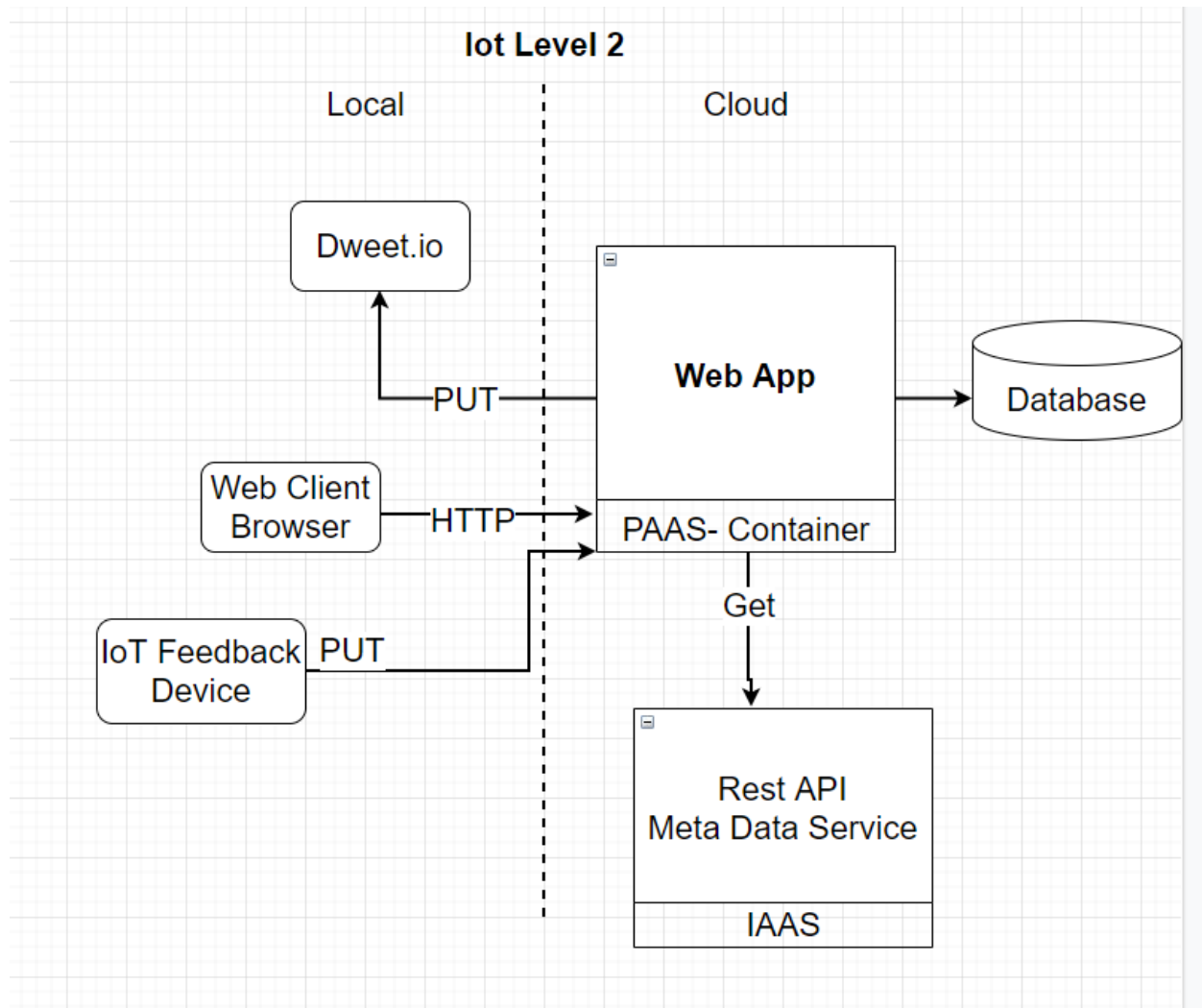


IoT Voting System

The IoT Voting system is a system where people can vote for a certain poll whether or not with a physical device. A registered user can create a poll and every user is able to vote on a poll. A poll could be either public or private. The poll also has a certain time frame.



Use Cases

These use cases give an understanding of what the functionalities are in our application.

