Exercise: Bootstrap

IRunes

Problems for exercises and homework for the ["C# Web Basics" course @ SoftUni](). Yoy can submit your solution in the course web page.

In the previous exercise you should have implemented the **IRunes application** – a simple **music store application**. Due to the fact that **exceptional design** was **not required**, the pages, which the application supported, were implemented – using only HTML. Thus, they looked like something implemented 50 years ago – ugly and simple.

But this time, the case is different... We now have a powerful friend to our side – **Bootstrap**. Let's use it to make the pages more beautiful.

Database Requirements

You have been tasked to implement a simple application, using the Web Server. The application imitates a **store** for **Music Albums** and **Music Tracks**. You will see the functionality – described below.

The first thing you need to do is implement the Database entities. Use Entity Framework Core, and implement the following entities:

User

- **`Id`** – a **string** (**GuID**).
- **`Username`** – a **string**.
- **`Password`** – a **string** (**encoded** in the database).
- **`Email`** – a **string**.

Album

- **`Id`** – a **string** (**GuID**).
- **`Name`** – a **string**.
- **`Cover`** – a **string** (a **link** to an **image**).
- **`Price`** – a **decimal** (**sum** of all **`Tracks`**' **prices**, **reduced** by **13%**).
- **`Tracks`** – a **collection** of **`Tracks`**.

Track

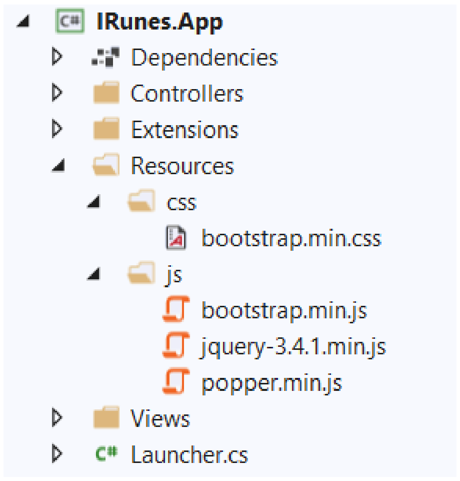
- **`Id`** – a **string** (**GuID**).
- **`Name`** – a **string**.
- **`Link`** – a **string** (a **link** to a **video**).
- **`Price`** – a **decimal**.

Initial Configuration

But is our Server capable of handling resource requests? Because requesting bootstrap.css is a resource request, which must be handled differently. Well, let's configure it to do so.

1. Resources Folder

The **`Resources`** Folder will be in the **`Application Project`**. Each application has its own resources.



```
IRunes.App
    Dependencies
    Controllers
    Extensions
    Resources
        css
            bootstrap.min.css
        js
            bootstrap.min.js
            jquery-3.4.1.min.js
            popper.min.js
    Views
    Launcher.cs
```

Create the **folder** and **divide** the **different resources**. (**download** the files shown above, locally, from their respective sites). **NOTE**: There is **no particular meaning** to the resource files being separated into folders named as the resources' extensions, it's just well-structured resource folder.

2. Inline Resource Result

First, we need to create a **`Response`**, which will be returned with the Resource File Contents.

Create a class named **`InlineResourceResult`**, which we will use for that purpose. It should look like this:

```
public class InlineResourceResult : HttpResponse
{
    public InlineResourceResult(byte[] content, HttpResponseStatusCode responseStatusCode)
        : base(responseStatusCode)
    {
        this.Headers.Add(new HttpHeader(HttpHeader.ContentLength, content.Length.ToString()));
        this.Headers.Add(new HttpHeader(HttpHeader.ContentDisposition, "inline"));
        this.Content = content;
    }
}
```

As you can see the **`content`** given in the constructor is **`byte[]`**. That's because the resource might be a media file. For example: **`favicon.ico`**.

We also set the **`Content-Length`** and **`Content-Disposition`** Headers. The **`Content-Disposition`** is **`inline`**, as this particular **`Result`** class is used to **return** only **inline resource**, which are used in the web pages.

NOTE: It is not quite good to depend on the **Client's Browser** to set the **`Content-Type`** header for you, but for now we will. (Will be fixed in the future).

