SVEUČILIŠTE U ZAGREBU

**FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA**

ZAVRŠNI RAD br. 454

**DEVELOPING A WEB APPLICATION FOR ORGANIZING SPORTING EVENTS**

Jura Darko Šlehan

Zagreb, lipanj 2022.

Content

[Introduction 1](#_Toc107255900)

[1. Application description 3](#_Toc107255901)

[2. Software specifications 7](#_Toc107255902)

[2.1. Logged out or unregistered user 7](#_Toc107255903)

[2.2. Logged in user 8](#_Toc107255904)

[2.3. Explore events 9](#_Toc107255905)

[2.4. Create a new event 10](#_Toc107255906)

[2.5. Event participation or cancelation 11](#_Toc107255907)

[2.6. View a profile 12](#_Toc107255908)

[3. System architecture and design 14](#_Toc107255909)

[3.1. Front-end 14](#_Toc107255910)

[3.2. Back-end 15](#_Toc107255911)

[3.3. Database 16](#_Toc107255912)

[3.3.1. SQL databases 16](#_Toc107255913)

[3.3.2. NoSQL databases 17](#_Toc107255914)

[3.3.3. SportEve database 17](#_Toc107255915)

[4. Implementation and user interface 20](#_Toc107255916)

[4.1. Technologies and tools 20](#_Toc107255917)

[4.1.1. React.js 20](#_Toc107255918)

[4.1.2. ExpressJS 20](#_Toc107255919)

[4.1.3. MongoDB 21](#_Toc107255920)

[4.1.4. Mongoose 21](#_Toc107255921)

[4.1.5. Other technologies and tools 21](#_Toc107255922)

[4.2. System implementation 21](#_Toc107255923)

[4.2.1. Registration 22](#_Toc107255924)

[4.2.2. Log in 23](#_Toc107255925)

[4.2.3. Event exploration 24](#_Toc107255926)

[4.2.4. My events 25](#_Toc107255927)

[4.2.5. Viewing an event 26](#_Toc107255928)

[4.2.6. New event 28](#_Toc107255929)

[4.2.7. Profile 29](#_Toc107255930)

[Conclusion 31](#_Toc107255931)

[Literature 32](#_Toc107255932)

[Sažetak 33](#_Toc107255933)

[Summary 34](#_Toc107255934)

# Introduction

The subject of this bachelor's thesis is the development of a web application for organizing sporting events. Sporting events include all kinds of athletic activities that a person can do in their free time. This application is targeted toward amateur athletes or just regular people who are looking for others who share their interest in recreational sports. This means that you can organize football games, find company for a jog in the morning and invite users to any other similar activity. The idea is that the people who use this app have a chance to meet other sports enthusiasts or just make the organization of sporting events with their friends easier.

The idea for this web application came from my friend who frequently organizes football games and has a difficult time finding participants and managing problems such as who to invite and how to handle cancelations. Applications that serve this purpose already exist, but the idea of recreating or perhaps even through time improving this kind of application sounded like an interesting challenge.

This application is going to be a web application because they are accessible to anyone who has access to the Internet. Developing a web application is a very in-demand skill in today's market and it is also a skill that I want to learn and improve, so this thesis is a great chance for me to do so. Most of the applications with this theme are mobile applications and that could be a good second step, but that is also why a web application could be refreshing.

An application that is strongly related to this is a mobile application called Tossli [1]. Tossli is an application where you can meet people to play sports with and reserve playgrounds. You can create and explore sports events, join them and play your favourite sports with other people. This app also allows you to explore sports venues and chat with other people. It is available on Android and iOS. We aim to develop an application that serves a similar purpose.

The name of this application is SportEve, which is a combination of the words sports and events. It is short and easy to memorize, but it also captures the spirit of this application, and that being the people's shared interest in sports. [4]

This document consists of 4 chapters:

1. Application description
2. Software specifications
3. System architecture and design
4. Implementation and user interface

# Application description

My goal is to develop a web application for organizing sporting events. This application will allow users to create sporting events, invite their friends or meet new people who share their interest in sports. Users will be able to create their accounts and once they have done so, they will have the ability to view their profile, as well as other user's profiles. Registered users can also view already created events and join them, or create new ones. Details about events and user profiles can be added and edited, that way people can share desired information about events or themselves. This application should allow users to simply view and manage their sports activities.

The first page that a user will see when starting this app should look like the one on Image 1.1.

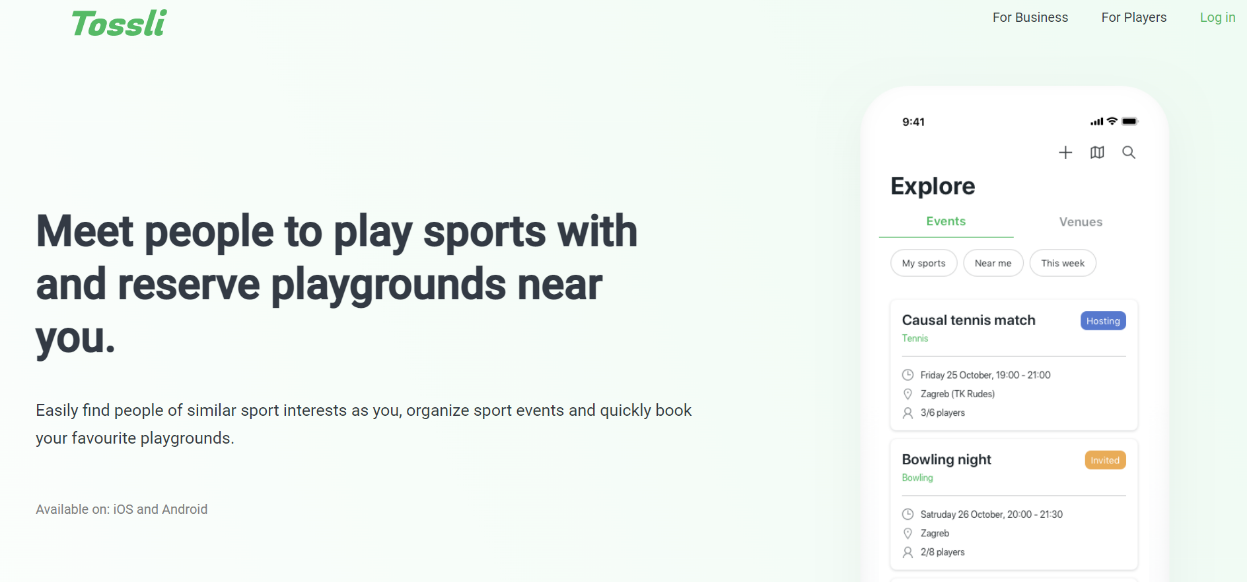


Image . Register/Login page example

This starting page should also have two main options:

1. Register
2. Login

If the user does not have a created account, logging in will not work because they do not have an account. By clicking on Register, the user will be taken to a registration page. Users have options to register with their email and password and with their google account. If they choose the first option, they can also enter other information about themselves (for example, their nickname). Once they enter all of the information correctly, the user can submit their registration form. The application will save their data, and they will be redirected to the login page.

However, if the user chooses the option to register with a google account, the application will take them to the page where they will have to choose their google account, and if they have done so successfully, the user will be registered. After that, they will be taken to a page where they can edit additional information about themselves. Once that is done, the user will be sent to the login page.

A registered user can log in to the application by clicking on the Login button, which will take them to the login page. There, the user will also choose their preferred login option (using their email and password or google account). A successful login will finally take them to the main page of this application which is only available to the logged-in users.

Once a user is logged in, they will have access to all of the application features. A logged-in user can view their profile, which is a page that shows information about the user, and the example is shown on Image 1.2.

