REPUBLIC OF TURKEY

YILDIZ TECHNICAL UNIVERSITY

DEPARTMENT OF COMPUTER ENGINEERING

EDUCATION ATTENDANCE SYSTEM

16011055 — Orkun AVCI

12011071 — Erhan AYAKKABICIOĞLU

COMPUTER PROJECT

Advisor

Assist. Prof. Dr. Ziya Cihan TAYŞİ

May, 2021

ACKNOWLEDGEMENTS

We would like to thank our advisor Assist. Prof. Dr. Ziya Cihan TAYŞİ for their indisputable and unquestionable help throughout the project.

Orkun AVCI
Erhan AYAKKABICIOĞLU

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

DDos	Denial of Service
ER	Entity-Relationship
KB	KiloBytes
KVKK	Kişisel Verilerin Korunması Kanunu
PCA	Personal Component Analysis
QR	Quick Response Code
RFID	Radio Frequency Identification
TL	Turkish Lira
URL	Uniform Resource Locator

LIST OF FIGURES

ABSTRACT

Education Attendance System

Orkun AVCI
Erhan AYAKKABICIOĞLU

Department of Computer Engineering
Computer Project

Advisor: Assist. Prof. Dr. Ziya Cihan TAYŞİ

Today, with the development of technology, the number of companies conducting online studies, trainings and events has gradually increased. Especially with the restrictions of pandemic, online events have gained even more importance. Online events are also more economical for companies in terms of time and money. This is the primary driving factor and the reason why the number of online events are expected to increase even more in the future. And because of this, the necessary tracking of attendees has become an even more important topic.

This work aims to provide a customizable and easy to use system for event organizers to help them plan events and track attendees. In the system, the organizers will be creating events and distributing the registration form of these events to their audience. Then, attendees would fill these forms to register to the event. When the registration completes, the system will give each attendee their unique code which they will use to enter the event. It is thanks to this code that we will be keeping track of them.

Event organizers will also be able to see the list of registered users, see the attendance status of each attendee, and create both blacklists and whitelists for registration forms.

Keywords: Event Scheduling, Attendance Validation, Attendance Tracking, Attendance Reporting, Attendance System

ÖZET

Eğitim Takip Sistemi

Orkun AVCI
Erhan AYAKKABICIOĞLU

Bilgisayar Mühendisliği Bölümü
Bilgisayar Projesi

Danışman: Dr. Öğr. Üyesi Ziya Cihan TAYŞİ

Günümüzde teknolojinin gelişmesiyle birlikte şirket çalışmalarının, eğitimlerin, etkinliklerin online olarak yapılma oranı da giderek artmıştır. Özellikle pandemi dönemiyle birlikte gelen zorunluluklar nedeniyle bu etkinliklerin online olarak yapılması büyük önem kazanmıştır. Ayrıca online yapılan etkinlikler zaman ve maliyet tasarrufu gibi konularda da oldukça sağlamaktadır. Bu nedenlerle sonraki dönemlerde de online etkinliklerin tercih edilme oranının daha da artacağı tahmin edilmektedir. Bu durumla birlikte online yapılacak etkinliklere katılım durumlarının takip edilmesinin gerekliliğini ortaya çıkmıştır.

Bu çalışma etkinlik düzenlemek isteyen kullanıcıların etkinlikleri planlamasının ve katılımcıların katılma durumlarını takip edebilmesini kolaylaştırmak ve düzenlemek adına yapılmıştır. Projede etkinlik düzenleyiciler, sistem üzerinde etkinlik oluşturduktan sonra katılımcılara etkinliğe kayıt olabilmeleri için kişisel bilgilerini girmelerinin gerektiği form linki vereceklerdir. Katılımcılar form üzerinde gerekli bilgileri doldurduktan sonra etkinliğe kayıt yaptırmış olacaklardır. Kayıt işlemi bittiğinde sistem, katılımcılara bir kod verecektir. Katılımcılar etkinlik zamanı geldiğinde etkinliğe giriş yapabilmek için bu kodu sisteme gireceklerdir. Bu sayede katılımcının etkinliğe katılım durumu takip edilebilecektir.

Sistem üzerinde etkinlik oluşturanların, katılımcıların devam durumlarını görebilmesi, etkinliğe kayıt yaptıranları sınırlandırabilmek için karaliste-beyazliste oluşturabilmesi gibi özellikler de eklenmiştir.

Anahtar Kelimeler: Etkinlik planlaması, Katılım kontrolü, Katılım takibi, Katılım raporlanması

1 Introduction

Since the start of the pandemic, many companies have been forced to shut down their offices and migrate over to working from home. This sudden change have led to an increased demand for remote work environments. And thanks to the increase in demand, online events like meetings, gatherings, classes, and similar cooperative activities have seen exponential growth in both number and magnitude over the course of the pandemic.

Study in [1] shows that, while the availability of online events positively impact the performance of attendees, it also negatively impacts the attendance.

In both face-to-face events and online events, if the attendance system requires an action from the instructor, then the event time will be disrupted each time the instructor allows a late attendee into the meeting[2]. The solution to this problem lies with the automation of the system.

There have been various research and proposed systems on the subject, but unfortunately most of these approaches could not provide an accurate and reliable way to validate attendance and keep track of it. We will examine two of these approaches in Chapter 2.

We would like to propose a new way of validating and keeping track of attendance using a code. This code will be generated with unique credentials of the attendee and the unique credentials of the event put through a hash function to create a one-time-use code.

Because the unique credentials in the system is used, each code will also be unique. This approach eliminates the problem of distinguishing between attendants per event and distinguishing between events per attendant.

There have been research on how we could increase the attendance. One such research is [3], and we have taken two of the bullet points from this paper.

1. **The system should have sound and reasonable attendance policies with consequences for absence,**
2. **Good attendance should be valued and rewarded.**

We have incorporated the first principle into the tracking system and the second principle into the filtering system.

2
Preliminary Examination

2.1 Related Work

Automation and digitization in attendance tracking systems can be achieved using a variety of technologies. For example, there have been researches utilizing smartphones and Bluetooth to track attendance. In this section we have examined the closest two approaches to ours: Face Detection and QR Code.

2.1.1 Face Detection Based Work

In [4], authors proposed a continuous observation based system over a single cycle face detection system as the face detection rate was not high enough in a single cycle. The results of the work revealed that in a single cycle of the face detection algorithm, the face detection rate was at 37.5%. On the other hand, in 79 minutes of continuous video feed the face detection algorithm completed total of 8 face detection cycles, and the face detection rate was improved to 80.00%.

While the face detection rate can be improved further with more advanced algorithms we have available today, authors concluded that face detection alone was not enough for estimating attendance and the system needed to be improved further with interaction among the system, students and the teacher.

Similarly in their work on [5], authors used a continuous video feed to detect the faces of students. Compared to the previous work, the results were improved to 95% for face detection and 85% for face recognition. Unfortunately, the algorithm struggled with diversification. When tested on people with beards, face detection dropped to 75% and face recognition dropped to 64%. Furthermore, when tested on people with veils, face detection went down to 40% and face detection rate plummeted to 2%. Authors recognized the need for improvement in the system and the need for algorithms that can recognize the faces in a veil.

To improve face detection rate, [6] paper proposed the PCA algorithm. PCA improved the face detection rate to 98.7% and face recognition was improved to 95%. But the problems started to arise when the face orientation of the subjects started to change. At 18° detection rate went down to 80% and recognition rate down to 78%. At 54° detection rate dropped to 59.2% and recognition rate dropped to 58%. Past 72°, both detection and recognition was 0% across.

The conditional reliability of face detection and face recognition algorithms leaves much to be desired.

2.1.2 QR Code Based Work

QR is a two-dimensional small image that contains data in the form of text. The stored data can be numerical, alphanumerical, and even binary code. It is most common in Japan and it is usually used to store URLs.

In [7], author proposes a QR code-based attendance system. In the system, the instructors will log in to their respective accounts and create QR codes. And then, the attendees have to log in to their respective accounts and scan the QR code. If the scan is successful then the attendee record of the attendants will be saved to the database.

While the contents of the QR code are not explicitly revealed in the paper, this is a step in the positive direction for authentication purposes.

The proposed system in [2] however, does reveal the contents of the QR code and how it is used for authentication. The QR codes in this system would be shown in the top left corner of the slides and the attendees would need to scan them. The created QR codes in the system contained:

- Course and section ID,
- Date and beginning time of the event,
- Instructor name,
- Some random passcode.

While this set of information was enough to distinguish between events, it wasn't enough to distinguish between attendees who scanned the QR code. Because the system didn't incorporate an accounting system for attendees. To overcome this problem, the authors have chosen to take a photo of the attendee and run a facial recognition algorithm back on the server.

As established in the previous section, face recognition algorithms are not always reliable and in need of improvement. Which in return, leaves this system in need of improvement, too.

2.2 Similar Systems

One similar system in the field is Clockify[8]. Clockify tracks time by using a timer, logs it in a timesheet, and categorizes it by projects. It is aimed towards teams and integrates itself into various workspaces.

To use the system, users have to start the timer when they start working on a project and stop it when they leave the workspace. The collected data is broken down into customizable reports and visual timesheets. These timesheets can be exported as PDF, CSV, or Excel.

Among the features of Clockify, we have taken inspiration from logging and reporting.