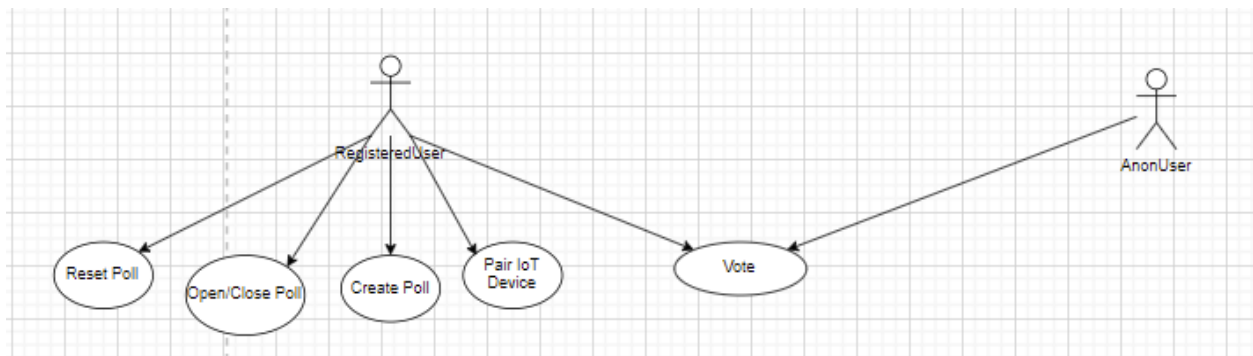
Create poll: A registered user is able to create a poll, with a certain time limit and with certain privacy settings.

Open/close poll: A registered user must be able to open or close their poll.

Reset poll: A registered user must be able to reset their poll.

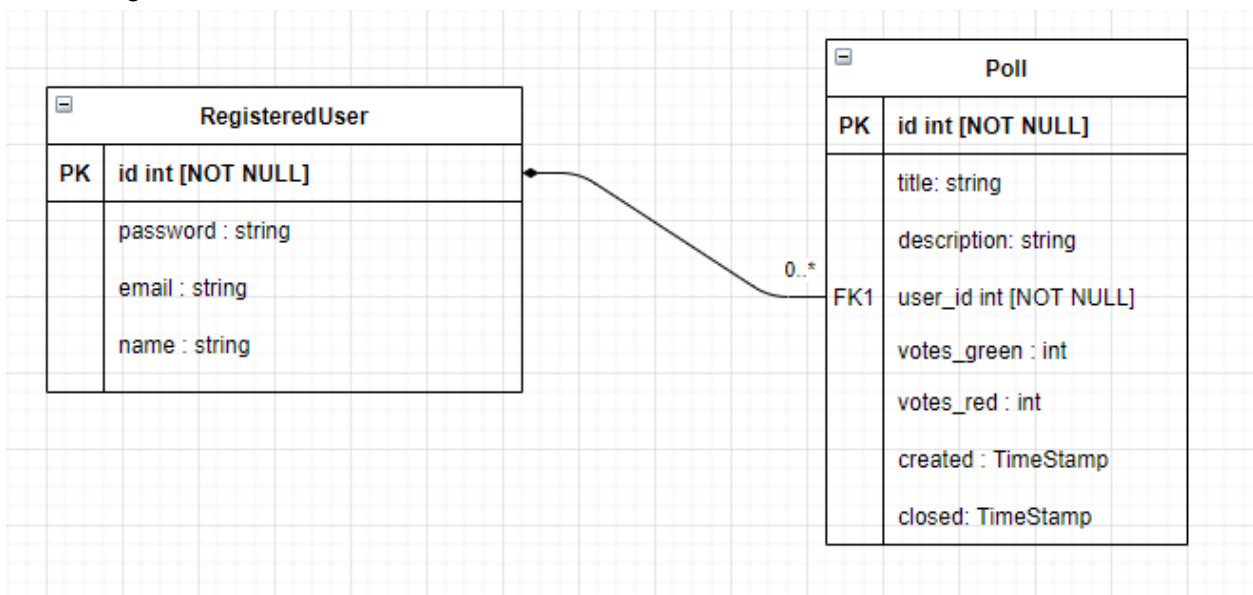
Pair IoT Device: A registered user must be able to pair the poll to an IoT device.

Vote: each user can vote on a poll.

