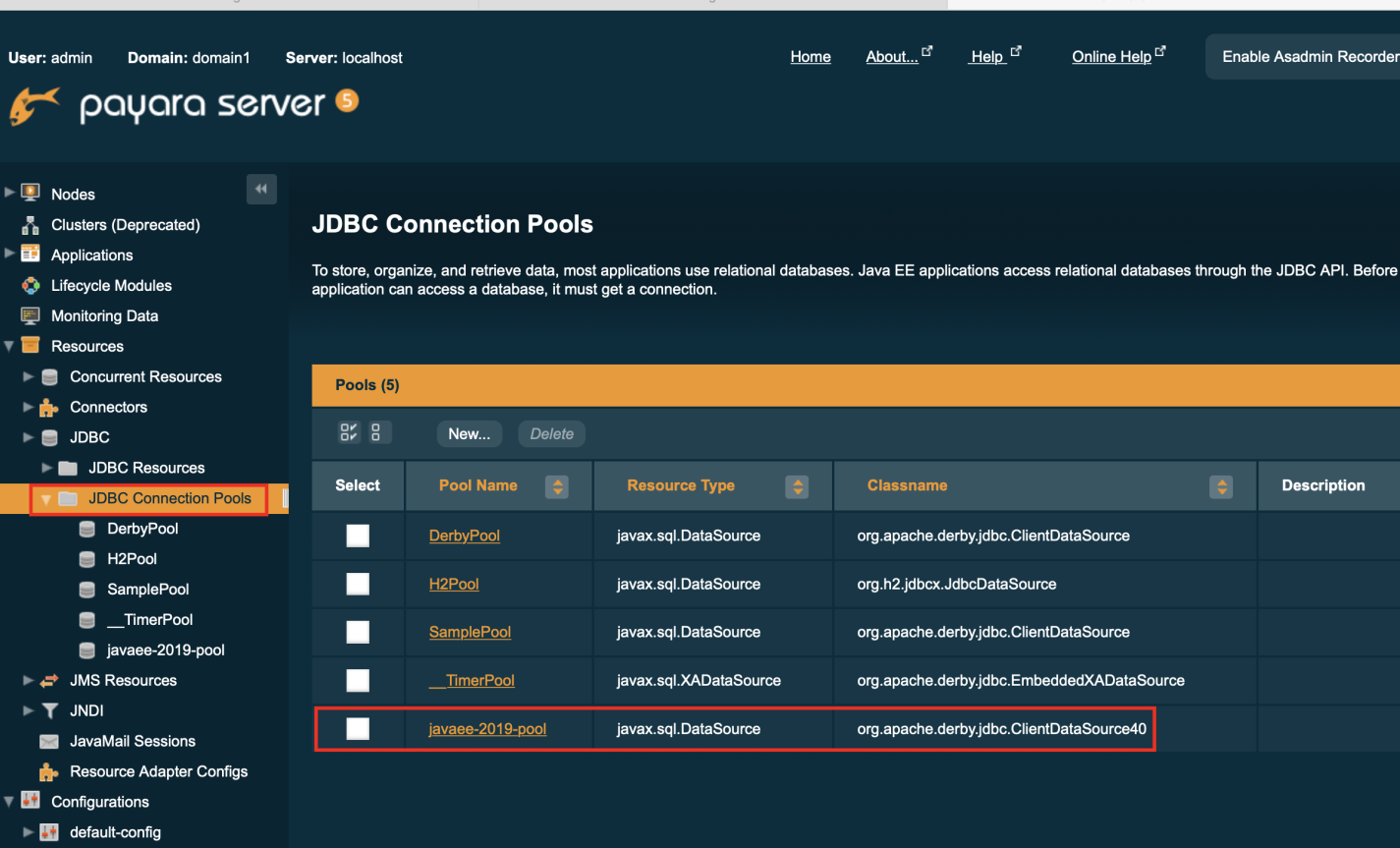
|  |  |
| --- | --- |
| databasename | javaee-2019-db(e.g dbname is ‘VotingDb`) |
| user | APP |
| password | APP |
| servername | localhost |
| portnumber | 1527 |