A more similar system in the field is Event Attendance Pro[9]. Event Attendance Pro uses RFID badges, Key Fobs, and Barcode badges for identity validation. Unlike Clockify, when chosen as an option at the time of event creation, this system captures the check-in and check-out times of attendants. Based on that data the system identifies and categorizes those who attended and those who did not attend. As soon as the attendance is captured, a receipt can be printed to be given to the attendee. This receipt can be used as evidence.

After the event, the system generates reports for event creator out of the extracted attendance data. It is also possible to segment the attendees by departments and tags.

From the features of Event Attendance Pro, we have taken inspiration from identity validation and smart tracking.

3 Feasibility

3.1 Technical Feasibility

In this section, how the sources selected to be used to make the project , workforce and time planning, comparison of economic incomes and expenses and legal responsibilities has been explained.

3.1.1 Software Feasibility

Which development programs and tools chosen and why those ones preferred has been explained in this section.

- **Backend Software Development Language :**
  We need a development language which easy to run different operating systems and environments, useful for database and frontend communication management. Connecting to database with JDBC and managing with hibernate is useful and we can use spring boot with Java for frontend communication. Therefore we preferred **Java 8** for backend development. Also we have more experience with Java.
- **Database Management System :**
  Database management system which we need is compatible with Java, open source, free to use and easy to manage. Therefore we decided to use **MYSQL 8.0**.
- **Frontend Software Development Languages :**
  To make our application publicly and readily available, we had decided to build the frontend as a website. For structure and look of the website, **HTML** and **CSS** was a must have. We needed to serve in many different browser environments so we needed a scripting language that is supported in almost all of them. We have chosen **Javascript** as our scripting language. Finally we decided to use a framework to streamline the development but since both developers had little

experience with frontend development we needed a framework that is widely used, has easily accessible learning material, and supports system's other needs. We have decided on **React** for our frontend framework.

- **Software Development Platform :**
  We should use Java, Spring and Hibernate together. We searched a development platform which we can use all together. We find out we can do it with **Intellij IDEA** and we preferred it.

- **Operating System :**
  Operating system to be used, should be able to work with programs given above and should be widely used for run the system different environments. For these reasons we preferred **Microsoft Windows 10**.

- **Other Development Tools :** Hibernate for object-relational mapping, Maven for dependencies, Spring Boot for rest APIs, and Node.js for React.

3.1.2 Hardware Feasibility

According to chosen versions of development programs above, minimum requirements of the system has given below.

- **Minimum Requirements:**

- CPU Clock Speed : 2 Ghz and 2 Core
  Recommended for MYSQL

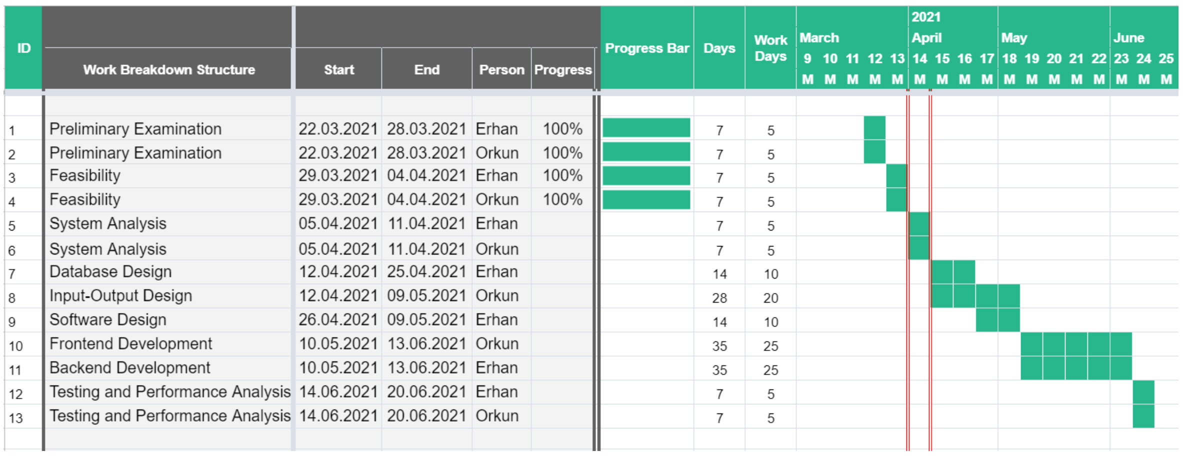
- System Type : 64-bit operating system
  Recommended for all programs

- RAM : 8 GB
  MYSQL needs 8 GB RAM and recommended for others

- Disk Capacity : 32 GB HDD
  500 MB for MYSQL, 128 MB for JDK 8, 500 MB program files and remaining disk capacity for datas will be saved.

3.2 Workforce and Time Planning

The Gantt diagram given below, how much workforce cost for the project, each developers responsibilities and which steps took how much time has shown.



ID	Work Breakdown Structure	Start	End	Person	Progress	Progress Bar	Days	Work Days	March					2021 April				May					June		
									9 M	10 M	11 M	12 M	13 M	14 M	15 M	16 M	17 M	18 M	19 M	20 M	21 M	22 M	23 M	24 M	25 M
1	Preliminary Examination	22.03.2021	28.03.2021	Erhan	100%	🟩	7	5				🟩													
2	Preliminary Examination	22.03.2021	28.03.2021	Orkun	100%	🟩	7	5				🟩													
3	Feasibility	29.03.2021	04.04.2021	Erhan	100%	🟩	7	5					🟩												
4	Feasibility	29.03.2021	04.04.2021	Orkun	100%	🟩	7	5					🟩												
5	System Analysis	05.04.2021	11.04.2021	Erhan			7	5						🟩											
6	System Analysis	05.04.2021	11.04.2021	Orkun			7	5						🟩											
7	Database Design	12.04.2021	25.04.2021	Erhan			14	10							🟩	🟩									
8	Input-Output Design	12.04.2021	09.05.2021	Orkun			28	20							🟩	🟩	🟩	🟩							
9	Software Design	26.04.2021	09.05.2021	Erhan			14	10									🟩	🟩							
10	Frontend Development	10.05.2021	13.06.2021	Orkun			35	25											🟩	🟩	🟩	🟩	🟩		
11	Backend Development	10.05.2021	13.06.2021	Erhan			35	25											🟩	🟩	🟩	🟩	🟩		
12	Testing and Performance Analysis	14.06.2021	20.06.2021	Erhan			7	5																🟩	
13	Testing and Performance Analysis	14.06.2021	20.06.2021	Orkun			7	5																🟩	

Figure 3.1 Gantt Diagram

3.3 Legal Feasibility

In the project we are saving and processing users personal data. According to [10], saving and processing personal data without users permission is a crime. Therefore users should be informed how we process and save their information and they should sign KVKK contract and allow us to save their data to be used in our system.

While signing up, we should have user contract for users who are creating events and tracking attendance. This contract contains that we save their data but never share any other third party person or application. When users decided to delete their account, their data will be deleted. Also contract should contains that this system is using only for validating and tracking attendance of online events. Users can not use the system any other purposes. We have no responsibilities for other usages. Users should accept this terms of use for using the system.

There should be also another contract for the guests who are registering for events. This contract contains that the system save their data but never share any other third party person or application. If guests want to register for event they should let the system save their data for tracking their attendance.

3.4 Economic Feasibility

This project lasted fifteen weeks and two developers worked in the project. Each of developer costed 100 TL per hour. Each of them worked four hours every day and five days in a week. Two developers daily cost was 800 TL. After fifteen weeks total developer cost was 60000 TL.

Each of developer needs a computer which fulfills the requirements described in the hardware feasibility part. A computer with these features costs 5000 TL. Two developers need two computer so total computer cost is 10000 TL.

Domain and hosting services are required to run this system on the web. Domain service cost 100 TL per month. Hosting service cost 150 TL per month. Total domain and hosting services cost 250 TL per month. After a year domain and hosting service will be 3000 TL.

According to described expenses above, projects expected yearly cost will be 73000 TL.

4
System Analysis

This analysis will be object-oriented and we will use use-case scenarios to summarize the expected usage.

4.1 Minimum Requirements and Guests

At the most basic level, we need to define an event. Events need to have a form associated with them, which people can fill to register for the event. The form should be able to have multiple of different types of fields depending on the event. This should be a customization option at the event creation time.

Events should keep track of their creation time, whom they were created by, what is the event time and what the event redirection link is. Creation time and creator information are needed for validation purposes. Event time and redirection links are needed for functionality purposes. Before the event time, we want to show the form page for registration, at event time we want to show the redirection page and after the event, we want to deactivate this event and show nothing.

We need to have an actor who can create these events and forms. We have decided to call this actor Organizer. An Organizer can create an event in the system, customize the form for the event, and enter the other credentials for the event.

We want these events and forms to be publicly available so we need an actor of Guest. Guests must be able to fill the forms in the system. Since we also want to keep track of attendance, we must get a unique data from the guests. We have decided to use the email address to log guest actions and keep track of guests.

With these minimum requirements, let's examine the guest use-case diagram in 4.1. Detailed steps of the diagram are given below the figure.

