Boğaziçi University

Department of Computer Engineering

SWE 573

Software Development Practice

Project Report

Hüseyin Kılıç

Table of Contents

1. Introduction

This project aims to create a browsing and notification system for upcoming social events based on selected categories and locations of users.

Github URL: *https://github.com/huseyin-kilic/Fall2017Swe573*

Deployment URL*: https://high-ace-181518.appspot.com/profile

*Please note that due to some technical restrictions of the deployment server, the deployed instance of the application is not functioning as expected. For testing puposes, you can follow the instructions in section-3 and use your local server instead.

2. Technology Stack

Following languages, frameworks and tools have been utilized throughout the project.

Language	Java (version 1.7.0_80)
IDE	Intellij Idea (version 2016.3.2)
Build Tool	Apache Maven (version 3.5.2)
Web Framework	Spring Boot 1.5.7, Spring Web MVC (version 4.3.11)
Database	H2 (version 1.4.196)
Twitter API Client	Spring Social Twitter (version 1.1.2)
ORM Framework	Spring Data JPA (version 1.11.7)
UI Rendering Engine	Thymeleaf (version 2.1.5)
Email Client	Spring Integration Mail (version 4.3.12)
Logging	SLF4J (version 1.7.25)
Scheduling	Spring Batch (version 3.0.8)
Web Server	Embedded Tomcat (version 8.5.20)
Deployment Server	Google Cloud AppEngine

3. Deployment Instructions and System Manual

Installation to Local Server

- Clone or download project to an appropriate directory in your computer.

```
git clone https://github.com/huseyin-kilic/Fall2017Swe573.git
```

- Make sure JDK and Maven binaries are installed, and "JAVA_HOME" and "M2_HOME" are defined as environment variables. You can follow this link for detailed instructions.
- Open a command line interface and navigate to project base directory.
- Build and install the project to your local maven repository.

```
mvn clean install
```

- Run the project.

```
mvn spring-boot:run
```

- Default port is 8080. To run on a different port, you can use the following command by changing *PORT_NUMBER* to an available port.

```
mvn spring-boot:run -Drun.jvmArguments='-Dserver.port=PORT_NUMBER'
```

- Open your browser and navigate to http://localhost:8080/profile to access to the application.

Installation to Google Cloud AppEngine

- Create a new account on Google Cloud Platform.
- Create a new AppEngine server and open the cloud shell utility and clone project to the remote server.

```
git clone https://github.com/huseyin-kilic/Fall2017Swe573.git
```

- Initialize project to run in Europe West-1 Region.

```
gcloud app create --region europe-west1
```

- Build and install the project to your local maven repository using the command

```
mvn clean install
```

- Deploy application into AppEngine environment.

```
mvn appengine:deploy
```

- More detailed information about deployment of a spring boot application to Google Cloud Platform can be found here.

4. Requirements

1. **Registration**: Users can register via their existing Twitter accounts. Username is set as the twitter username.

2. **Selection of preferred event categories**: Users should be able to view the pre-defined set of event categories and choose the list of categories that they are interested in.

3. **Specifying search keywords**: Users should be able to create a list of keywords that will be used by the application to search for the related events.

4. **Selection of preferred event locations**: Users should be able to view the pre-defined set of event locations and choose the list of locations that he/she would like to be notified about in case there is an upcoming event on those locations.

6. **Sending notification to users about new events**: On a daily basis, system will send a notification (email or direct message) to each user. This notification contains information about the upcoming events that take place in one of user's preferred event locations. The list of events contained in the notification should satisfy at least one of the following conditions:

    6.1. Event category matches with one of the preferred categories of the user.

    6.2. Event description contains at least one of the search keywords specified by the user.

If there are no upcoming events matching with these criteria, no notification is created.

7. **Notification channel selection**: There are two options for receiving notifications from the system.

    7.1. Email notification: System creates and sends an email to the user about the upcoming events.

    7.2. Direct message notification via Twitter: Direct message notification option is enabled only if the user has a public Twitter profile.

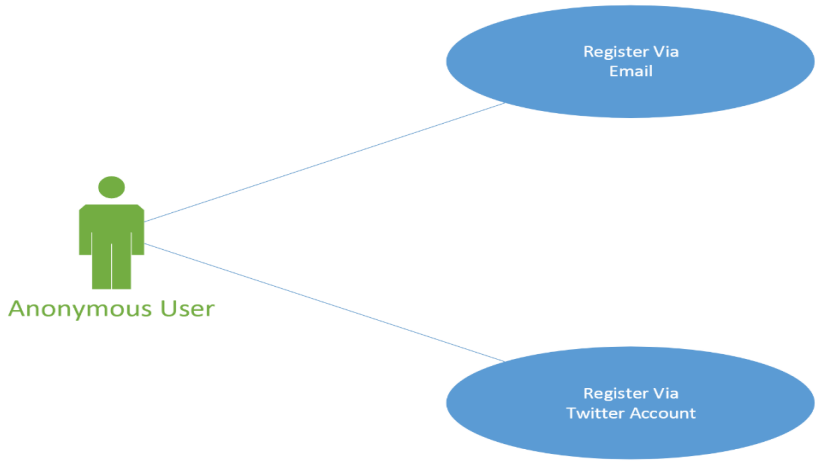
8. **Share on Twitter**: When they decide to attend to an event, users can share this on Twitter using a single share button. When the button is clicked, a tweet is created including a link to the corresponding event.