3. ConnectionHandler

The other class we need to change is the **`ConnectionHandler`**. Particularly, the way **`Requests`** are **Handled**. Modify the **`HandlerRequest()`** method to look like this:

```
private IHttpResponse HandleRequest(IHttpRequest httpRequest)
{
    if (this.serverRoutingTable.Contains(httpRequest.RequestMethod, httpRequest.Path))
    {
        return this.ReturnIfResource(httpRequest.Path);
    }

    return this.serverRoutingTable
        .Get(httpRequest.RequestMethod,httpRequest.Path)
        .Invoke(httpRequest);
}
```

The **`ReturnIfResource()`** method should check for a **resource** with the **given name** (in the path of the Request, for example: **`/css/bootstrap.min.css`**), in the **`Resources`** folder of the Application.

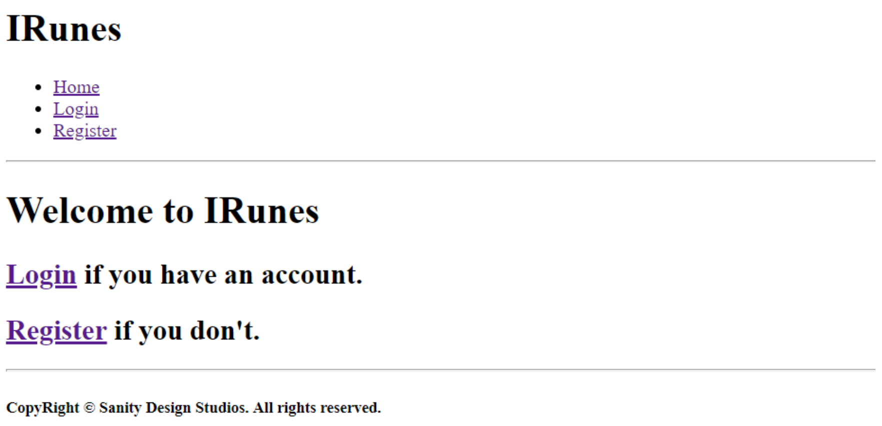
- If there is, an **`InlineResourceResponse`** should be returned, with the **`Resource File`**'s contents.
- If there isn't, a **Not Found `HttpResponse`** should be returned.

However, the method's **implementation** is up to **you**. 😊

Templates

You will be shown how the **template looked before**, and **how it should look now**. Style it **using Bootstrap**, to match the given screenshot. Be as precise as you can.

4. Index (guest, logged-out) (route="/Home/Index", route="/")



IRunes

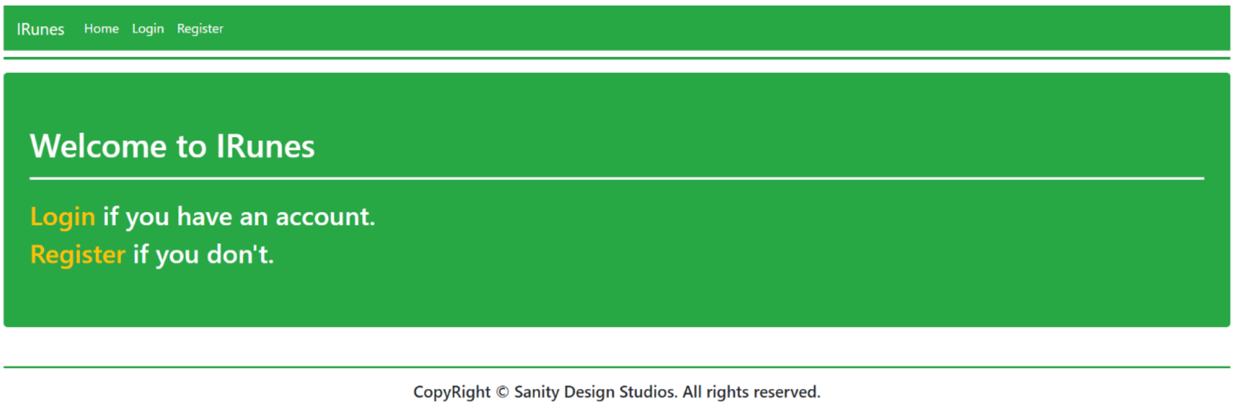
- Home
- Login
- Register

Welcome to IRunes

Login if you have an account.

Register if you don't.

CopyRight © Sanity Design Studios. All rights reserved.