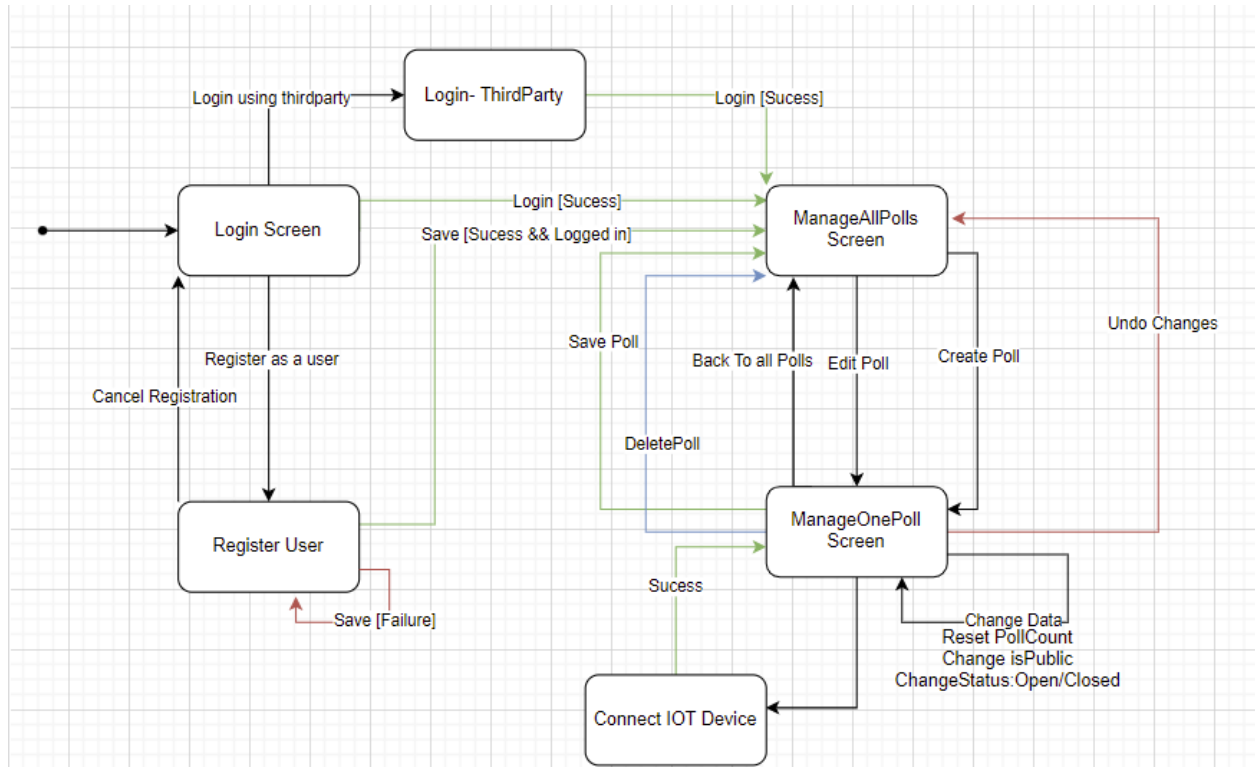


Domain Model

This domain model shows all the required concepts of our application including their relationships to each other. In this case we have a fairly simple domain model with only two tables RegisteredUser and Poll. These tables are also stored in a database.



Application Flow Diagram

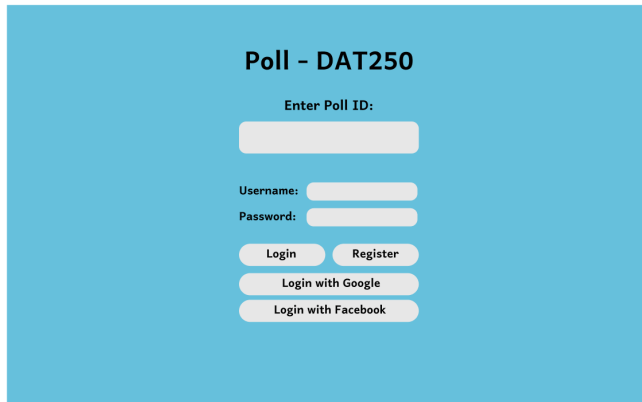


This application consists of 4 screens. Non-logged-in users have the ability to vote on public polls and they can also register as a user. Logged-in users can create, edit and view their polls and also vote on public and private polls. The registered users also have the ability to reset their polls and to connect an IoT device.