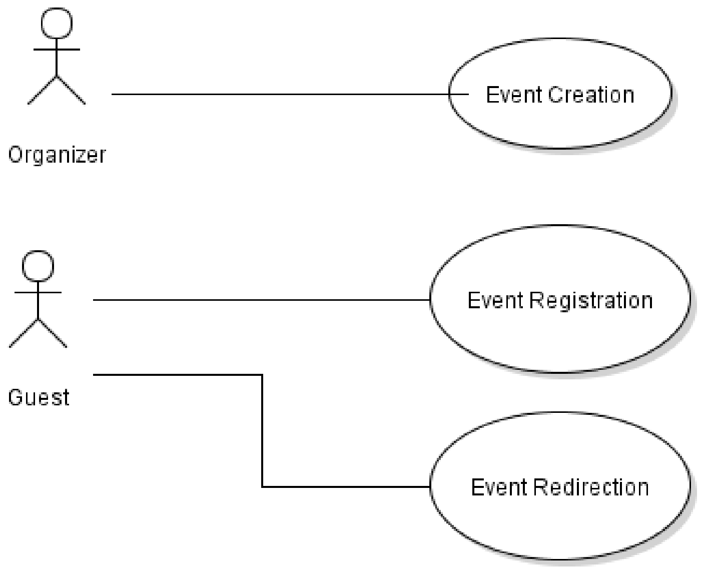


Figure 4.1 Guest Use-case

Organizer:

1. Organizer creates an event in the system along with its registration form.

Guest:

1. Guest registers to the event by filling the registration form of the event.
2. Guest gets their code after successful registration.
3. Guest uses redirection page at event time and enters their code.
4. Code is accepted by the system and guest is redirected to the event link.

4.2 Higher Level of Authentication with Users

The email information we have taken from guests alone is not enough to accurately and efficiently keep track of the attendees. We need a better way to identify people, so we need an actor called a user. This user will be able to log into the system, search for events within the system, register to them easily because we can pull some of the data from within the system, and see the record of previous attendances. This also helps us associate attendees with identities in the system.

As the baseline, we expect users to register to events in the system and keep track of their attendance. On a level above that, we expect some users in the system to be able to create these events. We have defined the Organizer actor previously but we didn't associate an account for the actor in the system, because we have integrated Organizer authorization to user accounts. Organizer is not a separate entity but an advanced user in the system. But the initial authorization of all new users is without Organizer rights. We don't want every new user to be able to create an infinite amount of events in the system. This helps with the security and robustness of the system.

Next, we need an actor who can give Organizer rights to users. We have called the actor System Admin. This level of authorization bypasses user-level restrictions and regulates the system. While Organizers are advanced users in the system, System Admins are not. They are separate entities from users.

Now that we have defined the minimum requirements for users to exist within the system, let's examine use-case diagram for users in 4.2. Detailed steps of the diagram are given below the figure.

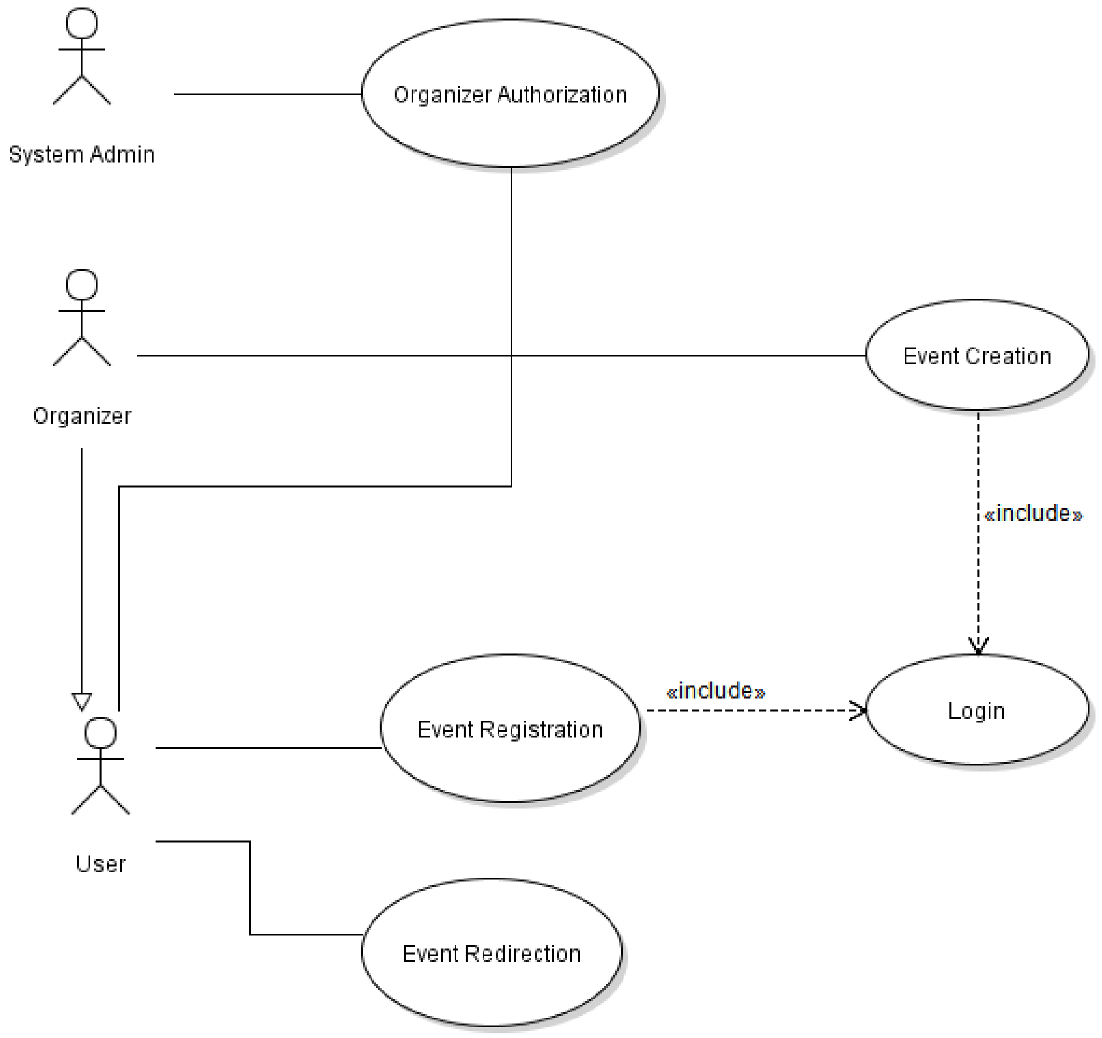


Figure 4.2 User Use-case

System Admin:

1. System Admin gives Organizer rights to User.

Organizer:

1. Organizer logs into the system through their User account.
2. Organizer creates an event in the system along with its registration form.

User:

1. User logs into the system.
2. User registers to the event by filling the registration form of the event.
3. User receives their code after successful registration.
4. User uses redirection page at event time and enters their code.
5. Code is accepted by the system and the user is redirected to the event link.

4.3 Other Subsystems

For privacy and safety reasons, Organizers might not want all events to be publicly available. We want to be able to house these private events in our system, too. To accomplish this, we have put an authorization level requirement at event creation time. By default, all events are created to be accessible by all users in the system. If the Organizer goes one level below, the event will be publicly available to everyone on the net, namely guests.

We have taken authorization one step further, however. We have defined a company field for users. At creation time this field is empty and the system doesn't expect all users to fill this field. To accompany this field we have defined a company table in our database. The company doesn't function as an actor but helps with the categorization and grouping of users. And thanks to this company field, we have added another level of authorization to the events. Now Organizer can go one level above the default authorization level and make events only accessible to users within the same company. This creates further private groups in the system and protects the privacy of such events.

The company keeps track of its creator, its owners, its workers by user IDs, and company-wide events by event IDs. The creator field for the company is stored at creation time and it is the ID of the user who created it. Users have a company field and the company has a user IDs field, so the relationship between them is a two-way relationship. However, events only keep track of the users who created them, not the company, so that's a one-way relationship. But event creator is identified by user ID so it can be used to trace back to the user who created it, and then the company itself.

Similarly, the creator is assigned as the first owner of the company at creation time. We have limited the maximum number of owners of a company to five. To ease up the load of the System Admins and to give company owners better control over their workers, we have enabled these company owners to give Organizer rights to their workers. The preliminary requirements before Organizer authorization can happen are, the user needs to be working in the company, the user doesn't have Organizer rights, one of the company owners is enacting this action, and the company owner have Organizer rights themselves.

Users without a company still can only gain Organizer rights by the approval of a System Admin.

In the end filtering system is finalized with three levels of authorization: guest level, user level, and company level.

The final use-case diagram for the system can be examined in 4.3. Detailed steps of a full cycle of successful and expected use of the system is given below the figure.