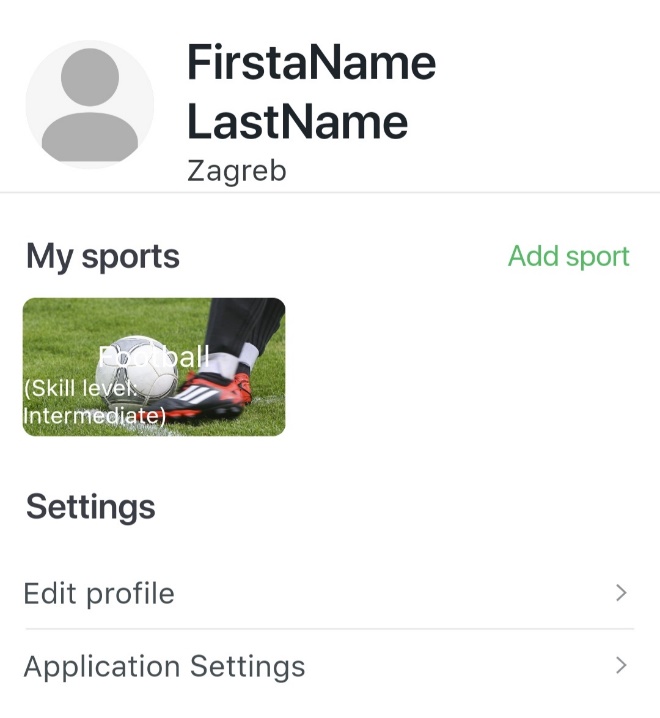


Image . Profile example

That way a user can choose which information to share with the other users. The information displayed on the profile can also be edited. Aside from viewing their own profile, users will be able to explore other users and view their profiles. Users can view events that other people have created, see details about those events and create their own events. Event exploration might look something like the one shown on the Image 1.3.

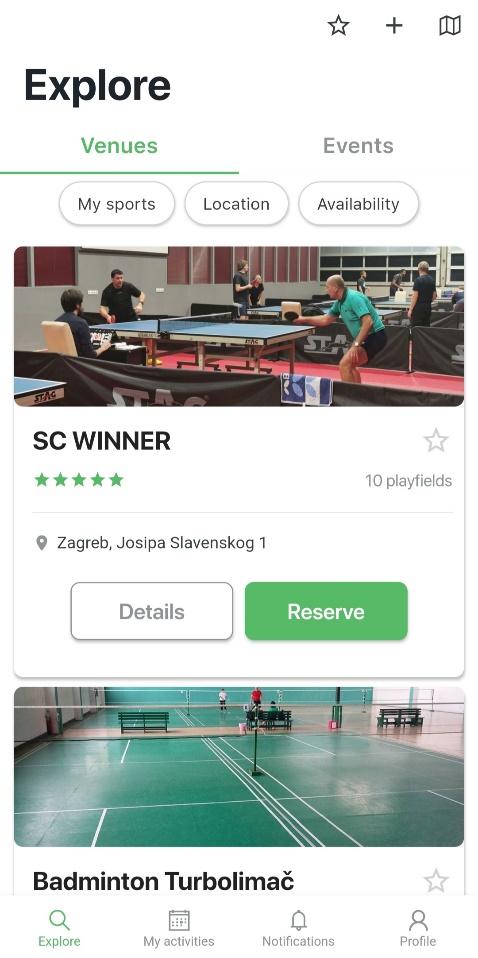


Image . Explore events page example

Creating events is a key feature and it should be done with a form where a user provides information about the event, as shown on the Image 1.4.

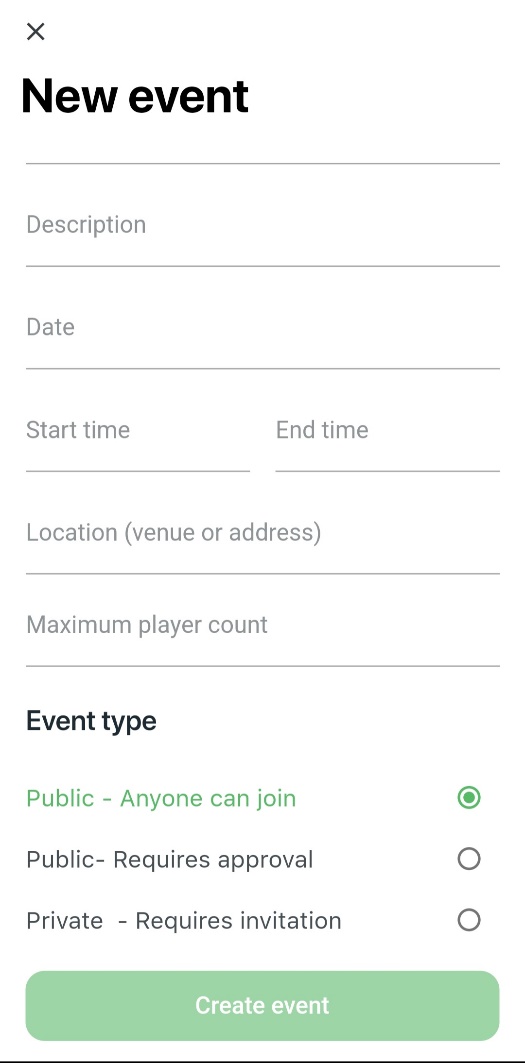


Image . Event creation example

They will also have an option to see all of their events, the one that they are taking part in or the ones they have created on a separate page.

# Software specifications

In order to make an application, a developer needs to know all of its desired functionalities. Use-case diagrams are a great way to visually represent those functionalities. They describe the high-level functions and scope of a system and identify the interactions between the system and its actors. The use cases and actors in their diagrams do not explain how the system works internally, it only describes what this system does and how it is used by the actors. The use-case diagrams are a visual representation of the use cases. [8]

## Logged out or unregistered user

If the user does not have an account, he can make it by clicking on Register which will take them to the register view which consists of a registration form and the google authentication button. This means that the user can register with their email and password by filling out the registration form with some information about their account. All of that information can be changed later on the user's profile page. Some of those attributes are optional, but the email, password and nickname fields are mandatory and the registration cannot be successful without them. If the user enters attributes that the application deems invalid, the user will be prompted to change them, and will not be able to submit them. Once the registration form is filled out and submitted, the user will be registered and a new account with their information will be created. The other registration option is via their Google account. This means that the application will use their Google profile information in order to register a new user. In this case a new password or email will not be required, but the user will have an option to add some other attributes (like their nickname, full name or their date of birth). After that, the user will be registered and they will have access to their new account. The use case diagram for this use case is shown on Image 2.1.

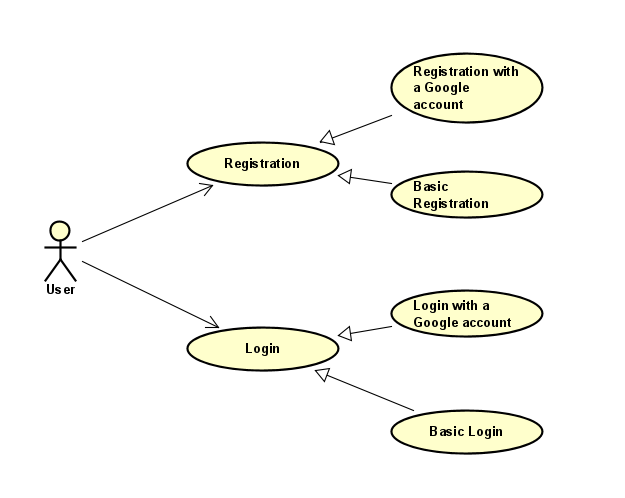


Image . Logged out or unregistered user

## Logged in user

The main features of this application are available to the user once they are logged in. A logged in user can explore events that the other users have created. This means that they can see other users events all in one place and view each one individually, which provides more detail about the event. The user is also able to join those events, and after that even cancel the participation. The main feature is the creation of new events. Users can do so by providing information about the event in the form and submitting it. After that, the event will be created and other users will be able to join them. The host of the event can also remove and add new participants to the event. If the host cancels his event (only the host of the event can cancel it), that event will no longer exist and users will not be able to see it. Users can see all of the events that they have created of are participating in. Events that have already taken place will be available in the users event history. Every user will have a profile page which will show certain information about them, and some of it will be open for editing when a user desires to do so. Finally, a logged in user can log out at any time, which means that their session has ended. The use case diagram that describes the abilities of a logged in user can be seen on Image 2.2.