User Screens

The user screens you'll see below is a prototype of the application it will give a good understanding of how the application will work.

The first screen a person would see is the homescreen, here you can enter the ID of the poll to vote on a certain public poll, you also have the ability to login or register.



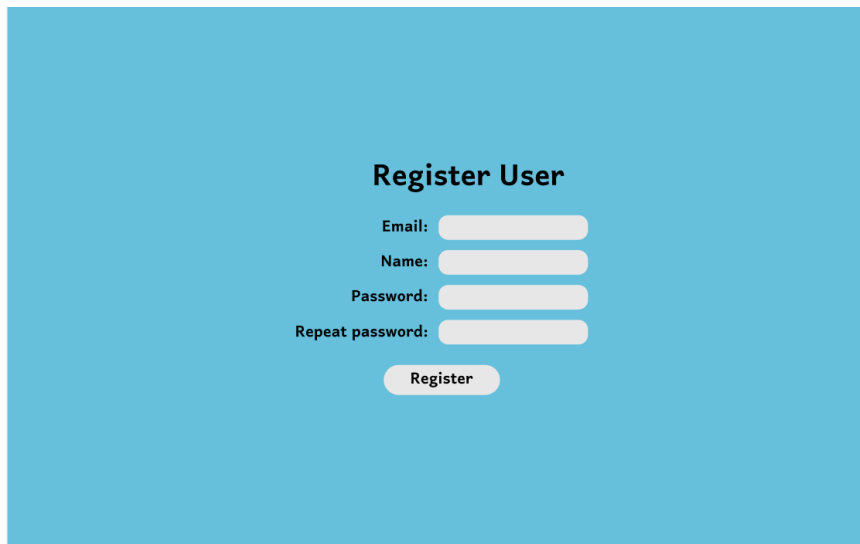
Poll - DAT250

Enter Poll ID:

Username:

Password:

In the register page a non-registered user is able to register as a user to have access to the creation of polls.



Register User

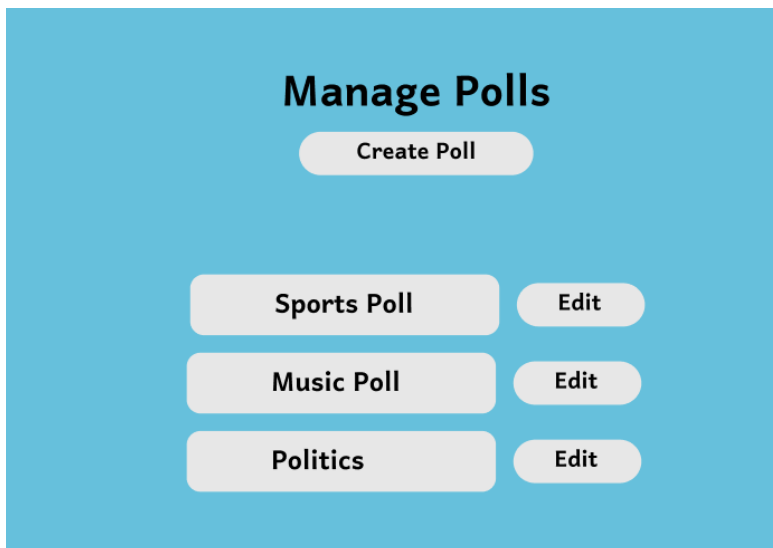
Email:

Name:

Password:

Repeat password:

On the manage polls page a user can watch all the polls he has created and has the option to view/edit them. The user also has the chance to create a poll on this page.



Manage Polls

<input type="button" value="Sports Poll"/>	<input type="button" value="Edit"/>
<input type="button" value="Music Poll"/>	<input type="button" value="Edit"/>
<input type="button" value="Politics"/>	<input type="button" value="Edit"/>

When we click on the create poll button we are redirected to a new page where we can create a brand new poll, with a certain title, description, time and options.

Create Poll

Title:

Description:

Time:

Option #1

Option #2

Create

We also have a page where we can edit the poll and also view the score of a certain poll. On this page we also have the ability to delete a poll and to start/stop it.

