



CS353 Database Systems

Final Report

Local Events Application

Group 1

Alper Mumcular - 21902740
Vesile İrem Aydın - 21902914
Ravan Aliyev - 21500405
Ece Kahraman - 21801879

Table of Contents

| | |
|---|----------|
| 1. Description | 3 |
| 2. Individual Contributions | 4 |
| 3. Final E/R Model | 4 |
| 3.1. Modifications to the Model | 4 |
| 3.2. The Model | 4 |
| 4. Final List of Tables as Relational Models | 6 |
| 4.1. User | 6 |
| 4.2. Admin | 6 |
| 4.3. Non Admin | 6 |
| 4.4. Participant | 6 |
| 4.5. Organizer | 6 |
| 4.6. Verified Organizer | 7 |
| 4.7. Event | 7 |
| 4.8. Paid Event | 7 |
| 4.9. Price | 7 |
| 4.10. Ticket | 7 |
| 4.11. Wallet | 8 |
| 4.12. Card | 8 |
| 4.13. Has | 8 |
| 4.14. Joins | 8 |
| 4.15. Purchase | 8 |
| 5. Implementation Details | 9 |
| 5.1. Frontend | 9 |
| 5.2. Backend | 9 |
| | 1 |

| | |
|---------------------------------------|-----------|
| 5.3. Database | 9 |
| 6. Advance Database Components | 10 |
| 6.1. View | 10 |
| 6.2. Triggers | 10 |
| 6.2.1. inc_quota | 10 |
| 6.2.2. dec_quota | 11 |
| 6.2.3. inc_points | 11 |
| 6.2.4. dec_points | 11 |
| 6.2.5. inc_purchase | 12 |
| 6.2.6. dec_purchase | 12 |
| 6.2.7. inc_popularity | 13 |
| 6.2.8. dec_popularity | 13 |
| 6.2.9. inc_paid_popularity | 13 |
| 6.2.10. dec_paid_popularity | 14 |
| 6.2.11. refund | 14 |
| 6.3. Reports | 15 |
| 6.3.1. Report 1 | 15 |
| 6.3.2. Report 2 | 15 |
| 6.3.3. Report 3 | 16 |
| 6.3.4. Report 4 | 16 |
| 6.3.5. Report 5 | 16 |
| 6.3.6. Report 6 | 16 |
| 7. User's Manual | 17 |
| 8. Website | 43 |

1. Description

Eventium is a local events application where users can find a variety of events to participate in, they can contribute by creating their own events. If they get verified, the users can monetize their own events as a choice. The application has two main types of users: admins and non admins.

Admins are assumed to be assigned by the application company, so they are not allowed to change their login information. Only admins have access to the whole list of users, they can ban a user or lift the ban. They can see the whole list of events, past or upcoming. They are able to force an event cancellation if needed.

Non admin users are participants, organizers, and verified organizers. All of them need to register to the system by entering their information like name, address, and telephone number. An unverified user is assumed as a participant upon login. They can see their whole upcoming events in a list on their homepage. They can view their active tickets on a different page, and they can increase the balance in the wallet assigned to them by entering their credit cards. They can search for a specific event and join it directly or purchase some tickets for it. Every participant can organize an event by switching to the organizer page, but they are not allowed to monetize their events. A participant gains 50 participation points on every join, and an organizer gains 1 popularity on every person joining their event. They also lose the same amount if a cancellation happens. An organizer can only be verified by directly reaching out to the assumed company and proving that they are affiliated with a reputable company or person. Once an admin verifies them, the organizer can no longer be a participant, and the organizer name for them becomes the organization name. They are also given the choice to monetize the event during the creation.

2. Individual Contributions

- Vesile İrem Aydın: İrem took care of the creation of the pages.
- Alper Mumcular: Alper mainly took care of the management of data in the database. He also helped the connection of the pages.
- Ece Kahraman: Ece connected the pages to the database.
- Ravan Aliyev:

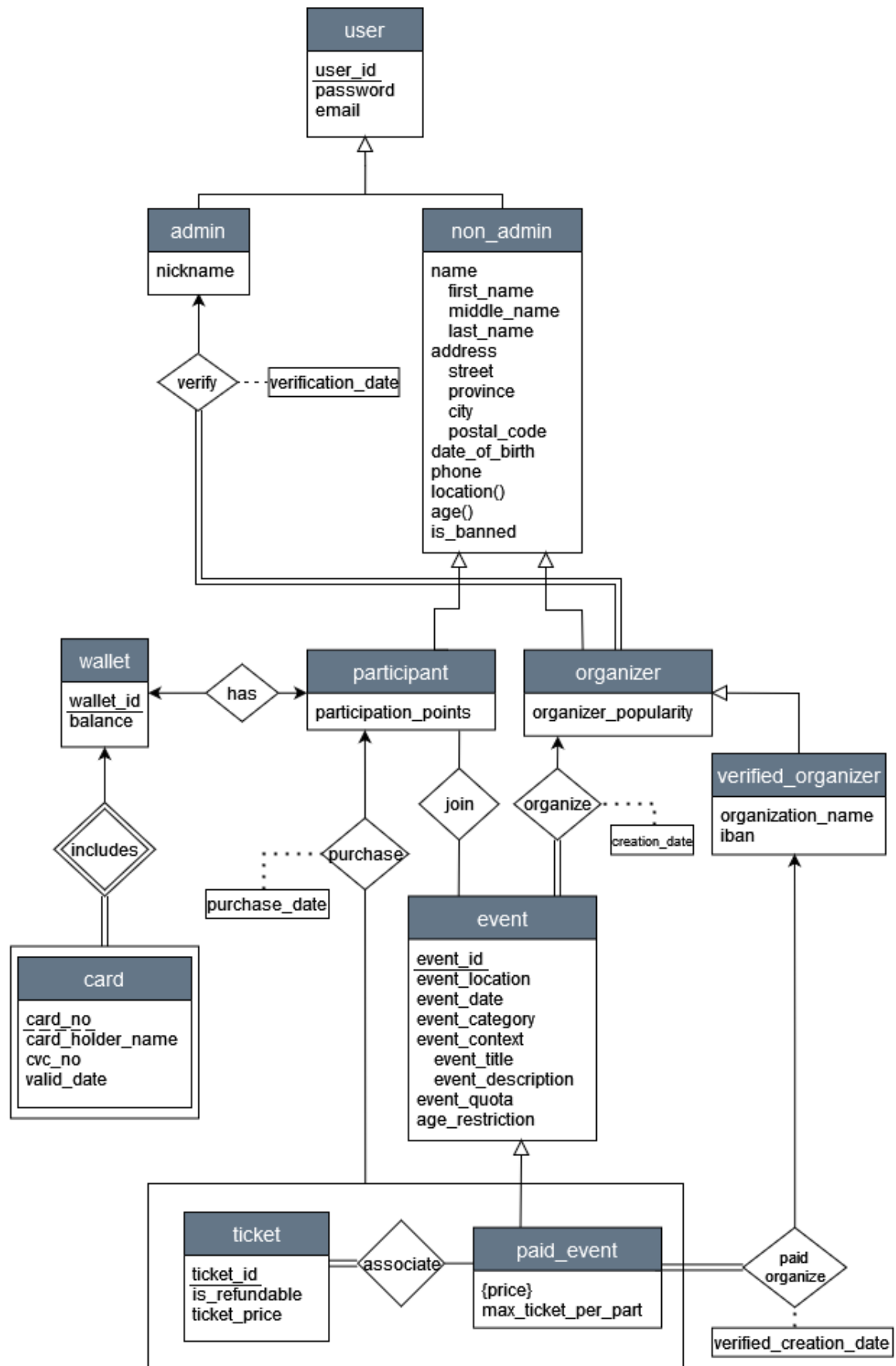
3. Final E/R Model

3.1. Modifications to the Model

- The *organize* relation between *event* and *organizer* was changed to many-to-one by adding an arrow pointing at *Organizer*.
- *Is_banned* attribute was added to the *non_admin* table to prevent banned users from logging in and accessing the platform.
- The weak entity *report* was removed

3.2. The Model

The E/R Model Diagram is given on the next page.



4. Final List of Tables as Relational Models

4.1. User

user(user_id, password, email)

Primary key: user_id

4.2. Admin

admin(user_id, nickname)

Primary key: user_id

Foreign key: user_id references user(user_id)

4.3. Non Admin

non_admin(user_id, first_name, middle_name, last_name, street, province, city, postal_code, date_of_birth, phone, is_banned)

Primary key: user_id

Foreign key: user_id references user(user_id)

4.4. Participant

participant(user_id, participation_points)

Primary key: user_id

Foreign key : user_id references non_admin(user_id)

4.5. Organizer

organizer(user_id, organizer_popularity)

Primary key: user_id

Foreign key: user_id references non_admin(user_id)

4.6. Verified Organizer

verified_organizer (user_id, organization_name, iban, admin_id, verification_date)

Primary key: user_id

Foreign key: user_id references organizer(user_id)

Foreign key: admin_id references admin(user_id)

4.7. Event

event(event_id, user_id, creation_date, event_location, event_date, event_category, event_title, event_description, event_quota, age_restriction)

Primary key: event_id

Foreign key: user_id references organizer(user_id)

4.8. Paid Event

paid_event(event_id, max_ticket_per_part, user_id)

Primary key: event_id

Foreign key: event_id references event(event_id)

Foreign key: user_id references verified_organizer(user_id)

4.9. Price

price(event_id, ticket_price)

Primary key: (event_id, ticket_price)

Foreign key: event_id references paid_event(event_id)

4.10. Ticket

ticket(ticket_id, is_refundable, event_id, ticket_price)

Primary key: ticket_id

Foreign key: (event_id, ticket_price) references price(event_id, ticket_price)

4.11. Wallet

wallet(wallet_id, balance)

Primary key: wallet_id

4.12. Card

card(wallet_id, card_no, card_holder_name, cvc_no, valid_date)

Primary key: (wallet_id, card_no)

Foreign key: wallet_id references wallet(wallet_id)

4.13. Has

has(user_id, wallet_id)

Primary key: user_id

Foreign key: user_id references participant(user_id)

Foreign key: wallet_id references wallet(wallet_id)

4.14. Joins

joins(user_id, event_id)

Primary key: (user_id, event_id)

Foreign key: user_id references participant(user_id)

Foreign key: event_id references event(event_id)

4.15. Purchase

purchase(ticket_id, user_id, purchase_date)

Primary key: ticket_id

Foreign key: ticket_id references ticket(ticket_id)

Foreign key: user_id references participant(user_id)

5. Implementation Details

5.1. Frontend

We decided to use HTML, CSS, JavaScript and Bootstrap libraries for the front-end. Initially, we tried to use the React framework. However, we encountered some problems while trying to connect the front-end and back-end. Then, we stopped using React and used technologies we used in homework 4 since we are more comfortable with them. First, we created HTML files and constructed the basics of the website. Then, CSS files are created and some Bootstrap attributes are added to style HTML files. After that, Javascript is used to add basic functionalities to the website such as adding extra components or a modal is displayed when a button is clicked. Since more complex functionalities are dealt with on the back-end side, Javascript is generally used for some front-end related tasks.

