# Express.JS Retake Exam – Shared Trip

## Exam rules:

1. You have **4 hours**
2. When you are ready, delete the ***node\_modules***folder, make sure all dependencies are listed in the ***package.json*** file and submit your archived project.
3. You are provided with **HTML & CSS** resources, which you can **modify**
4. Use **Express.js** as a back-end framework
5. Use **MongoDB** as a database with **mongoose**
6. You can use whatever **view engine** you like (Handlebars, EJS, Pug etc.…)
7. It is **forbidden** to **use** **React**, **Vue**, **Angular** etc…

## Application Overview

Get familiar with the provided **HTML & CSS** and create an application for **shared trip's plans**.

## Functional Requirements

The Functional Requirements describe the functionality that the **Application** must support.

### Guest (not logged in)

The **application** should provide Guest (not logged in) users with the functionality to login, register and **view** the Home page.

### Users (logged in)

The **application** should provide **Users** (logged in) with the functionality to:

* **View home page and all other pages with logged-in navigation**
* **View all shared trips**
* **Create new trip [Offer Tripp!]**
* **Access trip details page [Find Out More!]**
* **Deleting or joining some trip depending of user's authentication**

## Database Models (10 pts)

The **Database** of the **Shared Trip** application needs to support **2 entities**

### User

* Email - string (required), unique
* Password - string (required)
* **Trips History – a collection of Trips (reference to the Trip Model)**

Note: When a user creates a new trip, a reference to that trip is added to that collection.

### Trip

* **Start Point - string (required),**
* **End Point – string (required)**
* **Date – string (required),**
* **Time – string (required),**
* **Seats – number (required),**
* **Description – string (required),**
* **Car Image – string (required),**
* **Creator – string (required)**
* **Buddies – a collection of Users (reference to the User model)**

Note: When a user joined the given trip, a reference to that user is added to that collection.

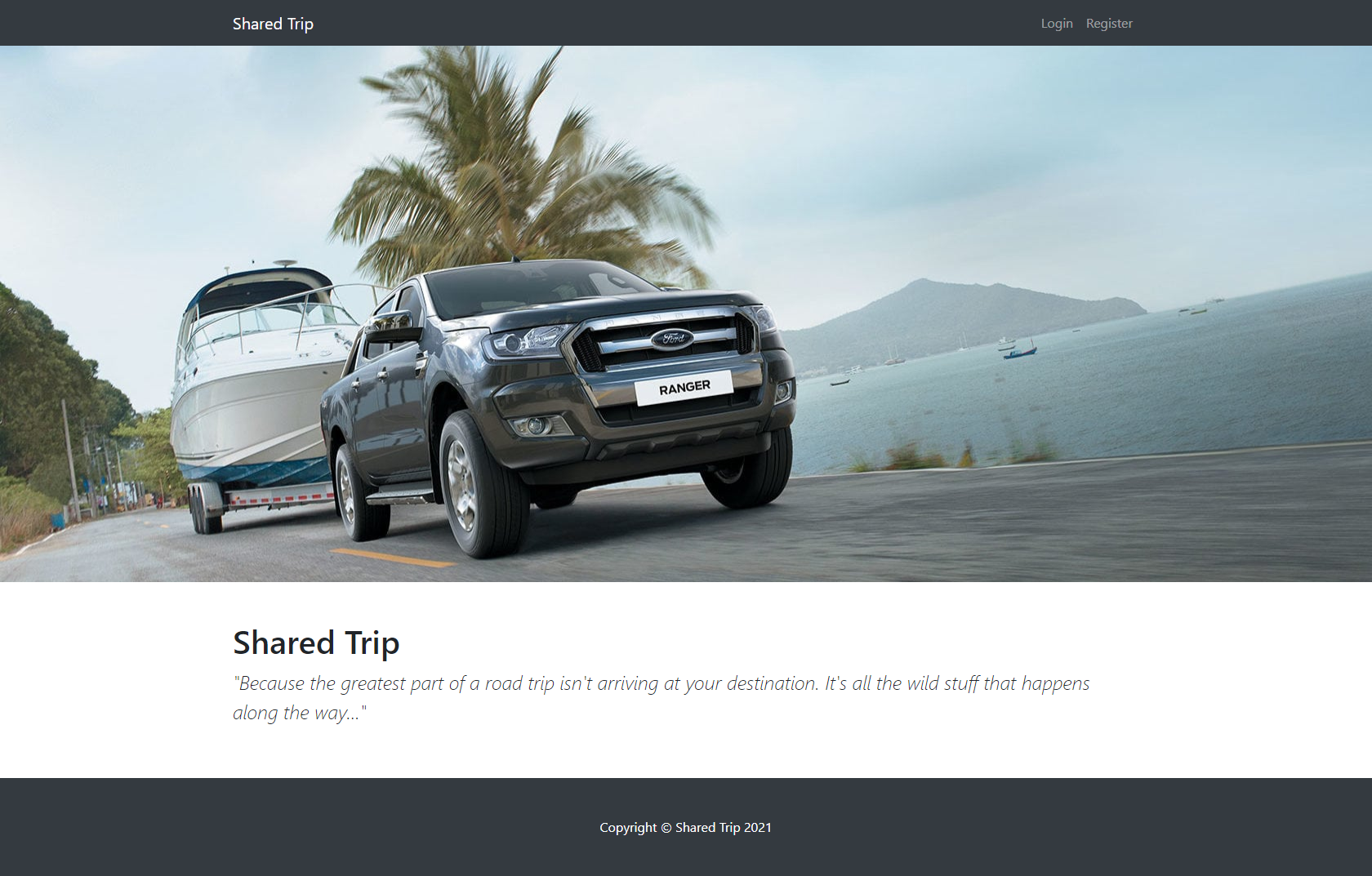
Implement the entities with the correct data types.