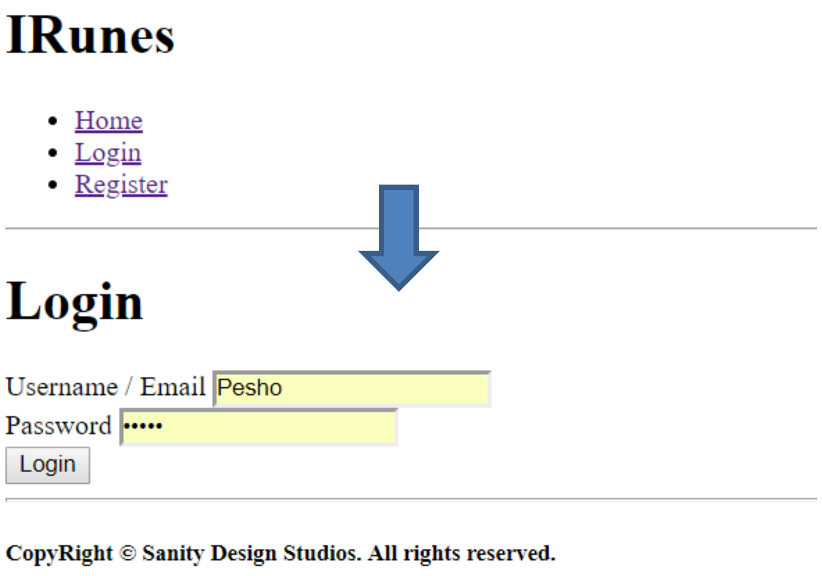
IRunes Home Login Register

Welcome to IRunes

Login if you have an account.
Register if you don't.

CopyRight © Sanity Design Studios. All rights reserved.

5. Login (guest, logged-out) (route="/Users/Login")



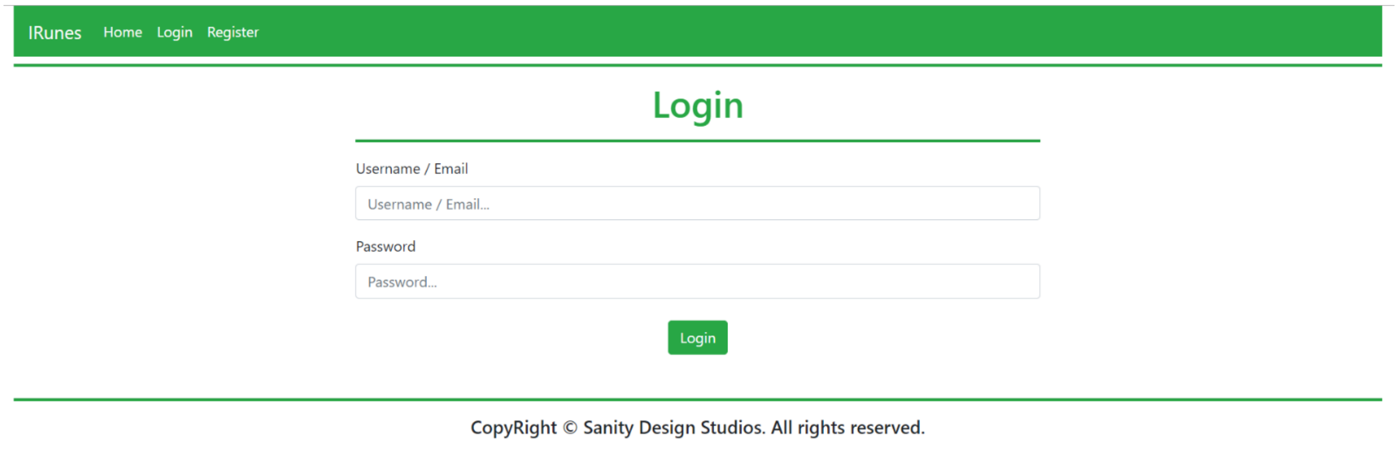
IRunes

- Home
- Login
- Register

Login

Username / Email Pesho
Password •••••
Login

CopyRight © Sanity Design Studios. All rights reserved.