5.2. Backend

For the connection between the frontend and the database, we decided to use PHP. One of our team members became comfortable with using PHP and its mysqli library with the object-oriented style during the implementation of the fourth homework, therefore we believed using a language at least one of us is used to would be beneficial for the project implementation. The user id is added to the session variables during the successful log in so that we can access it wherever we can. In one or two cases, event id is also added to the session variables but it is not accessed by any other files except where it was genuinely needed. Except for those cases, the inputs taken from the user are passed by the get or post variables.

5.3. Database

During the design stage, we wanted to use the local servers created by the MySQL workbench. Due to the mismatch of some PATH variables on the Windows OS, we could not connect to the server on either of our team member's computers. Therefore, since we were familiar with the MariaDB server on Dijkstra, we decided to use that for our project. The SQL statements were passed to the server as a string with PHP. To initialize our server during the implementation, we wrote a Java file using Java's SQL library. When we were done with the implementation, instead of writing on the Java file one by one, we used our connected project to populate our database.

6. Advance Database Components

6.1. View

While inserting each event into the database, we also create a view for that event. By doing this, the organizer can list the people participating in the event and even remove the participants from the event. Briefly, it is used to hide the information of the participants that we do not want to be seen from the organizer.

inputs: given_eventID

```
CREATE VIEW view AS
  SELECT P.first_name, P.middle_name, P.last_name,
 P.date_of_birth, P.phone, P.user_id
  FROM joins J NATURAL JOIN non_admin P
 WHERE J.event_id = given_eventID AND P.user_id = J.user_id
```

6.2. Triggers

We created triggers for the mundane and repeated tasks which were increasing and decreasing participation points, organizer popularity, and event quota for both free and paid events after joining an event, purchasing some tickets for paid events and the cancellation of both.

6.2.1. inc_quota

This trigger is called when a deletion on the *joins* table occurs, indicating a cancellation of a free participation of the event. Upon removal, the remaining quota held in the *event_quota* attribute is incremented by 1.

```
DROP TRIGGER IF EXISTS inc_quota

CREATE TRIGGER inc_quota AFTER DELETE ON joins
FOR EACH ROW
BEGIN
  UPDATE event
  SET event_quota = event_quota + 1
  WHERE event_id = OLD.event_id;
END;
```

6.2.2. dec_quota

This trigger is called when an insertion on the *joins* table occurs, indicating a new free participation for the event. Upon insertion, the remaining quota held in the *event_quota* attribute is decremented by 1.

```
DROP TRIGGER IF EXISTS dec_quota

CREATE TRIGGER dec_quota AFTER INSERT ON joins
FOR EACH ROW
BEGIN
UPDATE event
SET event_quota = event_quota - 1
WHERE event_id = NEW.event_id;
END;
```

6.2.3. inc_points

This trigger is called when an insertion on the *joins* table occurs, indicating a new free participation for the event. Upon insertion, the participation points of the participant held in the *participation_points* attribute is incremented by 50.

```
DROP TRIGGER IF EXISTS inc_points

CREATE TRIGGER inc_points AFTER INSERT ON joins
FOR EACH ROW
BEGIN
UPDATE participant
SET participation_points = participation_points + 50
WHERE user_id = NEW.user_id;
END;
```

6.2.4. dec_points

This trigger is called when an deletion on the *joins* table occurs, indicating a cancellation of a free participation of the event. Upon deletion, the participation points of the participant held in the *participation_points* attribute is decremented by 50.

```
DROP TRIGGER IF EXISTS dec_points

CREATE TRIGGER dec_points AFTER DELETE ON joins
FOR EACH ROW
BEGIN
UPDATE participant
SET participation_points = participation_points - 50
```

```
WHERE user_id = OLD.user_id;
END;
```

6.2.5. inc_purchase

This trigger is called when an insertion on the *purchase* table occurs, indicating purchases of some tickets of a paid event. Upon insertion, the remaining quota held in the *event_quota* attribute is incremented by 1. Since one participant can buy multiple tickets at once, updating the quota for each ticket means at most the same amount of people will attend the event.

```
DROP TRIGGER IF EXISTS inc_purchase

CREATE TRIGGER inc_purchase AFTER DELETE ON purchase
FOR EACH ROW
BEGIN
UPDATE event
SET event_quota = event_quota + 1
WHERE event_id = (select event_id from ticket T where
OLD.ticket_id = T.ticket_id);
END;
```

6.2.6. dec_purchase

This trigger is called when a deletion on the *purchase* table occurs, indicating cancellation of some tickets of a paid event. Upon deletion, the remaining quota held in the *event_quota* attribute is decremented by 1. Since one participant can buy multiple tickets at once, updating the quota for each ticket means at most the same amount of people will not attend the event.

```
DROP TRIGGER IF EXISTS dec_purchase

CREATE TRIGGER dec_purchase AFTER INSERT ON purchase
FOR EACH ROW
BEGIN
UPDATE event
SET event_quota = event_quota - 1
WHERE event_id = (select event_id from ticket T where
NEW.ticket_id = T.ticket_id);
END;
```

6.2.7. inc_popularity

This trigger is called when an insertion on the *joins* table occurs, indicating a new free participation for the event. Upon insertion, the popularity of the organizer held in the *organizer-popularity* attribute is incremented by 1.

```
DROP TRIGGER IF EXISTS inc_popularity

CREATE TRIGGER inc_popularity AFTER INSERT ON joins
FOR EACH ROW
BEGIN
    UPDATE organizer
    SET organizer_popularity = organizer_popularity + 1
    WHERE user_id = (select E.user_id from event E where E.event_id
= NEW.event_id);
END;
```

6.2.8. dec_popularity

This trigger is called when a deletion on the *joins* table occurs, indicating a cancellation of a free participation of the event. Upon removal, the popularity of the organizer held in the *organizer_popularity* attribute is decremented by 1.

```
DROP TRIGGER IF EXISTS dec_popularity

CREATE TRIGGER dec_popularity AFTER DELETE ON joins
FOR EACH ROW
BEGIN
    UPDATE organizer
    SET organizer_popularity = organizer_popularity - 1
    WHERE user_id = (select E.user_id from event E where E.event_id
= OLD.event_id);
END;
```

6.2.9. inc_paid_popularity

This trigger is called when an insertion on the *purchase* table occurs, indicating a selling of a ticket. Upon insertion, the popularity of the verified organizer held in the *organizer_popularity* attribute is incremented by 1.

```
DROP TRIGGER IF EXISTS inc_paid_popularity

CREATE TRIGGER inc_paid_popularity AFTER INSERT ON purchase
FOR EACH ROW
BEGIN
```

```

UPDATE event NATURAL JOIN organizer
SET organizer_popularity = organizer_popularity + 1
WHERE event_id = (select event_id from ticket T where
NEW.ticket_id = T.ticket_id);
END;

```

6.2.10. dec_paid_popularity

This trigger is called when a deletion on the *purchase* table occurs, indicating a cancellation of a ticket. Upon removal, the popularity of the verified organizer held in the *organizer_popularity* attribute is decremented by 1.

```

DROP TRIGGER IF EXISTS dec_paid_popularity

CREATE TRIGGER dec_paid_popularity AFTER DELETE ON purchase
FOR EACH ROW
BEGIN
UPDATE event NATURAL JOIN organizer
SET organizer_popularity = organizer_popularity - 1
WHERE event_id = (select event_id from ticket T where
OLD.ticket_id = T.ticket_id);
END;

```

6.2.11. refund

This trigger can be called on three different cases. When a participant cancels their refundable tickets, when a verified organizer cancels the whole event, or when an admin cancels the whole event, this trigger is called and it updates the balances of all the participating users with the amount they have spent on the tickets.

```

DROP TRIGGER IF EXISTS dec_paid_popularity

CREATE TRIGGER refund AFTER DELETE ON purchase
FOR EACH ROW
BEGIN
UPDATE participant P NATURAL JOIN has H NATURAL JOIN wallet W
SET W.balance = W.balance + (select ticket_price from ticket TI
where OLD.ticket_id = ticket_id)
WHERE OLD.user_id = P.user_id;
END;

```

6.3. Reports

We implemented the reports as a feature for the admin users. We determined 6 reports to be displayed. Once one of the reports is chosen, a pop-up appears if some inputs are needed to be given, and then the results of the reports are displayed on a different page which can be downloaded as a PDF file.