## Application Pages (70 pts)

### Guest Pages (15 pts)

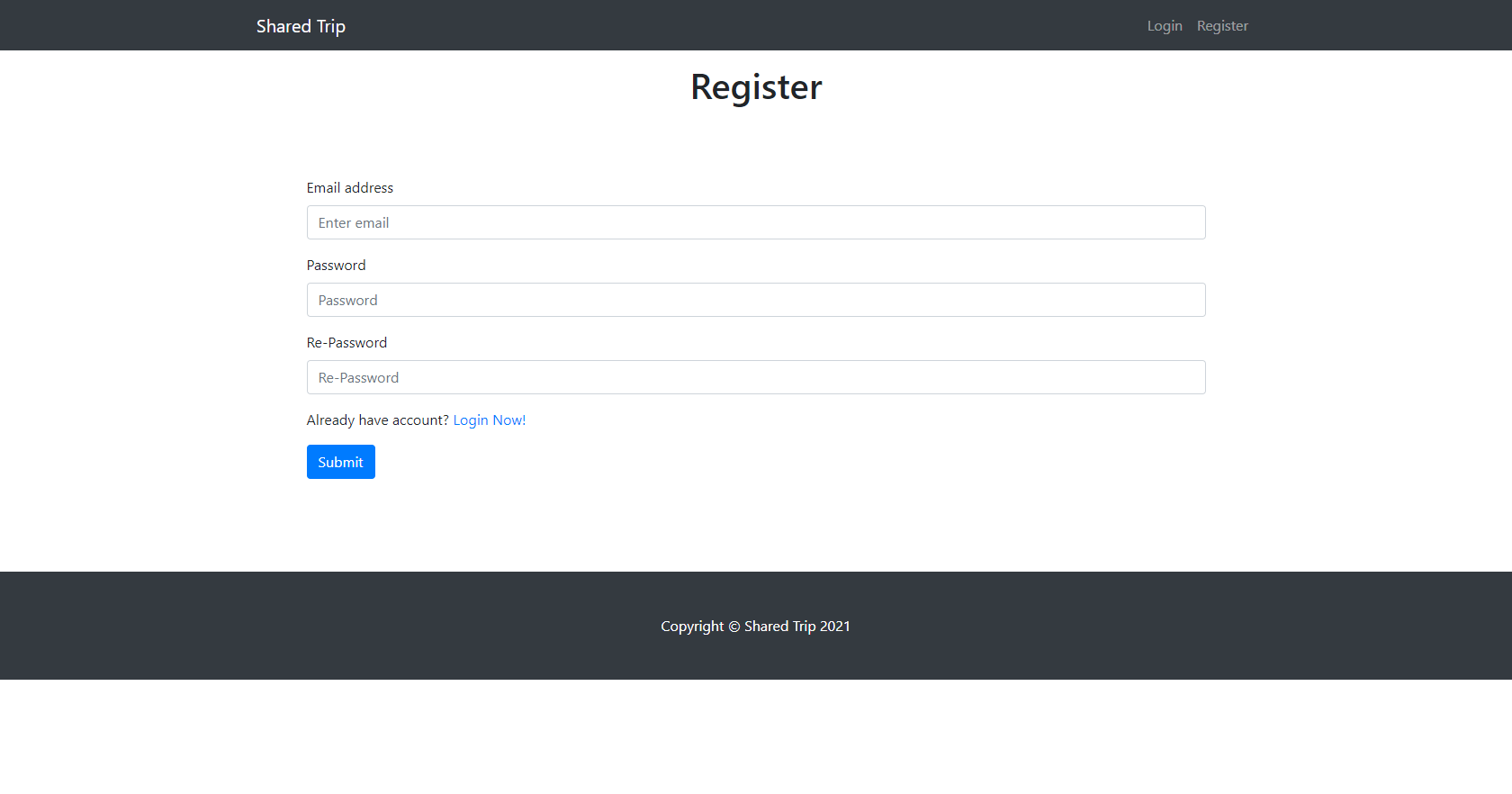
These are the pages and functionalities, accessible by **Guests (logged out users).**

#### Home Page (logged out user)



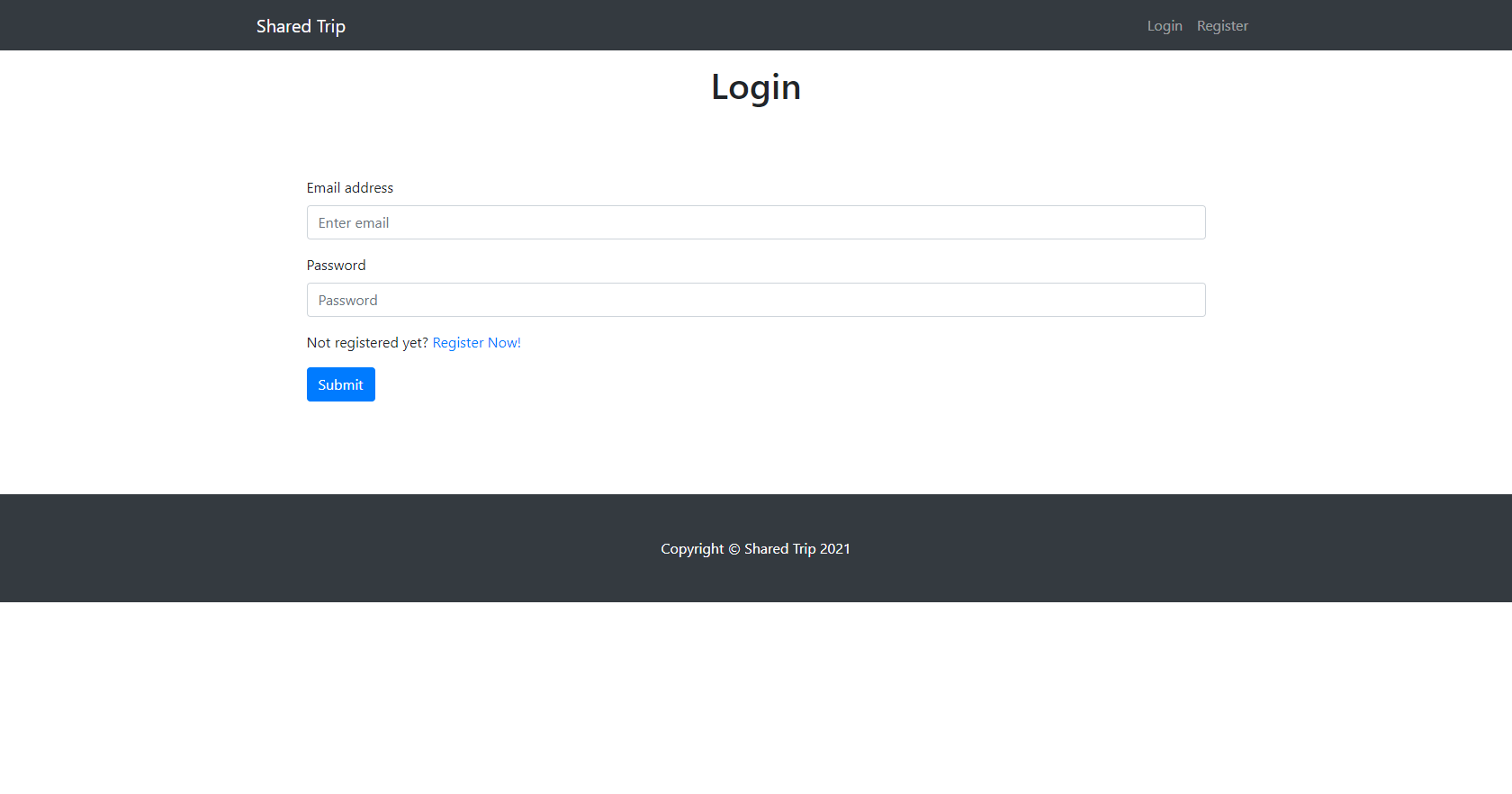
#### Register Page (logged out user)

Register a user inside the database with **email**, **password**. Both **passwords** must **match**! After successful registration **redirect to home page, with already logged in user**.