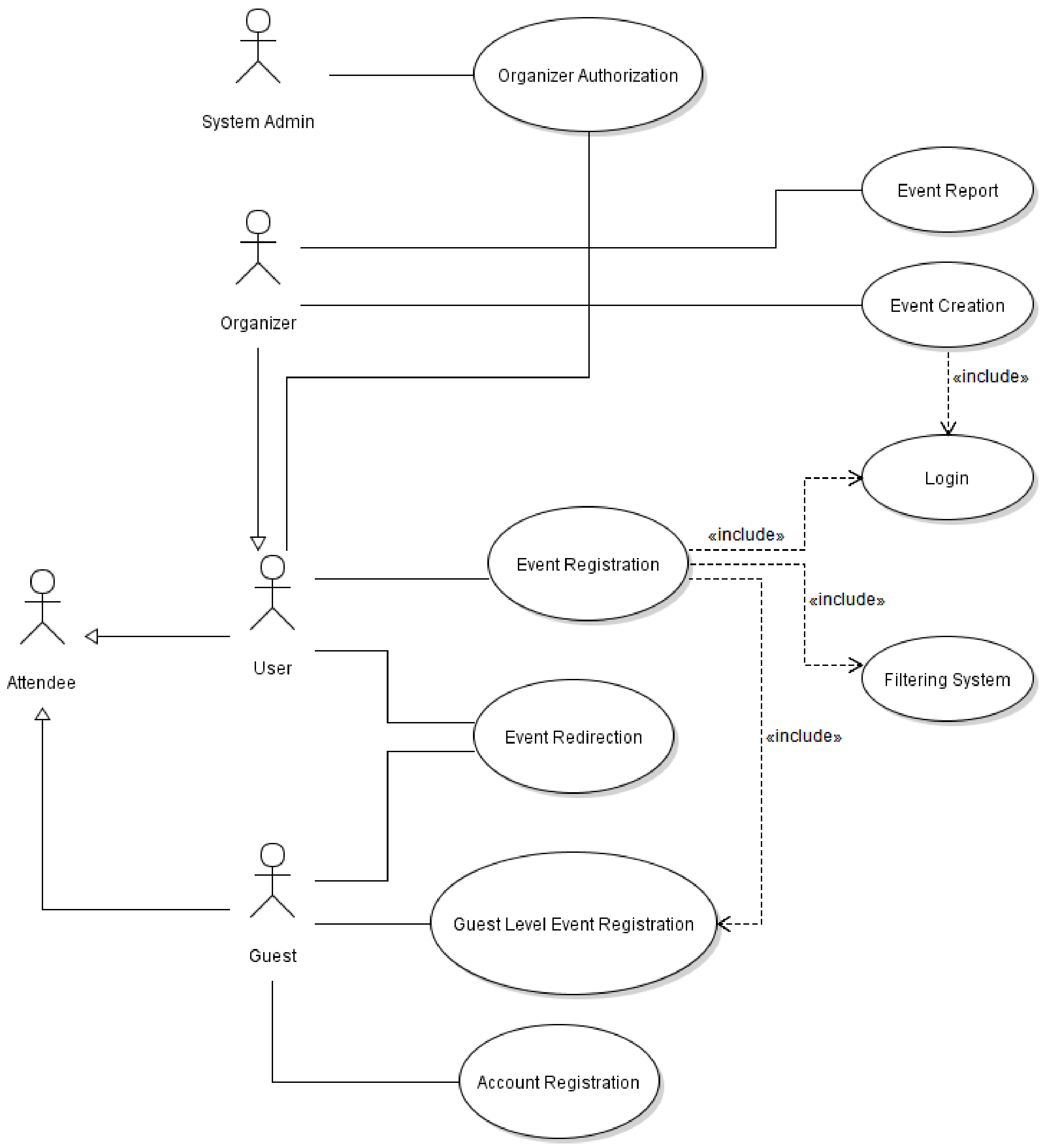


Figure 4.3 System Use-case

Guest:

1. Guest enters the system and moves to sign up page.
2. Guests fills the account registration form.
3. After successful registration guest is given permission of log into their account.

System Admin:

1. System Admin gives Organizer rights to User.

Organizer:

1. Organizer logs into the system through their User account.
2. Organizer creates an event in the system along with its registration form.
3. Organizer sets the required authorization level for the event at creation time.

User:

1. User logs into the system.
2. The filtering system only shows events up to the same authorization level as the user.
3. User registers to an event by filling the registration form of the event.
4. User receives their code after successful registration.
5. User uses redirection page at event time and enters their code.
6. Code is accepted by the system and the user is redirected to event link.

5
System Design

5.1 Software Design

In this section, user roles and processes shown given figure 5.1 and explained below the figure.

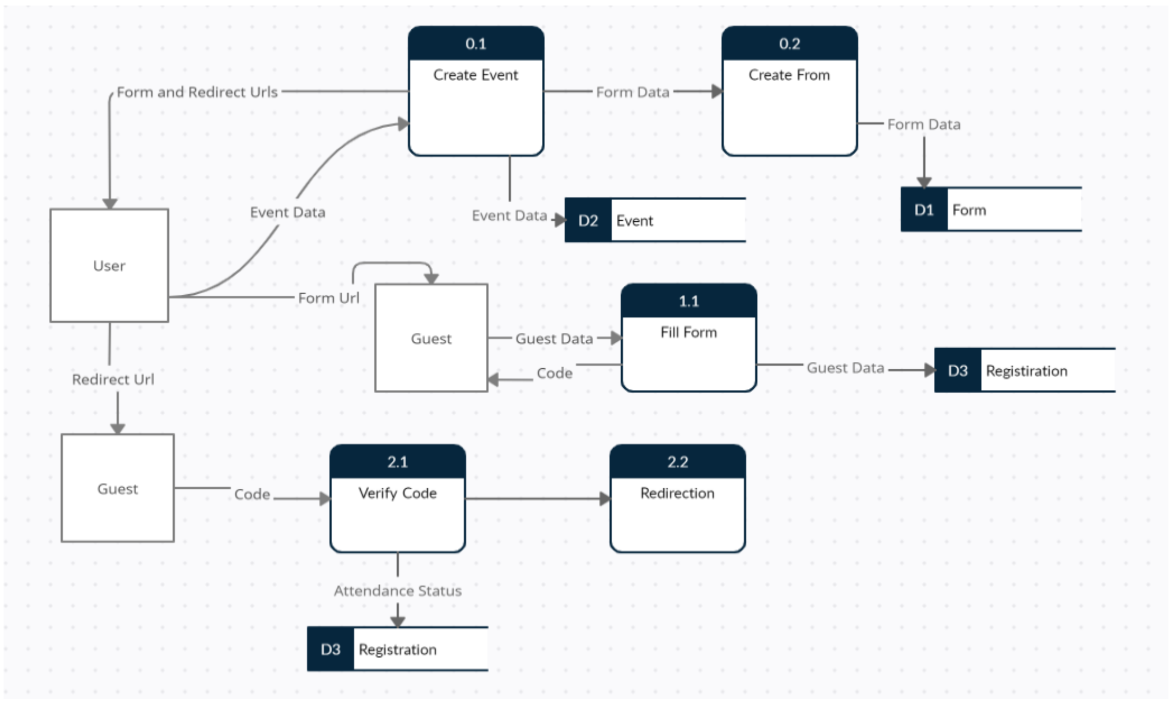


Figure 5.1 Data Flow Diagram

As seen on given figure 5.1, user can create an event in the system. While user is creating event, can choose the form fields for taking wanted personal information from event participants. After this process system gives back form URL and redirect URL to user. User shares form URL with guest for registering events.

Guest reaches form page with using given form URL. When guest fills the form page system gives guest a code for attending the event.

When the event time comes, guest can reach redirection page with using given redirect

URL. On this page guest enters given code for verifying and tracking attendance. If entered code is true, system redirects the guest to event environment.

5.2 Database Design

In database design, entities which their data should be kept and their relationships with other entities has shown in ER diagram given below.

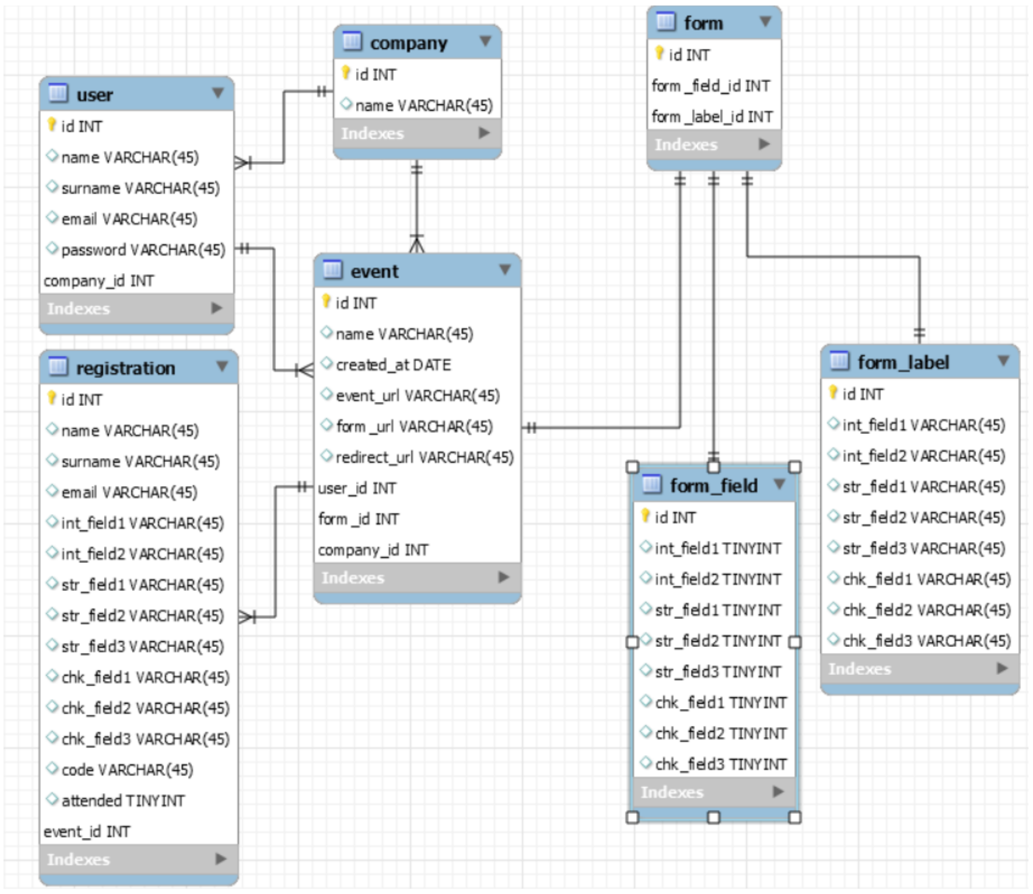


Figure 5.2 ER Diagram

5.3 Input-Output Design

Since React is a component based single page framework, all the interactive and static elements are shown dynamically cycled through on our empty canvas in Figure 5.3. All the components we will go over are shown in the middle of the screen, both vertically and horizontally.