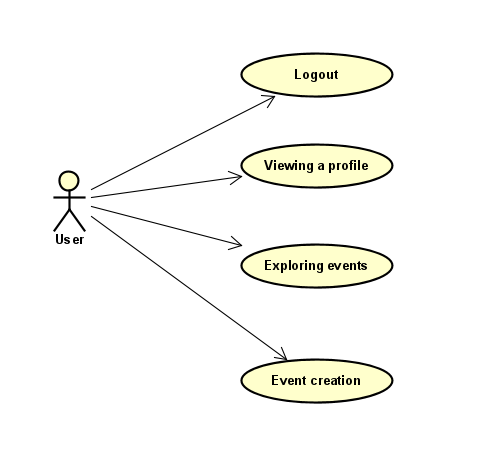


Image . UC2 Logged in user

## Explore events

Event exploration consists of exploring new events to participate in and browsing your own events. The purpose of looking through the list of new events is to join and then participate in them. The users events are listed because that way the user can track in which events they are participating in (in the future) or the ones they have already participated in. The idea of exploring events is shown on the use case diagram on Image 2.3 UC3 Exploring events.

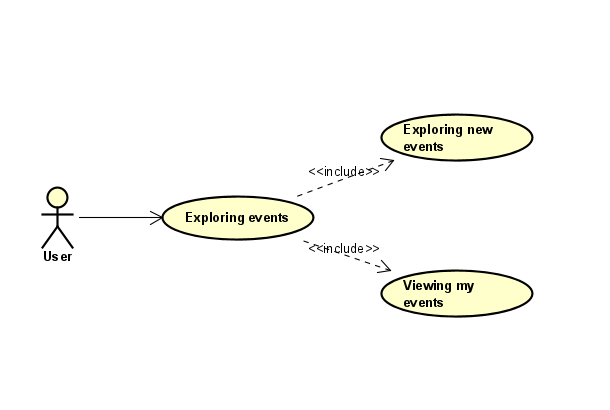


Image . UC3 Exploring events

## Create a new event

The creation of a new event is basically filling a form that consists of some crucial information that users have to provide in order to make it. That way the event becomes available for other users to view or even join. Use case diagram number 4 describes this functionality on Image 2.4.

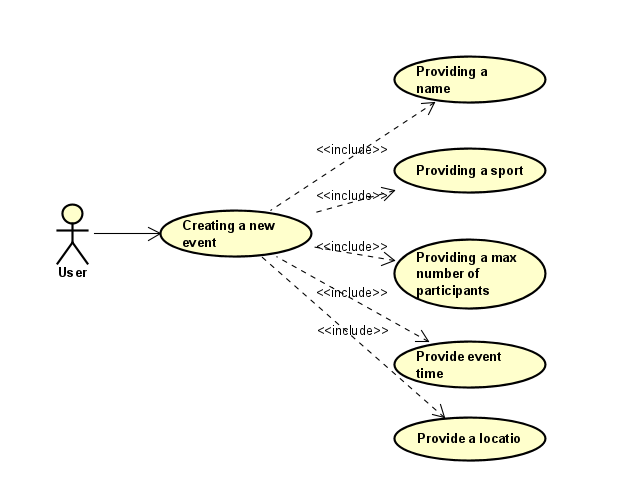


Image . UC4 Event Creation

## Event participation or cancelation

If a user chooses an event for viewing, there are some actions that the user can do, depending on their role in that event. A host of a certain event is the person who created that event. Once they have created it, the host can add new participants to the event, remove participants that have already been added or even cancel that event, as seen on Image 2.5. A user can become a participant of an event if the host has added them or if they have joined the event themselves, as shown on Image 2.6. Participants of an event can cancel their participation.

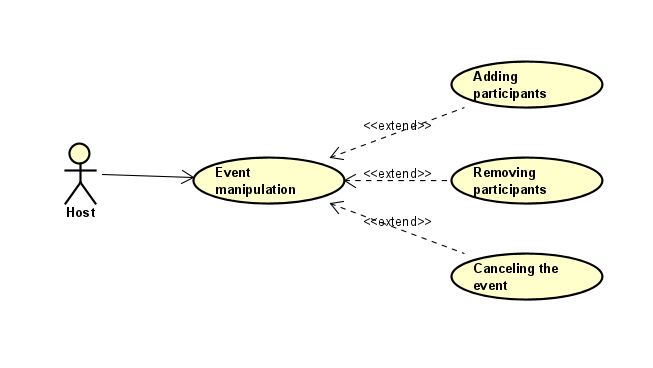


Image . UC5 Hosts event actions

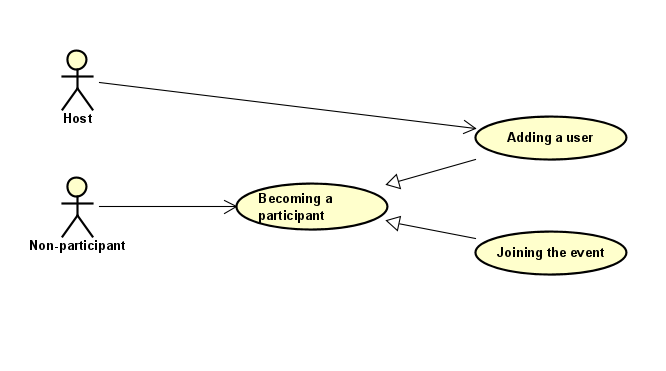


Image . UC6 Becoming a participant of an event

## View a profile

Each user has their own profile and the ability to view other user's profiles. If the users views their own profile, they will also be able to edit certain information about themselves. That, of course, cannot be done with the profiles that belong to another user. Those are only available for viewing. All of this is explained by the diagram on Image 2.7.

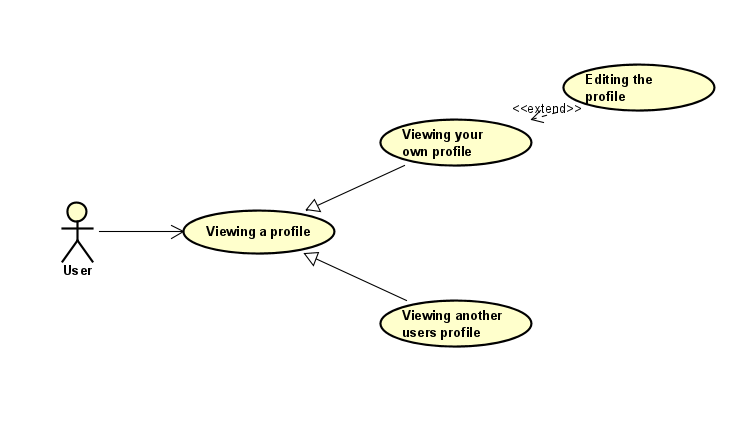


Image . UC7 Viewing a profile

# System architecture and design

The architecture of this system can be divided in three components:

* Front-end
* Back-end
* Database

This three-tier architecture is a very popular application architecture in which all of the components have their role; the front-end or the presentation tier with which the user interacts with, the back-end or the application tier where the data processing happens and the database where the data is stored and managed. [3]

## Front-end

The role of the front-end or the presentation tier of the application is to show the user interface to the users and enable them to interact with the application. This tier, in most cases, does not handle any businees or process the data that the user has provided. However, the front-end usually validates users' data and passes that data to the back-end. The communication between the presentation tier and the application tier is classically done through the use of HTTP requests. The HTTP protocol is the main and the most used data transfer method. This communication goes both ways, as the back-end can also send data to the front-end and provide it to the user. All of this is done through HTTP requests and responses. The code that is used to make the presentation tier and its content is HTML (HyperText Markup Language). It offers a wide array of options to structure the content, from paragraphs to data tables or images, but in order to style the content better the language that is used is CSS. CSS (Cascading Style Sheets) describes how HTML elements are displayed on the screen. A front-end framework that is also frequently used to complement HTML and CSS is Bootstrap. Bootstrap provides templates for the creation of the user interface. To provide more complex features to the front-end of an application, developers use a programming language called JavaScript. It enables us to create dynamically updating and much more. A famous JavaScript library is React, which is a user interface library and a tool for building components of the user interface. It allows us to build interactive elements on the presentation tier and more about React will be explained in chapter 4.1.1.