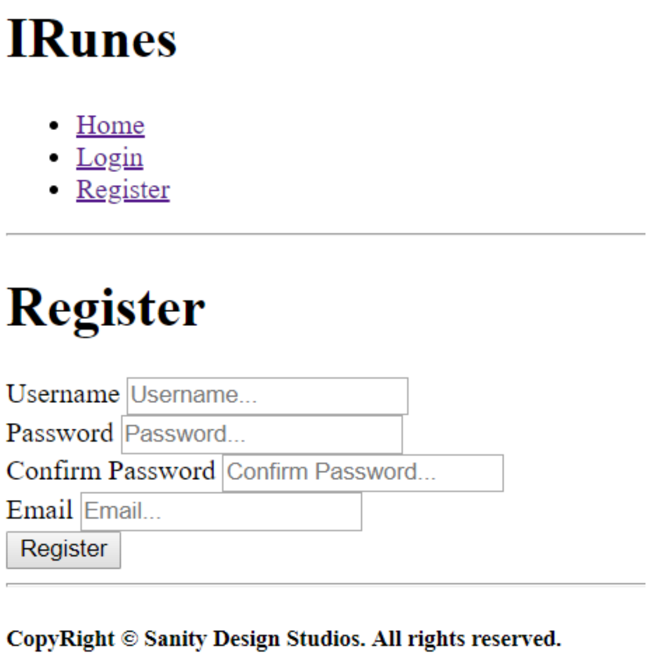
IRunes Home Login Register

Login

Username / Email
Username / Email...
Password
Password...
Login

CopyRight © Sanity Design Studios. All rights reserved.

6. Register (guest, logged-out) (route="/Users/Register")



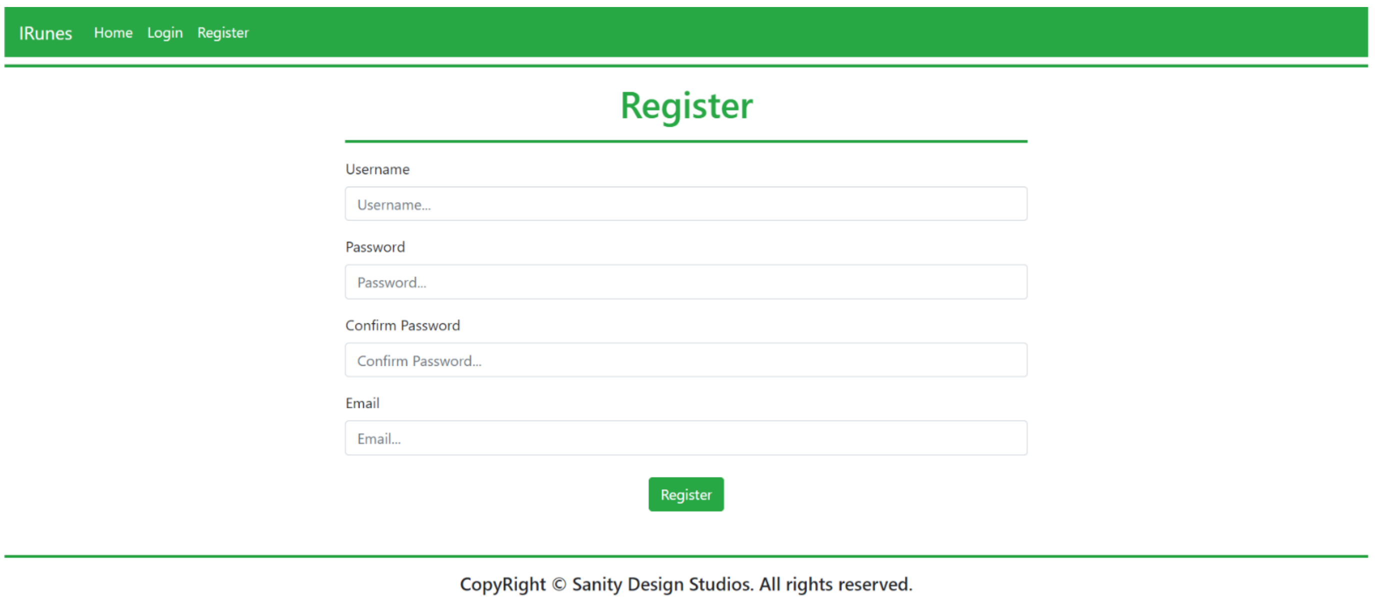
IRunes

- Home
- Login
- Register

Register

Username [Username...]
Password [Password...]
Confirm Password [Confirm Password...]
Email [Email...]
Register

CopyRight © Sanity Design Studios. All rights reserved.



IRunes Home Login Register

Register

Username [Username...]

Password [Password...]