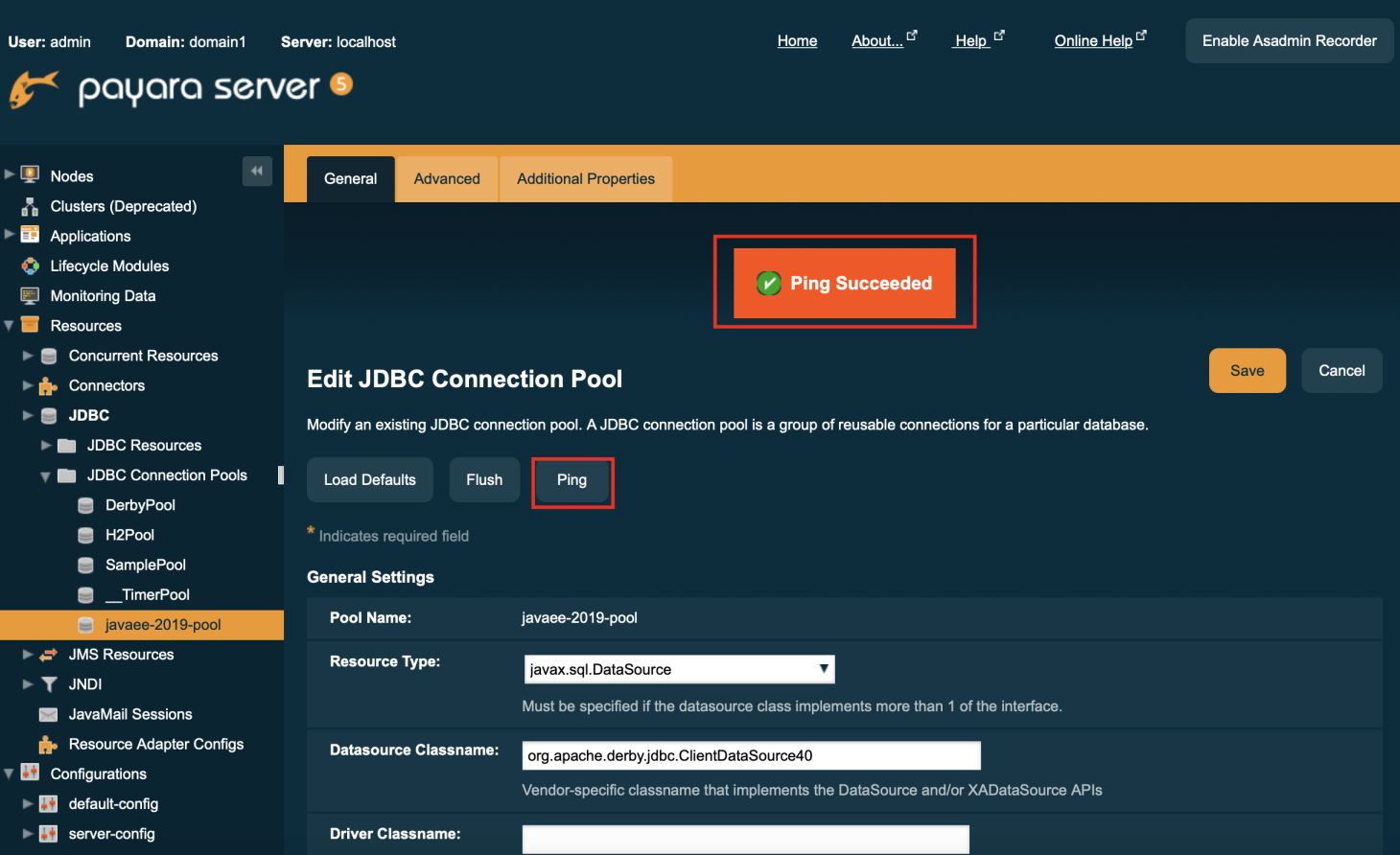
**click "Finish"**



- Verify the connection

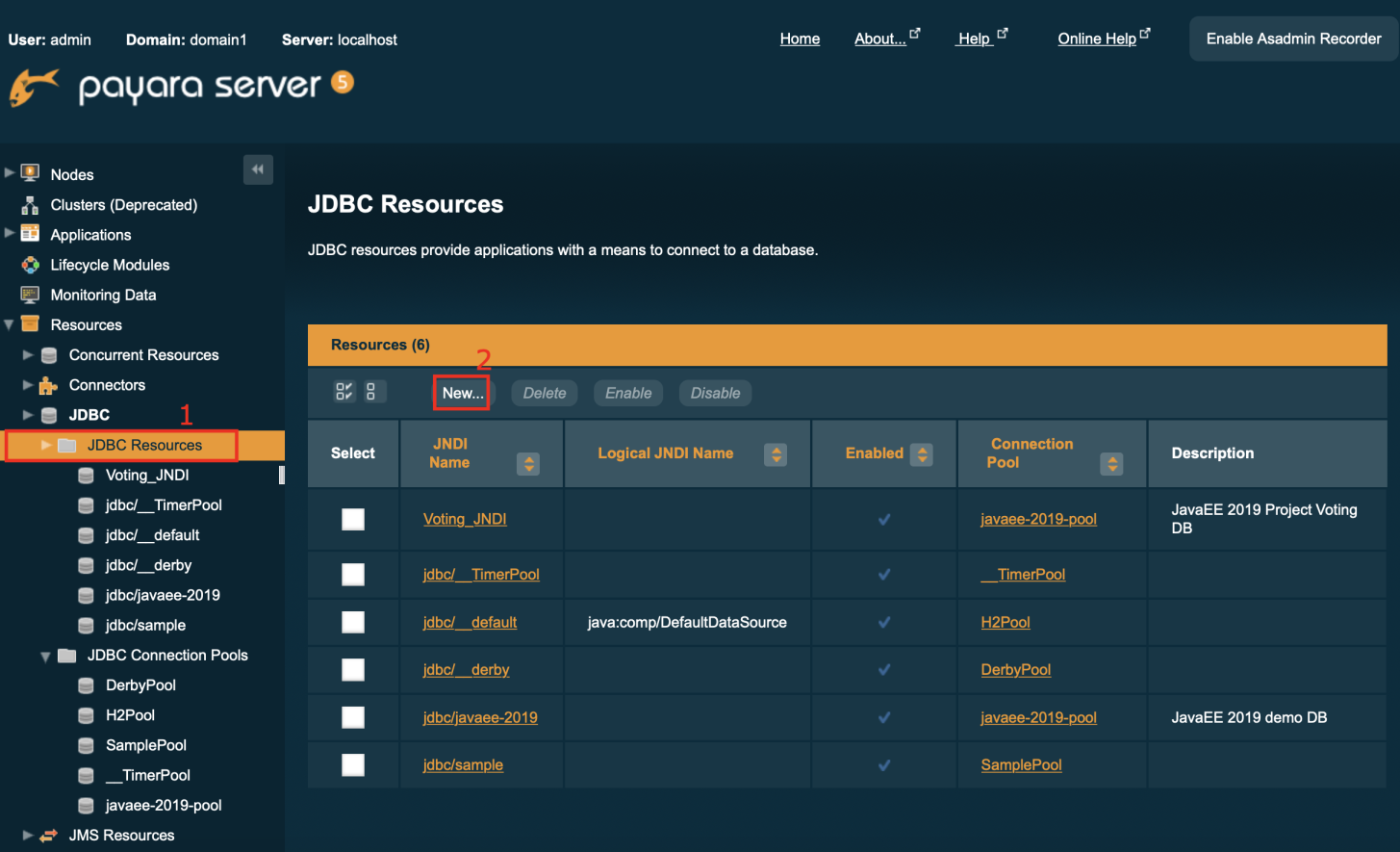
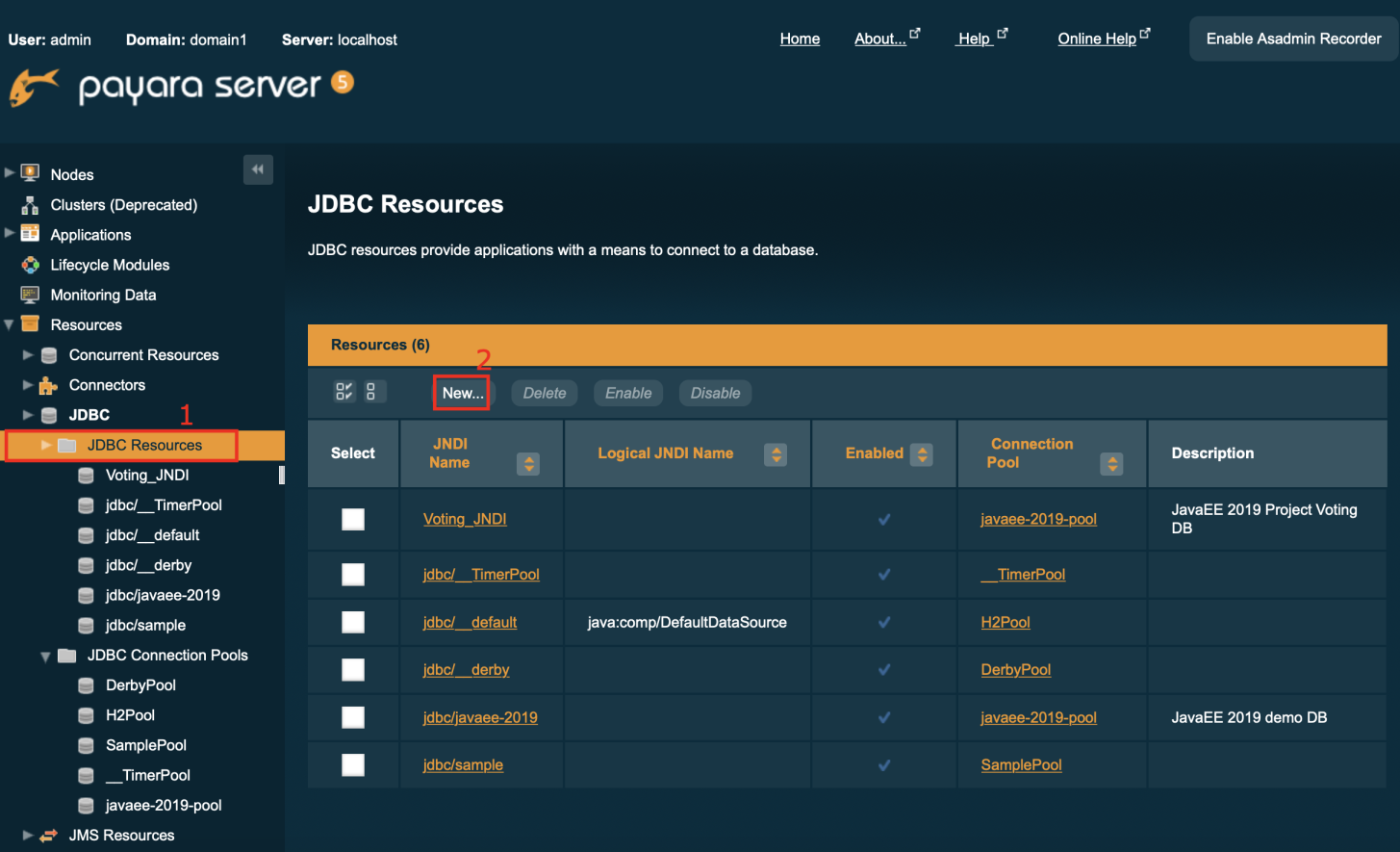
Select the created connection Pool