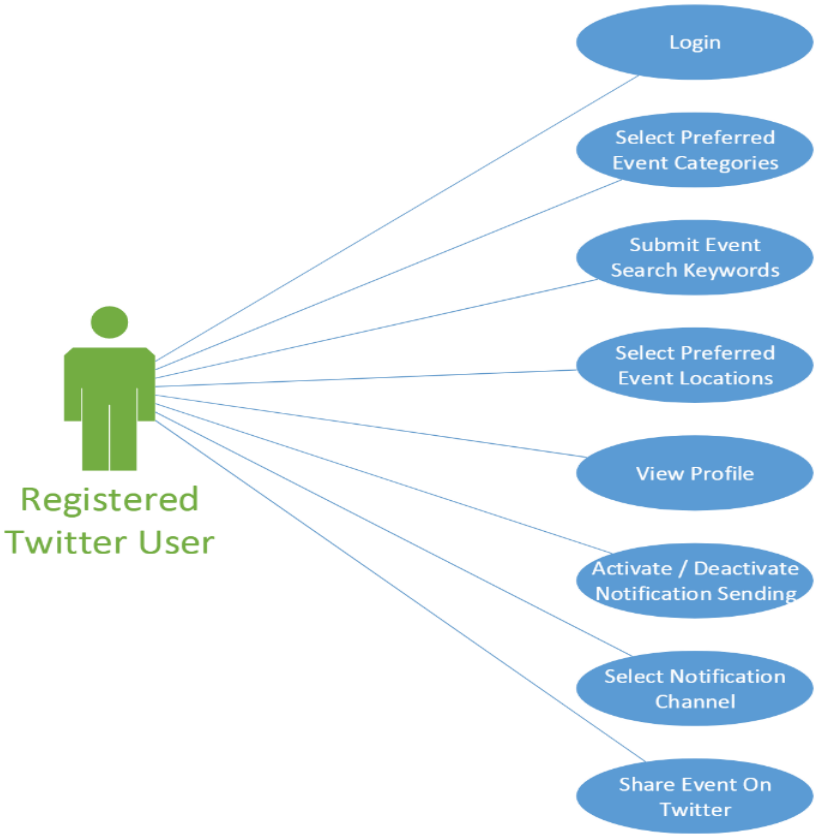
9. **Profile overview**: Users should be able to view their profile and update preferences.

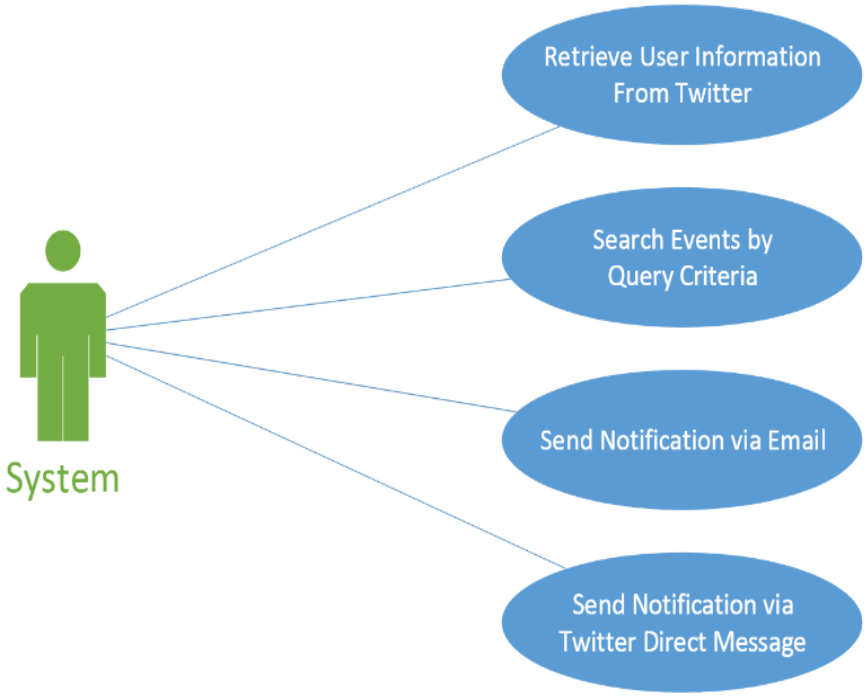
10. **Notification deactivation/activation**: Users should be able to deactivate and activate notifications from the application.

5. Design Documents

Use Case Diagrams

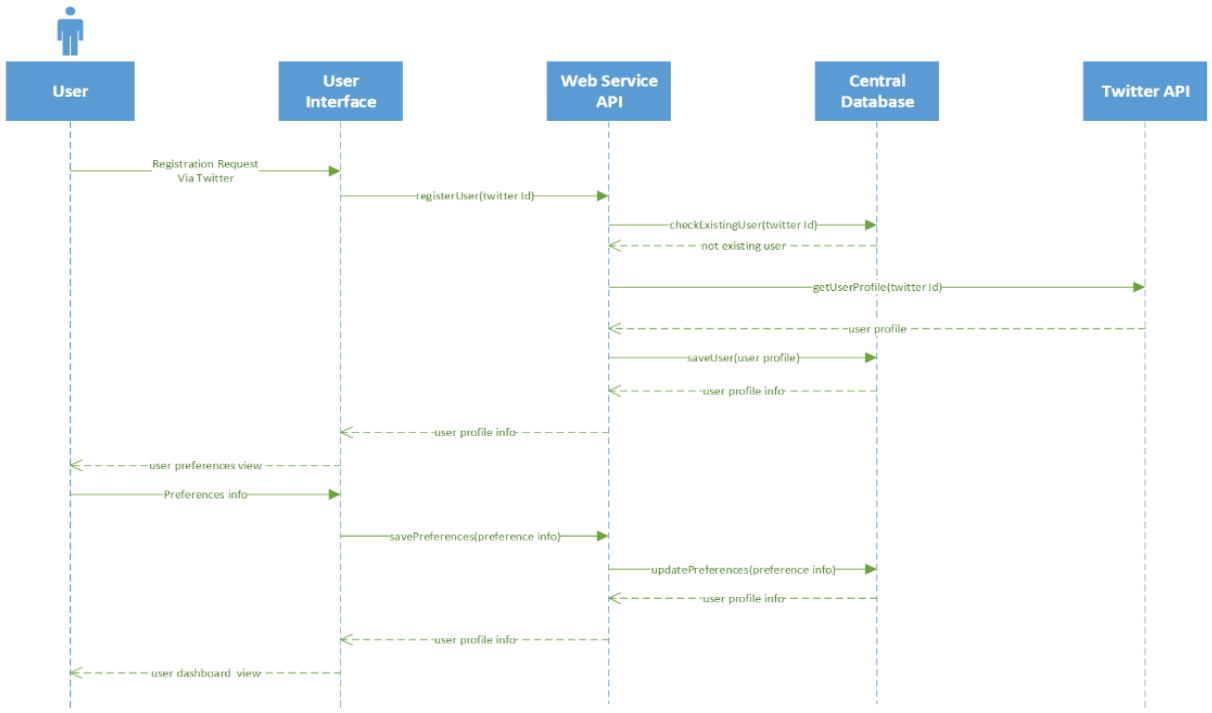



Login
Select Preferred Event Categories
Submit Event Search Keywords
Select Preferred Event Locations
View Profile
Activate / Deactivate Notification Sending
Select Notification Channel
Share Event On Twitter
Registered Twitter User