#### Login Page (logged out user)

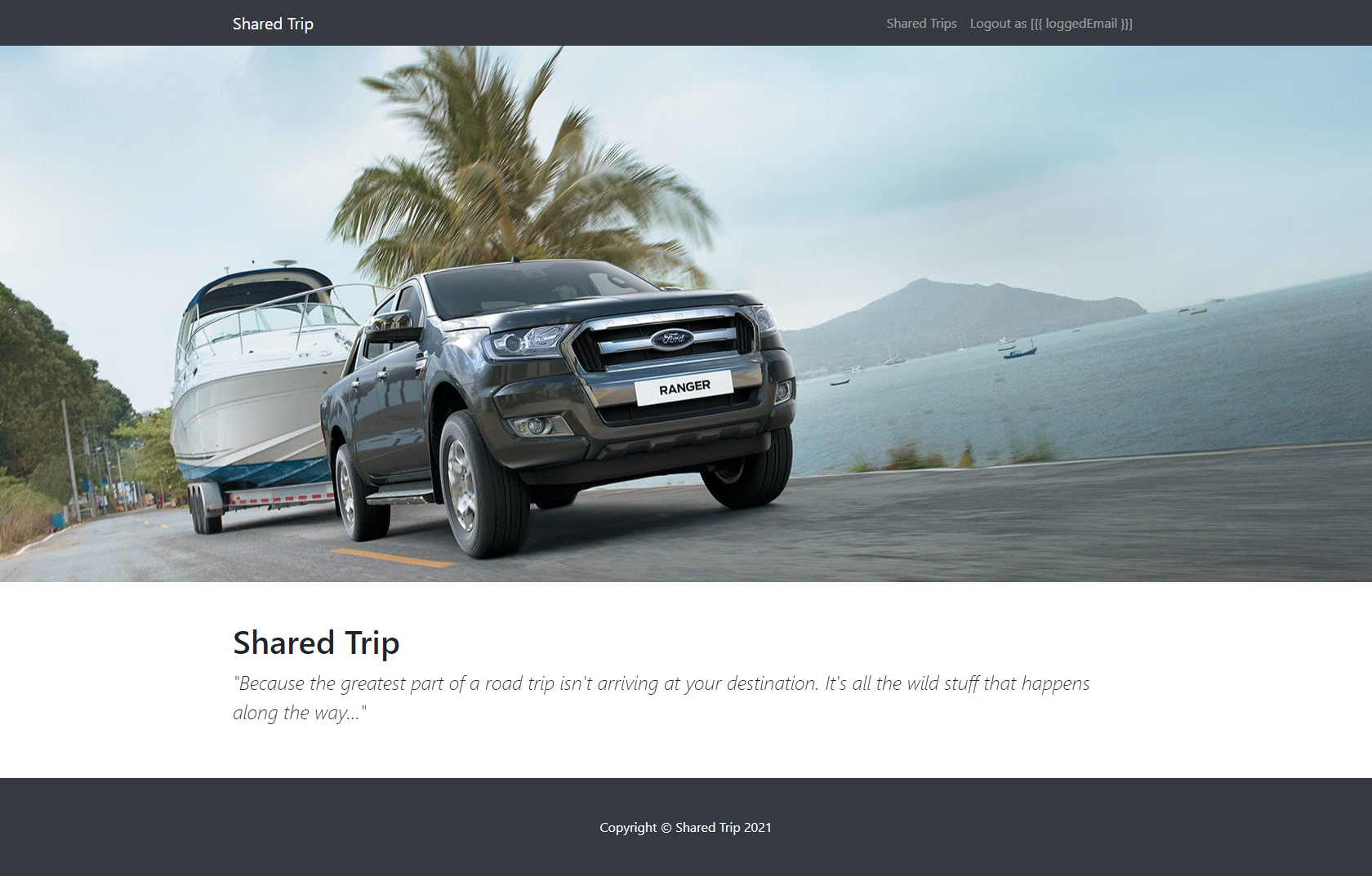
Logging an already registered user with correct email and password. After successful login **redirect to home page, with already logged in user**.



### User Pages (55 pts)

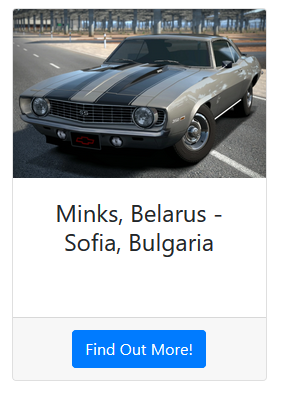
These are the **templates** and **functionalities**, accessible by Users (**logged in** users).