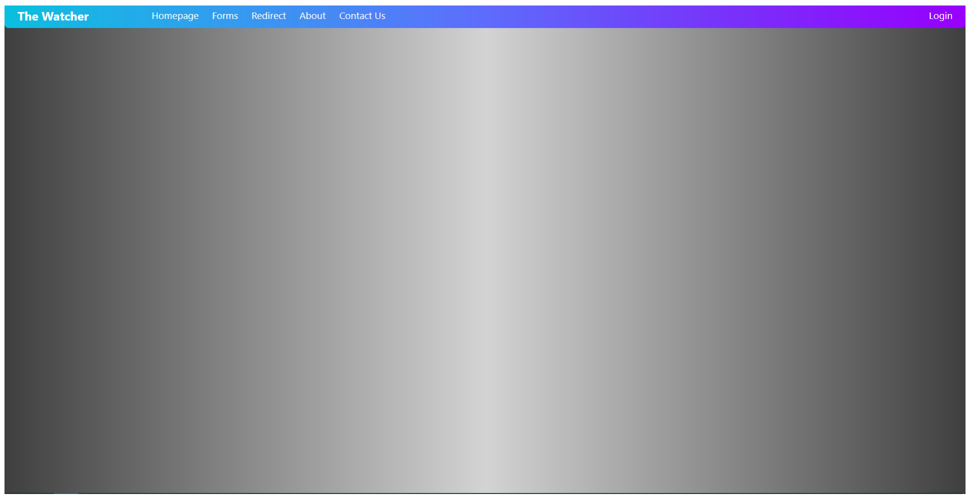


Figure 5.3 Empty Canvas

Two of the most frequently interacted components would be login and redirection. Login takes email address and password of the user as input and redirection takes email address and hash code as the input. Output of the login page is an error message above the button on errors and a redirection on success. The output of the redirection page is similarly an error message above the button on error and a redirection to event link on success.

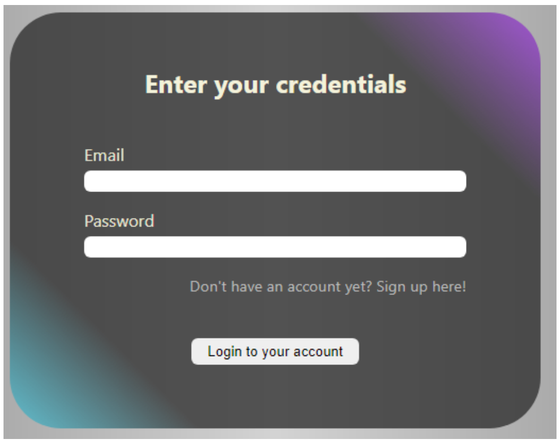


Figure 5.4 Login Component

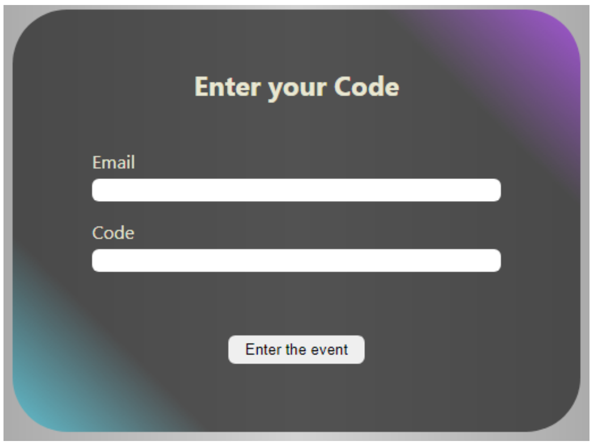


Figure 5.5 Redirection Component

Another commonly interacted component would be the form page. Since it is event dependant, we have placed the bare minimum size of this interactive component in Figure 5.6. A form with maximum number of fields is showcased in Chapter 6. The output of this component is an invisible div that we have enabled for showcase purposes. On successful event registration text will be the hash code and on errors it will be an error message, both colored red.

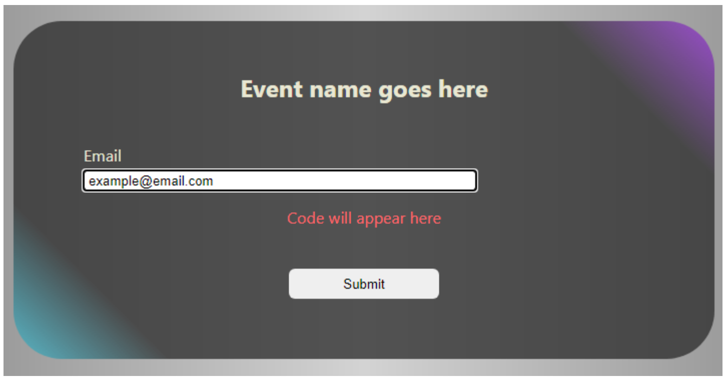


Figure 5.6 Minimum Sized Form

For interactive components we used top-right and bottom-left coloring like in Figure 5.7.

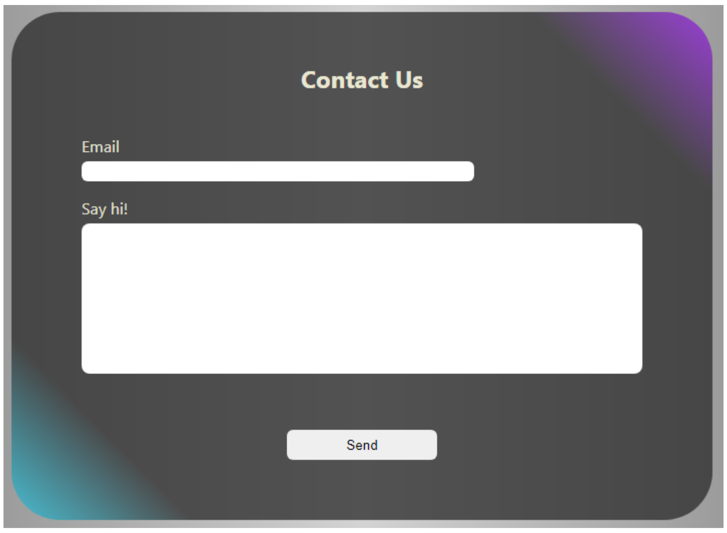
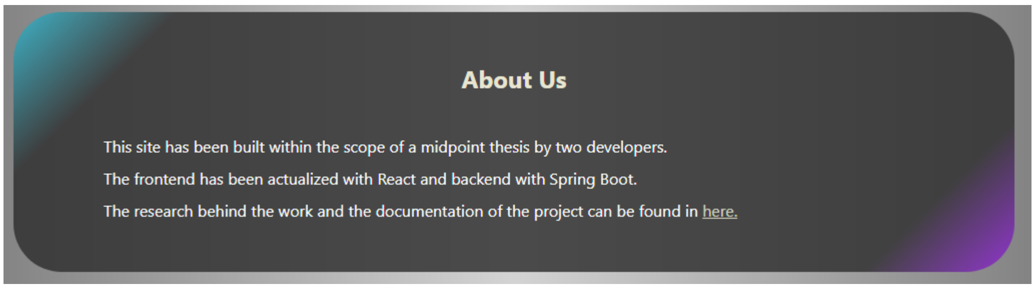


Figure 5.7 Contact Us Component

And for non-interactive static components we used top-left and bottom-right coloring to visually separate them, like in Figure 5.8.



About Us

This site has been built within the scope of a midpoint thesis by two developers.
The frontend has been actualized with React and backend with Spring Boot.
The research behind the work and the documentation of the project can be found in here.

Figure 5.8 About Us Component

6 Application

In this chapter, we will follow some of the steps of Chapter 4.3 and provide screenshots from the running state of the system.

Guest actor enters the system. We don't expect guests to have accounts in the system so Guest moves onto the sign-up page. Sign-up page can be seen in Figure 6.1.