Edit Poll

Delete Start/stop

Title:

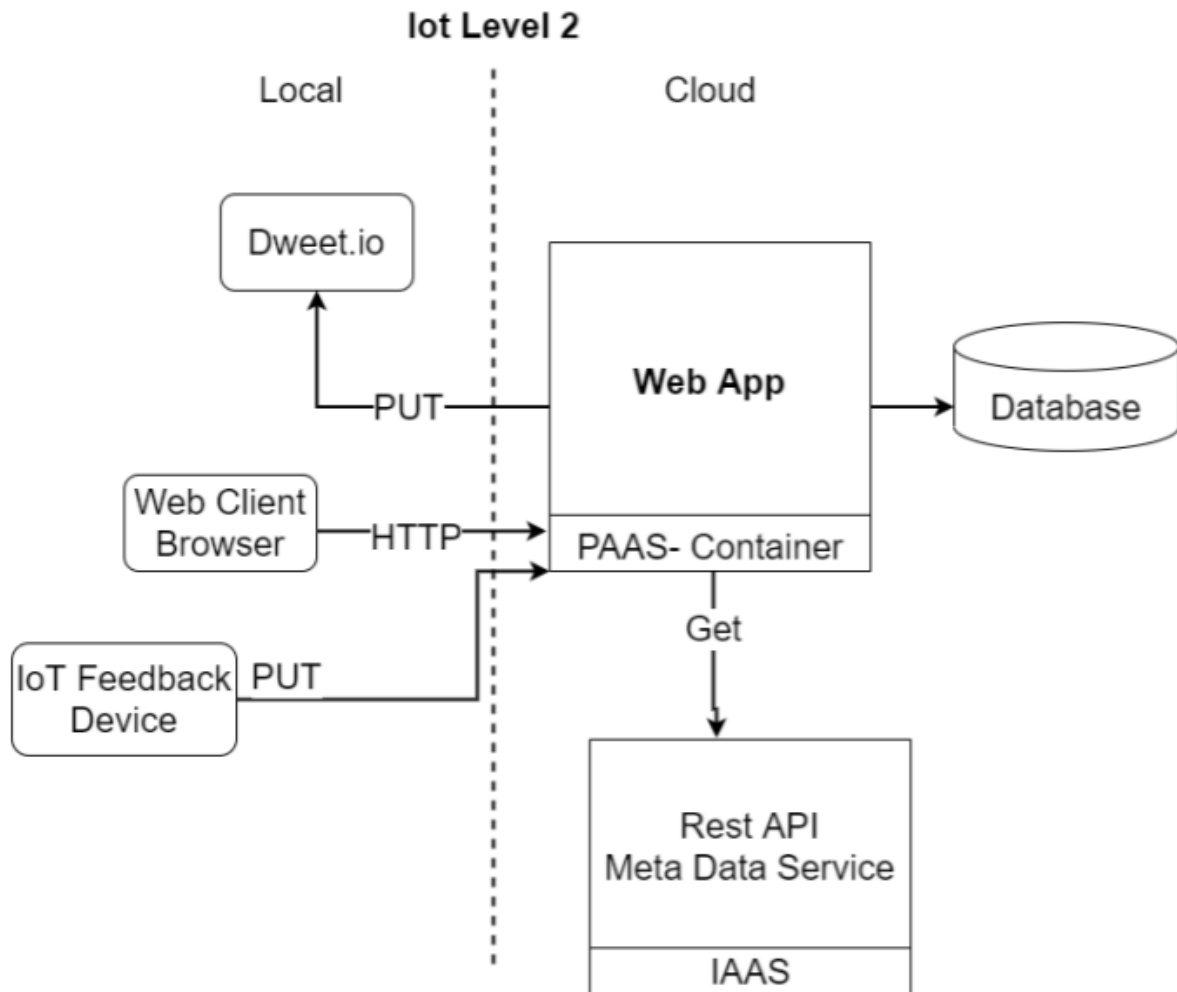
Description:

Time:

Option #1	<input type="text"/>	Votes: 530
Option #2	<input type="text"/>	Votes: 475