6.3.1. Report 1

This report displays the organizers in the selected location who have more popularity points than the given number.

inputs: given_num, given_location

```
SELECT  UNIQUE  O.user_id,  CONCAT(N.first_name,  '  ',
N.last_name) as full_name, O.organizer_popularity
FROM organizer O, event E, non_admin N
WHERE O.organizer_popularity>given_num AND E.event_location
= 'given_location' AND O.user_id = E.user_id AND N.user_id =
O.user_id;
```

6.3.2. Report 2

This report displays the free events with the most participants in every category in the selected location.

inputs: given_location

```
WITH temp(event_id,cat,cnt,title) as (
        SELECT  E.event_id,  E.event_category,
COUNT(J.user_id), E.event_title
        FROM joins J, event E
        WHERE  J.event_id  =  E.event_id  AND
E.event_location = 'given_location'
        GROUP BY E.event_id, E.event_category)
, temp2(cat, cnt) as (SELECT cat, MAX(cnt)
FROM temp GROUP BY cat)
SELECT cat, event_id, cnt, title
FROM temp NATURAL JOIN temp2
```


6.3.3. Report 3

This report displays the full names and the IDs of the participants who attended events last year in a given category.

inputs: given_category, and a PHP function to calculate last year “date('Y-m-d', strtotime('-1 year'))”

```
SELECT DISTINCT P.user_id, CONCAT(N.first_name, ' ', N.last_name) as full_name
FROM participant P NATURAL JOIN non_admin N, event E, joins J
WHERE P.user_id = J.user_id AND E.event_id = J.event_id AND
E.event_category = 'given_category' AND E.event_date > 'date('Y-m-d',
strtotime('-1 year'))'
```

6.3.4. Report 4

This report displays the events in the next month in the chosen category in the chosen location.

inputs: given_category, given_location, and PHP functions to calculate the next month's time period “date('Y-m-d', strtotime('+1 month'))” and “date('Y-m-d', strtotime('+2 month'))”

```
SELECT E.event_title, E.event_date
FROM event E
WHERE E.event_category = 'given_category' AND E.event_location =
'given_location' AND E.event_date > 'date('Y-m-d', strtotime('+1
month'))' AND E.event_date < 'date('Y-m-d', strtotime('+2 month'))'
```

6.3.5. Report 5

This report displays the most expensive and the cheapest ticket prices for each category.

```
SELECT E.event_category, max(P.ticket_price) as maxPrice,
min(P.ticket_price) as minPrice
FROM event E NATURAL JOIN price P
GROUP BY E.event_category
```

6.3.6. Report 6

This report displays the most popular and unpopular organizers which created at least 5 events in at least two different cities for all cities.

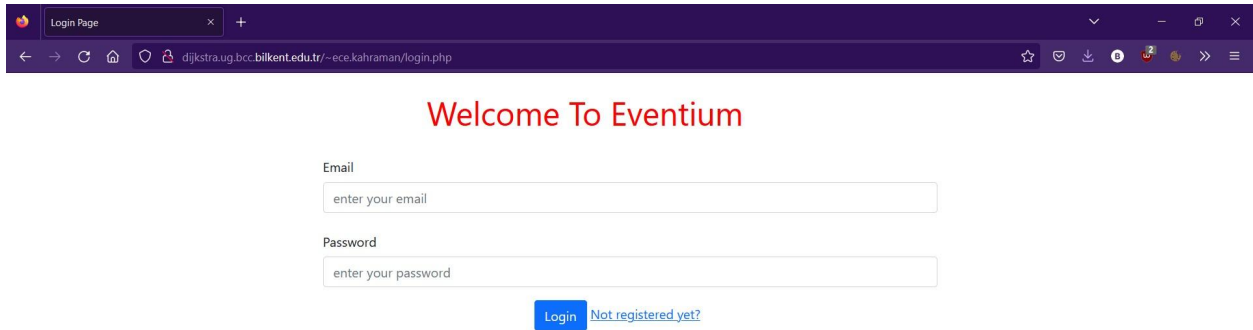
```

with maxp(event_location, organizer_popularity) as (SELECT N.city,
MAX(O.organizer_popularity)
      FROM organizer O NATURAL JOIN non_admin N
      WHERE 5 <= (SELECT COUNT(*)
                  FROM event E
                  WHERE O.user_id = E.user_id AND 2 <= (SELECT COUNT(
DISTINCT Ev.event_location)
                                                    FROM event Ev NATURAL
JOIN organizer Org
                                                    WHERE Org.user_id =
Ev.user_id AND E.user_id = Org.user_id
                                                    GROUP BY Org.user_id)
                  )
      GROUP BY N.city ),
minp(event_location, organizer_popularity) as (SELECT N.city,
MIN(O.organizer_popularity)
      FROM organizer O NATURAL JOIN non_admin N
      WHERE 5 <= (SELECT COUNT(*)
                  FROM event E
                  WHERE O.user_id = E.user_id AND 2 <= (SELECT COUNT(
DISTINCT Ev.event_location)
                                                    FROM event Ev NATURAL
JOIN organizer Org
                                                    WHERE Org.user_id =
Ev.user_id AND E.user_id = Org.user_id
                                                    GROUP BY Org.user_id)
                  )
      GROUP BY N.city ),
maxPerson(event_location, maxfullname) as (SELECT event_location,
CONCAT(first_name, ' ', last_name, ' ( ', MAX(organizer_popularity), ' ) ' )
as maxPerson
                                                    FROM maxp
NATURAL JOIN organizer NATURAL JOIN non_admin
                                                    GROUP BY event_location),
minPerson(event_location, minfullname) as (SELECT event_location,
CONCAT(first_name, ' ', last_name, ' ( ', MAX(organizer_popularity), ' ) ' )
as maxPerson
                                                    FROM minp
NATURAL JOIN organizer NATURAL JOIN non_admin
                                                    GROUP BY event_location)

SELECT *
from maxPerson NATURAL JOIN minPerson
      GROUP BY event_location

```

7. User's Manual



Login Page

dijkstra.ug.bcc.bilkent.edu.tr/~ece.kahraman/login.php

Welcome To Eventium

Email

Password

Login [Not registered yet?](#)

Figure 1: View of the login screen

When you enter the website you can login to the system by entering your email and password. If your variables are matched with the database values, you are directed to the homepage screen of your user type. If you do not have an account by clicking “Not registered yet?” you will be directed to the registration page.

The screenshot shows a web browser window with the title 'Register Page' and the URL 'dijkstra.ug.bcc.bilkent.edu.tr/~ece.kahraman/register.php'. The page has a red header 'Register To Eventium'. Below the header, there are several input fields for registration: 'Email' and 'Password' at the top; 'First Name', 'Middle Name', and 'Last Name' in the second row; 'City', 'Province', and 'Street' in the third row; and 'Postal Code', 'Phone', and 'Birth Date' in the fourth row. The 'Birth Date' field has a placeholder 'mm / dd / yyyy'. A blue 'Register' button is located below the 'Postal Code' field.

| | | | |
|-------------|-------------|------------|--|
| Email | | Password | |
| First Name | Middle Name | Last Name | |
| City | Province | Street | |
| Postal Code | Phone | Birth Date | |

Register

Figure 2: View of the register screen

If you do not have a Eventium account , you can register the system by entering your information. Middle name value is optional. Postal code and birthdate values are checked before creating the account. For instance, birth date cannot be a future date.

Update Profile

Welcome Bora! Participation points: 300 Your Information Organize Event

Change Email
bora@mail

Change Password
New Password
Confirm Password

Change Information

First Name: Bora Middle Name: Baran Last Name: Kima

City: Ankara Province: Çankaya Street: Universiteler

Postal Code: 6800 Phone: 05398089405 Birth Date: mm / dd / yyyy

Update

Go back to the home page

Figure 3: View of the edit profile screen

User information can be changed after creating the account.

Participant Home

Welcome Alper! Participation points: 0 Your Info Organize Event

Tickets Wallet

Search an event Search

Your Upcoming Events

You have no active participations!

Figure 4: View of the main page of participants

Participants are directed to the main page after login. On this page, they can see their upcoming events and cancel them. The search button is used to see nearby events and directs users to filter events page. The Organize Event button in the header is used to switch from participant to organizer. Users can log out using the arrow located on the left of the header.

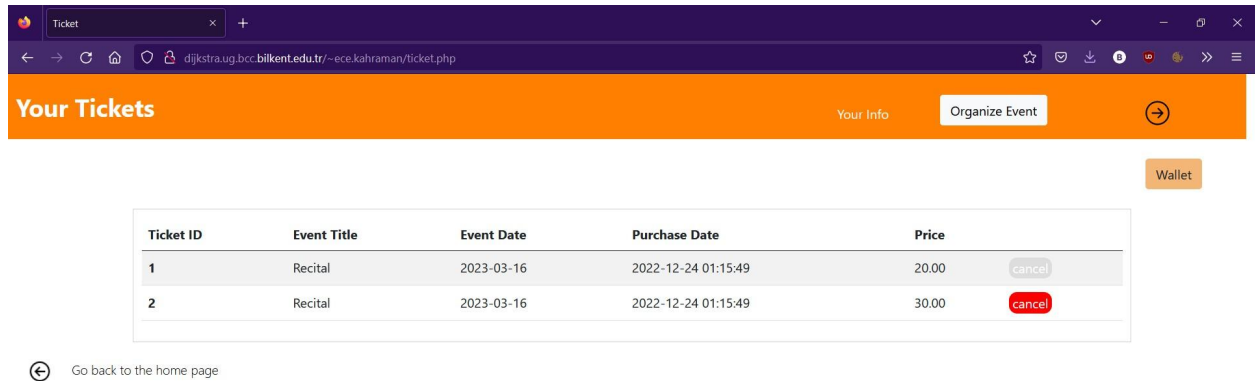


Figure 5: View of the your tickets screen

Tickets page of users displays all active and inactive tickets of users. Inactive tickets represent past activities. Tickets can be canceled. Their balance is automatically updated by trigger after canceling a participation.