## Back-end

The back-end is the component of a web application that processes the requests and responds to the client. It usually consists of three parts:

* The server (the computer that receives the requests)
* The app (the application that runs on the server and listens for requests, gets the data from the database and sends responses)
* The database (can be seen as an individual component, organizes data)

Servers are machines that are optimized for the purpose of listening to requests. This can be any computer that is connected to a network. In the case of this application, the server will be my personal computer.

An app is run on the server and it has the logic that is required for the task of responding to HTTP and URI requests. The HTTP and URI are called a route and if you match them for a certain request, you are doing an action called routing.

To handle some functions, an app will need a middleware, which is a code that is executed after the request has been received and before the response has been sent. Using the middleware means doing any of the following functionalities like; getting data from the database, modifying the object that has been requested or in any way processing the request. These middleware functions often give control to the next middleware function before sending the response. In the end there will have to be a middleware function that sends the response to the client.

Frameworks help the programmers doing the task of routing, which can be done by frameworks like Ruby on Rails or Express (which we will be using). We just need to keep in mind that if a request has been made to a route, there will be some handler functions that are triggered by that request and executed.

There are different kinds of data that are sent by the server. That can be a JSON format, HTTP status code or even an HTML file. [5]

## Database

Databases can be seen as a separate part of a web application or just a part of the back-end. They give the user an option to persistently save the data in the memory. This allows the server to run more easily, as the database takes the task of storing the data and gives the option of retrieving data even when the server crashes.

A lot of requests that are sent to the server also have a query to the database. This means that the client has requested some data that is in the database, or event wants to add some data to it.

A database can be a SQL database or a NoSQL database.

### SQL databases

The SQL language is an ANSI standard which also includes some dialects like T-SQL or PL/SQL. If you use SQL that is compatible to the ANSI standard, it can transfer your scripts to some other SQL database more easily.

Some of the more popular SQL databases are MySQL, Oracle, Microsoft SQL Server and PostgreSQL. Now we will go into some of the features of SQL databases.

The SQL databases are relational because of the unique tuples that are in the collection, but some people think that the name comes from the ability to define relations between entities in the database.

The relation in a SQL database is represented by a table, where a row of the table is a tuple. Defining rows and tables is called defining a schema.

Data in the relational database should always be normalized before storing, because it saves storage. If a user has a schema, the pro is that you know what values and entities you expect in your app, but that also has its cons, and those are the lack of the ability to deal with dynamic data. The schema also allows you to validate the data, and force some relationships in you database. These databases are usually quick, robust and reliable.

### NoSQL databases

The meaning of the name NoSQL means that these database do not only use SQL. There is some SQL to be found in NoSQL databases.

There are four categories of NoSQL databases:

* Document stores
* Key-value stores
* Graph databases
* Wide-column data stores

The NoSQL databases usually belong in one of these categories, but there are some, like Cosmos DB that belongs in multiple categories.

The pro of these databases is scalability and speed, but it also sacrifices robustness.

The most popular type of NoSQL databases are document stores. Document stores are kind of like SQL databases, but there is no normalization or schema. The collection is whatever you decide it is, or what you put in it, as opposed to the rows and columns.

It is easy to define a new entity, but it also means that there will be fields that are defined, and some that are not. There is also the ability to store the same entity as many times as you like, with the values that differ.

These databases are very good with dynamic data and they as opposed to the SQL databases, the NoSQL ones can run on more than one server.

Some of the most popular document stores are DynamoDB, Firebase, Cosmos DB and the one we will be using; MongoDB. [7]

### SportEve database

The database for this web application is going to be a NoSQL database. Data will be stored in a JSON format, which will contain attributes that describe each schema.

The main two schemas that will be used are users (Image 3.1) and events (Image 3.2), as the whole application is based around them. An additional schema will be the one that describes the rating (Image 3.3) between two users.

This NoSQL database will be achived with the help of MongoDB and Mongoose, which we will discribe in the later chapter.

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

Image . Database schema for a user

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

Image . Database schema for an event

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

Image . Database schema for rating

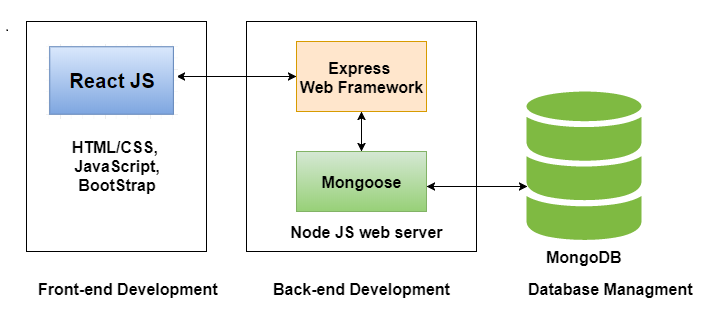


Image . Architecture schema

# Implementation and user interface

## Technologies and tools

In order to create a three tier architecture web application, we need to use some technologies and tools to satisfy all of the requests needed to create each of the components of this application. This chapter will describe the technologies and tools that were used to create this web-application. It is also important to note that the programming language that was used in the development of this application is JavaScript, so the technologies and tools are going to be based around it. The reason for using JavaScript is that it is one of the most, if not the most used programming language in web development.

### React.js

Right now, React.js is one of the most popular, if not the most popular front-end library based on Javascript. It has a large community and a strong foundation.

It is efficient, declarative, flexible and it is used for building UI components that can be reused. React.js is only responsible for the view layer of an application, and it is also component-based and open source. React was used for developing applications like Facebook, Instagram and WhatsApp.

React.js User Interfaces are developed with the goal of speeding up the applications. Virtual DOM is a JavaScript object that improves the overall performance of the application. React.js can also be used with other frameworks. It helps in maintaining larger apps as well as improving the readability by using data and component patterns. [6]

### ExpressJS

ExpressJS is a flexible and minimal NodeJS framework which provides a large set of features for the development of mobile and web applications. With a myriad of HTTP utility methods and middleware at your disposal, creating a robust API is quick and easy. Express provides a thin layer of fundamental web application features, without obscuring Node.js features that you know and love. Many popular frameworks are based on Express. [10]

### MongoDB

MongoDB is a document-oriented NoSQL database used for high volume data storage. Instead of using tables and rows as in the traditional relational databases, MongoDB makes use of collections and documents. Documents consist of key-value pairs which are the basic unit of data in MongoDB. Collections contain sets of documents and function which is the equivalent of relational database tables. MongoDB is a database which came into light around the mid-2000s. [11]

### Mongoose

Mongoose is a library for Node.js and MongoDB that uses Object Data Modeling (ODM). It helps managing the relationship between the data, providing the validation of a schema and the translation object that are in code and their representation in the MongoDB database. [12]

### Other technologies and tools

Other technologies and tools that were user are Passport, GoogleOAuth, Git and Visual Studio Code.

We used Passport for authentification and hashing and salting the passwords, GoogleOAuth was used for the register and login function by using a Google account. Git was used for version control, with the help of GitHub. Visual Studio Code was the code editor that we chose for the development of this application.