#### Home Page (logged in user)



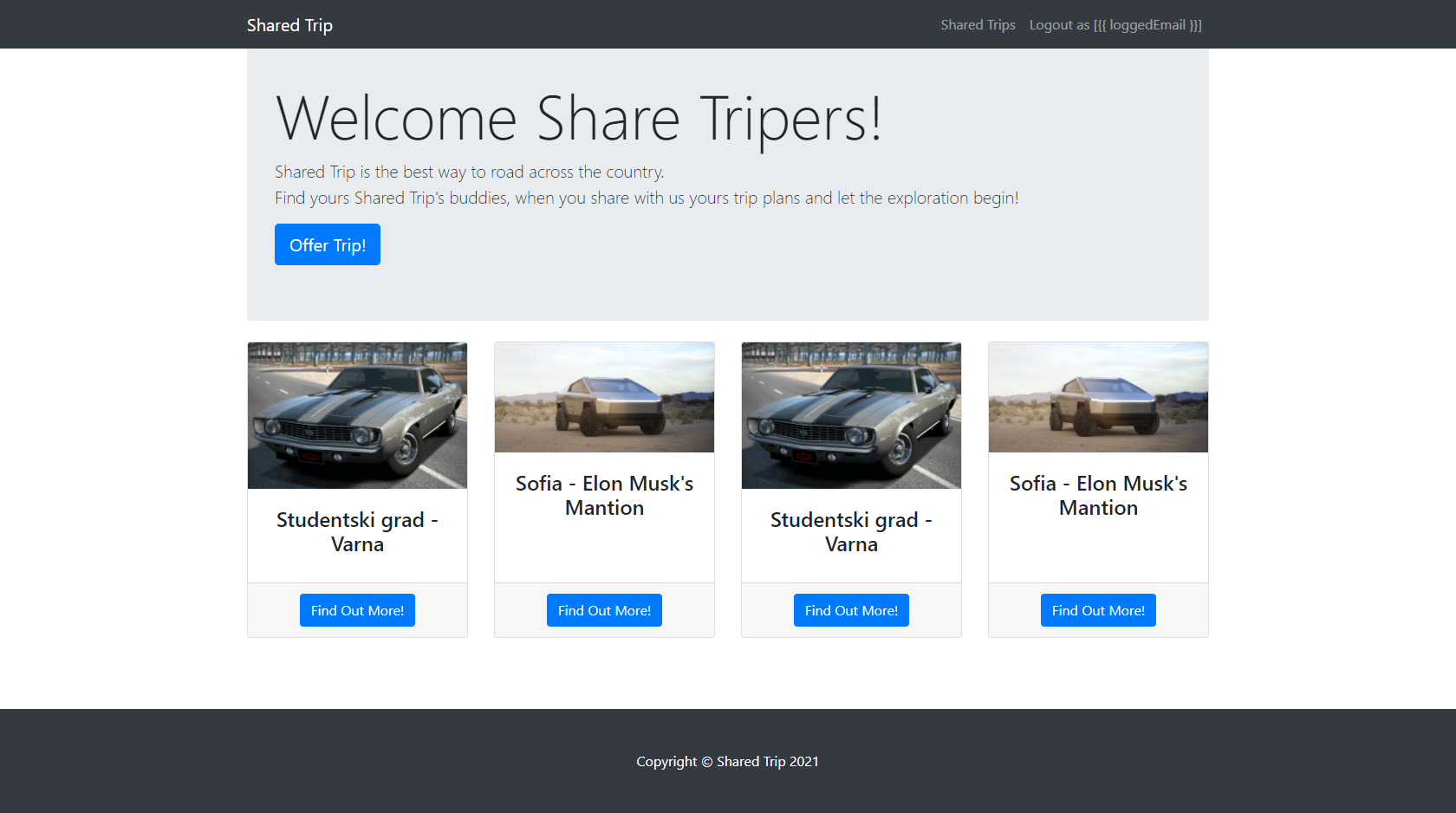
#### Shared Trips

List all **shared trips**. **Each trip** should be showing information about **trip's car image, destination** (**from** - **to**), Like in the picture below:

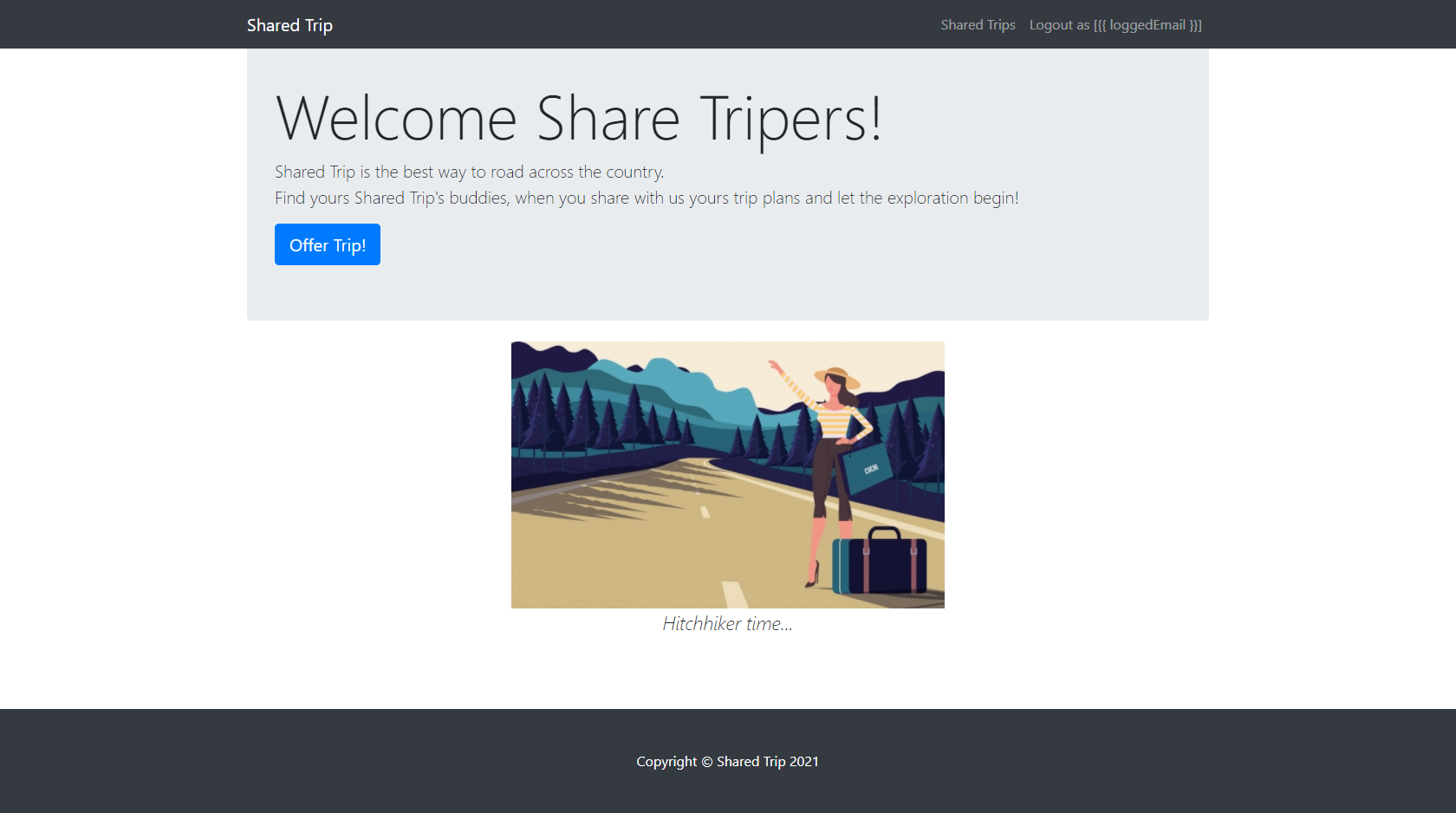


[**Find Out More!**] button should be a link to the **details** **page** for the current trip.