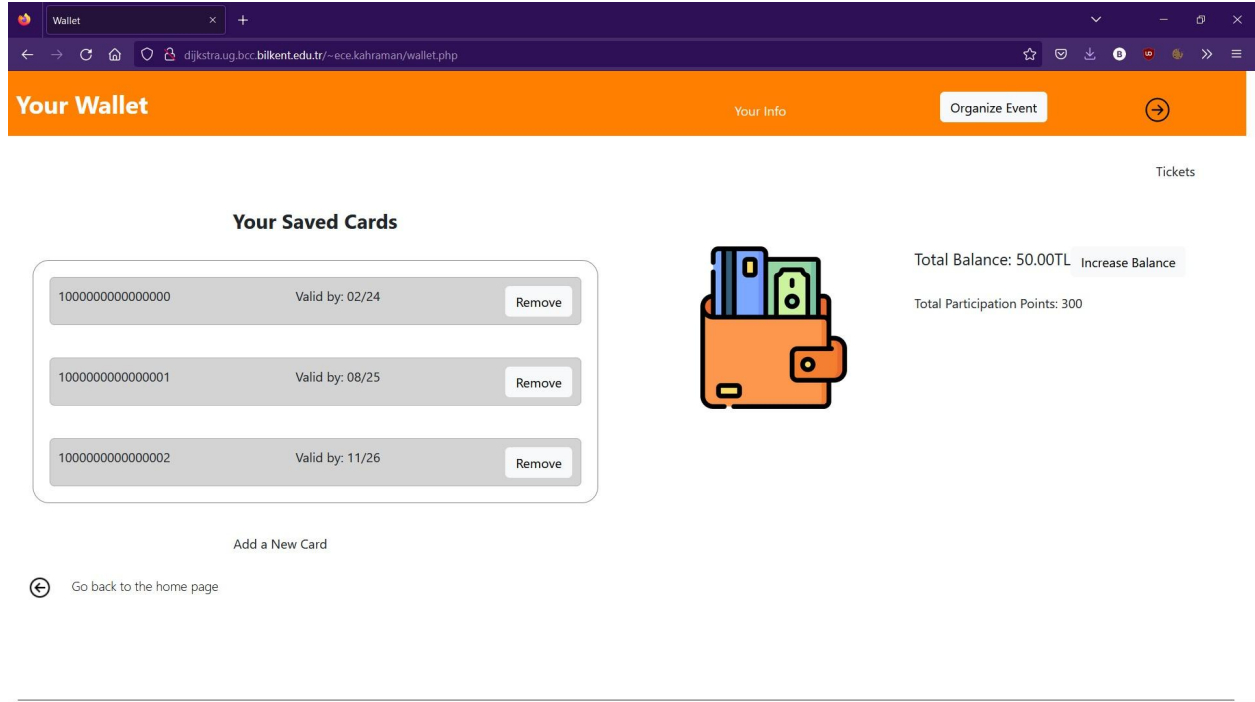


Figure 6: View of the your wallet screen

Wallet displays the amount of balance the user has. Next to the balance, the button prompts the user to increase their balance. By clicking the button, the user is redirected to that page to increase the balance from one of the saved cards. The page lists the saved credit cards, these cards can be edited or completely removed from the database. The user can also add a new card by clicking the button below the list. It redirects to the card addition modal. The user can reach their own personal data from the top bar, they can switch to the organizer mode, and log out by clicking the icon on the rightmost side of the bar. The user can move to the previous filtering list by clicking the arrow on the lower left corner.

Add Card

dijkstra.ug.bcc.bilkent.edu.tr/~ece.kahraman/add_card.php

Your Wallet

Your Info

Organize Event

Card Holder's Name

Name

Card No

Card No

Validation Date

mm / dd / yyyy

CVC

CVC

Add

Go back to wallet

Figure 7: View of the add card screen

The wallet page is used to add a new card to the system.

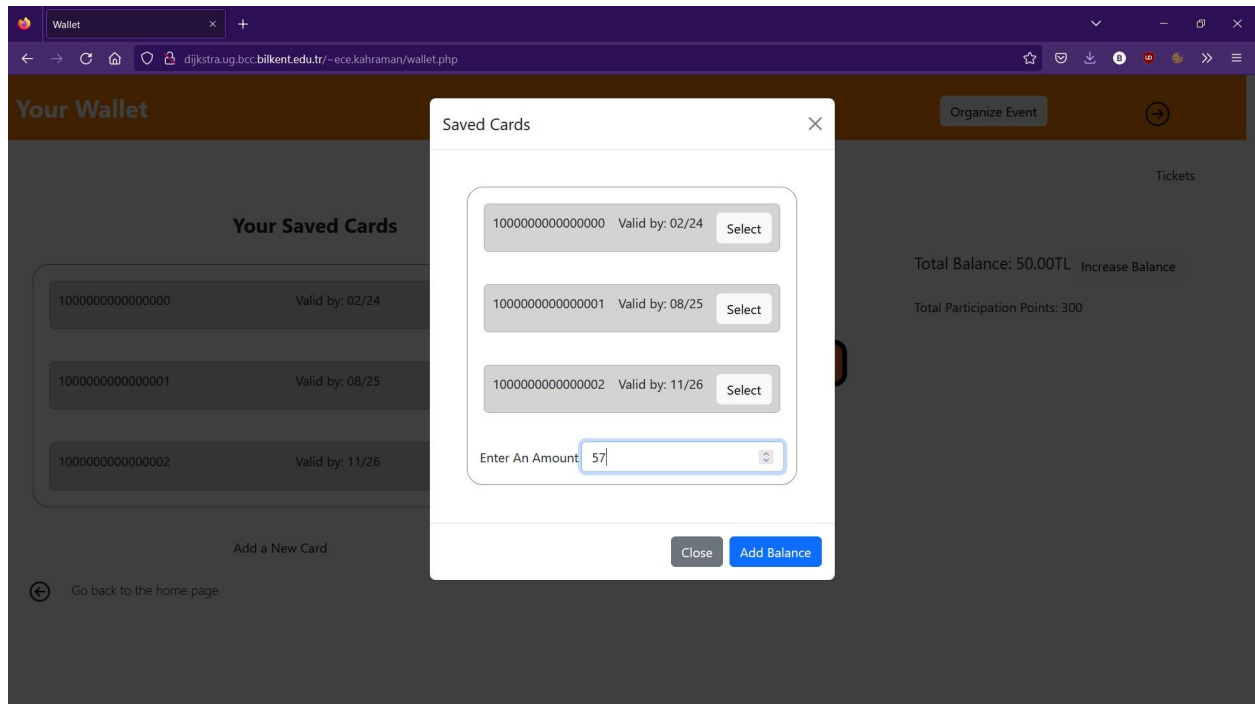


Figure 8: View of the add balance modal

In this modal, participants can increase their balance by selecting a card and entering an amount.

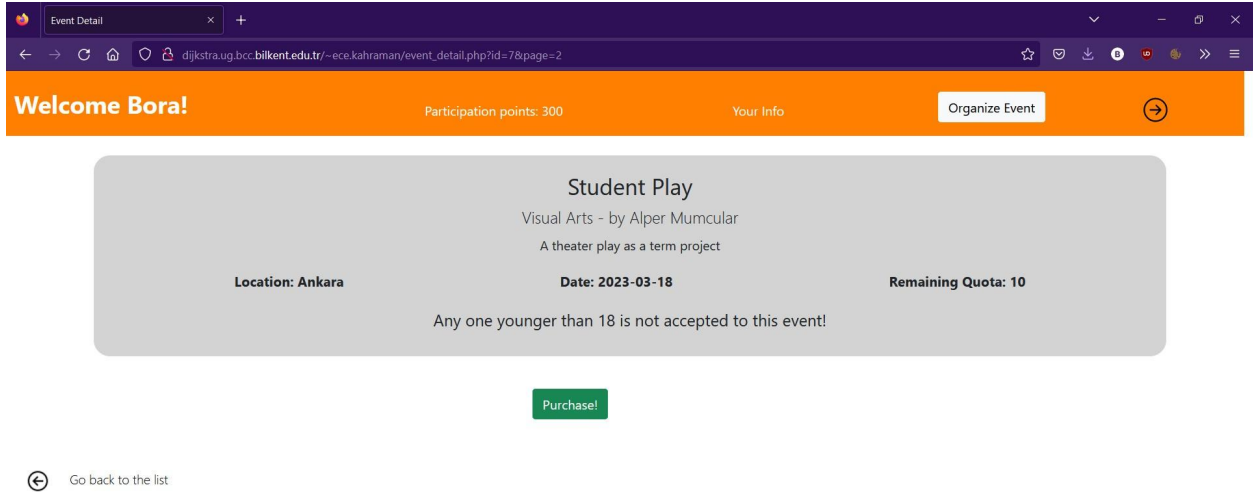


Figure 9: View of the view event screen

An event can be viewed in more detail when they are clicked on, in the filtering list. All details related to the event are displayed: its title, category, organizer, description, location, date, remaining quota, and age restriction. The user can join the event by clicking on the green button, and it will increment their participation points by 50 by a trigger. When the user joins, the green button turns gray. The user can reach their own personal data from the top bar, they can switch to the organizer mode, and log out by clicking the icon on the rightmost side of the bar. The user can move to the previous filtering list by clicking the arrow on the lower left corner.