## System implementation

After all the work of developing an application is done, the app becomes ready for usage. The user interface is one of the most important parts of the web-application, as the user directly interacts with it. This chapter will explain the basic features of this application and how the user interface works.

The starting page is shown on Image 4.1 and that is the first page that the user will see. After that the user can choose whether to register or login, which will take them to the pages that serve that purpose.



Image . Starting page

### Registration

If the user clicks the register button, the app will take them to the page shown on the Image 4.2. The page shows a form where the user has to provide data in order to create a new account. Aside form the form, there is also a Sign Up with Google button, which gives the user an option to register using their Google account information. If they choose that option, the application will take them to the Google Sign Up page, and if they successfully sign up, a new account will be created and they will be taken to the Login page where they will have to log-in if they want to access the main functionalities of the web-application.

The second option is to fill out the form and register with their email and password, while also providing a nickname that has to be unique, otherwise the Register button will be disabled. Once the user provides the correct data, their account will be created, and they will be prompted to log-in.

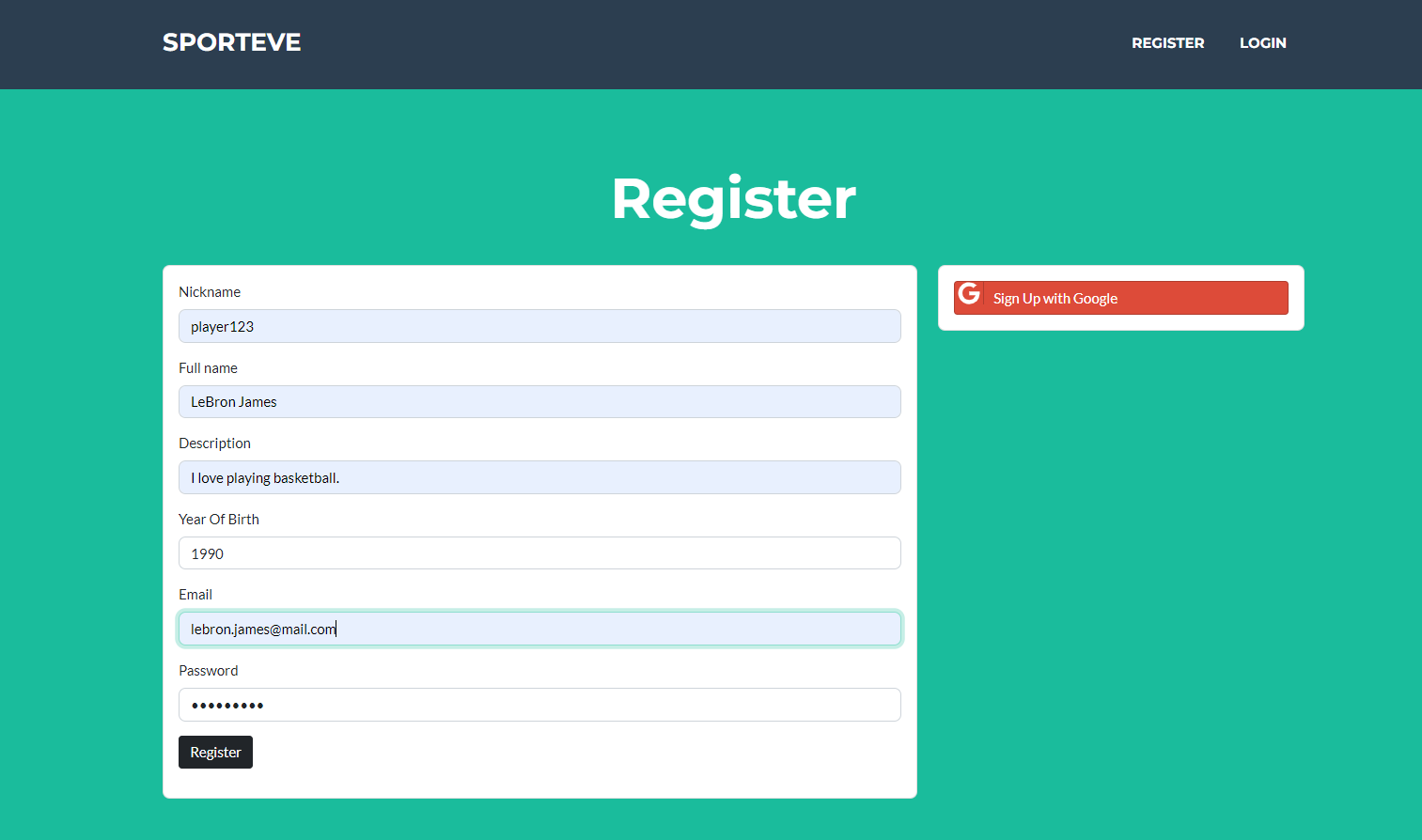


Image . Register page

### Log in

The starting page also provides access to the login page through the Login button. Should the user press that button, they will see the page on Image 4.3. Users that have already registered need to log-in to use this application. If the users does not have an account, they will not be able to log-in. In a similar fashion to the register functionality, a user has two options to choose from when logging-in. Pressing a Sign In with Google button simply loggs the user in using their Google account. That takes them to the main page of the application and unlocks the main functionalities of the application. To log-in with their email and password, the user has to fill the form, and if the data is correct, they will be logged in. The user that has provided incorrect data will be prompted to log in again and will see a message that says that their log-in has failed.

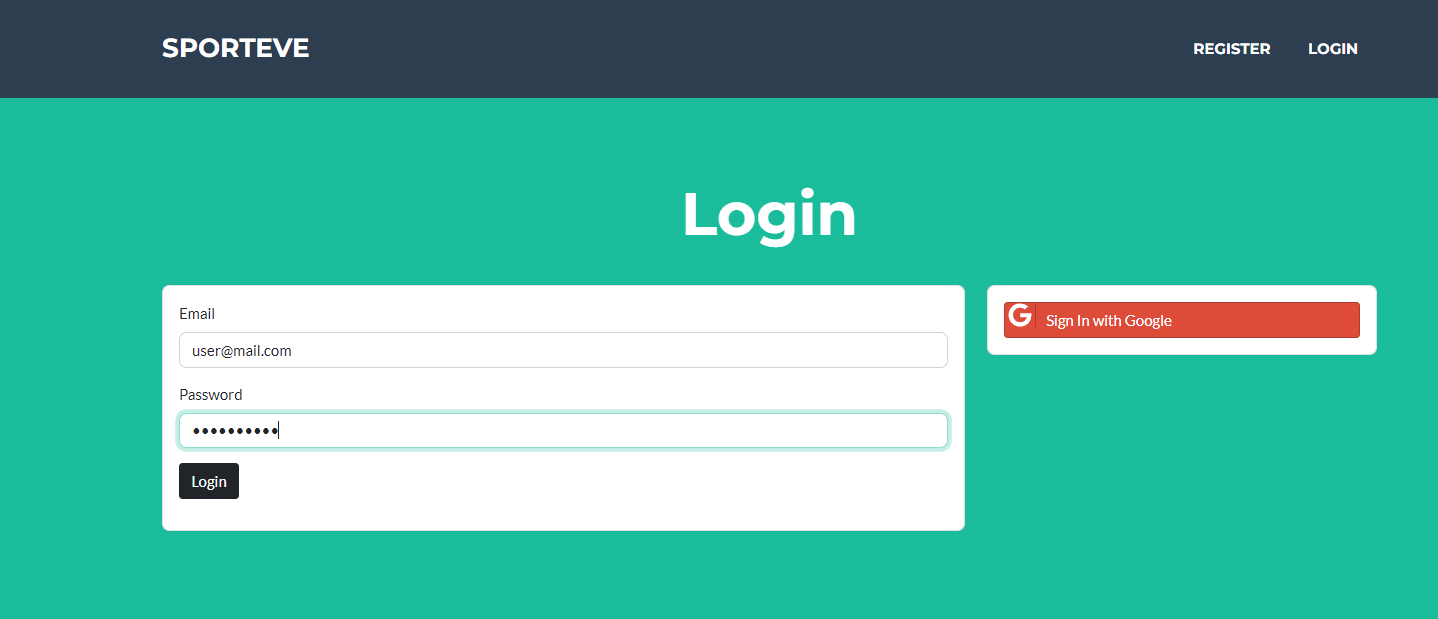


Image . Login page

### Event exploration

The first page a logged in user will see is the event exploration page (Image 4.4). On this page a user will be able to browse through all of the events that the other users have created. Each event is represented with its name and details about it, and by clicking on its name the user will see the event page.