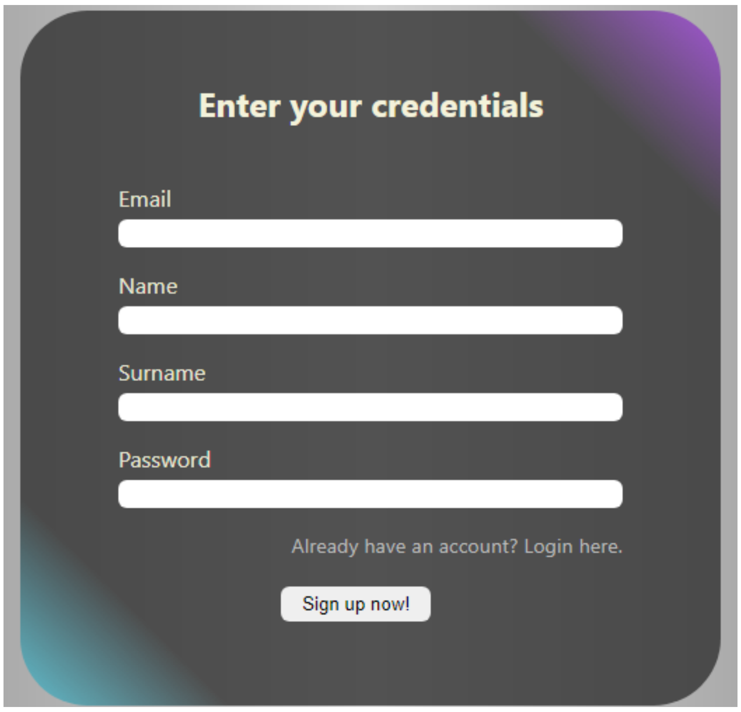


Figure 6.1 Sign-up Page

On this page, they enter their credentials and these credentials are sent to the back end. The back end returns a success or an error. Error is shown to the Guest for three seconds and the Guest is sent back to entering their credentials stage. On success, a success message is shown and the Guest is redirected to the login page. Login page can be seen in Figure 6.2.

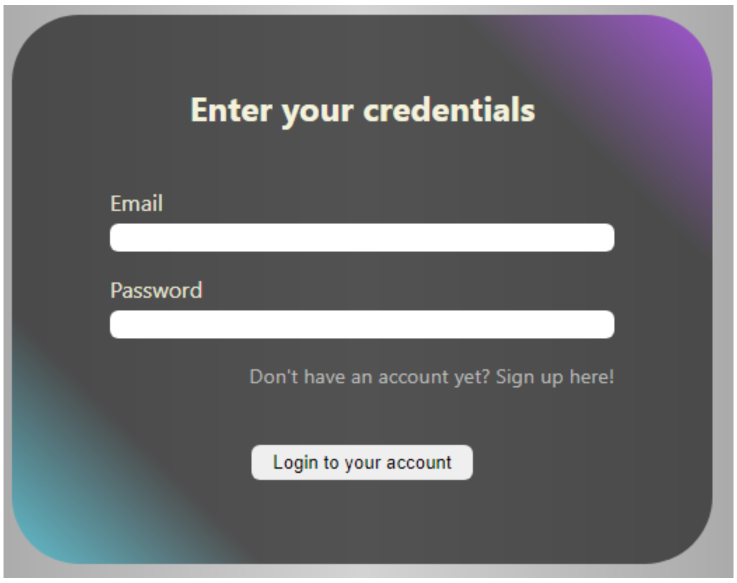


Figure 6.2 Login Page

The Guest then enters their email and password on the login page. On successful login, they are redirected to the homepage with their account stored in session. We will call this actor User from now on.

But before we move onto what a User can do let's examine what a User with admin rights can do. We will call this actor Admin for short.

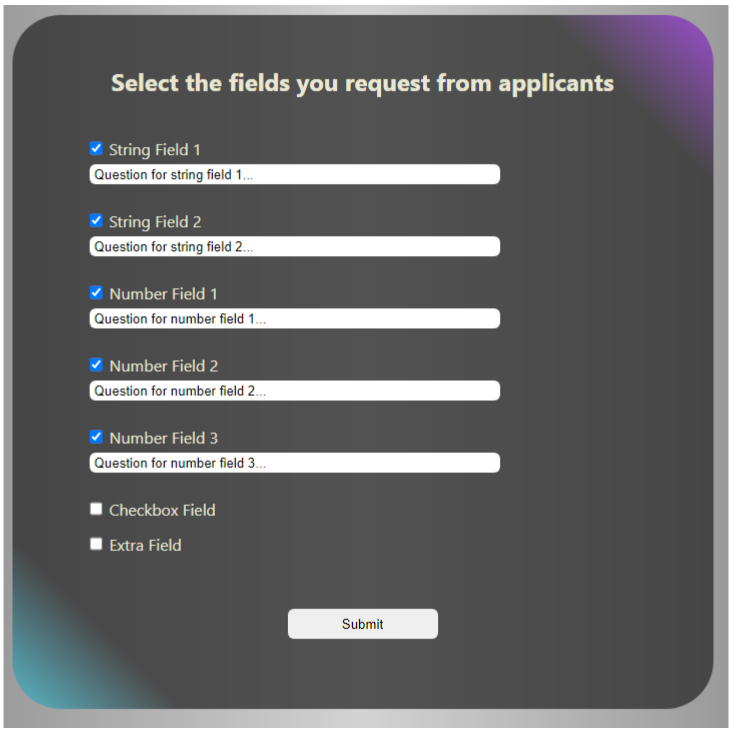
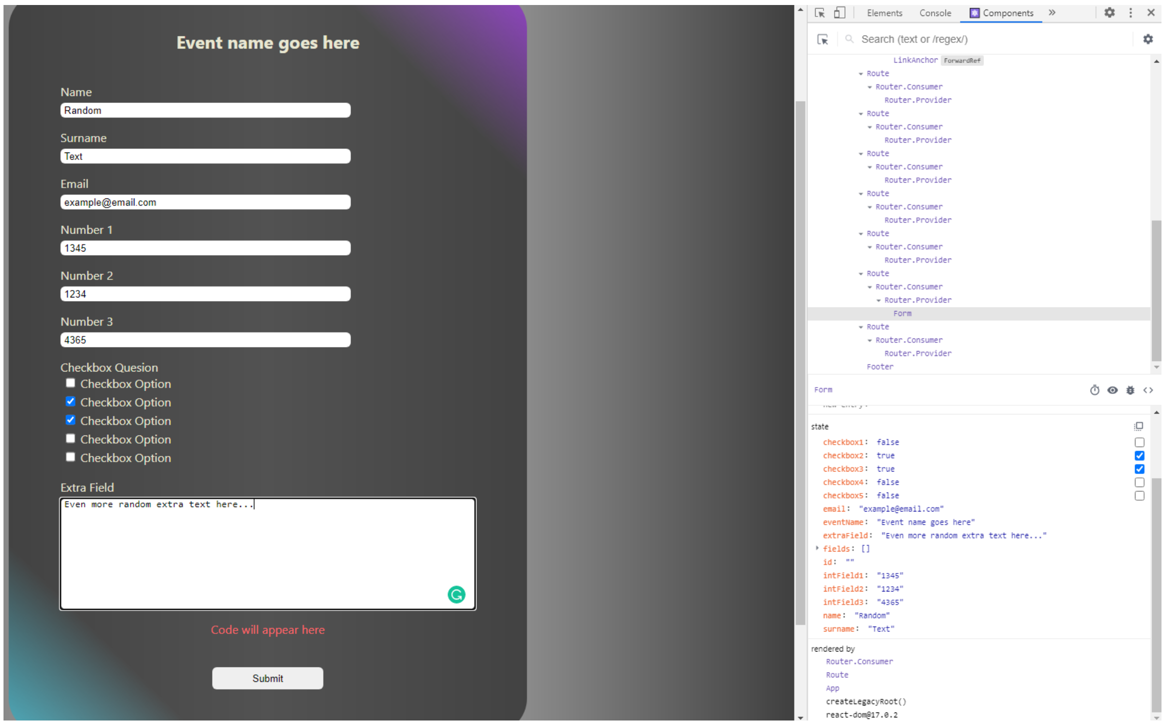

Select the fields you request from applicants
String Field 1
Question for string field 1...
String Field 2
Question for string field 2...
Number Field 1
Question for number field 1...
Number Field 2
Question for number field 2...
Number Field 3
Question for number field 3...
Checkbox Field
Extra Field
Submit

Figure 6.3 Creating a Form

Admin enters the system with their credentials just like any other user. Unlike other users, Admin can create events in the system. Admin moves to the event creation page and enters the credentials of the event. Along with the event Admin also has to create a form in the system. This is the interactive part of the system. Admin will choose which fields will be requested in the event registration form and what the labels of those fields will be. A form in the making can be examined in Figure 6.3.

After the form has been created, the creation of the event concludes. And now Users can fill the form and register for the event. A filled form can be seen in Figure 6.4. On the left side, the user interface shows a form we created with all the possible fields enabled, and on the right in React Developer extension from Google Chrome, you can see all the input from the User being captured and processed by React. Normally the red text showing "Code will appear here" would not be there until the answers are submitted but because submitting the answers clear the form we have enabled it for demonstration purposes. When answers are submitted, a hash code of 256 characters will be shown to the User here.



Event name goes here

Name: Random

Surname: Text

Email: example@email.com

Number 1: 1345

Number 2: 1234

Number 3: 4365

Checkbox Quesion
- [ ] Checkbox Option
- [x] Checkbox Option
- [x] Checkbox Option
- [ ] Checkbox Option
- [ ] Checkbox Option

Extra Field: Even more random extra text here...