 -click "Ping" -> Should display "**Ping Succeed"**



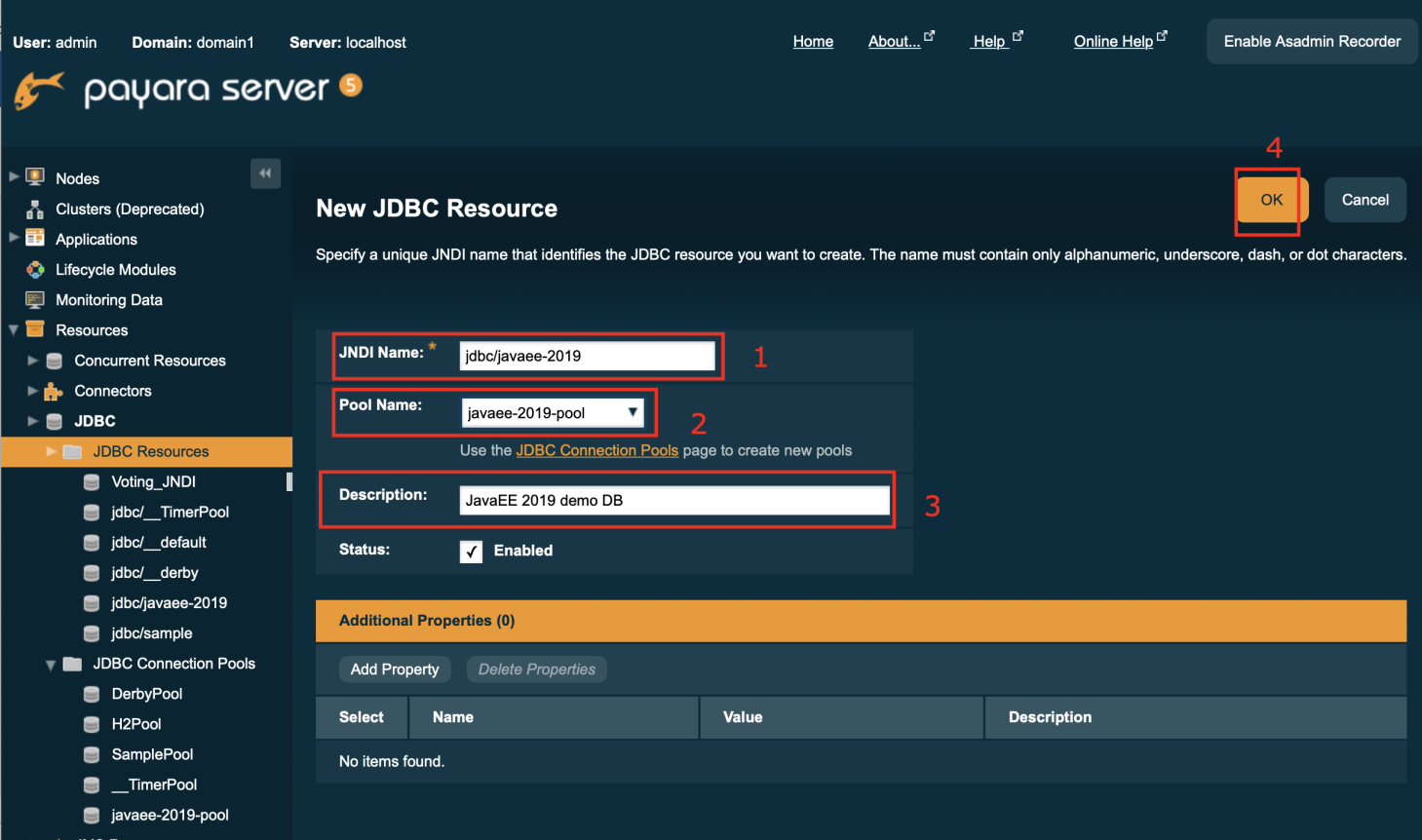
**5)**  
 - Navigate to "JDBC/JDBC Resources"