If **there are** shared trips in the **database**, display the following view:



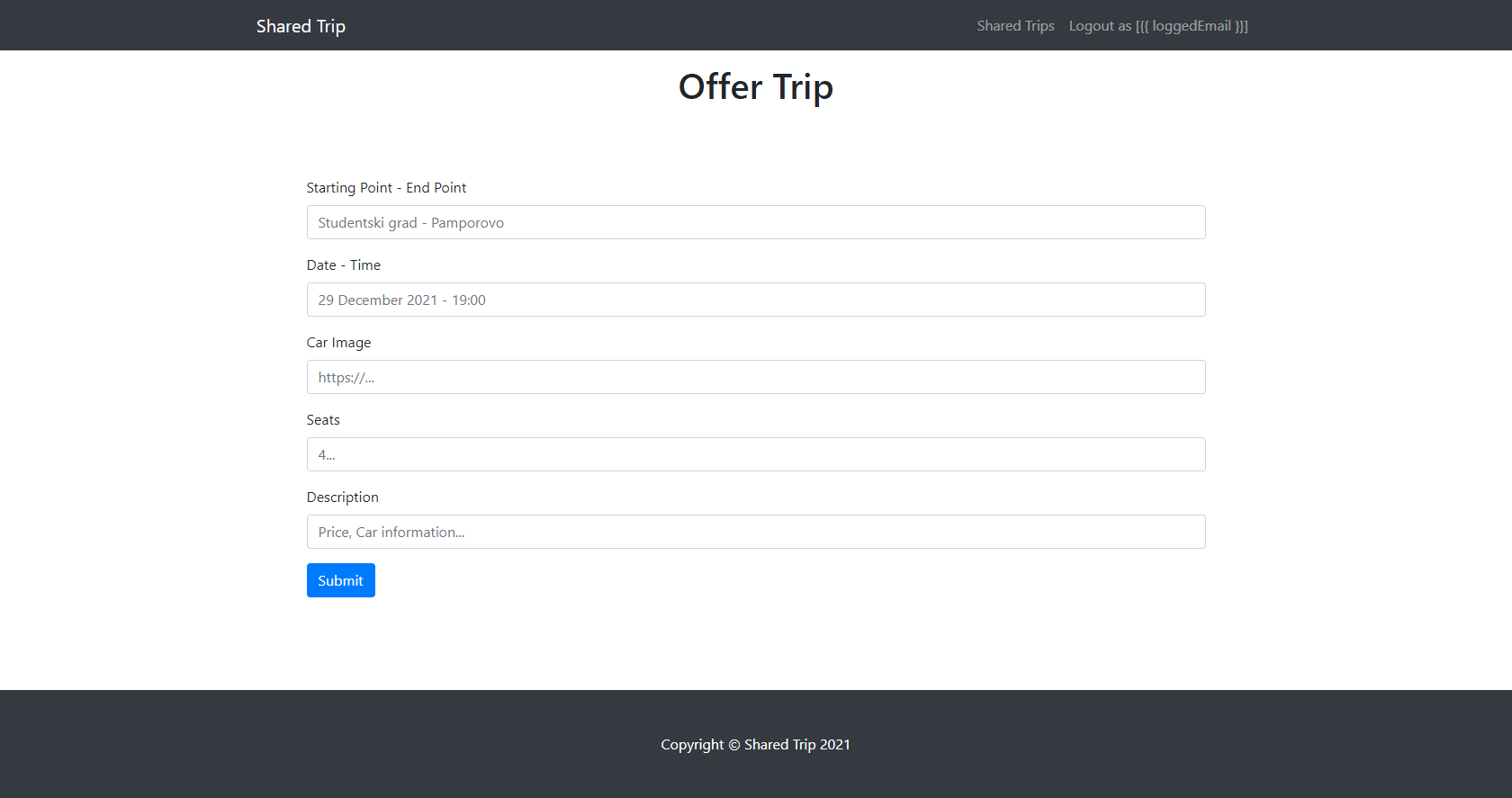
If **there are** **no** shared trips in the **database**, display the following view:



#### Offer New Trip Page (logged in user)

Enter **Trip**:

* **Destination -** in (**starting point - end point**) format
* **DateTime** -in (**date** **-** **time**) format
* **CarImage** - **http** or **https** **link** to the car
* **Seats** - **available** **seats** for this trip
* **Description** - A little description or something specific for the trip



#### Find Out More (Trip Details Page) - (logged in user)

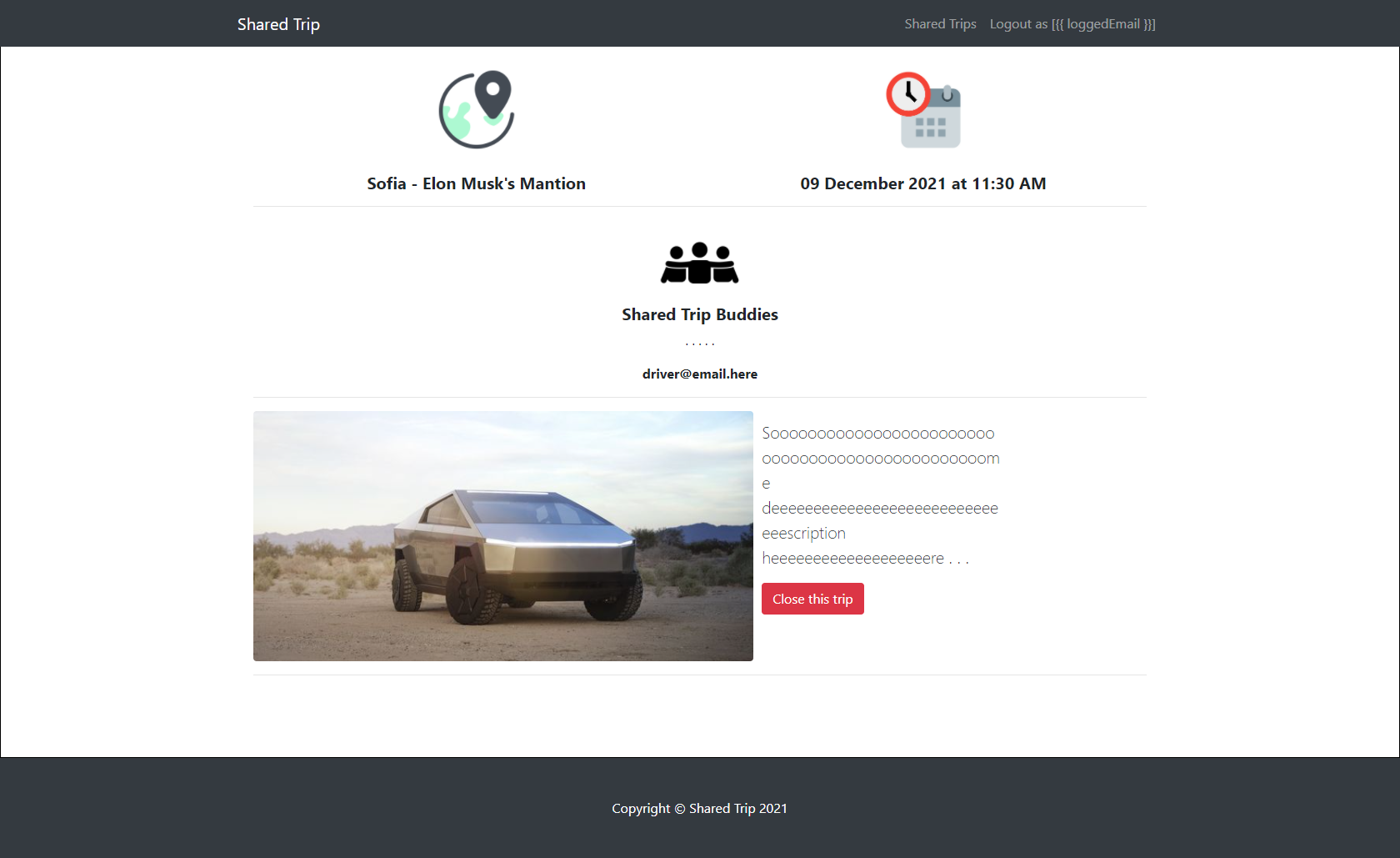
**The find out more** page is accessed by clicking over the [**Find Out More**] **button's** Shared Trip (if any) and there are shown the trip's:

* **Destination (from - to)**
* **Date and Time (date - time)**
* **Shared Trip Buddies**
  + If any, separate them with **space** and **comma** (", ")
  + If not, display 5 dots separated with space (**". . . . ."**)
  + Below that, the **email** **on** **the** **driver** should be displayed
* **Car Image**
* **Description**
* **Button** (Depending the status of the currently logged in user)

Check the pictures below for more clarity:

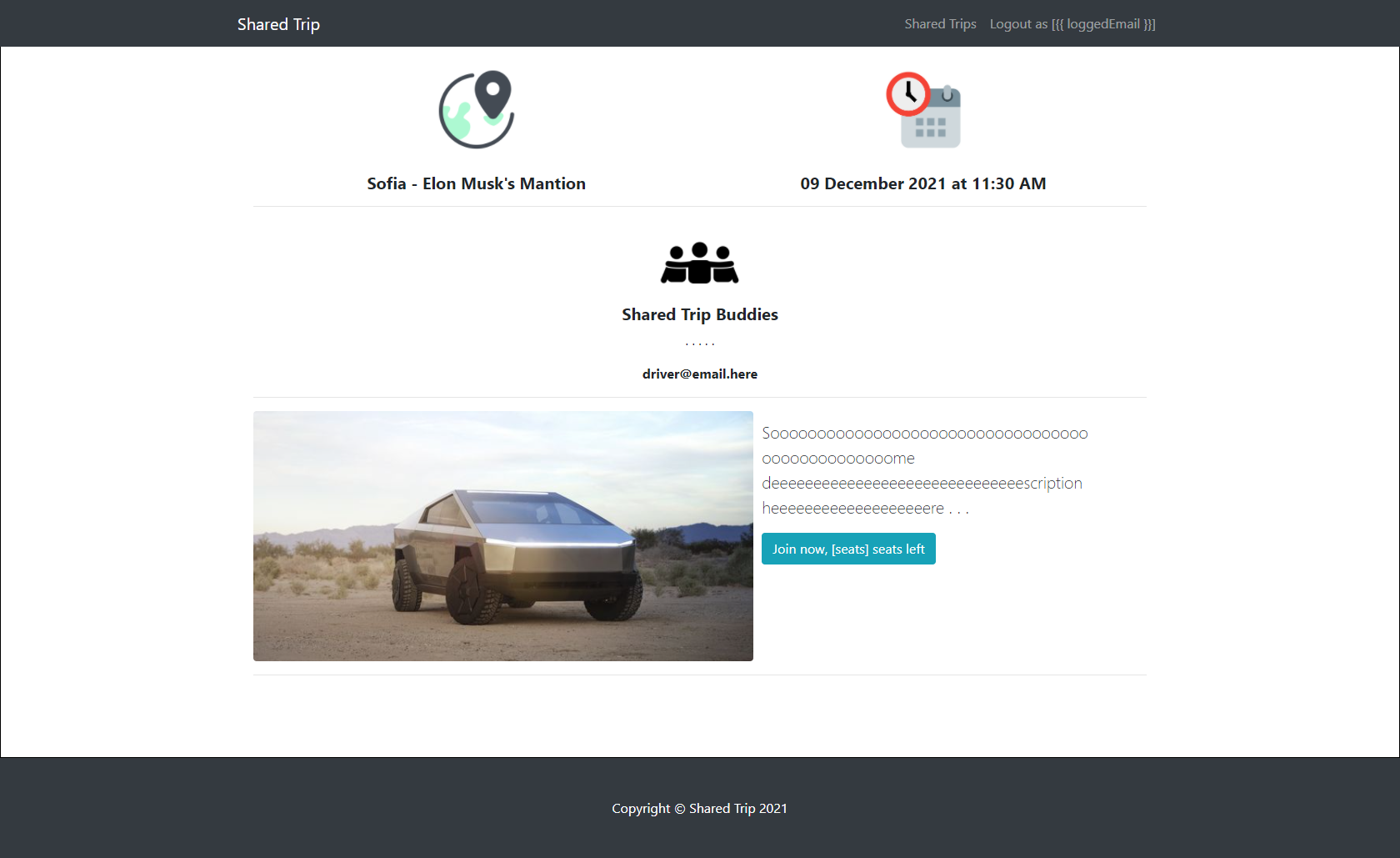
#### Shared Trip Details - Driver view

If the currently logged-in user is the driver (user who's **created** the trip), he should see the [**Close this trip**]which **deletes** the **current** **trip**.