Code will appear here

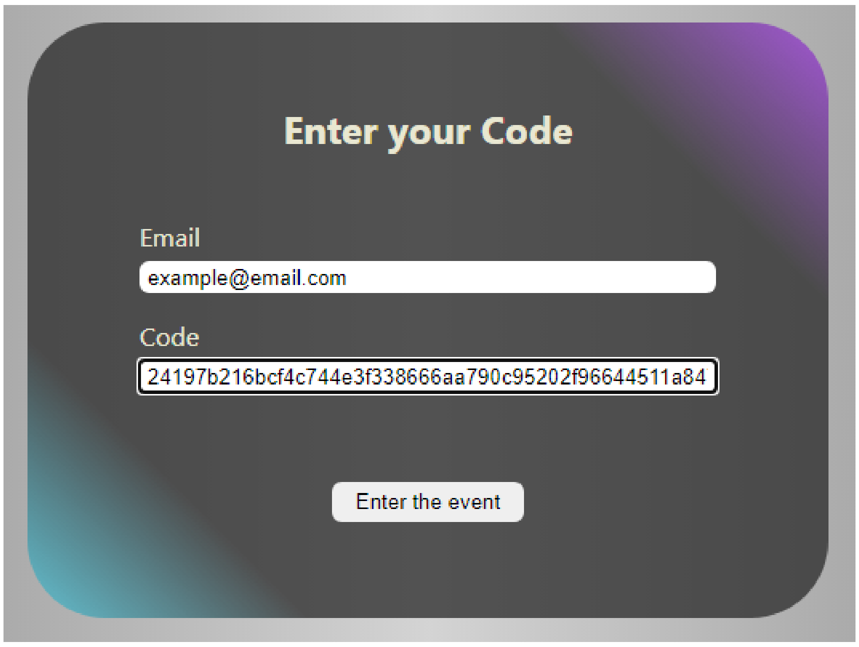
Submit

```
      LinkAnchor ForwardRef
    ▾ Route
      ▾ Router.Consumer
          Router.Provider
    ▾ Route
      ▾ Router.Consumer
          Router.Provider
    ▾ Route
      ▾ Router.Consumer
          Router.Provider
    ▾ Route
      ▾ Router.Consumer
          Router.Provider
    ▾ Route
      ▾ Router.Consumer
          Router.Provider
    ▾ Route
      ▾ Router.Consumer
        ▾ Router.Provider
            Form
    ▾ Route
      ▾ Router.Consumer
          Router.Provider
    Footer

Form
state
  checkbox1: false
  checkbox2: true
  checkbox3: true
  checkbox4: false
  checkbox5: false
  email: "example@email.com"
  eventName: "Event name goes here"
  extraField: "Even more random extra text here..."
▸ fields: []
  id: ""
  intField1: "1345"
  intField2: "1234"
  intField3: "4365"
  name: "Random"
  surname: "Text"
rendered by
  Router.Consumer
  Route
  App
  createLegacyRoot()
  react-dom@17.0.2
```

Figure 6.4 Filling a Form

After User has registered to an event, they don't need to do anything else until event time. At event time they need to direct to the redirection page or if they try to access the event page, they will be redirected by the system. On the redirection page, the system will ask for their email address and their hash code they have gotten in the previous step.


Enter your Code

Email

example@email.com

Code

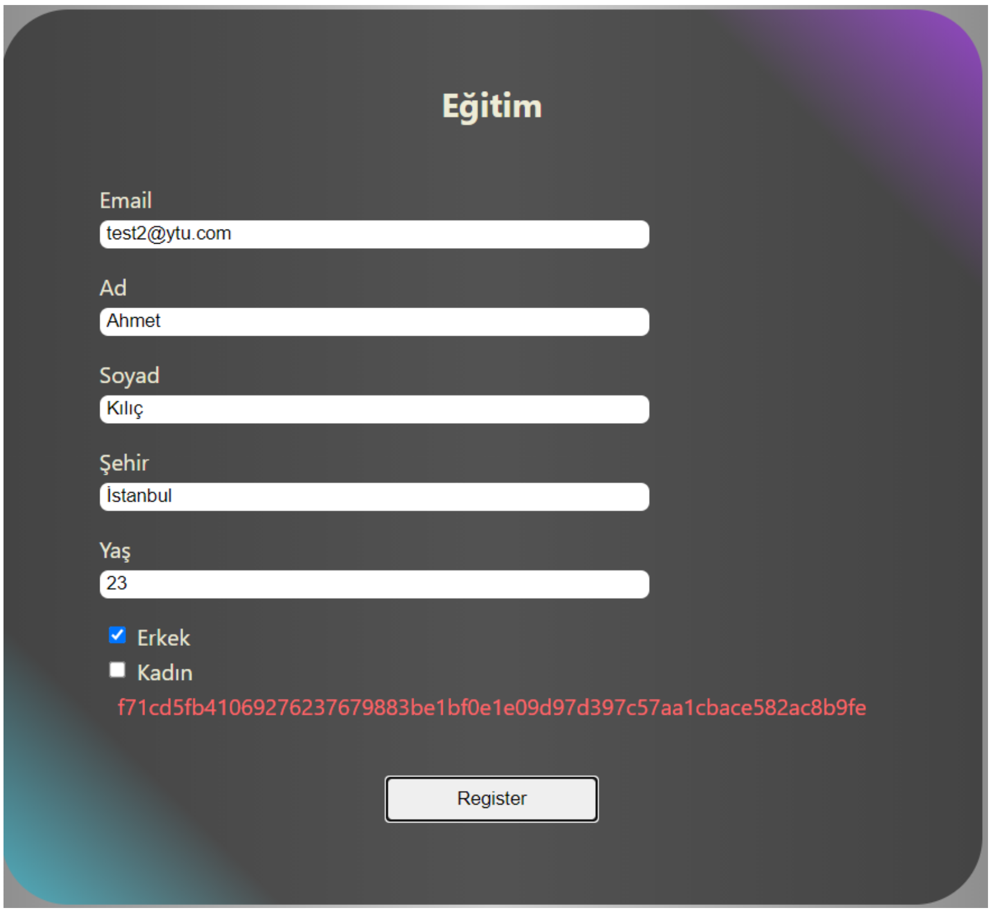
24197b216bcf4c744e3f338666aa790c95202f96644511a84

Figure 6.5 Redirection Page

In Figure 6.5 an example of a redirection page usage can be seen. While the email address is a fake one, the code is a 256 character long hash code created by the hash function of the system. When the code is accepted, the system will mark the User as attended and redirect them to the event link. Which moves the User out of the system's scope, finishes the responsibilities of the system, and concludes the use-case scenario.

7
Experimental Results

We tested different registration actions for an event. We expected different verifying codes for different input and it worked accordingly and accurately.