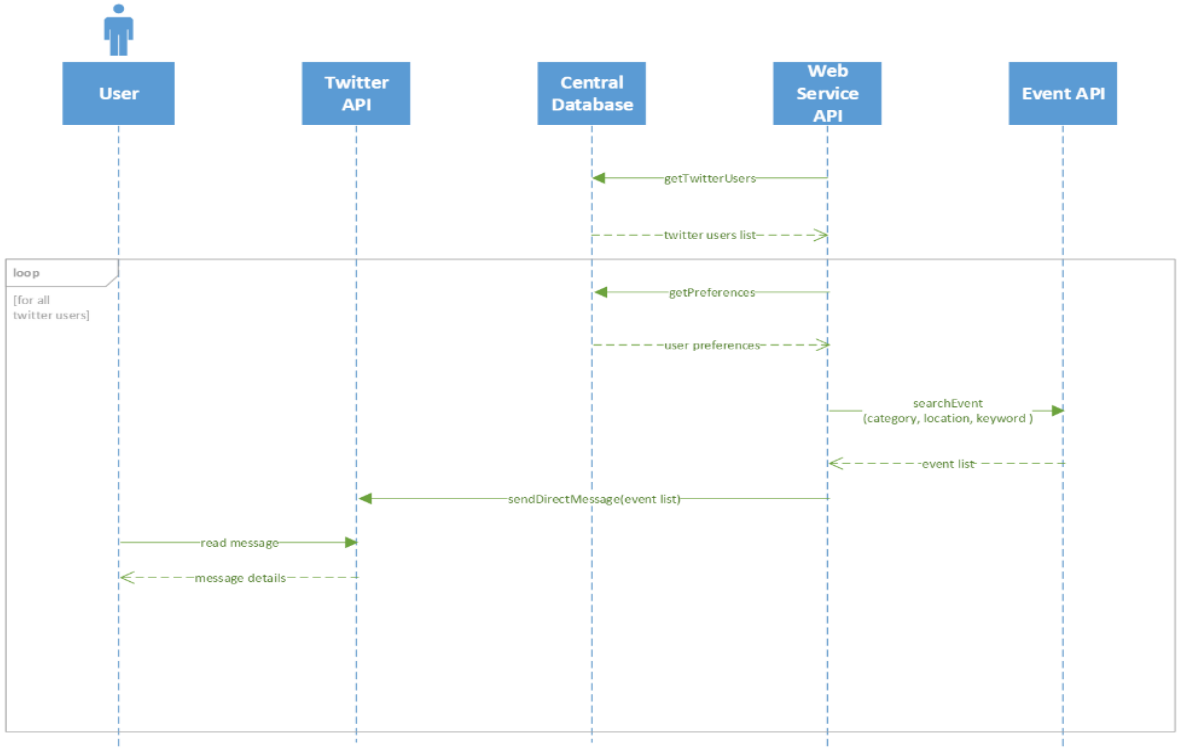

Retrieve User Information From Twitter
Search Events by Query Criteria
Send Notification via Email
Send Notification via Twitter Direct Message
System

Sequence Diagrams

User Profile Creation Sequence Diagram



User Notification Sending Sequence Diagram



6. APIs Used

Following two APIs have been used in this project:

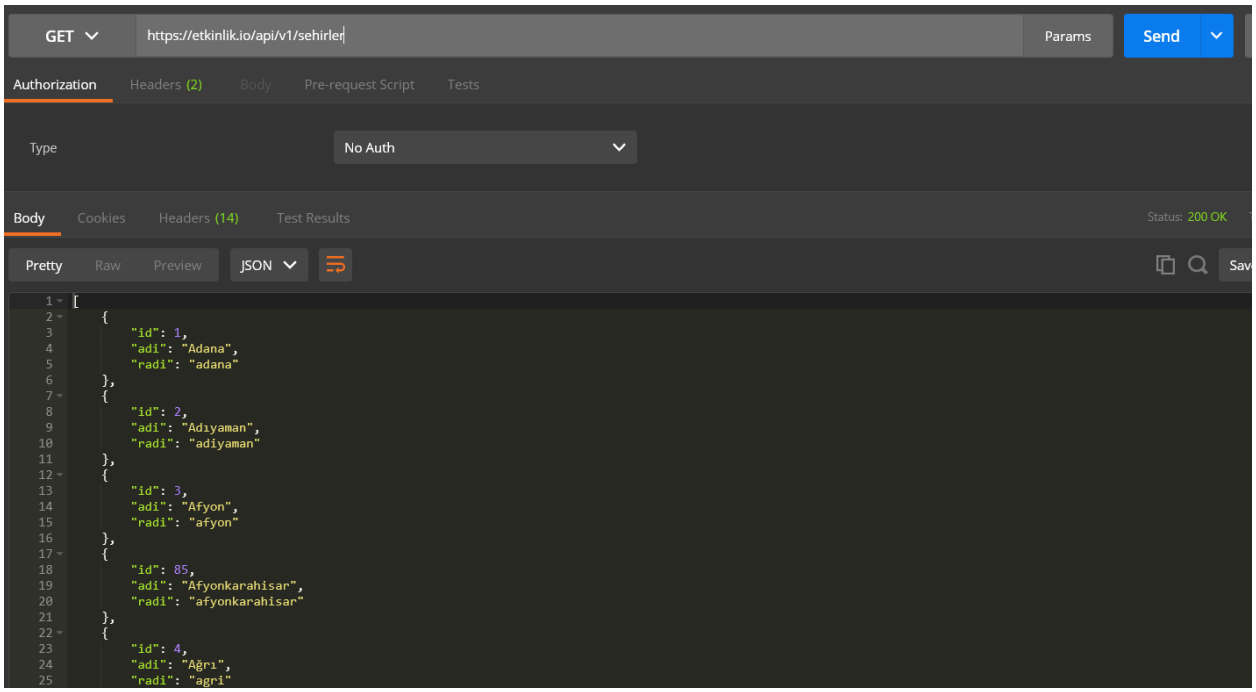
1. Twitter API

Used for single sign on and user profile information retrieval.