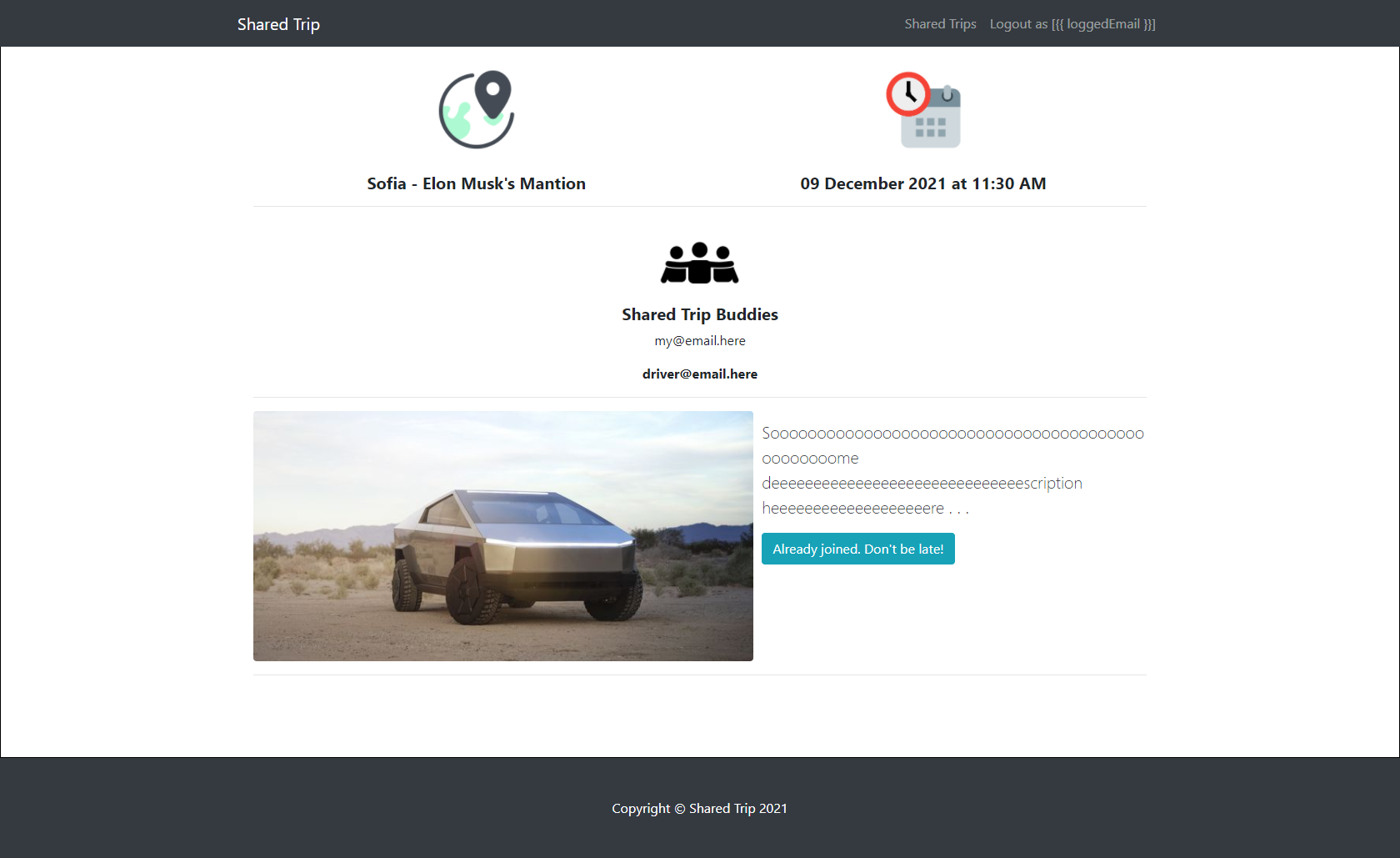
#### Shared Trip Details with available seats

If the currently logged-in user is **not the driver** (user that is not the creator for that trip) he should see button like [**Join now, {availableSeats} left**]. If there is at **least 1 seat left**



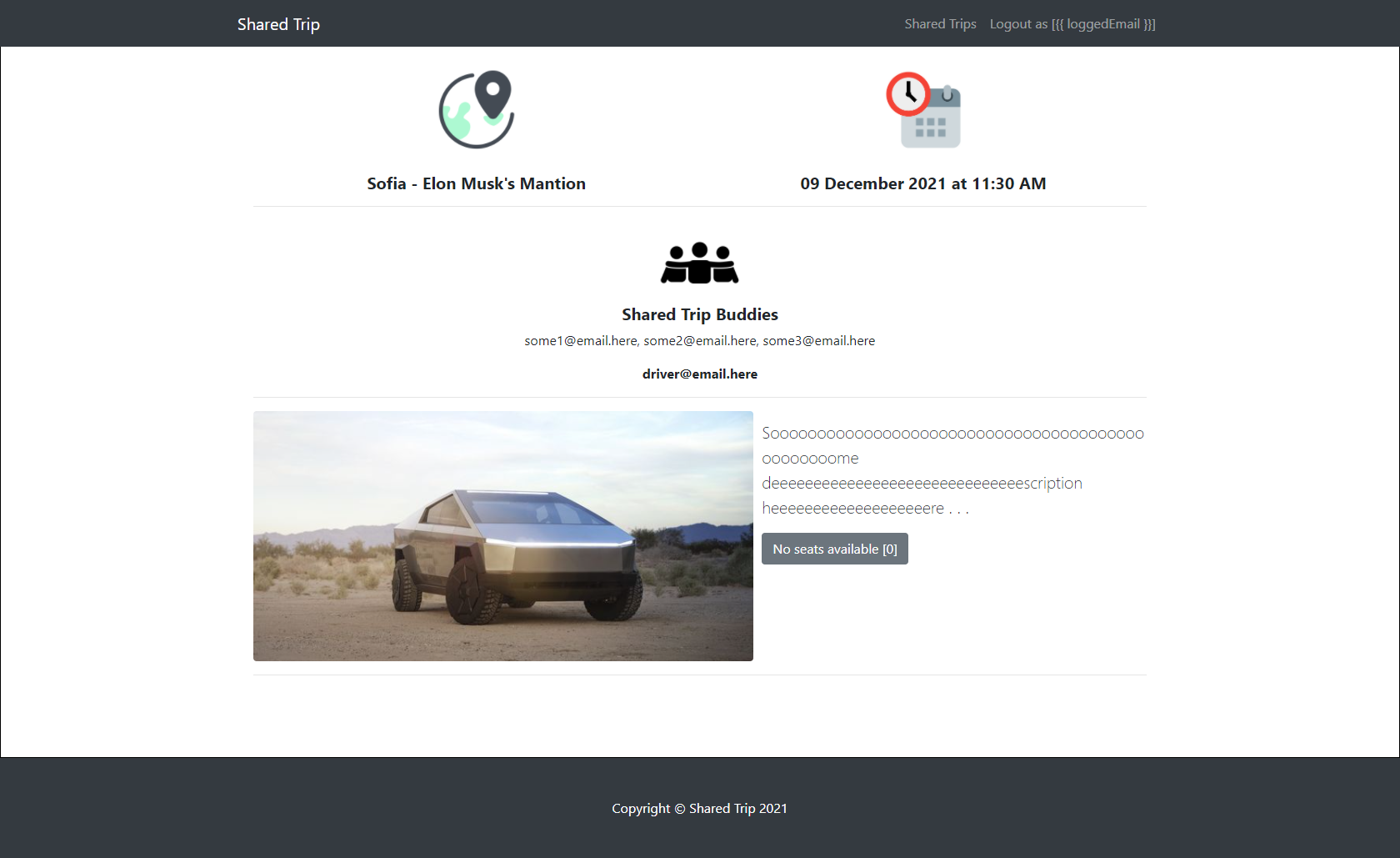
#### Shared Trip details - Already joined the trip

If the currently logged-in user is **already joined** some trip, he should see the [**Already joined. Don't be late**] button.