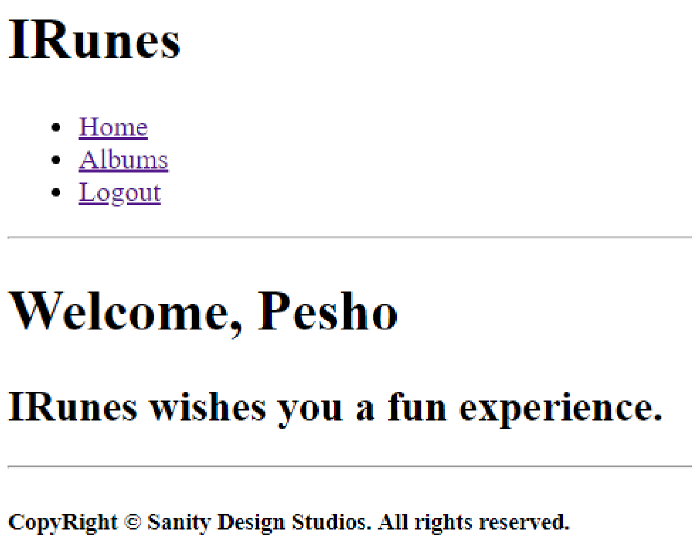
Confirm Password [Confirm Password...]

Email [Email...]

Register

CopyRight © Sanity Design Studios. All rights reserved.

7. Index (user, logged-in) (route="/Home/Index", route="/")



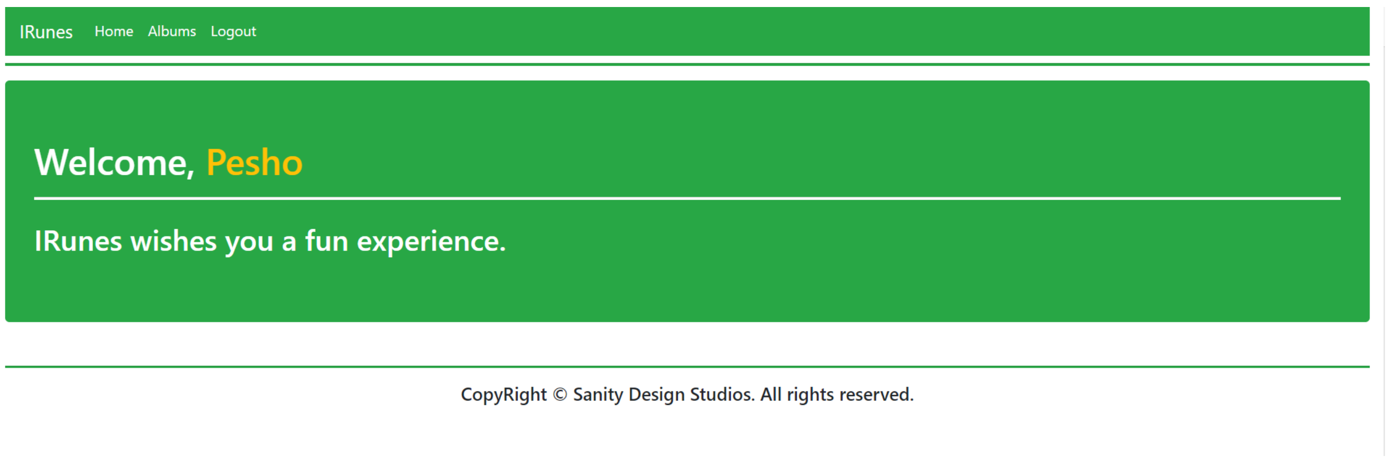
IRunes

- Home
- Albums
- Logout

Welcome, Pesho

IRunes wishes you a fun experience.

CopyRight © Sanity Design Studios. All rights reserved.