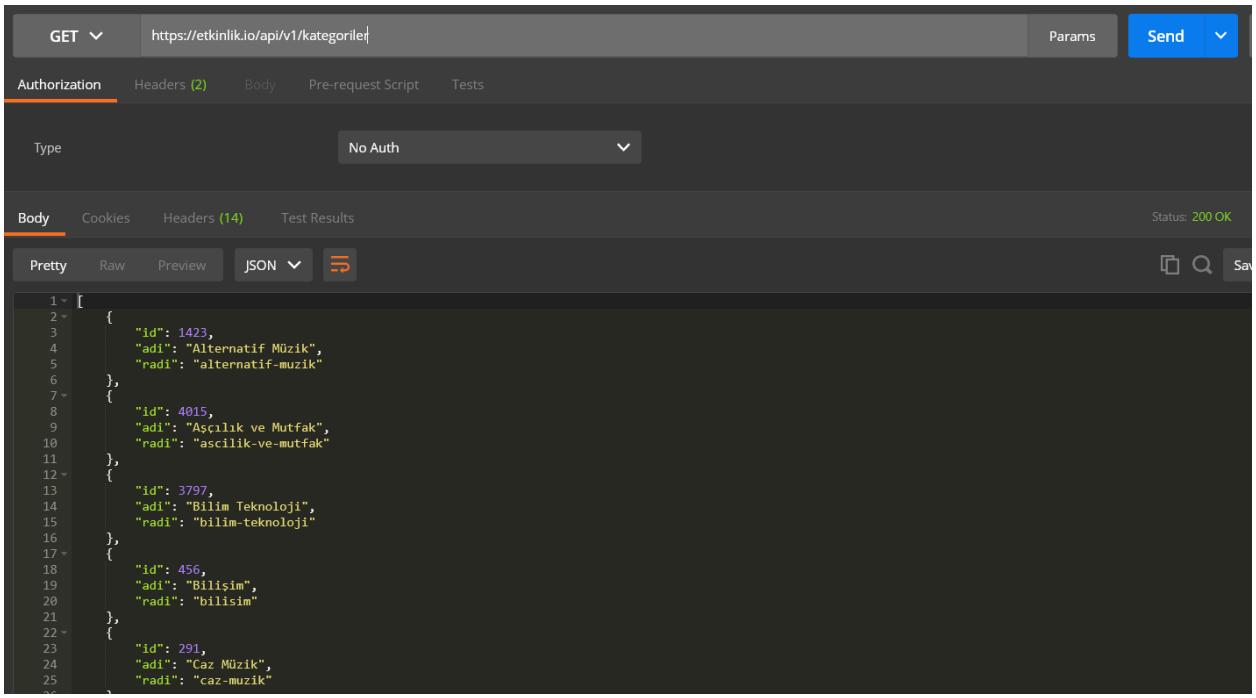
2. Etkinlik.io API

Used for retrieving upcoming event details according to specified search criteria. Some sample request/response pairs are as follows:

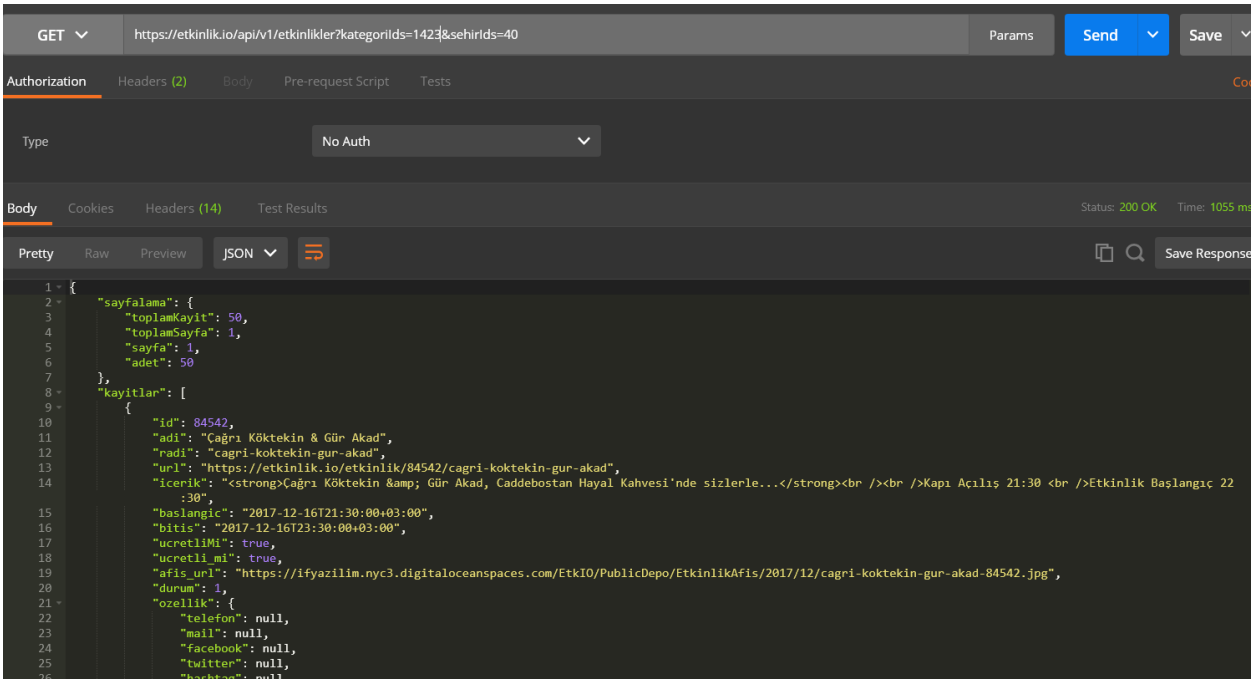
- Retrieve Predefined Locations



- Retrieve Predefined Event Categories



- Filter events by category and location



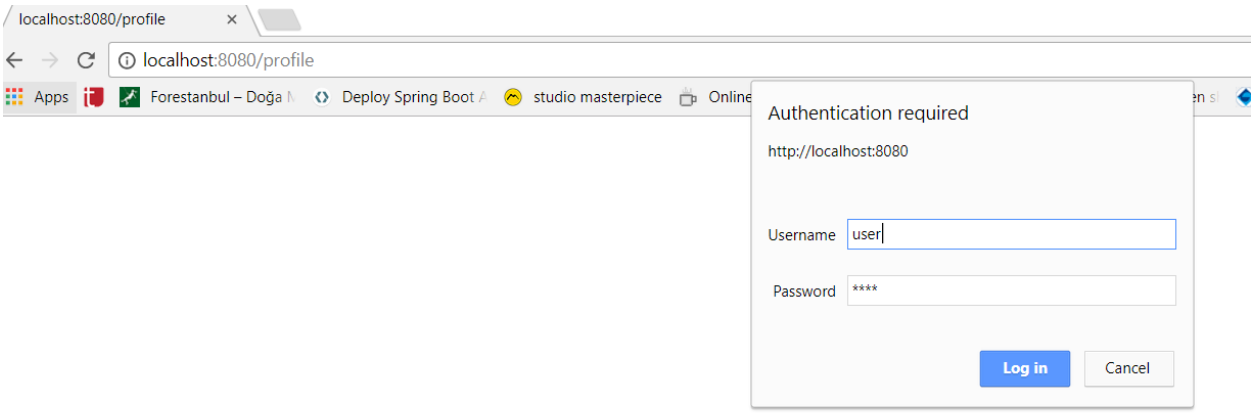
7. Known Issues

Following requirements have not been implemented yet.

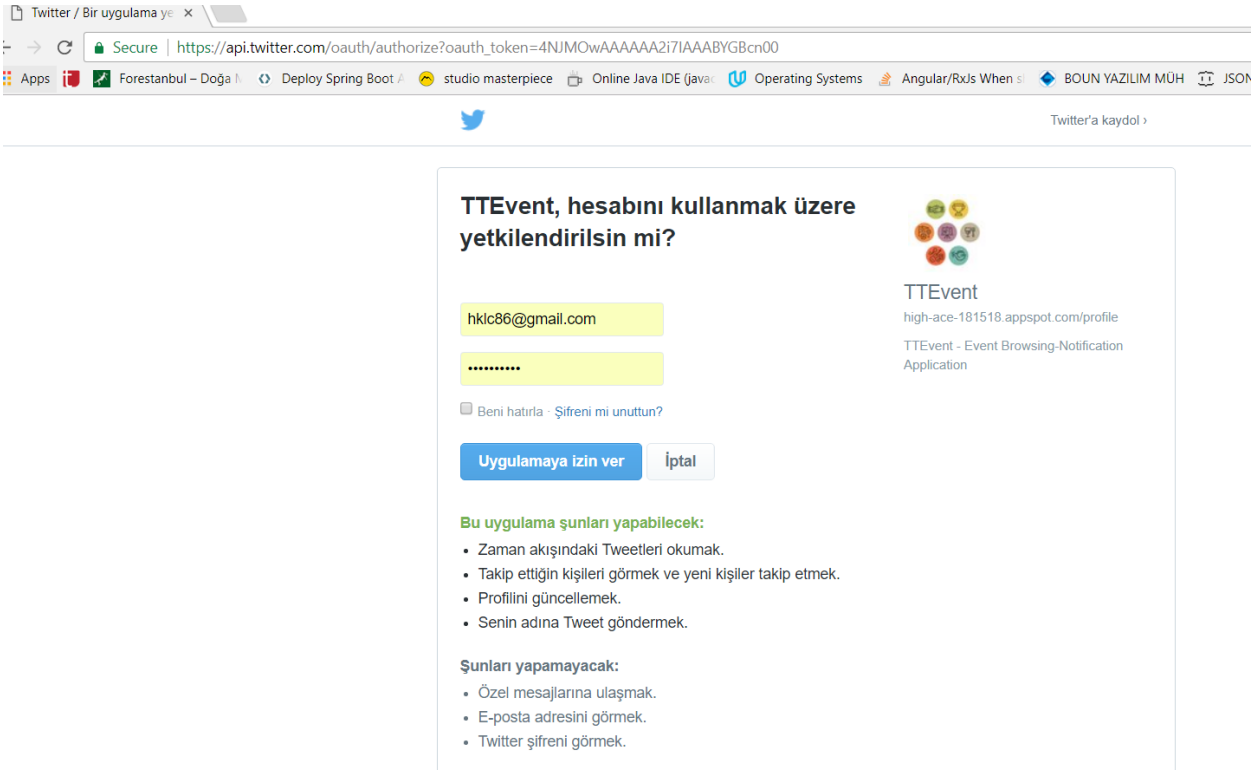
- 6.2 Filter Events by Selected Keywords
- 7.2 Send Notification via Twitter Direct Message