-Create a new JDBC Resource



|  |  |
| --- | --- |
| JNDI Name | jdbc/javaee-2019 ( e.g ‘Voting\_JNDI’) |
| Pool Name | javaee-2019-pool |
| Description | JavaEE 2019 demo DB |

**click "OK"**



## Installation

1. Clone the project

<https://gitlab.uni-koblenz.de/yankee/votingsystem.git>

1. Open the project
2. Clean and build the project
3. Run the project

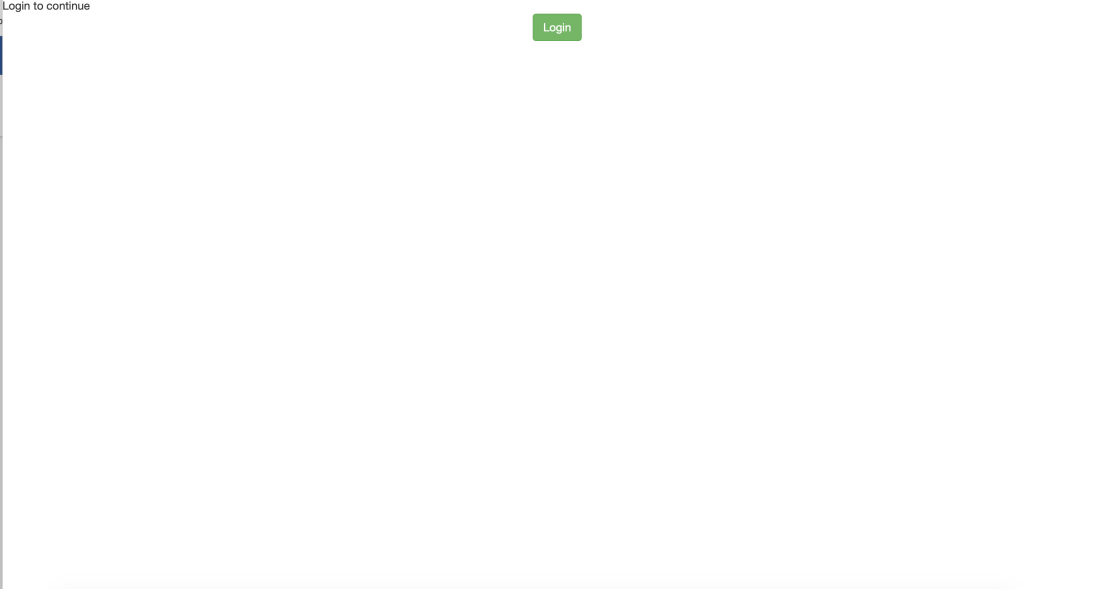
## Beginning

Making the authentication module and making the entities of the system simultaneously.

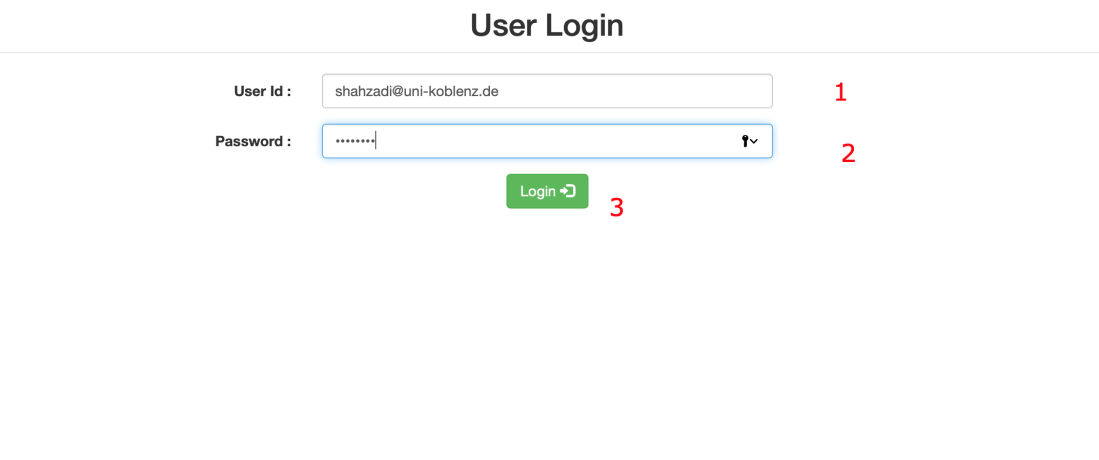
Later, merge the authentication module with the secure pages.