This page, and the other pages (after the log-in) all have a navigation bar. The Logout button on the navigation bar simply logs the user out and ends their session. Other buttons on the navigation bar take the user to certain pages; the event creation page, profile page, my events page and the main page which is the event exploration page.

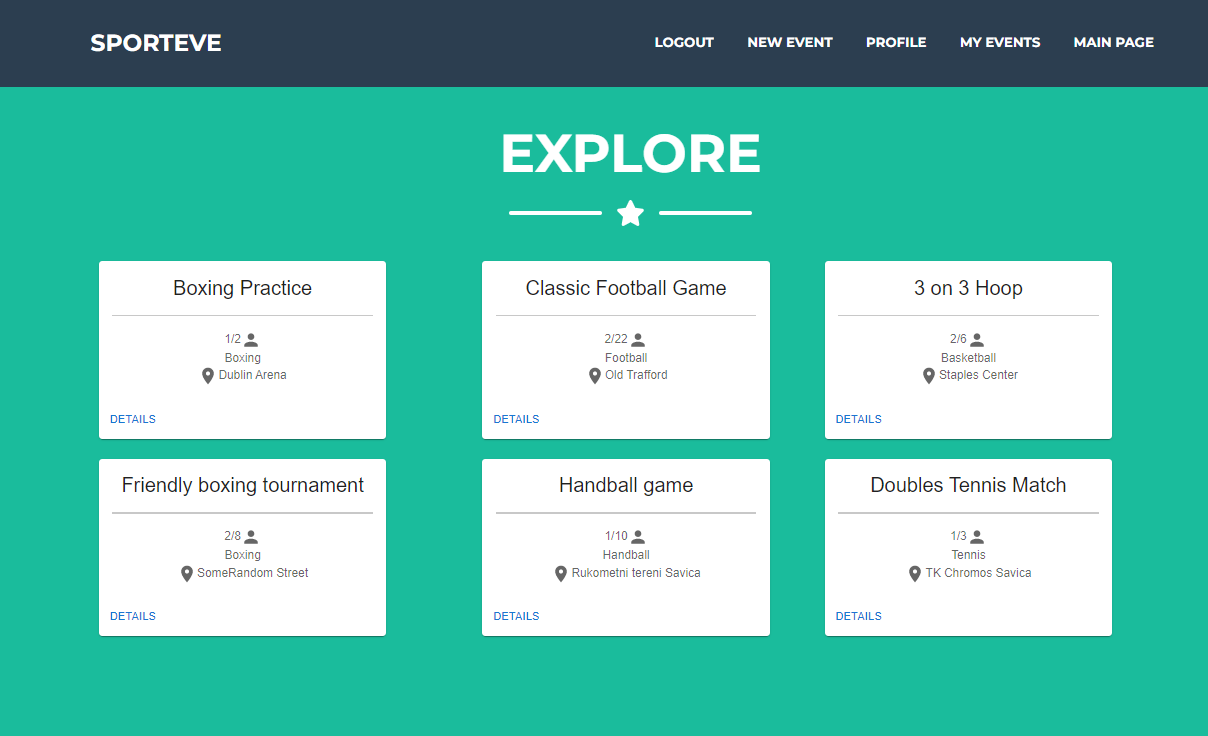


Image . Event exploration page

### My events

My events page shows all of the users events (Image 4.5). Those events include the ones that the user is hosting or participating in. The events are represented by their name and some details about them, all of the event information can be shown by clicking on its name.

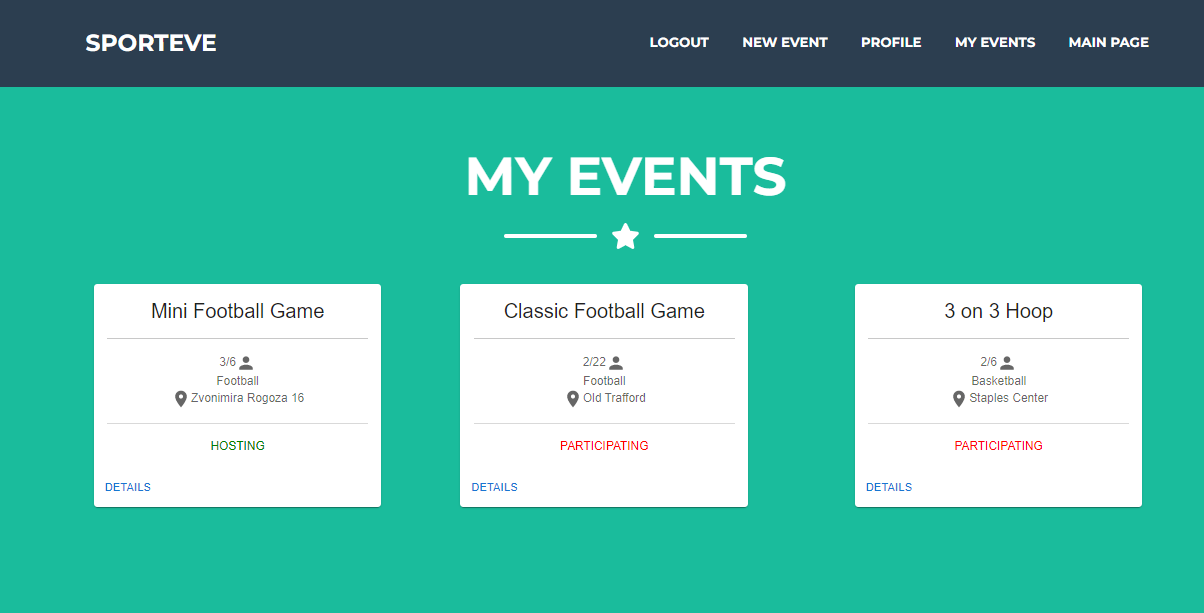


Image . My events page

### Viewing an event

The event page shows the details about the event. Every user can see some of the details about the event, like its title, sport, description, participants, etc. Depending on the role of the user in a given event, they will have access to certain functionalities concerning that event. If a host clicks on the event they are hosting they will be able to add or remove participants and cancel that event (Image 4.6.). Participants can be removed by clicking on the remove icon, and added by clicking on Add Participants where the host can write the nickname of a user in an autocomplete textbox, choosing the user and adding them. The Cancel event button serves the purpose of deleting that event and all the data that surrounds it.