8. User Manual

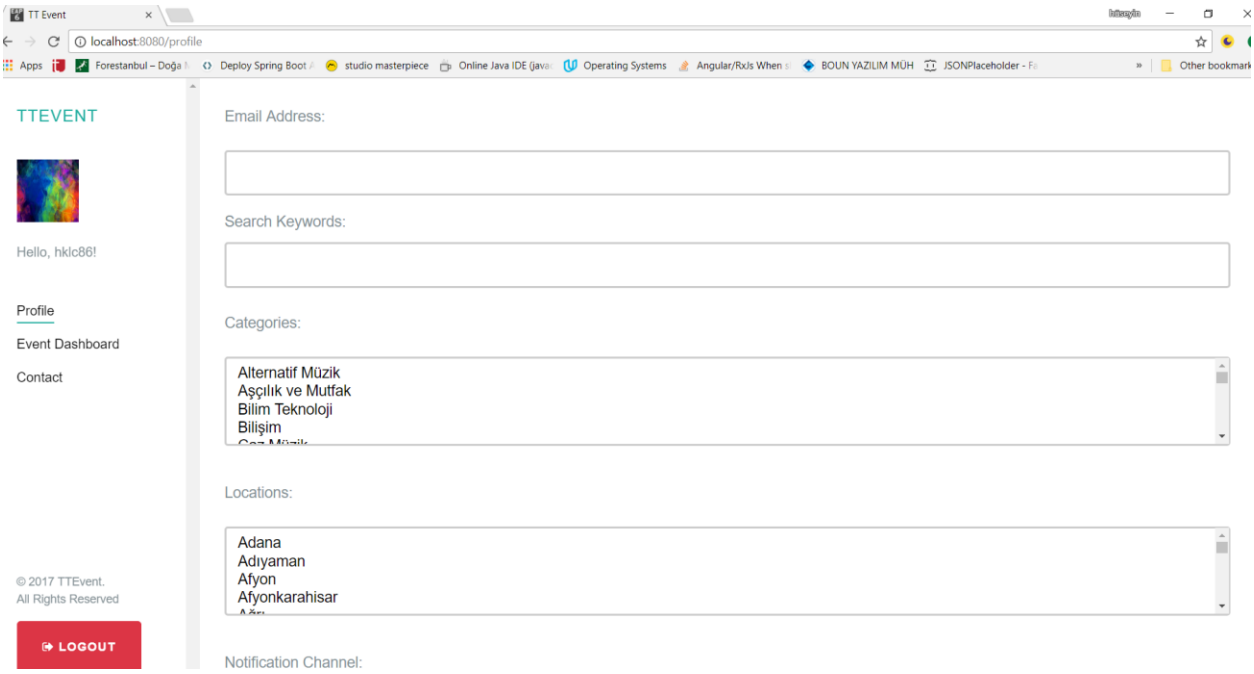
Once the application starts, an authentication popup shows up. For testing purposes, both username and password are set as "*user*".



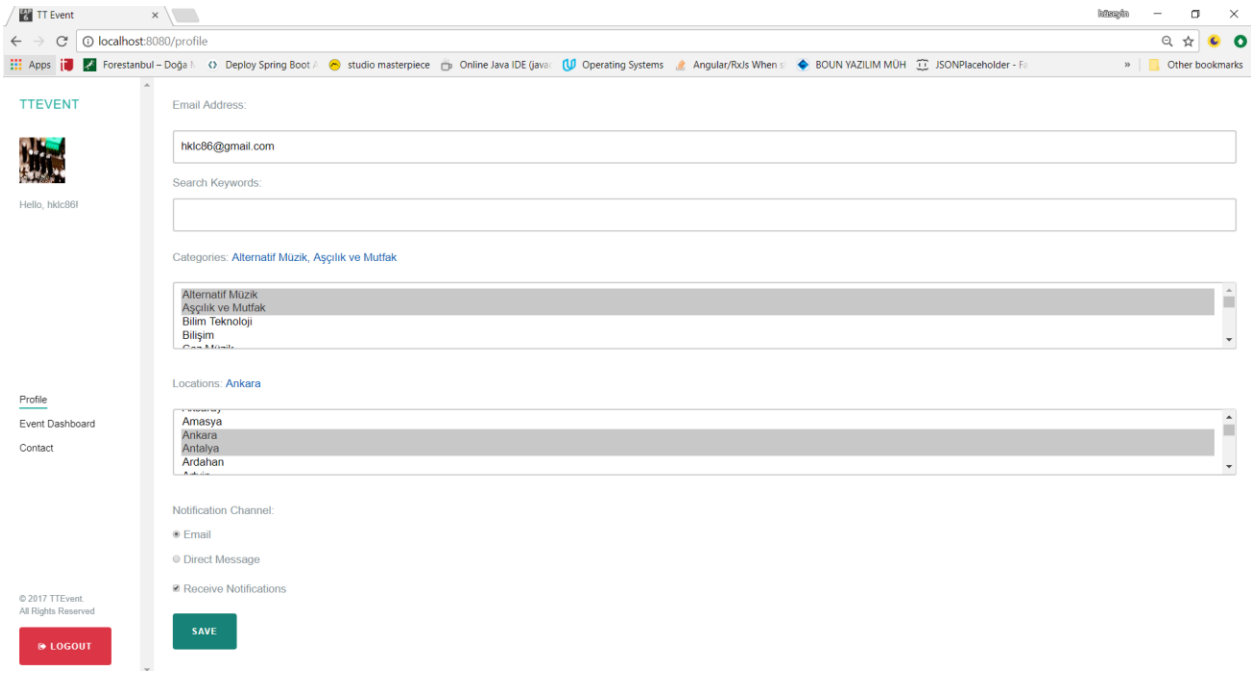
In order to be able to use the application, connect with twitter button should be clicked and requested permissions should be granted to the application.