##### Shared Trip Details with no available seats

If the currently logged-in user is not **the driver** and there are **no available seats** for the current trip, he should see the [**No seats available [0]**] button.



#### Join Trip (logged in user)

**Every logged-in user** which is **not** the current **trip** **creator** (driver) should be able to join trip (**if seats available**). If he manages to **join successfully** to some trip, his **email** should be **added** to the current trips **shared Trip buddies**' collection and the **seats** for the current trip should be **decreased by 1**.

If some user once joined a current trip, he should see the "**already joined button"**.

If there are no seats available, he should see the "**no seats available button**".

#### Close Trip (logged in user)

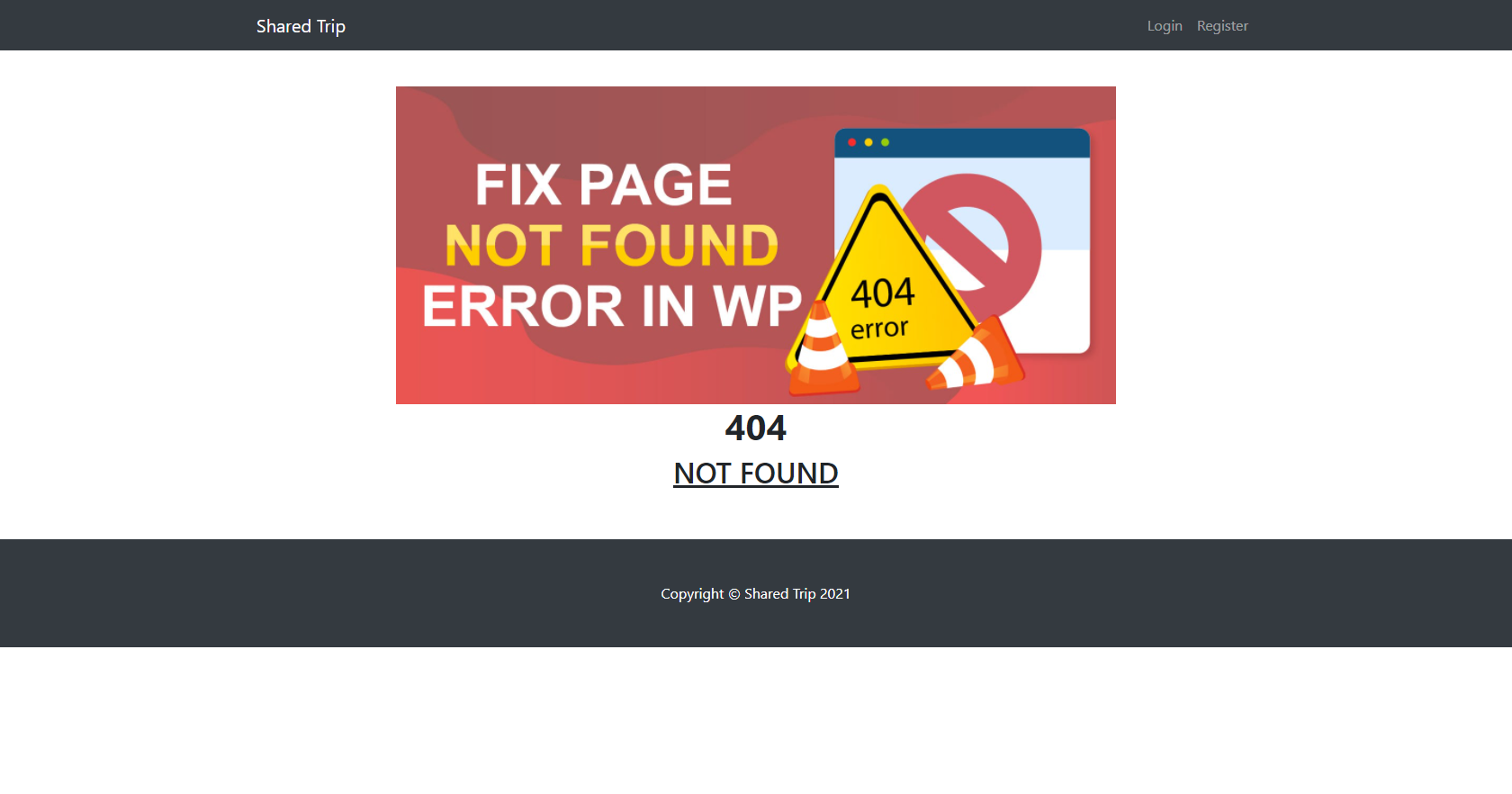
**Every trip creator** should be able to click over [**Close this trip**] **button** and deleting the current trip from the **database**.

## Security Requirements (10 pts)

The Security Requirements are mainly **access** requirements. Configurations about which users can access specific functionalities and pages.

* Guest (not logged in) users can access Home page (where users can register and login)
* Guest (not logged in) users can access Login page and functionality
* Guest (not logged in) users can access Register page and functionality
* Users (logged in) can access Shared Trips page (Listed all trips) page and functionality
* Users (logged in) can access **Offer Trip** page and functionality
* Users (logged in) can access **Find Out More** page and functionality
* Users (logged in) can acces to Join tripp and Delete trip (depending on the user authorization) functionality in the home page
* Users (logged in) can access Logout functionality

Use the following view for **invalid** **paths**:



## Validation and Error Handling (10 pts)

The application should notify the users about result of their actions.

### Login / Register

You should make the following validations:

* The **email** should be **in the following format** (mailboxname **@** domainname) - "**username@domain.bg**"
* The **password** should be **at least 4 characters** long
* The **repeat password** should be **equal to the password**



### Trip

You should make the following validations while **offering trip**:

* The **Starting Point - End Point** - **Starting** and **End** point should be **at least 4 characters long (each)** and should be separated with single space, dash and another single space (**" - "**).
* The **Date - Time - Date** and **Time** should be **at least 6 characters long** (**each**)and should be separated with single space, dash and another single space (**" - "**).
* The **Seats** should be positive number.
* The **description** should be minimum **10 characters** long.