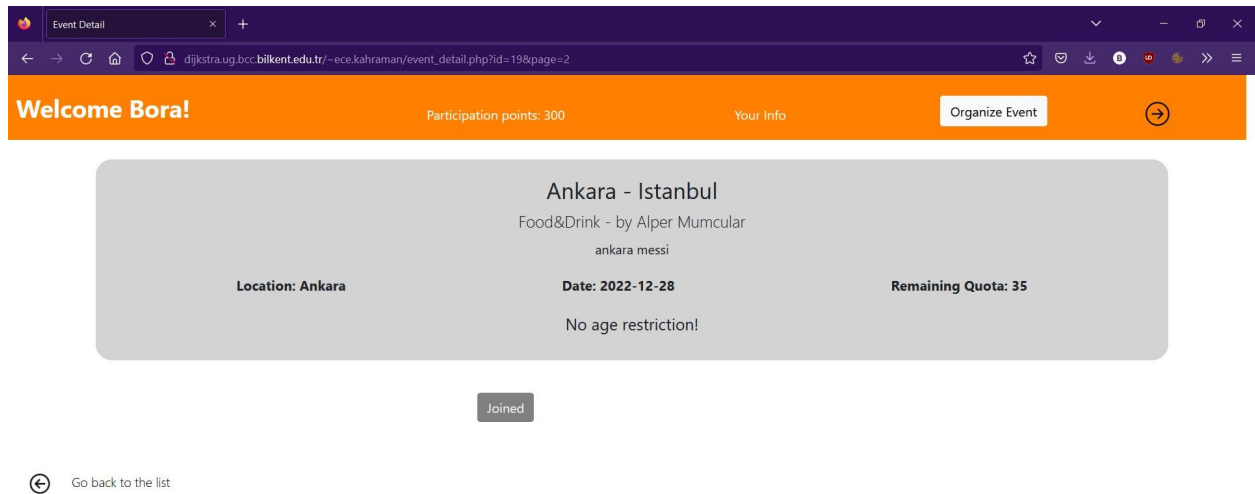


Figure 10: View of the view event screen after joining

A view event page after participating in the event.

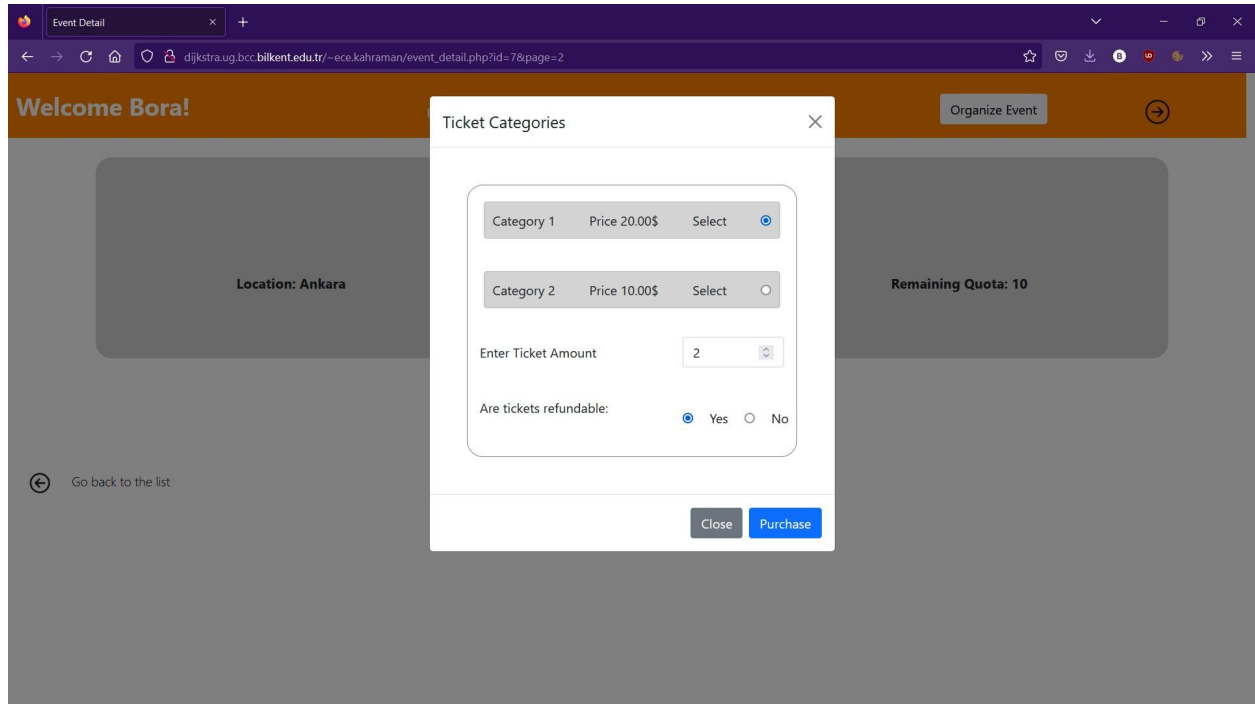


Figure 11: View of the ticket categories screen

For the priced events, a category must be selected. Users can buy the tickets by entering a ticket amount, and selecting if the ticket is refundable. If the ticket is refundable, users can cancel their ticket and get the money they paid for the event. If the ticket amount is more than the ticket per participant amount determined by the organizer, the system gives an error.

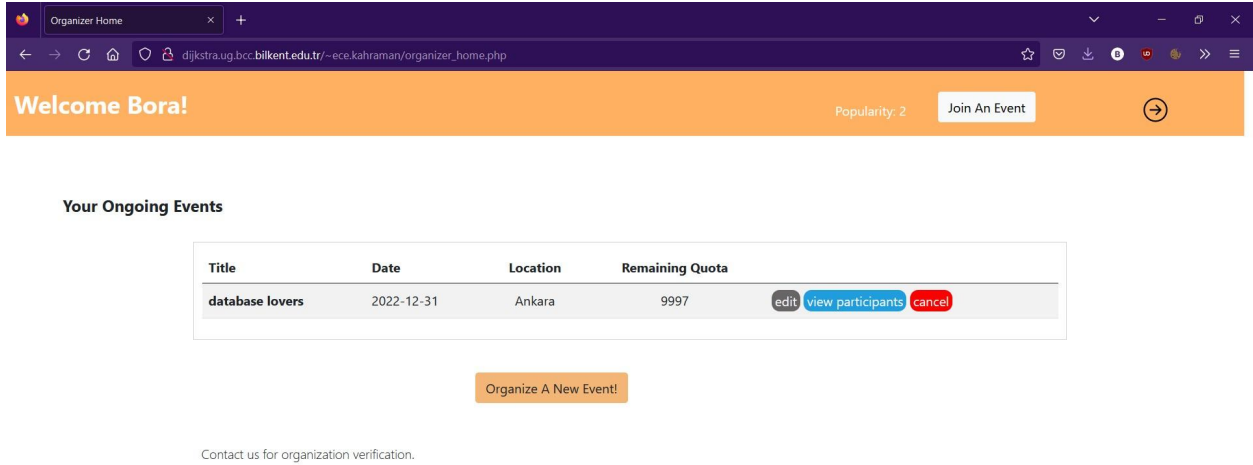


Figure 12: View of the main page of organizers

The main page of unverified organizers displays ongoing events of the organizer. On this page, organizers can edit the event, view participants and cancel the event. When the 'Organize A New Event!' button is clicked, the user is directed to the 'organize event' page. The 'Join An Event' button on the header is used to switch from organizer profile to participant profile. The popularity index of the organizer is shown on the top of the page. Organizers must contact the admins to get verification.

The screenshot shows a web browser window with the address bar displaying `dijkstra.ug.bcc.bilkent.edu.tr/~ece.kahraman/organize_event.php`. The page title is "Add Event". The main heading is "Create An Event". The form is a light orange box with the following fields:

| Field Label | Field Value / Placeholder |
|---------------------------|--|
| Title | Title |
| Details | Please explain details of the event... |
| Location | Ankara |
| Date | mm / dd / yyyy |
| Choose A Category | Music |
| Enter A Quota | Quota |
| Choose An Age Restriction | None |

Below the form is a "Create!" button. At the bottom left, there is a link with a circular arrow icon and the text "Go back to the home page".

Figure 13: View of the create an event screen

The create event page for unverified organizers requires all fields listed above to create a new event. A category in the list must be selected. The system has three age restriction options: none, +7 and +18. The necessary one for the event must be selected to create the event.

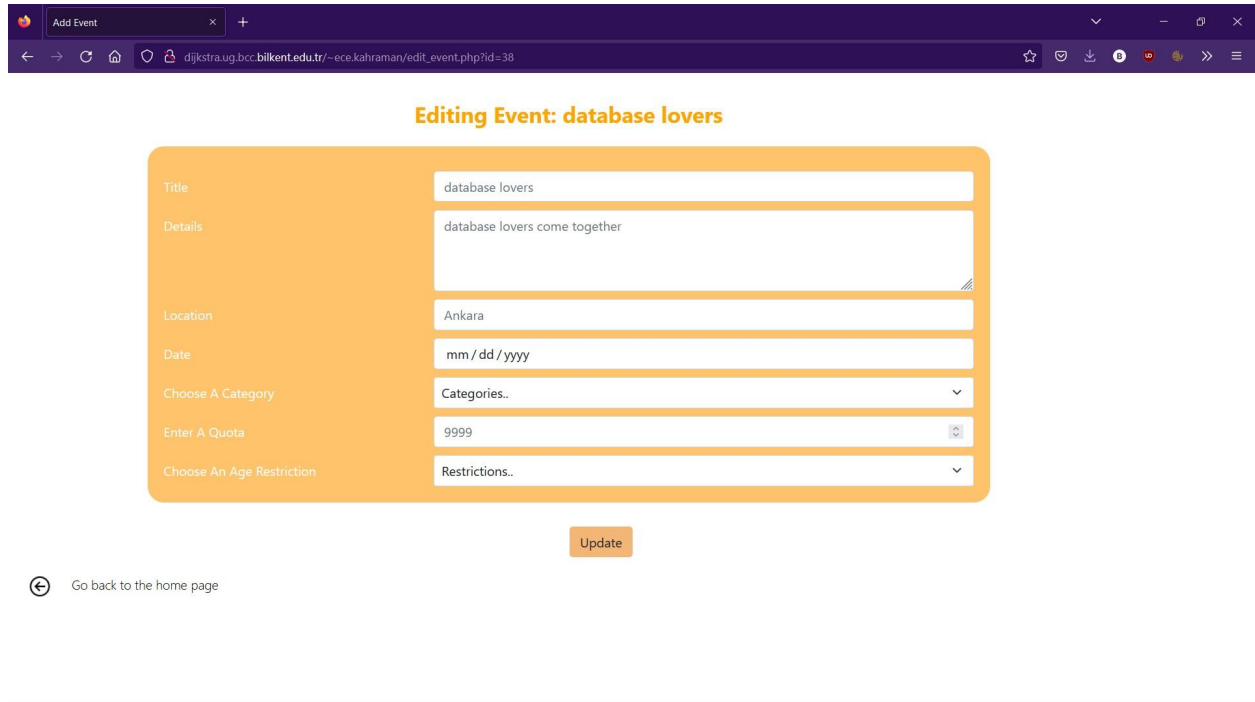


Figure 14: View of the edit an event screen

Created events can be edited by their organizers. The current values for the event is given as a hint on the text fields.