# Application

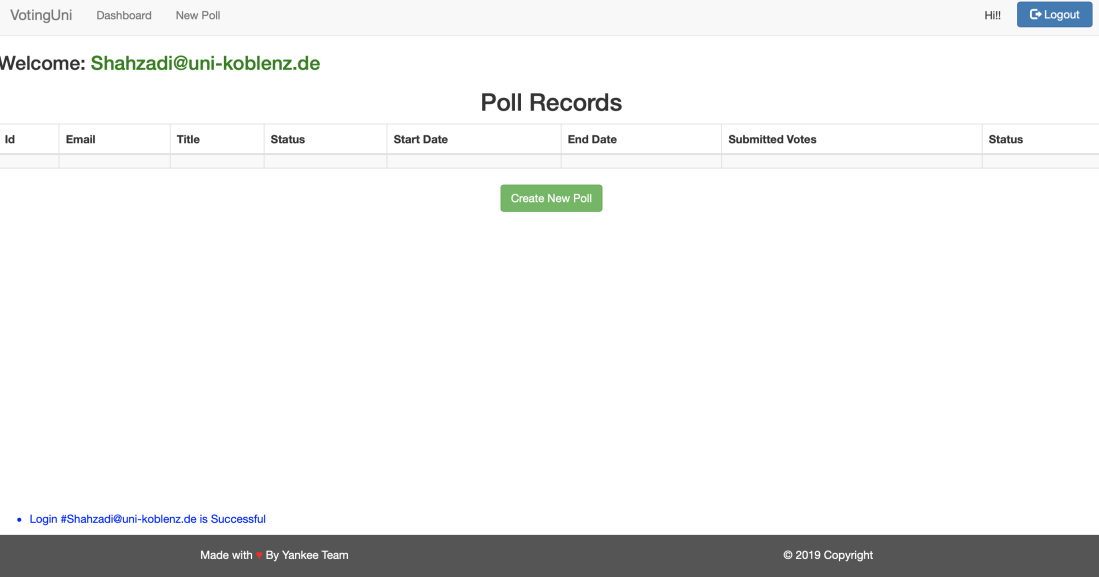
1st Screen



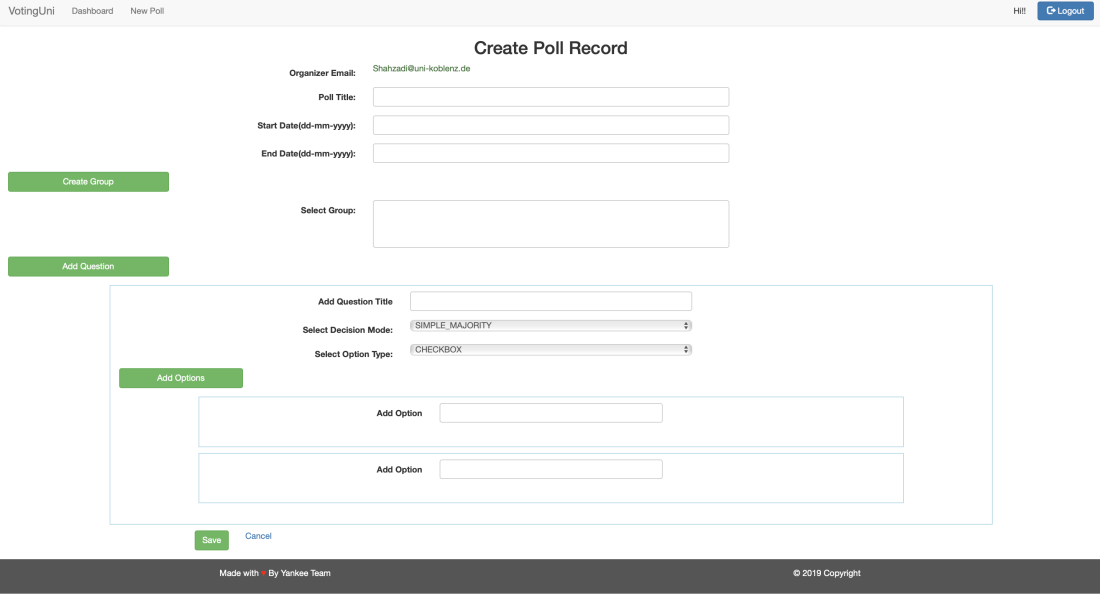
Enter you registered the login Details

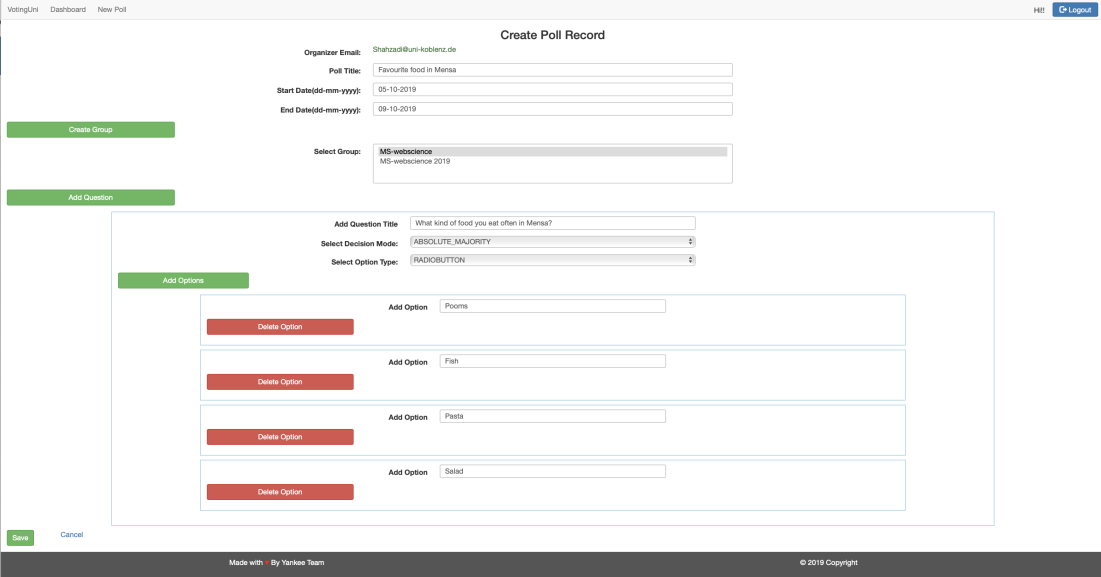


After Login Dashboard Screen will appear where the user can create new poll and see existing pool information (Id, Email, Title, Status, StartDate, EndDate, SumittedVote, Status)

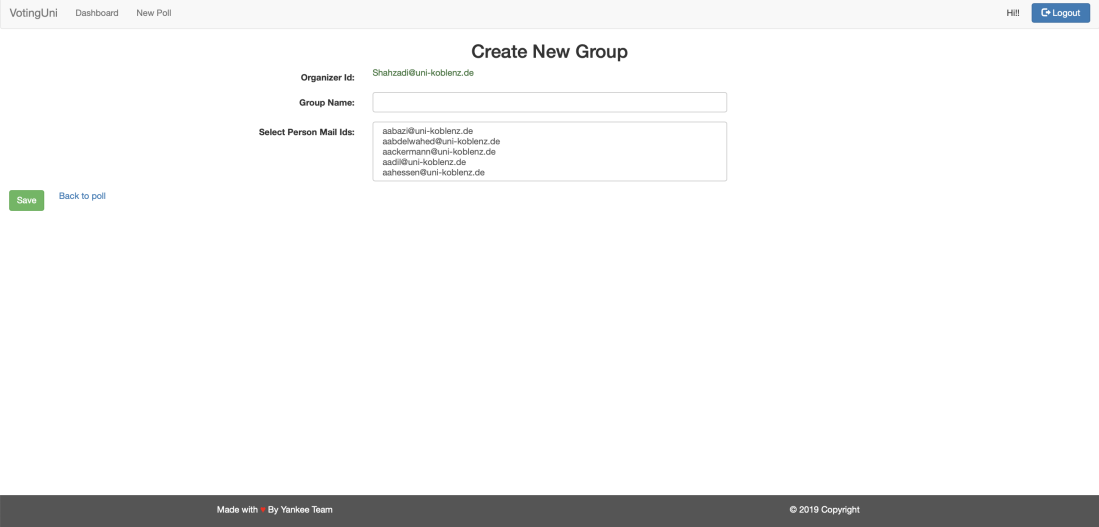


After click Create new Pool button



If no group is created before, User need to fill all the field before Creating Group(s)