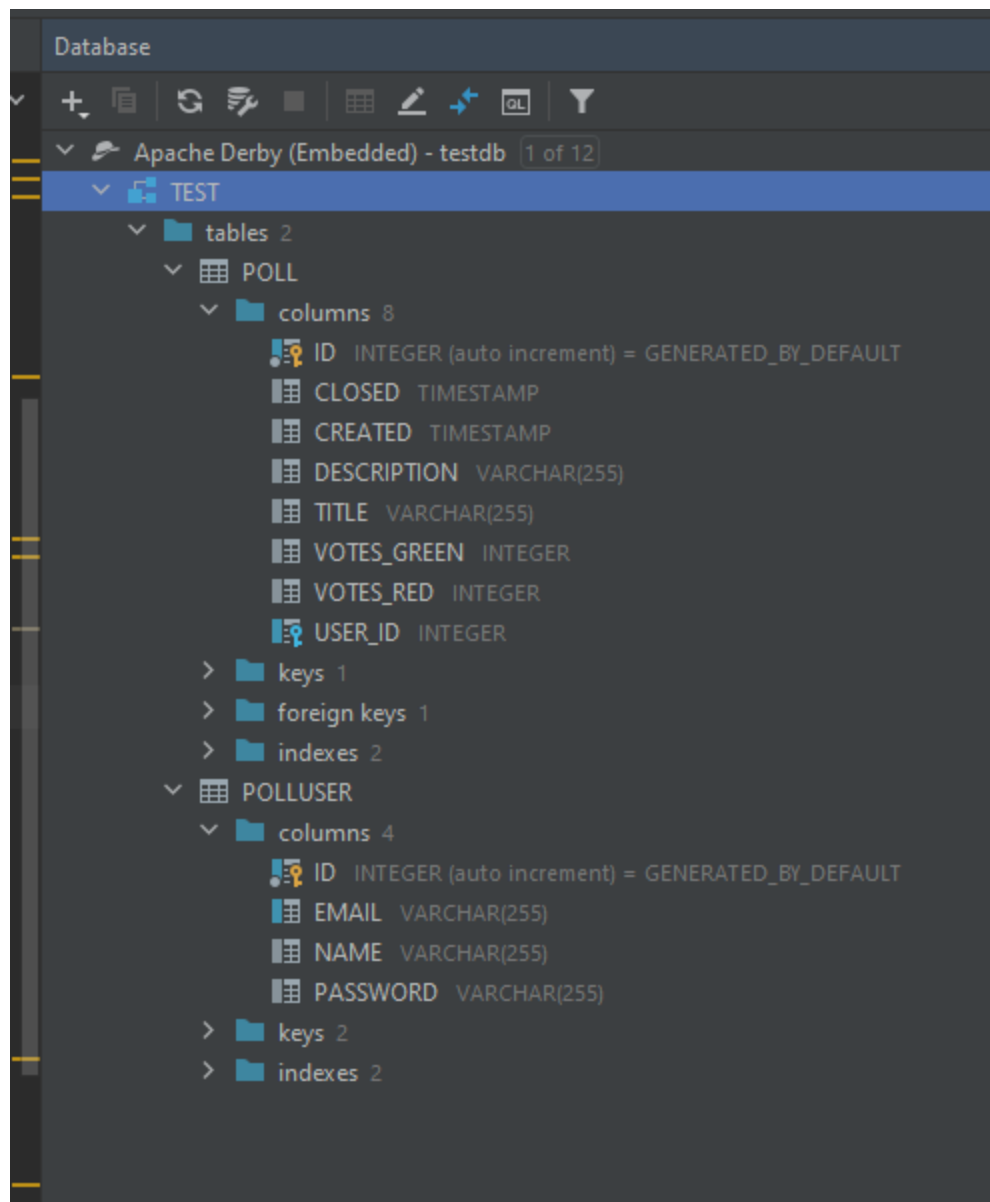
Save

Architectural Diagram



In the diagram above we laid the foundation for our application. We have two different possible sources of input, namely an IoT Feedback device or a web browser. This is all being sent to the web app which is the core of our application (using a PAAS - Container). On our web app service we can connect to our database which stores the results of the polls and also the registered users. Next to the database we also have a Rest API so other applications can easily integrate information from our polls. Finally we also send activity from the polls to an external service called Dweet.io.