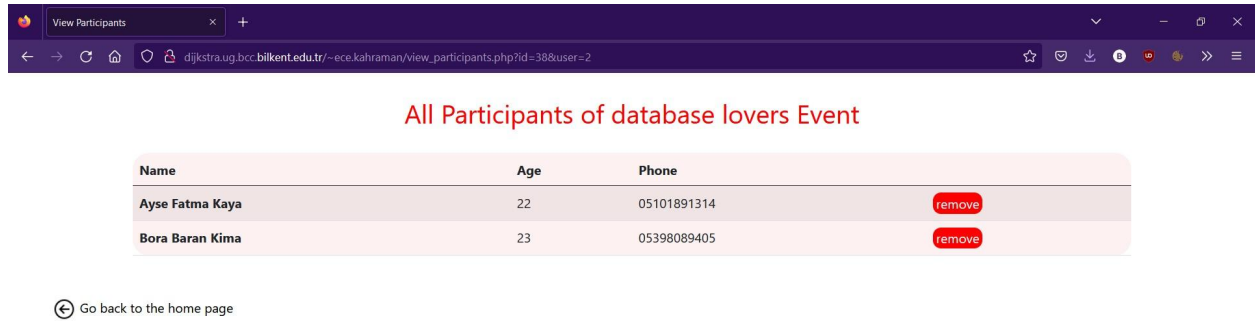


Figure 15: View of the all participants of an event

Organizers can see all participants for their events. Only name, age and phone number information are listed on the list to ensure privacy. They can remove a participant by clicking the remove button.

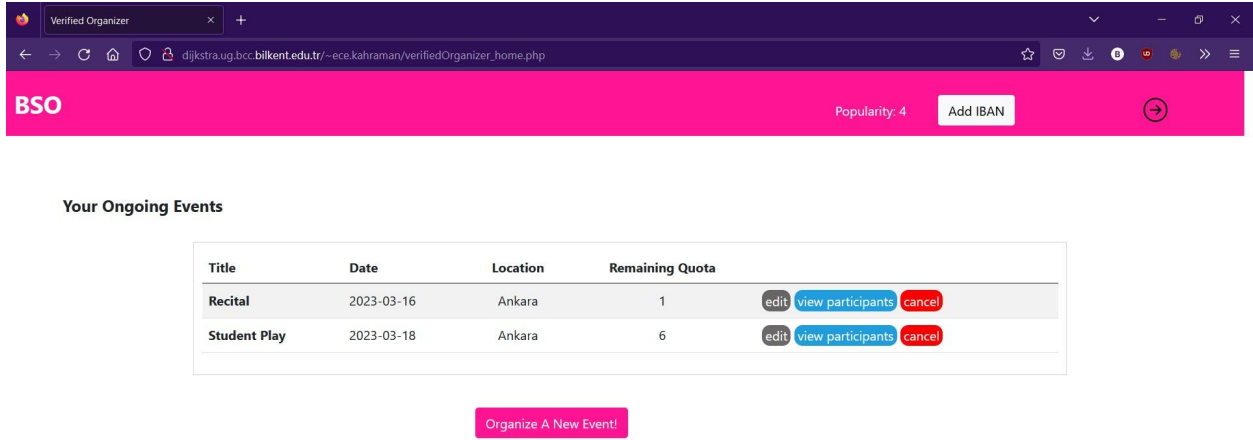


Figure 16: View of the main page of the verified organizer

Verified organizers can do all the things an organizer can do, plus, they can create paid events and sell tickets to those events. A verified organizer can enter their International Bank Account Number (IBAN) to get a payment for a paid event.

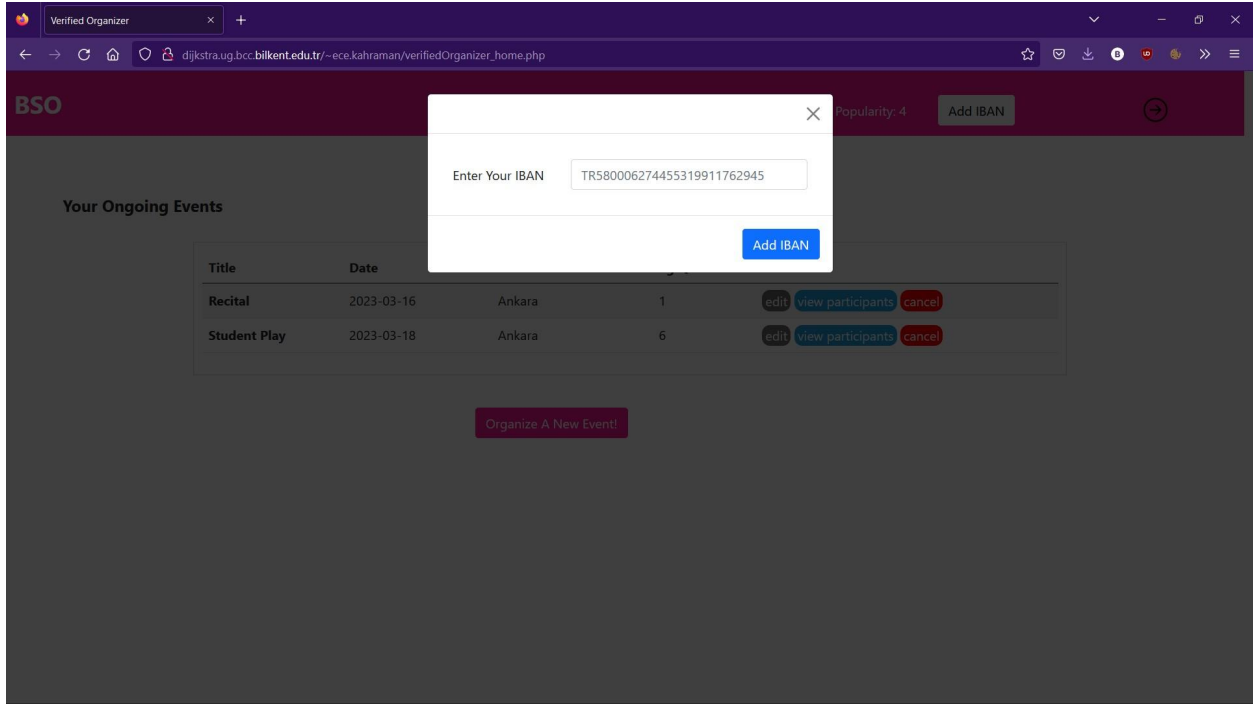


Figure 17: View of the enter IBAN modal

Verified organizers can enter their IBAN in this popup. The number is saved in the database.

Create An Event

Title:

Details:

Location:

Date:

Choose A Category:

Enter A Quota:

Choose An Age Restriction:

Maximum tickets bought by one person:

Add Ticket Category:

Enter ticket prices for categories:

Ticket Category 1:

Ticket Category 2:

[Go back to the home page](#)

Figure 18: View of the create an event screen of verified organizers

The create event page for verified organizers contains all components of the unverified create event page; plus, ticket information for the event. Verified organizers decide if the event is paid or free. If it is paid, they enter the maximum ticket amount that can be purchased by a user. They add categories for the ticket with different prices, delete all categories or just delete the last added category.

The screenshot shows a web browser window with the title 'Verified Organizer Add Event'. The address bar shows the URL 'dijkstra.ug.bcc.bilkent.edu.tr/~ece.kahraman/verified_add_event.php'. The page has a dark purple header and a light pink main content area. The title 'Create An Event' is centered at the top in pink. Below it is a form with a light pink background. The form has a left sidebar with labels: 'Title', 'Details', 'Location', 'Date', 'Choose A Category', 'Enter A Quota', and 'Choose An Age Restriction'. The main area contains input fields: a text field for 'Title', a text area for 'Details' with placeholder text 'Please explain details of the event...', a text field for 'Location', a text field for 'Date' with placeholder 'mm / dd / yyyy', a dropdown menu for 'Categories', a text field for 'Quota' with a clear button, and a dropdown menu for 'Age Restriction' with 'None' selected. At the bottom of the form are two buttons: 'Paid Event' and 'Free Event'. Below the form is a pink 'Create!' button. At the bottom left of the page is a link 'Go back to the home page'.

Verified Organizer Add Event

dijkstra.ug.bcc.bilkent.edu.tr/~ece.kahraman/verified_add_event.php

Create An Event

Title

Details

Please explain details of the event...

Location

Date

mm / dd / yyyy

Choose A Category

Categories

Enter A Quota

Quota

Choose An Age Restriction

None

Paid Event

Free Event

Create!

Go back to the home page

Figure 19: View of the create an event screen of verified organizers without tickets

When the event is free, there is no need to create tickets. Therefore ticket parameters become invisible. By filling the other fields a free event can be created.

The screenshot shows a web browser window with the address bar displaying `dijkstra.ug.bcc.bilkent.edu.tr/~ece.kahraman/verified_edit_event.php?id=7`. The page title is "Editing Event: Student Play". The form is a light pink box with the following fields:

| Field | Value |
|-----------------------------|----------------------------------|
| Title | Student Play |
| Details | A theater play as a term project |
| Location | Ankara |
| Date | mm / dd / yyyy |
| Choose A Category | Categories... |
| Enter A Quota | 6 |
| Choose An Age Restriction | Restrictions |
| Ticket Price for Category 1 | 20.00 |
| Ticket Price for Category 2 | 10.00 |

Below the form is a pink "Update" button. At the bottom left, there is a link with a left arrow icon and the text "Go back to the home page".