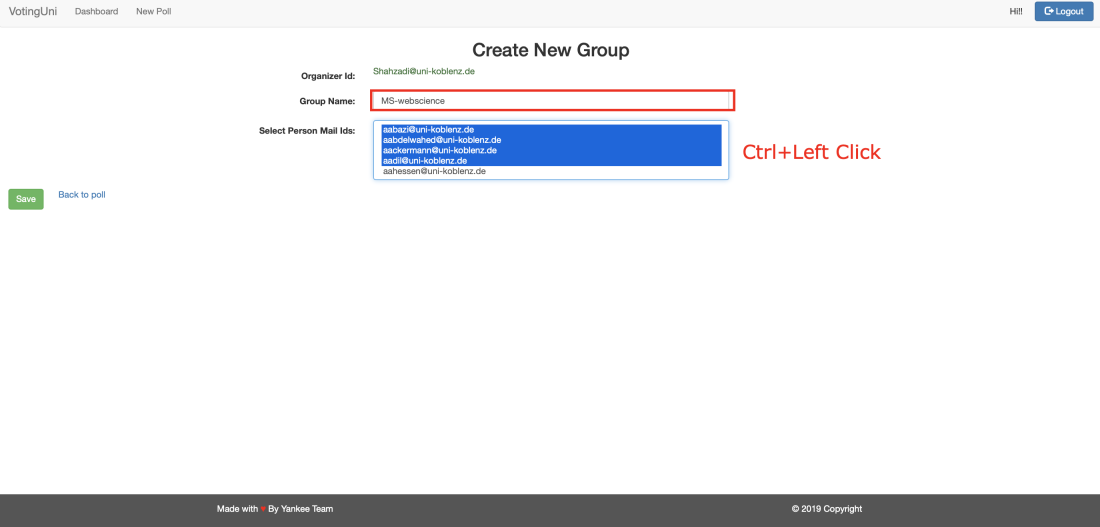
Create New Group

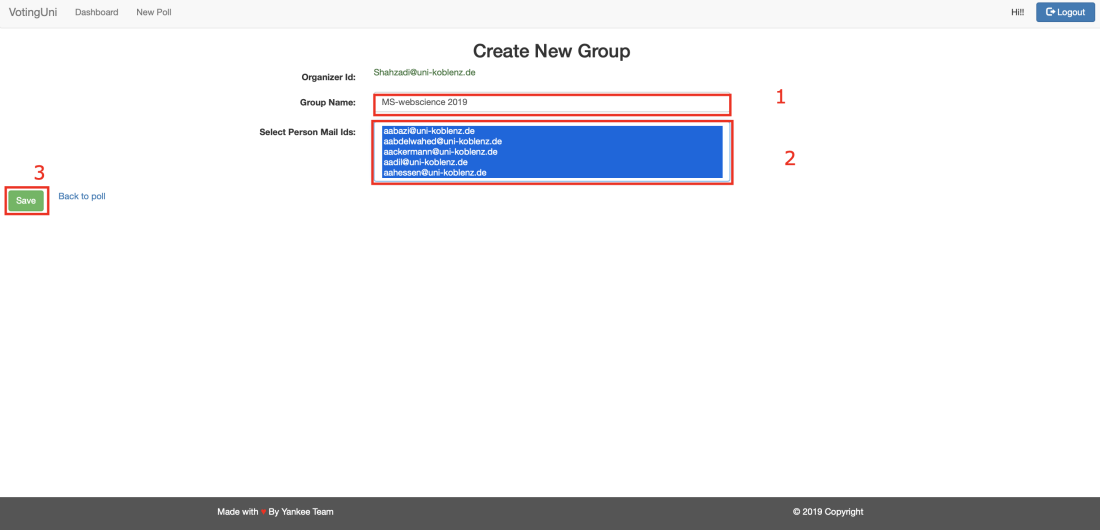


1) Write the name of the group

2) Ctrl+leftClick to select Multiple user-ids

3) Click save



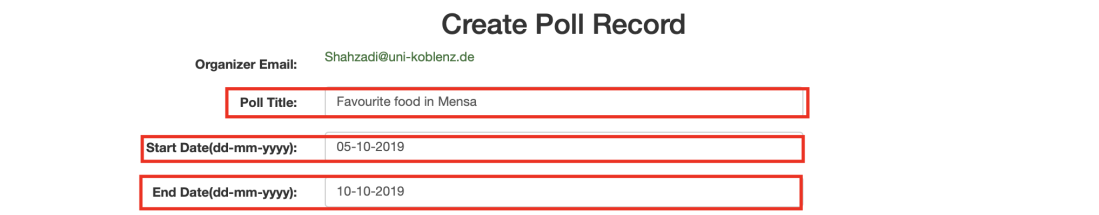


For Creating a pool

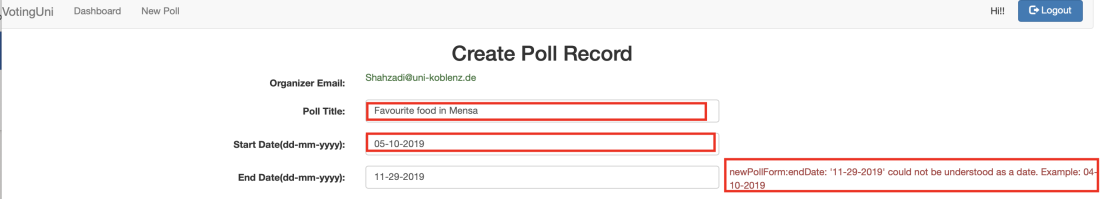
1) Write the Pool title \*

2) Start date \*

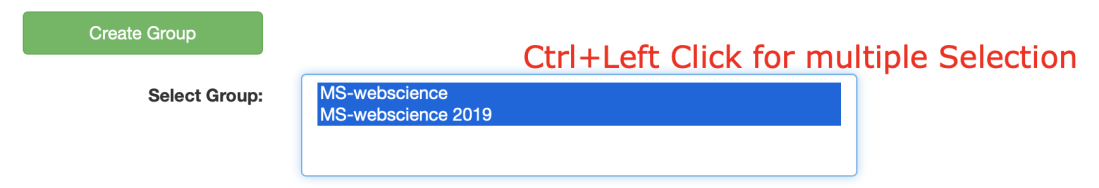
3) End Date \*



Note: The date should be in dd-mm-yyyy as it appears in the field name else the error will appear when you click on the save button as you can see in the following picture