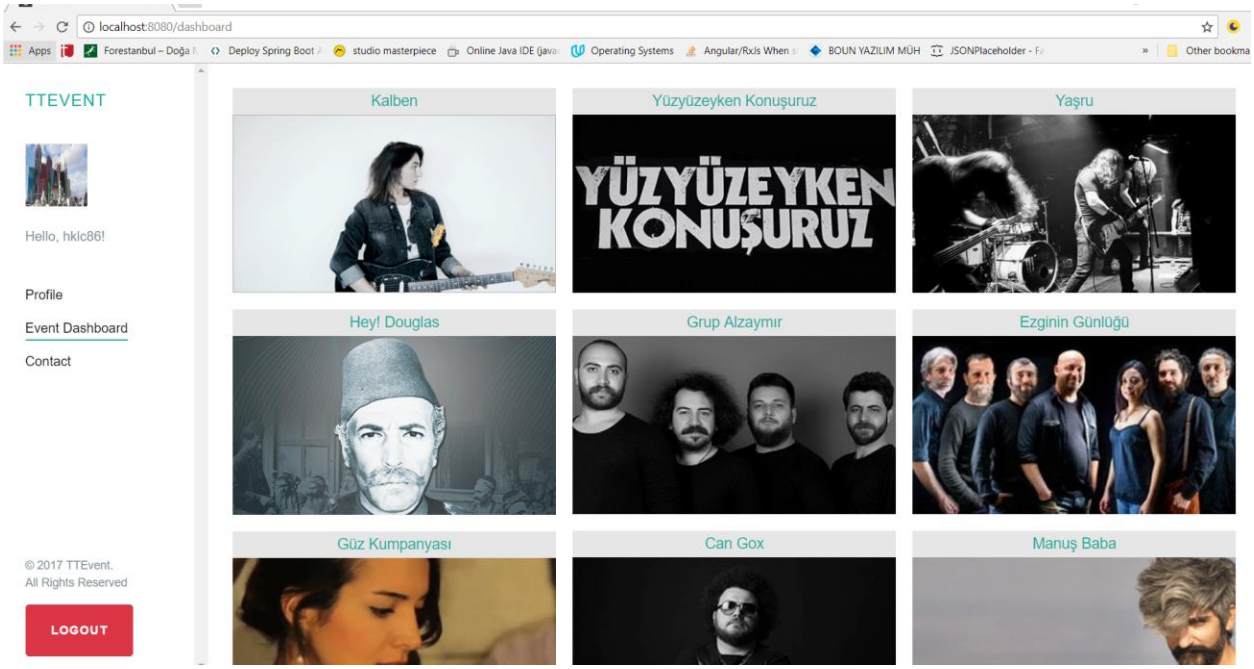
The post-login page is the profile management page, in which user should specify his email address and preferences.



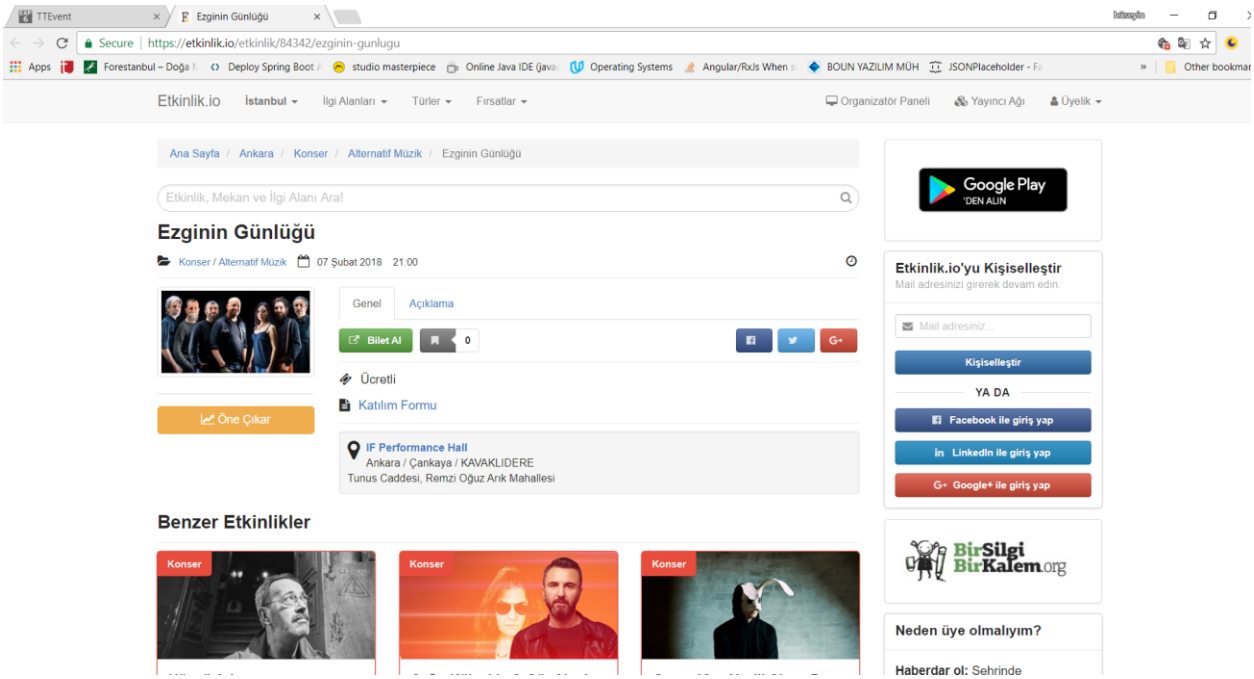
Users can select up to 5 categories and 5 locations. Please note that notification via direct message is not available yet. Cliking on the save button is the final step of registration.