Figure 20: View of the edit an event screen of verified organizers

Paid and free events can be edited by their organizers. Tickets can be removed or their prices can be changed for paid events.



Figure 21: View of the all participants of an event

Verified organizers can see all participants for their events. Only name, age and phone number information are listed on the list to ensure privacy. They can remove a participant by clicking the remove button.

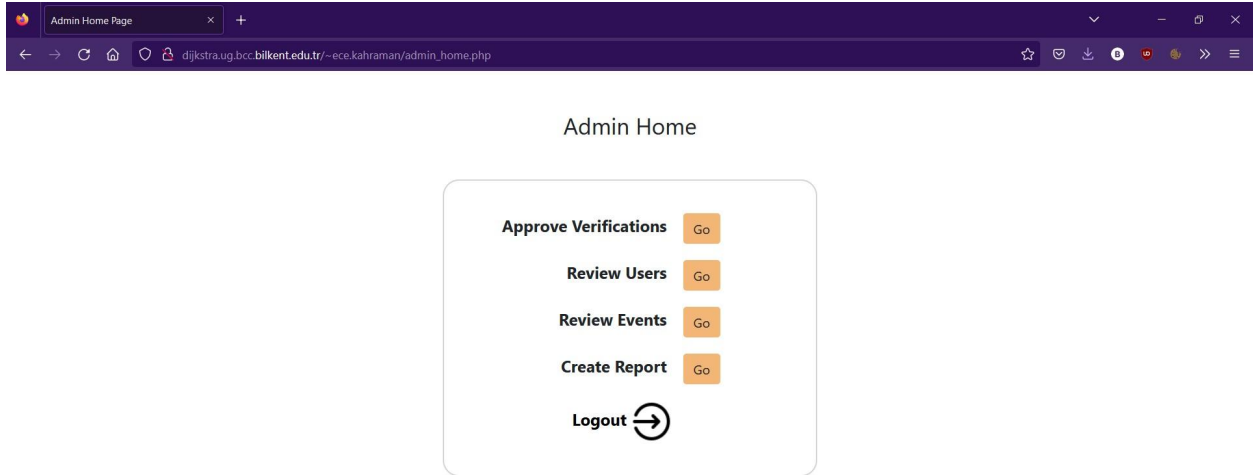


Figure 22: View of the main page of admin

The main page for admin user type contains buttons for approve verification, review users, review events, create reports pages. Also the log out button is placed at the bottom of the table.

Approve Verifications

dijkstra.ug.bcc.bilkent.edu.tr/~ece.kahraman/admin_approve_verifications.php

Verify Organizer

Enter Organizer ID

Enter Organizer Company Name

Verify

Go back to the home page

Figure 23: View of the verify organizer page

Admin users can verify an organizer by entering the id number and the name of it.

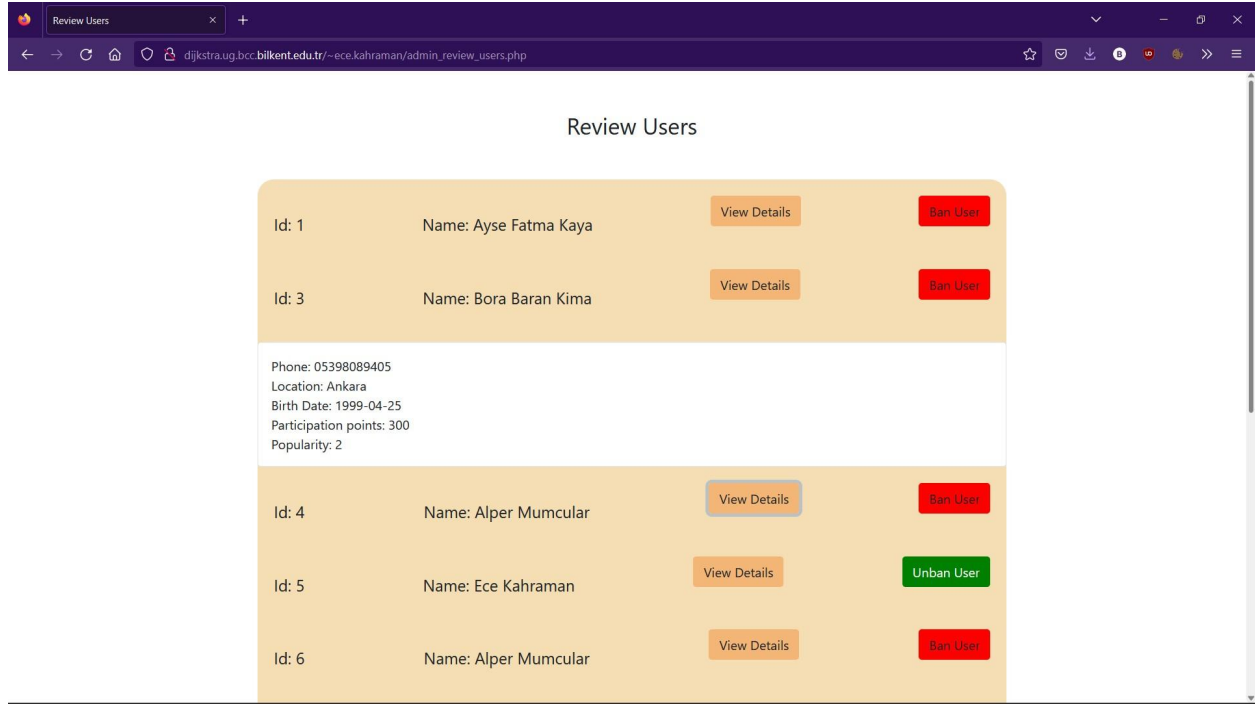


Figure 23: View of the review users page

Admins can see all users and their information. When the 'View Details' button is clicked, a detailed information card for the user is displayed on the table. Also admins can ban a user from the system. If a user is banned, the system does not allow him/her to log in the system.

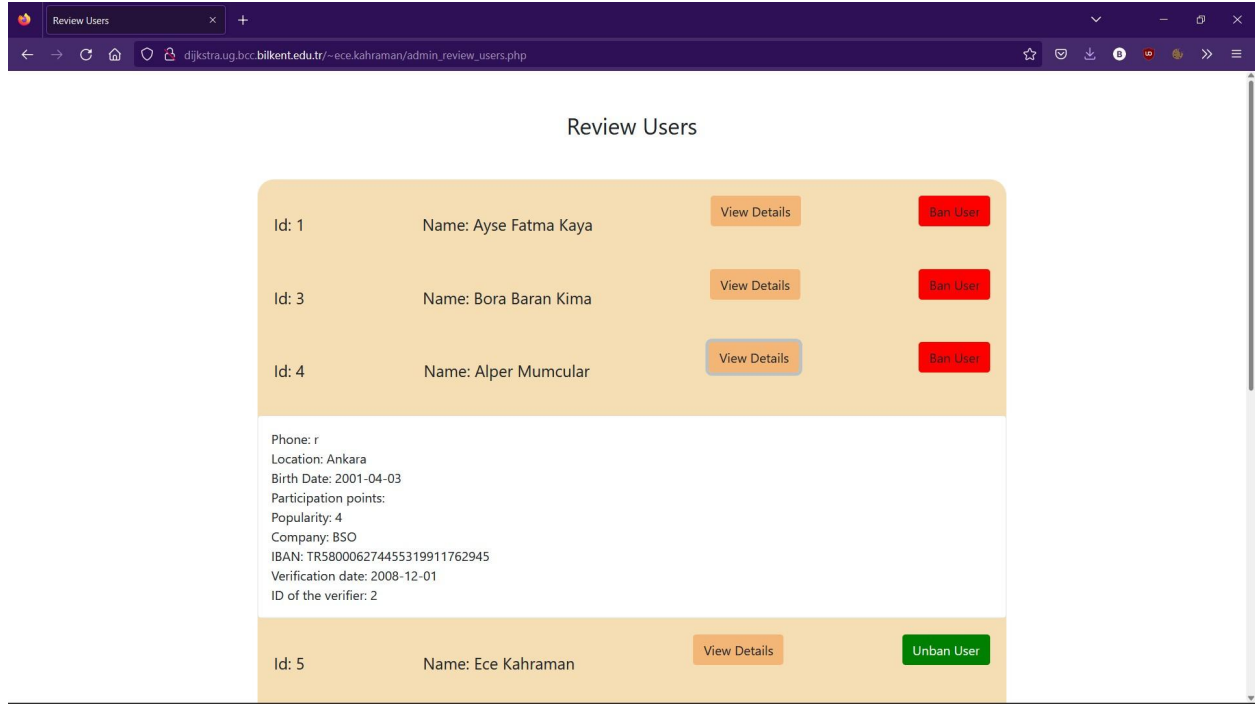


Figure 24: View of the review users page

Ban of the users can be lifted by admin users.