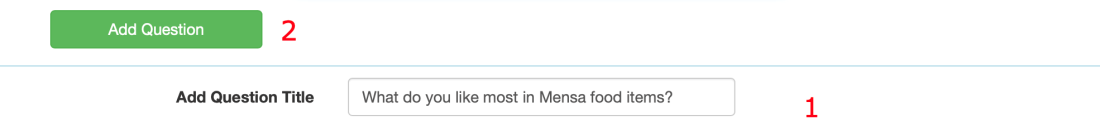
User can Select one or Multiple group(ctrl+leftClick) and can create a new group for the new pool voters if its already is not listed



Add Question(s)

1) In add Question Title write the question of the pool

2) You can add Multiple question by clicking on the add Question button



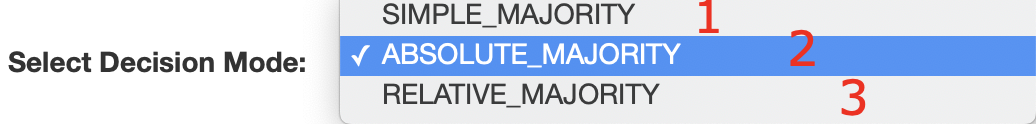
**Questions Properties**

**Select Decision Mode**

1) Simple Majority

2)Absolute Majority

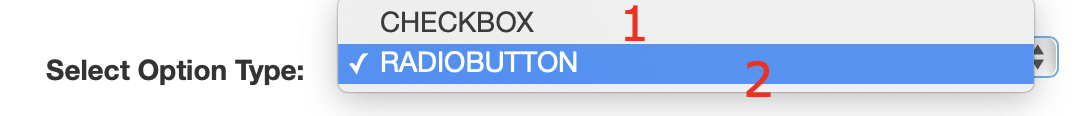
3) Relative Majority



**Select Option Type of the question**

1)Check-boxs

2)Radio-Buttons

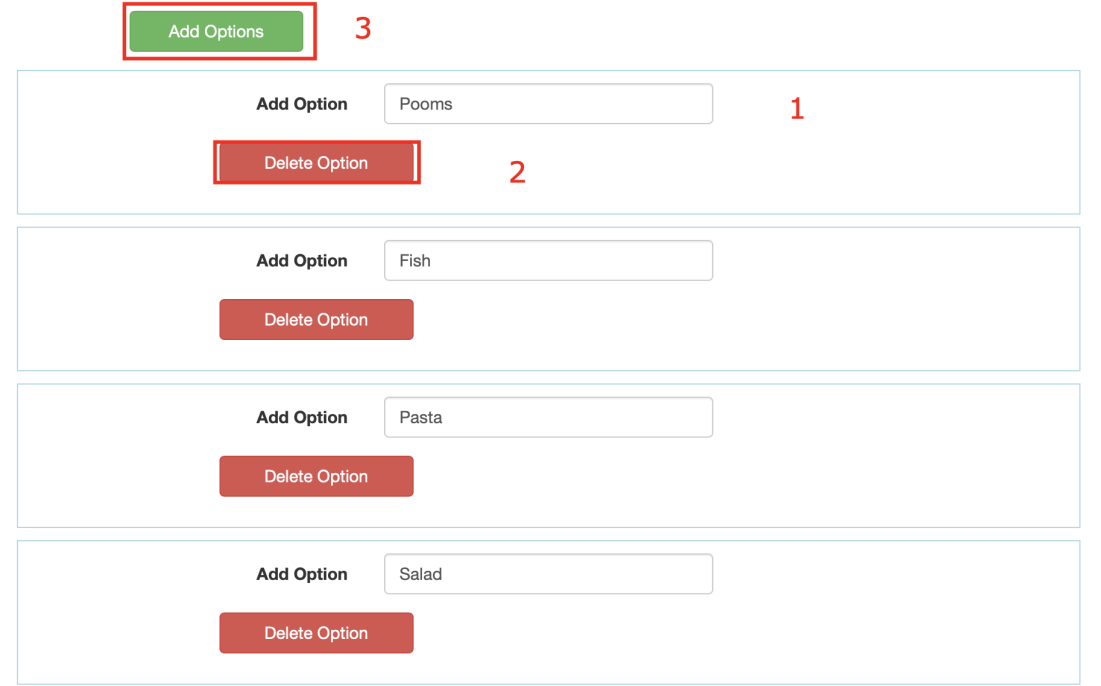


If you select Radio-Box the following window will show

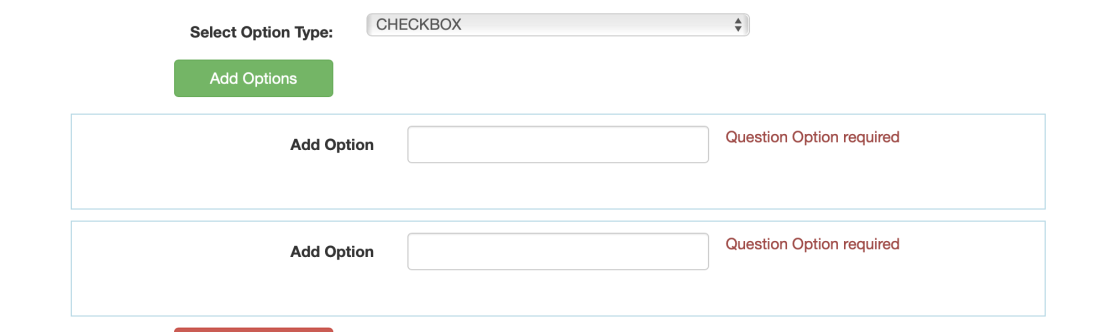
1) write Options in front of Add Option field