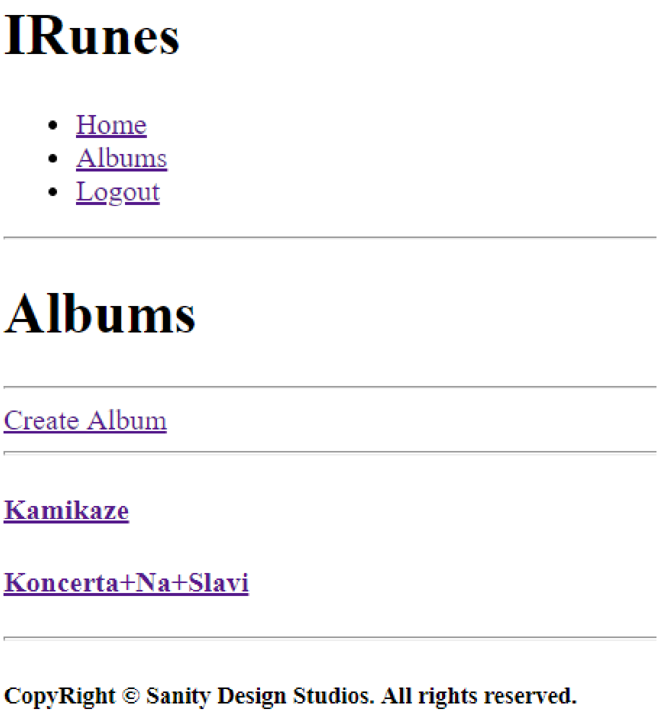
IRunes Home Albums Logout

Welcome, Pesho

IRunes wishes you a fun experience.

CopyRight © Sanity Design Studios. All rights reserved.

8. All Albums (user, logged-in) (route="/Albums/All")



IRunes

- Home
- Albums
- Logout

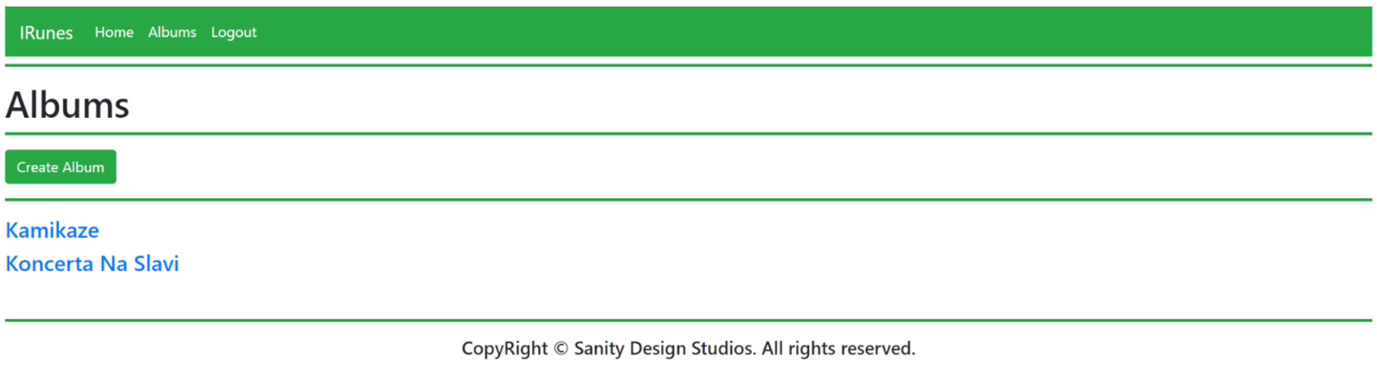
Albums

Create Album

Kamikaze

Koncerta+Na+Slavi

CopyRight © Sanity Design Studios. All rights reserved.



IRunes Home Albums Logout

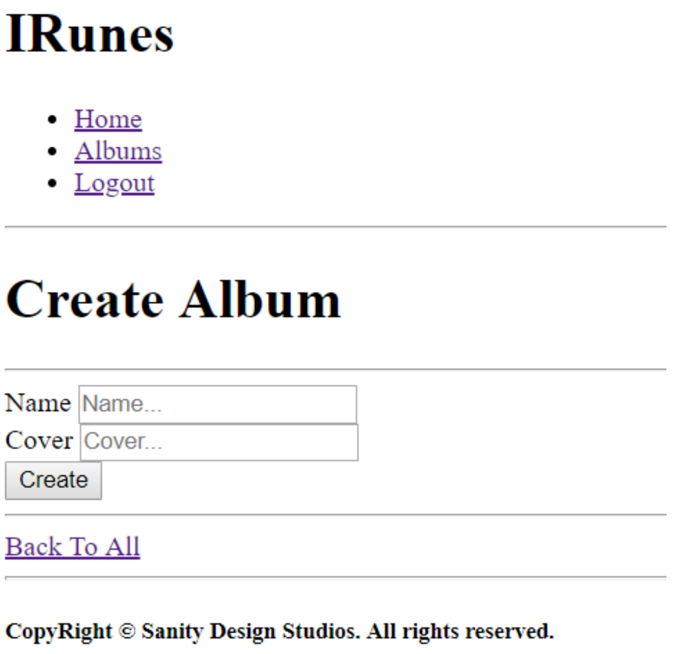
Albums

Create Album

Kamikaze
Koncerta Na Slavi

CopyRight © Sanity Design Studios. All rights reserved.