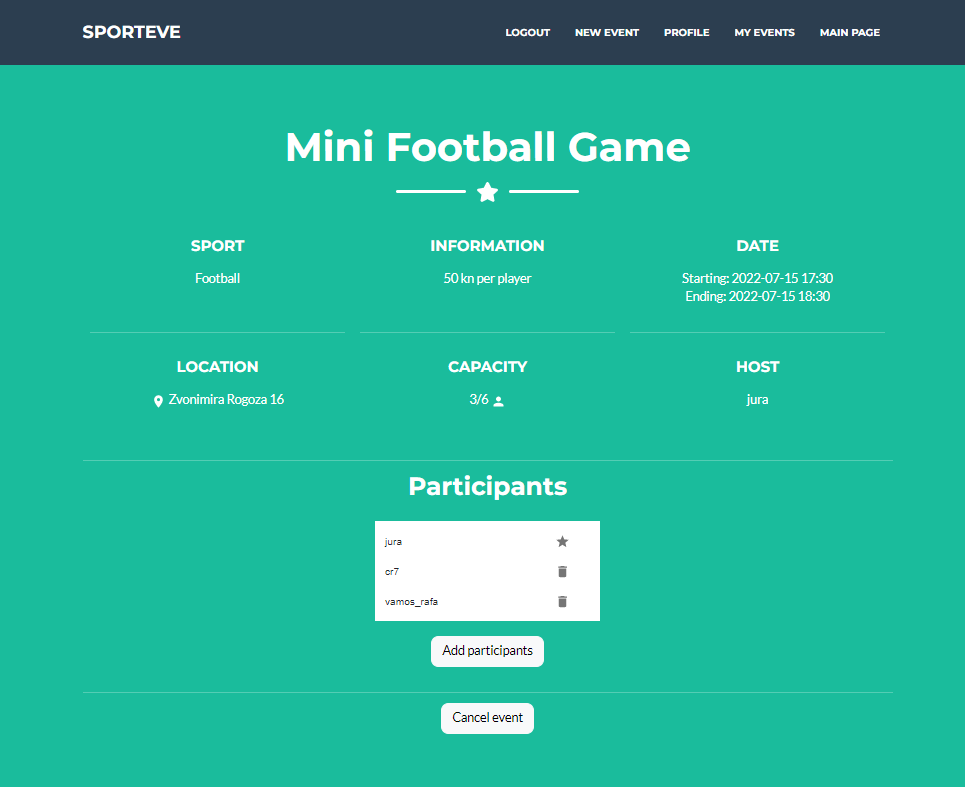


Image . Event page – host

A user that does not participate in the event that they are viewing has the option to join that event and become a participant (Image 4.7). Participants and non-participants cannot add or remove users from an event. A participant can always cancel their participation in an event by clicking the Cancel button (Image 4.8).



Image . Event page - non participant

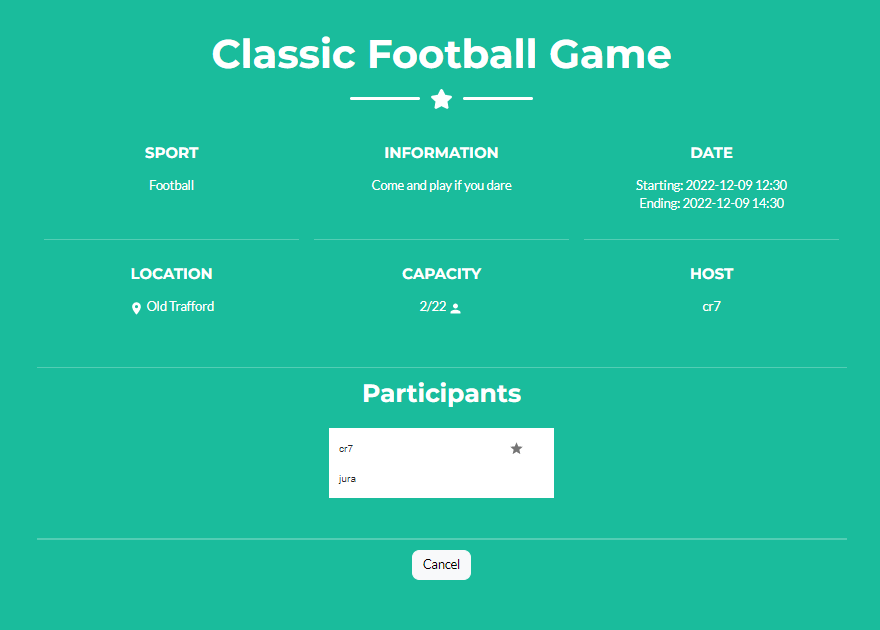


Image . Event page - participant

### New event

To create a new event a user has to fill the form with all the information concerning that event. An event has attributes like the name, sport, maximum number of participants, the description, and the starting and ending time and date. The user has to fill in the form correctly in order to submit the new event. The page that shows the event creation page and the filled out form can be seen on Image 4.9.

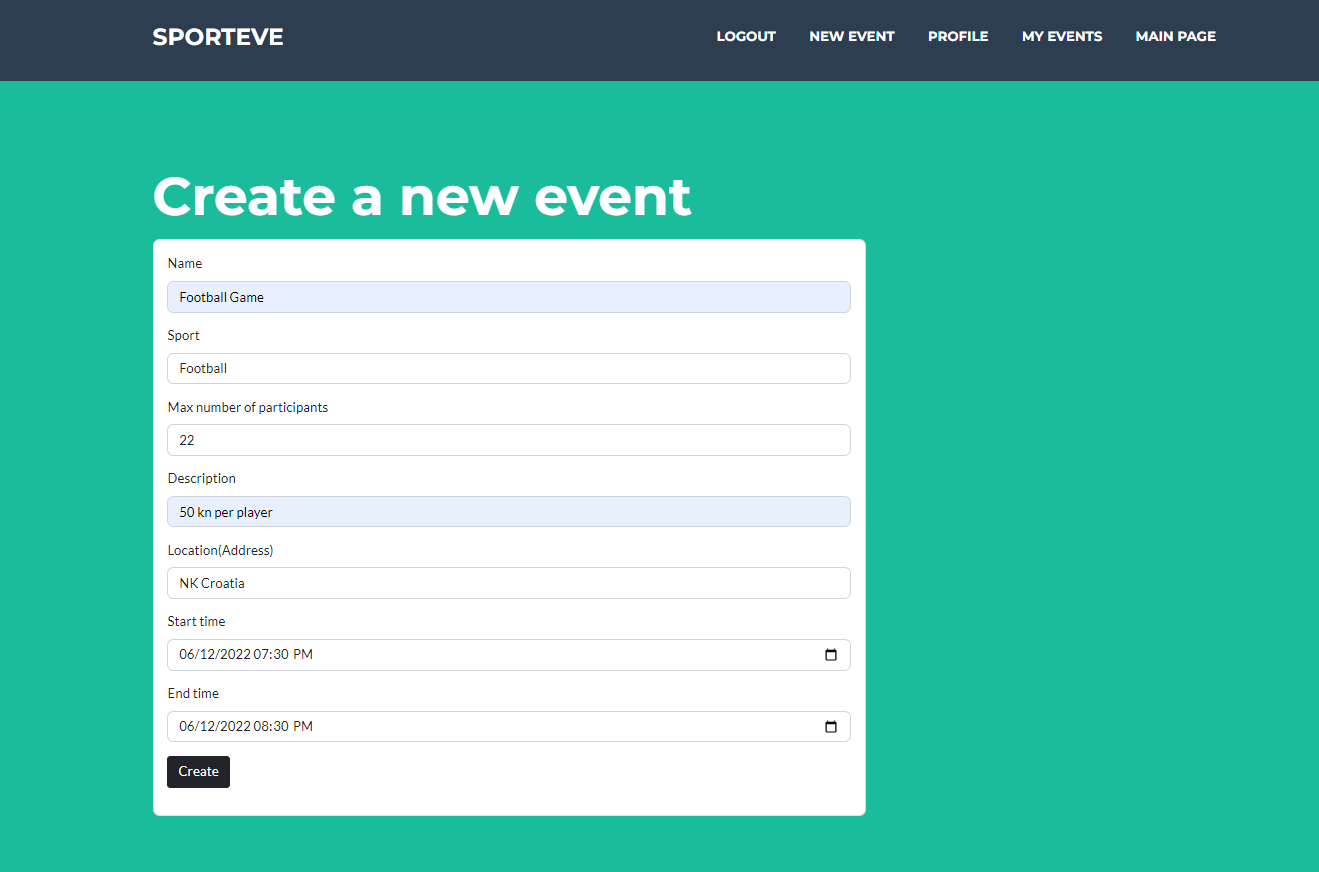


Image . Create new event

### Profile

A profile provides information about the user, like their email, nickname and other. The user can edit certain information by clicking on the edit profile button. Besides viewing their own profile, users can view profiles of other people, but of course, without the option of editing them. A profile can look like the one on Image 4.10.