DB view in intellij:



PollUser Class and DAO:

```

@Entity
@Data
public class PollUser {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;
    private String password;

    @Column(unique=true)
    private String email;

    private String name;

}

```

```

public class PollUserDAO {

    private EntityManager entityManager;

    public PollUserDAO(EntityManager entityManager) { this.entityManager = entityManager; }

    public PollUser createNewPollUser(String email, String password, String name){...}

    public Optional<PollUser> getUserByEmail(String email){...}

    public Optional<PollUser> getUserbyName(String name){...}

    public PollUser editPollUser(PollUser PU,String email, String password, String name){...}

    public void deletePollUser(PollUser PU){...}

}

```

Poll Class and DAO:


```

6
7 @Entity
8 @Data
9 public class Poll {
10     @Id
11     @GeneratedValue(strategy = GenerationType.IDENTITY)
12     private int id;
13
14     @ManyToOne
15     @JoinColumn(name = "user_id")
16     private PollUser user;
17
18     private String title;
19     private String description;
20
21     private int votes_green;
22     private int votes_red;
23
24     @Temporal(TemporalType.TIMESTAMP)
25     private java.util.Date created;
26
27     @Temporal(TemporalType.TIMESTAMP)
28     private java.util.Date closed;
29
30
31 }

```

```

public class PollDAO {

    private EntityManager entityManager;

    public PollDAO(EntityManager entityManager) { this.entityManager = entityManager; }

    /** @param PU ...*/
    public void createNewPoll(PollUser PU, String title, String description){...}

    /** @param PU ...*/
    public void createNewPoll(PollUser PU, String title, String description, Date dateClosed){...}

    public Optional<Poll> getPollbyTitle(String title){...}

    public List<Poll> getPollsbyUser(PollUser PU){...}

    public Poll editPoll(Poll P,String title, String description, java.util.Date dateClosed){...}

    public void deletePollUser(Poll P){...}

}

```

Sample Data:

```
private static void createSampleData(PollUserDAO PUDA0, PollDAO PDA0){

    PollUser jan = PUDA0.createNewPollUser( email: "jan@email.com", password: "mlk", name: "Jan");
    PollUser peter = PUDA0.createNewPollUser( email: "peter@email.com", password: "azerty", name: "Peter");
    PollUser hild = PUDA0.createNewPollUser( email: "hild@email.com", password: "qwerty", name: "Hild");
    PollUser tiril = PUDA0.createNewPollUser( email: "tiril@email.com", password: "tiril1", name: "Tiril");
    PollUser frank = PUDA0.createNewPollUser( email: "frank@email.com", password: "frank1", name: "Frank");

    PDA0.createNewPoll(jan, title: "Sports Poll", description: "description of sports poll");
    PDA0.createNewPoll(hild, title: "Music Poll", description: "description of music poll");
    PDA0.createNewPoll(peter, title: "Golf Poll", description: "description of golf poll");
    PDA0.createNewPoll(tiril, title: "Beer Poll", description: "description of beer poll");
    PDA0.createNewPoll(jan, title: "f1 Poll", description: "description of f1 poll");

}
```

PollUser Table:

	ID	EMAIL	NAME	PASSWORD
1	1	jan@email.com	Jan	mlk
2	2	peter@email.com	Peter	azerty
3	3	hild@email.com	Hild	qwerty
4	4	tiril@email.com	Tiril	tiril1
5	5	frank@email.com	Frank	frank1

Poll Table:

ID	CLOSED	CREATED	DESCRIPTION	TITLE	VOTES_GREEN	VOTES_RED	USER_ID
1	2021-09-27 07:05:51.532000000	2021-09-20 07:05:51.532000000	description of sports poll	Sports Poll	0	0	1
2	2021-09-27 07:05:51.543000000	2021-09-20 07:05:51.543000000	description of music poll	Music Poll	0	0	3
3	2021-09-27 07:05:51.545000000	2021-09-20 07:05:51.545000000	description of golf poll	Golf Poll	0	0	2
4	2021-09-27 07:05:51.546000000	2021-09-20 07:05:51.546000000	description of beer poll	Beer Poll	0	0	4
5	2021-09-27 07:05:51.548000000	2021-09-20 07:05:51.548000000	description of f1 poll	f1 Poll	0	0	1