Eğitim

Email
test2@ytu.com

Ad
Ahmet

Soyad
Kılıç

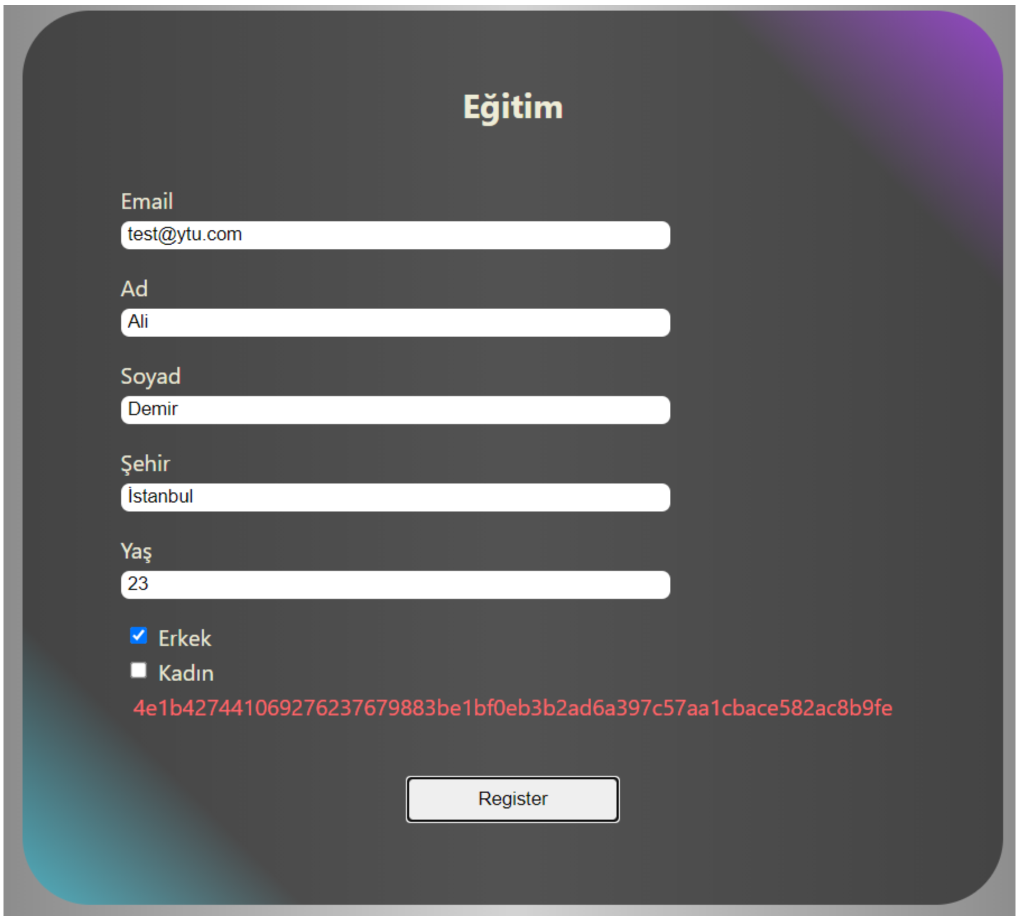
Şehir
İstanbul

Yaş
23

[✓ Erkek]
Kadın

f71cd5fb41069276237679883be1bf0e1e09d97d397c57aa1cbace582ac8b9fe

Figure 7.1 Experiment 1



Email
test@ytu.com

Ad
Ali

Soyad
Demir

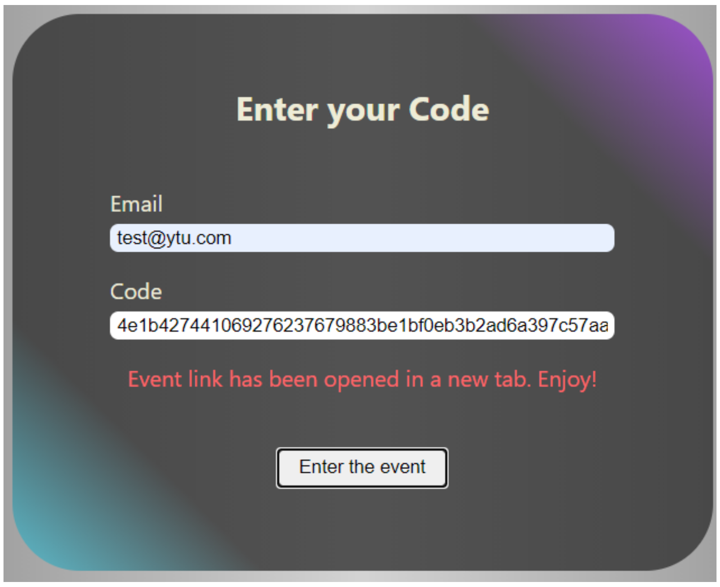
Şehir
İstanbul

Yaş
23

[✓ Erkek]
Kadın

4e1b427441069276237679883be1bf0eb3b2ad6a397c57aa1cbace582ac8b9fe

Figure 7.2 Experiment 2

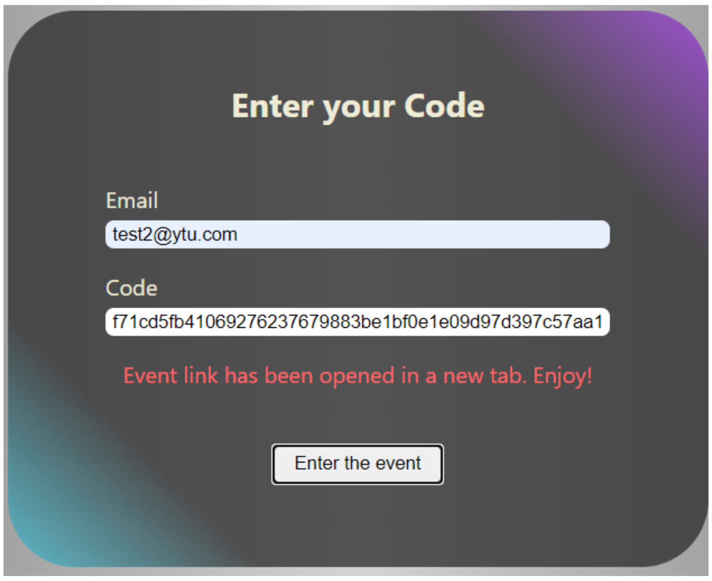


Email
test@ytu.com

Code
4e1b4274410692762376798883be1bf0eb3b2ad6a397c57aa

Event link has been opened in a new tab. Enjoy!

Figure 7.3 Experiment 3



Email
test2@ytu.com

Code
f71cd5fb41069276237679883be1bf0e1e09d97d397c57aa1

Event link has been opened in a new tab. Enjoy!

Figure 7.4 Experiment 4

8 Performance Analysis

The system design is optimized to keep track of only the essentials on memory. Session information for a user for example is tracked by the user id which is kept in the React state of front end. Since it is the front end that is keeping track of sessions, the back end doesn't have to.

On the other hand front end also has to send user id whenever it wants to execute registration or creation on the database so we can keep track of who did what.

This separation reduces the RAM needs on both client and server. We took this approach because if we wanted to store information we would need to be storing duplicates of the same information on front end, back end, and in database. Database is given but the rest is not necessary immediately.

Thanks to this design choice, the number of active account are not directly limited by the RAM of the server. Now the most limiting factor would be connection speed and bandwidth. But the system doesn't send big chunks of data across the internet. Some of the most taxing actions are registering to an event, creating and event, and searching for a list of events. First two of these are hard capped at 2.5KB of data maximum for any package, because each field only accepts up to 256 characters at most. The last one is not hard capped by the technical limits but by us. The biggest load on the system would be to give wildcard as the event name to try to get all of the events in the system onto the client. The search function doesn't accept empty strings or special characters as valid search inputs so it is also capped by the maximum number of event with a specific name, which we don't have an immediate solution to. These can be tracked by system admins in the future of course.

One possible vulnerability is DDoS attacks on the server. Since it relies on data flow from the client, there are some function on the front end that makes multiple calls to back end. Malicious people can trigger these function in quick succession on multiple devices to overwhelm the processing power of the server and bring it to a halt.

9
Conclusion

The system has hierarchical user structure starting from guest, user and up to organizer. Organizers in the system are the only entity able to create a new event. While creating event, organizers can choose the registration form fields dynamically for taking personal information from attendees. After creating an event, organizer should share the form URL with attendees who then will fill the form and register to the event. Form URL will be active until the form end date provided by Organizer. Organizer also will share the redirect URL with attendees which verifies their code and redirects them to event URL.

Users can sign up to the system. When logged in, Users can register to the events and can see which events they registered to. Some events can let only only be registered if you have an account.

Guests can only register to the events which has guest level registration permission. When guests or users register to the event, system will give them a unique code. When event time arrives, redirect URL will be active and they can attend to the event using their email and unique code. If their email and code verification are correct, system confirm their attendance and redirects them to event.

References

[1] T. Traphagan, J. V. Kucsera, and K. Kishi, “Impact of class lecture webcasting on attendance and learning,” *Educational technology research and development*, vol. 58, no. 1, pp. 19–37, 2010.

[2] F. Masalha, N. Hirzallah, *et al.*, “A students attendance system using qr code,” *International Journal of Advanced Computer Science and Applications*, vol. 5, no. 3, pp. 75–79, 2014.

[3] J. Railsback, “Increasing student attendance: Strategies from research and practice.,” *Northwest Regional Educational Laboratory NWREL*, 2004.

[4] Y. Kawaguchi, T. Shoji, W. Lin, K. Kakusho, and M. Minoh, “Face recognition-based lecture attendance system,” in *The 3rd AEARU workshop on network education*, Citeseer, 2005, pp. 70–75.

[5] N. K. Balcoh, M. H. Yousaf, W. Ahmad, and M. I. Baig, “Algorithm for efficient attendance management: Face recognition based approach,” *International Journal of Computer Science Issues (IJCSI)*, vol. 9, no. 4, p. 146, 2012.

[6] N. Kar, M. K. Debbarma, A. Saha, and D. R. Pal, “Study of implementing automated attendance system using face recognition technique,” *International Journal of computer and communication engineering*, vol. 1, no. 2, p. 100, 2012.

[7] S. Sarana, A. Sadida, A. Mansyur, and A. Suwondo, “Design and development of student attendance information system using qr code in accounting department of politeknik negeri semarang,” in *IOP Conference Series: Materials Science and Engineering*, IOP Publishing, vol. 1108, 2021, p. 012 012.

[8] (2021). “Clockify website,” [Online]. Available: `https://clockify.me/` (visited on 05/10/2021).

[9] (2021). “Event attendance website,” [Online]. Available: `http://www.event-attendance.com/` (visited on 05/10/2021).

[10] K. V. K. Kurulu, “Kişisel verilerin korunması kanunu ve uygulaması,” *Ankara: Kişisel Verileri Koruma Kurumu. Erişim adresi: https://www. kvkk. gov. tr/yayinlar/K% C4% B0% C5% 9E% C4% B0SEL% 20VER% C4% B0LER% C4% B0N% 20KORUNMASI% 20KANUNU% 20VE% 20UYGULAMASI. pdf*, 2018.

Curriculum Vitae

FIRST MEMBER

Name-Surname: Orkun AVCI
Birthdate and Place of Birth: 28.09.1998, Balıkesir
E-mail: orkunavci28@gmail.com
Phone: 0537 563 36 18
Practical Training: ElectroPazar Bilgisayar

SECOND MEMBER

Name-Surname: Erhan AYAKKABICIOĞLU
Birthdate and Place of Birth: 08.02.1993, İzmir
E-mail: erhanayakkabicioglu@gmail.com.tr
Phone: 05323100743
Practical Training:

Project System Informations

System and Software: Windows Operating System, Google Chrome(8.0+), Java(SDK 8+), MySQL
Required RAM: 2GB
Required Disk: 1GB