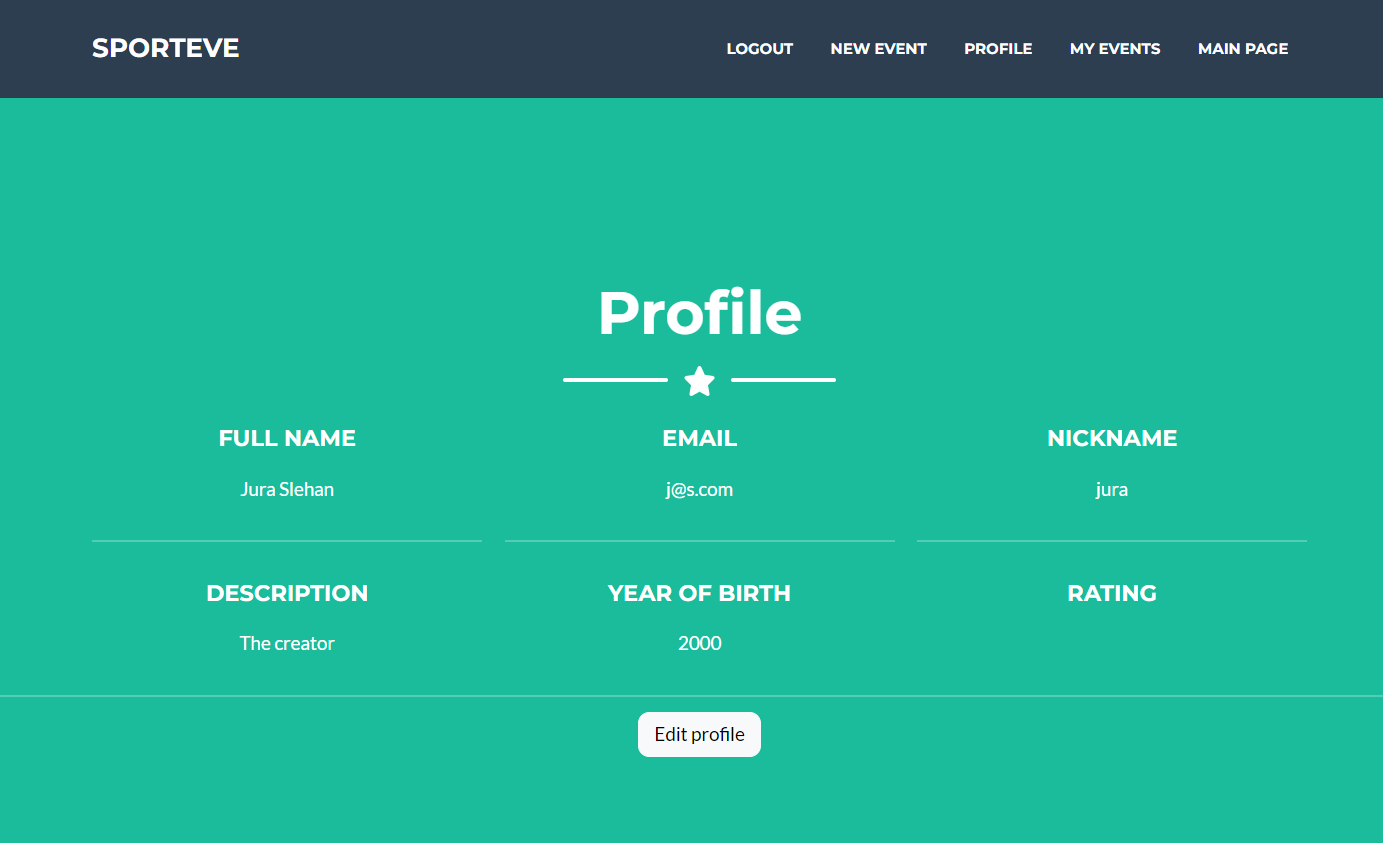


Image . Profile example

# Conclusion

The web application SportEve is an application for creating and joining new sporting events. For the development of this application, we used a three tier architecture.

The three tiers are:

* Front-end
* Back-end
* Database

The programming language that was used for the development of this application is JavaScript. This web application was made using a lot of tools and technologies that are commonly used for web development. Some of the technologies are ReactJS, Express.js and MongoDB. The application uses a NoSql database, which is an alternative to SQL databases.

SportEve provides functionalities that enable users to create accounts, and make new sporting events that they can enjoy with other people who also share their interest in sports. It makes organizing those events easier, which is enabled by the easy to use user interface. This should encourage people to try this and start managing their sporting events more easily.

# Literature

1. Tossli d.o.o., Tossli, (2020). Link: <https://www.tossli.com/>; accessed May 21, 2022.
2. Angela Yu, The Complete 2022 Web Development Bootcamp. Poveznica: <https://www.udemy.com/course/the-complete-web-development-bootcamp/>; pristupljeno March 7, 2022.
3. IBM Cloud Education, Three Tier Architecture, IBM, (2020, October). Link: <https://www.ibm.com/cloud/learn/three-tier-architecture>; accessed May 25, 2022.
4. Šalković, *Web-aplikacija za preporuku društvenih događaja,* Diplomski rad, Sveučilište u Zagrebu Fakultet Elektrotehnike i Računarstva, 2019
5. Code Academy, Back-end architecture, Link: <https://www.codecademy.com/article/back-end-architecture>; accessed: June 1, 2022
6. Javapoint, ReactJS Tutorial, Link: <https://www.javatpoint.com/reactjs-tutorial>; accessed: May 22, 2022
7. CircleCi, SQL vs NoSQL Databases, Link: <https://circleci.com/blog/SQL-vs-NoSQL-databases>; accessed, June 3, 2022
8. IBM, Use-case diagrams, Link: <https://www.ibm.com/docs/en/rational-soft-arch/9.7.0?topic=diagrams-use-case>; accessed, June 4, 2022
9. TechTerms.com, Frontend, Link: <https://techterms.com/definition/frontend>; accessed, June 1, 2022
10. Express, Express, Link: <https://expressjs.com/>; accessed, June 7, 2022
11. MongoDB, What is MongoDB, Linke: <https://www.mongodb.com/what-is-mongodb>; accessed, June 2, 2022
12. Freecodecamp; Introduction to Mongoose for MongoDB, Link: <https://www.freecodecamp.org/news/introduction-to-mongoose-for-mongodb-d2a7aa593c57/>; accessed: June 2, 2022

# Sažetak

Cilj ovog završnog rada je prikazati proces razvoja web aplikacije za organizaciju sportskih događaja koju koriste registrirani korisnici. Svrha ove aplikacije je olakšati organiziranje događaja kao i pronalaženje osoba sličnih interesa i prikupljanje određenih informacija o njima. Aplikacija će omogućiti korisniku da napravi profil i poveže se s drugim korisnicima. Korisnici će moći kreirati nove sportske događaje poput rekreativnih nogometnih utakmica i pozivati druge korisnike da sudjeluju. Nadalje, omogućit će svojim korisnicima da međusobno ocjenjuju i pregledavaju svoje profile. Prilikom kreiranja sportskih događaja, korisnici će morati dati određene informacije o događaju poput mjesta, vremena i trajanja.

# Summary

The goal of this thesis is to show the process of developing a web application for organizing sporting events that is used by registered users. The purpose of this application is to facilitate organizing events as well as finding people with similar interests and gathering certain information about them. The application will allow the user to make a profile and connect with other users. Users will be able to create new sporting events such as recreational football games and invite other users to participate. Furthermore, it will allow its users to grade each other and view their profiles. When creating sporting events, users will have to provide certain information about the event like the location, time, and duration.