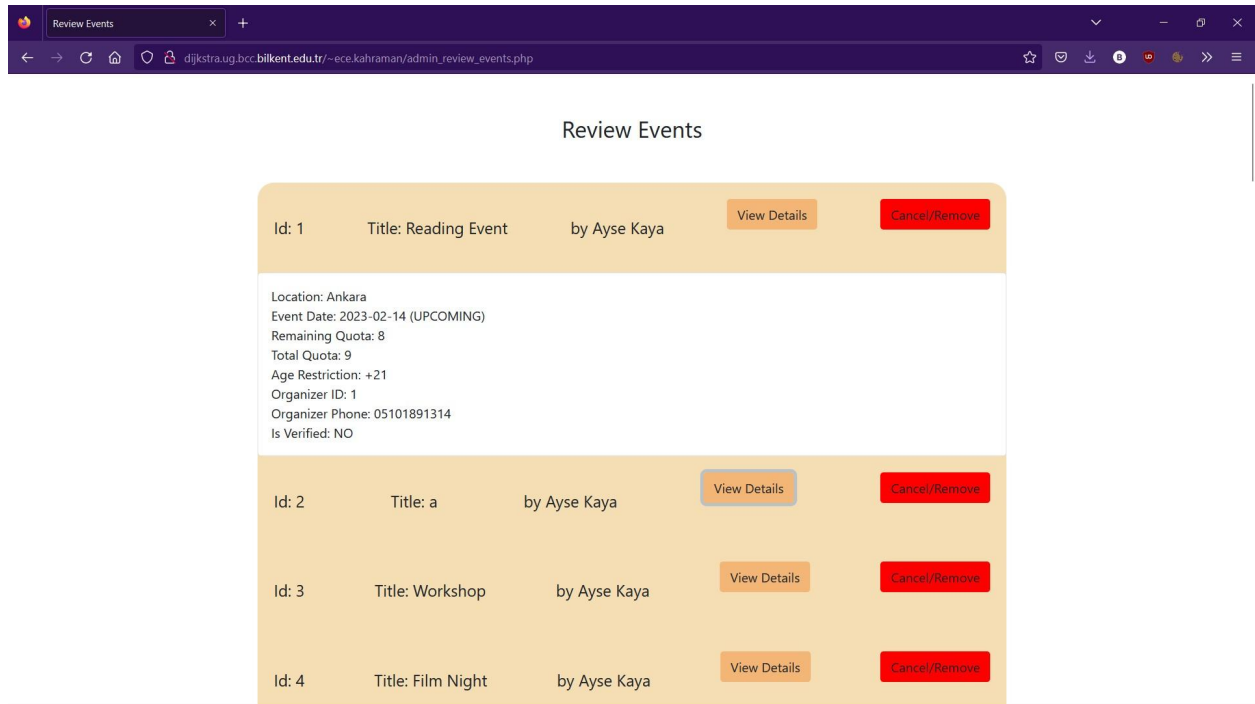


Figure 25: View of the review events page

Admins can see all events and their information. When the 'View Details' button is clicked, a detailed information card for the event is displayed on the table. Also admins can cancel an event.

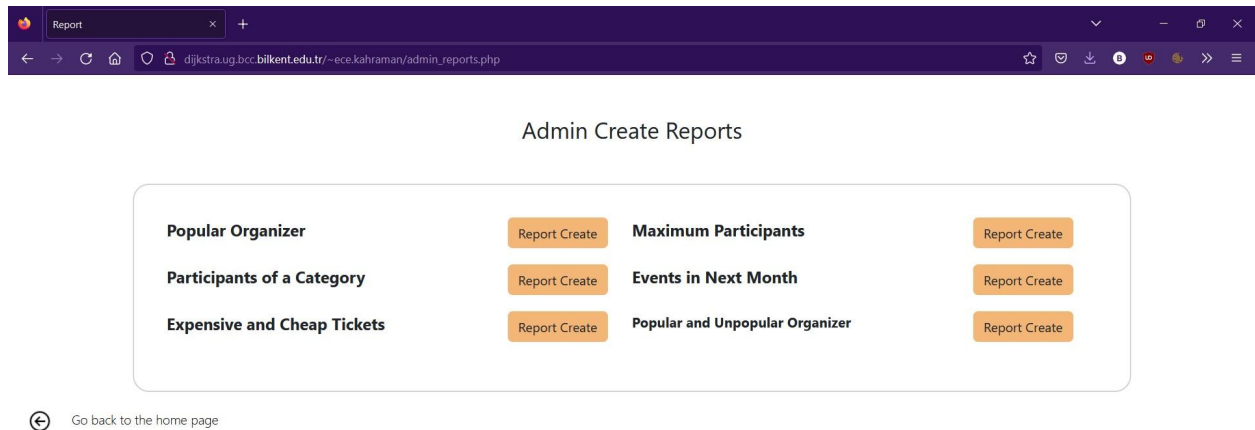


Figure 26: View of the create reports screen

Admins can create different types of reports. “Popular organizer” report lists all events in the selected location whose organizer has a popularity rate higher than the entered popularity rate. “Maximum participant” report retrieves the free event which has the maximum number of participants for each category located in the selected location. “Events in next month” report lists the participants who attended the selected category events last year. “Expensive and cheap tickets” lists the maximum and minimum ticket price for each category. “Popular and unpopular organizer” displays the most popular and unpopular organizers which created at least 5 events in at least two different cities for all cities.

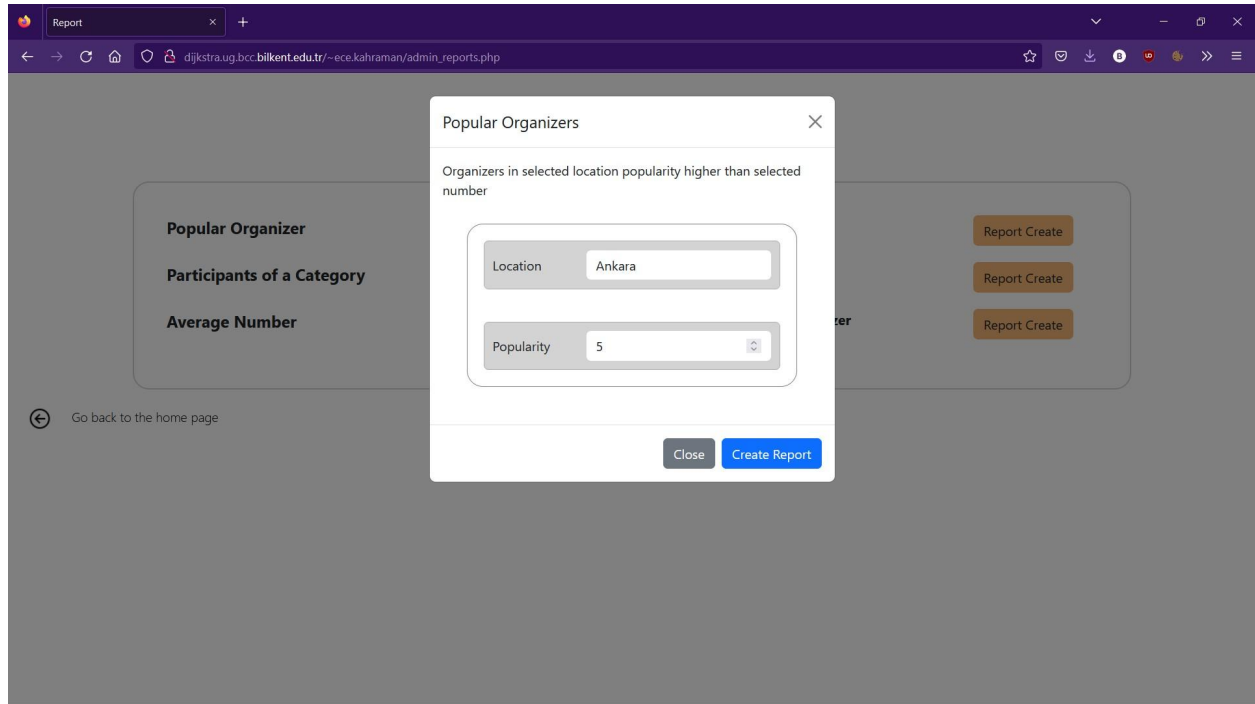


Figure 27: View of the popular organizers modal

If a report requires some values from the admin, a model shows up to get these parameters. When the “create button” is clicked, the admin is directed to the report page.

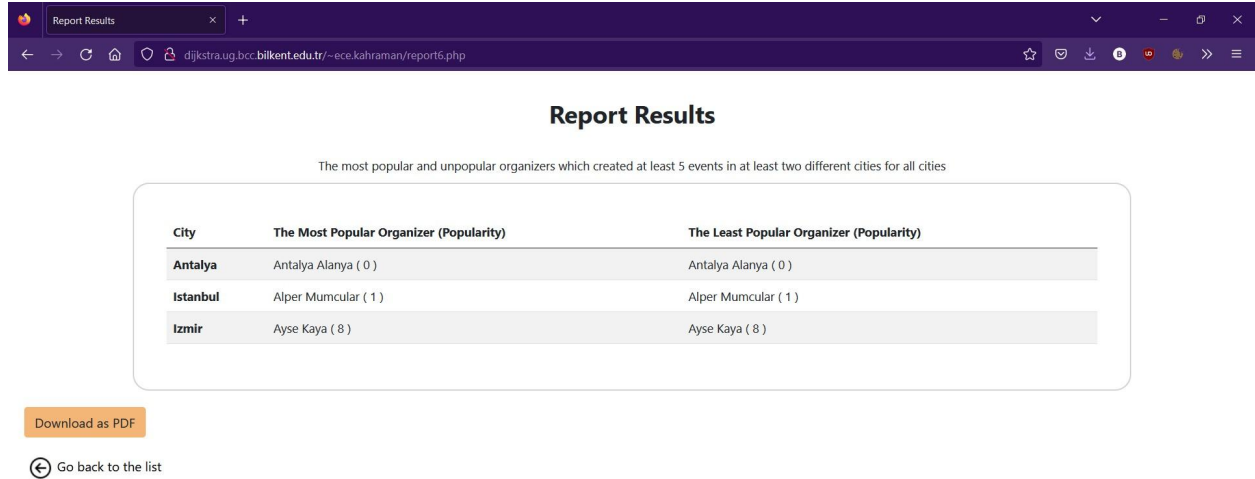


Figure 28: View of the report results

Report results are retrieved from the database and displayed on the report results page. The results can be downloaded as a pdf file.

8. Website

The website is available on

<http://dijkstra.ug.bilkent.edu.tr/~ece.kahraman/login.php>