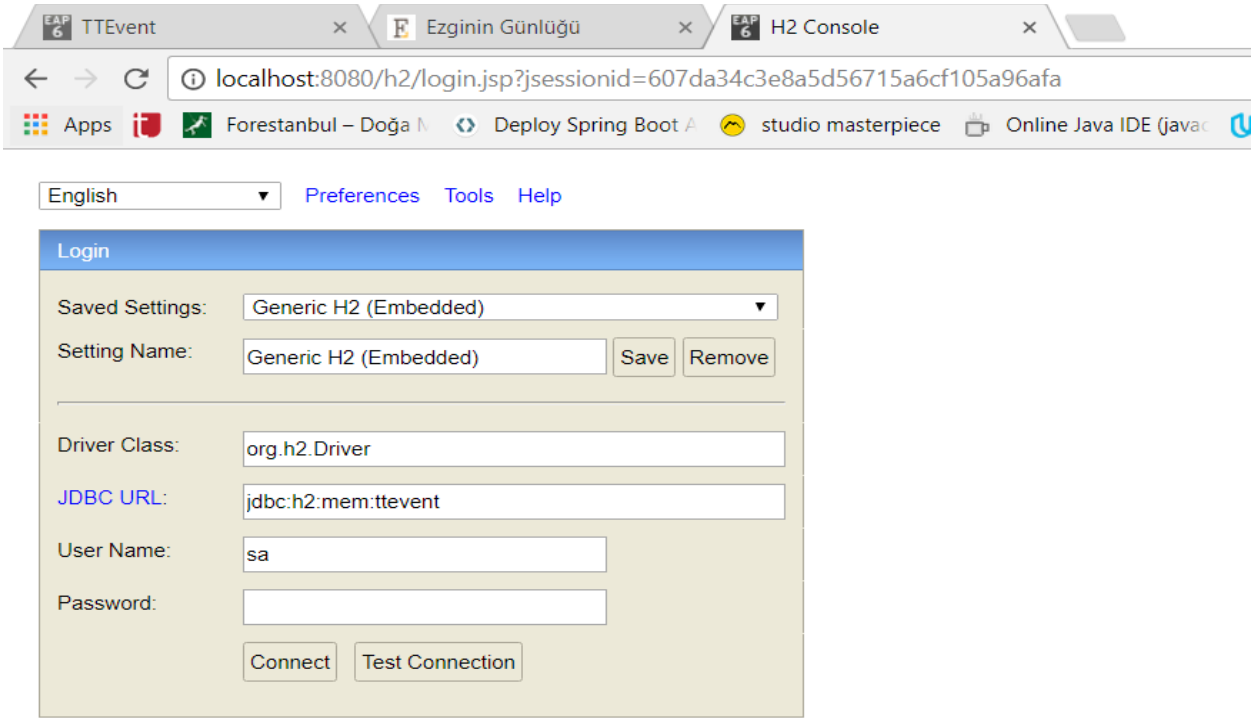
Following the completion of user profile and preferences, user is redirected to the dashboard page. In this page, events that match the selected categories/locations are listed.



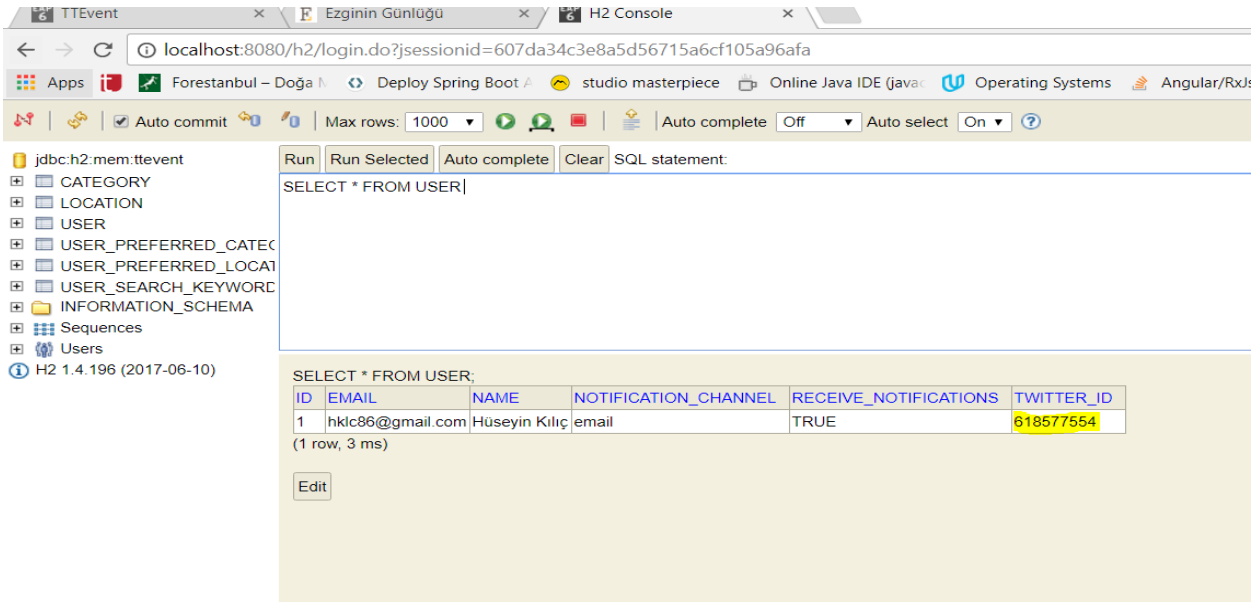
Users can click on any of the events and visit the external event details page.



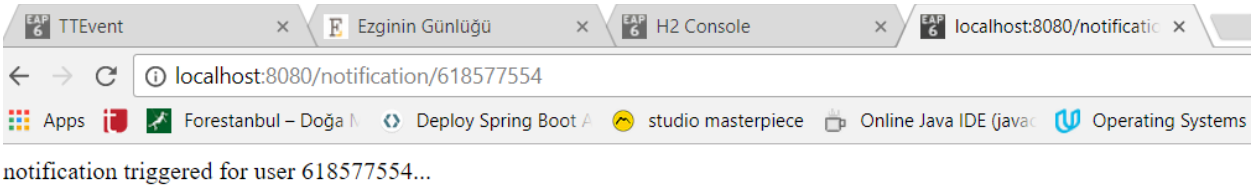
To be able to see the data stored in the database, you can access the H2 console of the application by visiting localhost:8080/h2. This console is currently enabled for everyone for testing purposes. Please note that JDBC URL and user name should be exactly the same as in the screenshot. No password is required for this time.



Once you connect to the database, you can access the details of the registered user with a simple sql query. Please note down the twitter id of the user as it will be user in the next step.



For the moment, in order not to create spam emails, the application is configured to run the scheduled email notification sending job once in an hour. Still you can test notification sending functionality manually using the endpoint as in the screenshot below. Please note that the numeric path parameter should be the twitter_id obtained from the database in the previous step.



After the notification triggering, user is sent an email containing the list of events that the user may be interested in.