2)You can Delete option

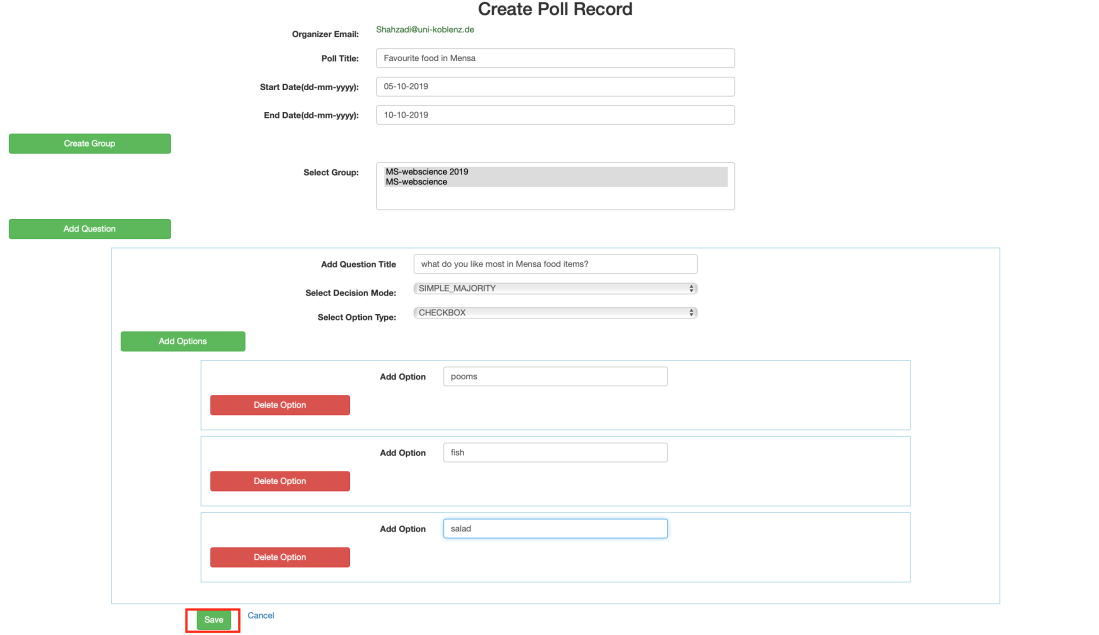
3)You can add More option by clicking on the add options button



For checkbox's type question User can select multiple options

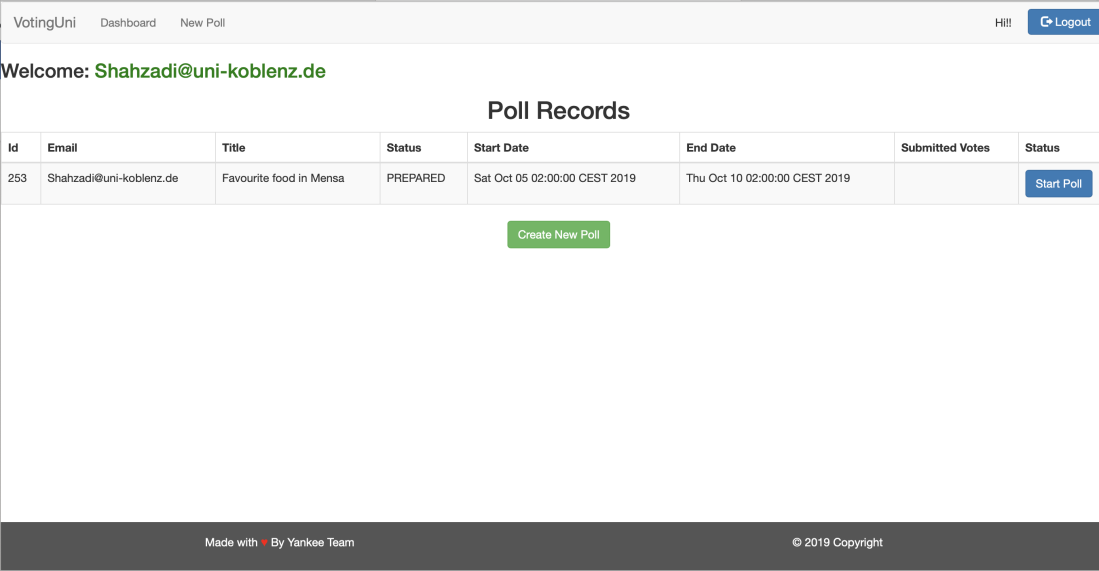


Now Click on save Button

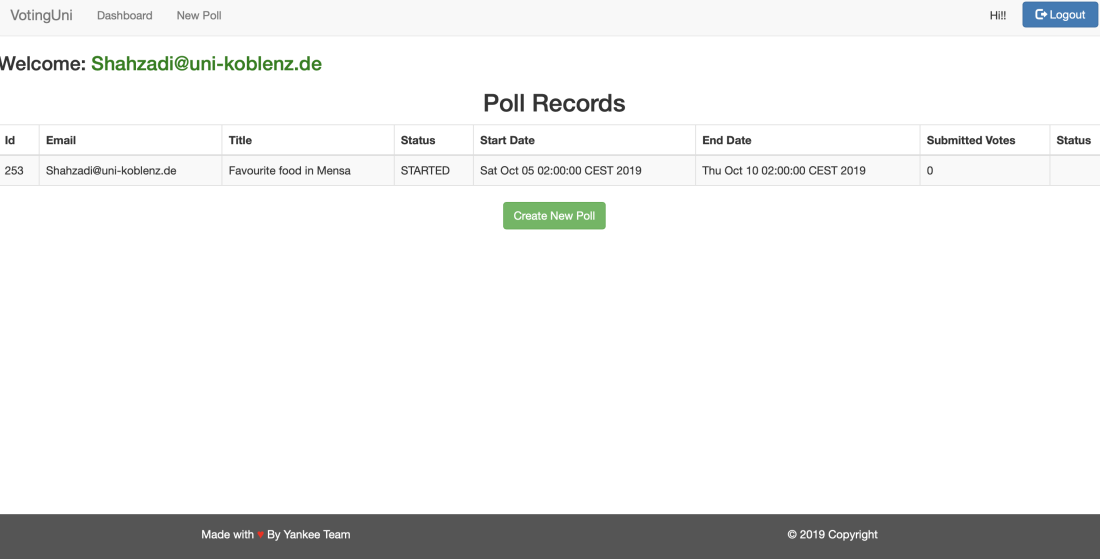


On dashboard Pool status will appear

User can start the pool by clicking on start poll



After Clicking on start Button this you can see the status of submitted votes



# Glossary

|  |  |
| --- | --- |
| **Term** | **Definition** |
| Organizer | Member of the university that is able to create a poll and edit it. |
| Token | Every time a participant needs to vote a token is issued. |
| Item | An option in a poll that participants can decide to vote for it. |
| Abstain | A participant refuses to vote for or against an option in a poll. |
| Option | A choice made by the participant weather he/she would like to vote or not. |
| Participant | A person who can take part in the voting and is a part of making a decision. |
| Participant Tracking | Keeping a track of all the online voters, also maintaining the anonymity at the same time. |
| Poll | A poll is a voting process that collects data from the participants to make a decision. |
| Poll State | Each poll includes states. Our Voting System includes states as: PREPARED, STARTED, VOTING and FINISHED. |
| Vote | Expressing the decision of a choice among many other options. |