9. Album Create (user, logged-in) (route="/Albums/Create")



IRunes

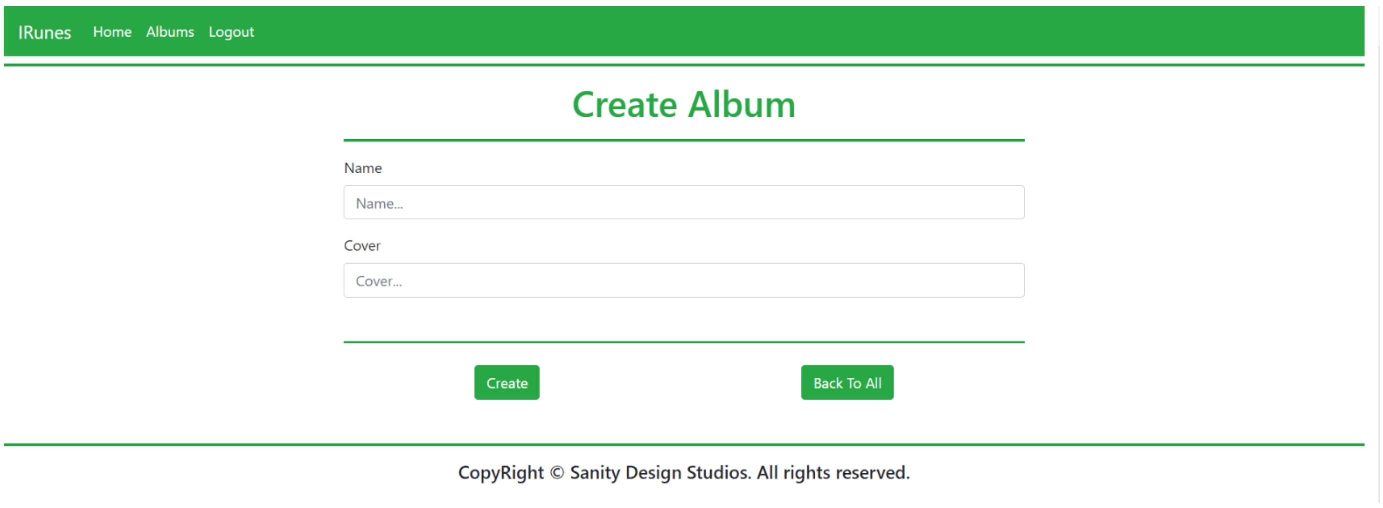
- Home
- Albums
- Logout

Create Album

Name Name...
Cover Cover...
Create

Back To All

CopyRight © Sanity Design Studios. All rights reserved.



IRunes Home Albums Logout

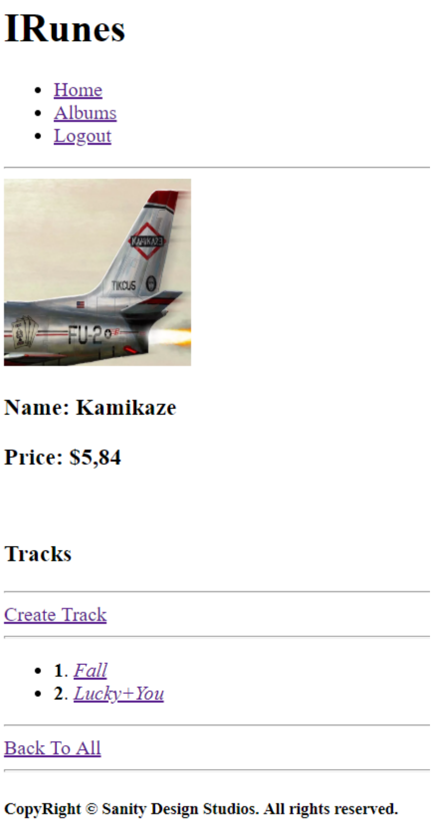
Create Album

Name
Name...
Cover
Cover...

Create Back To All

CopyRight © Sanity Design Studios. All rights reserved.

10. Album Details (user, logged-in) (route="/Albums/Details?id={albumId}")


IRunes

- Home
- Albums
- Logout

Name: Kamikaze

Price: $5,84

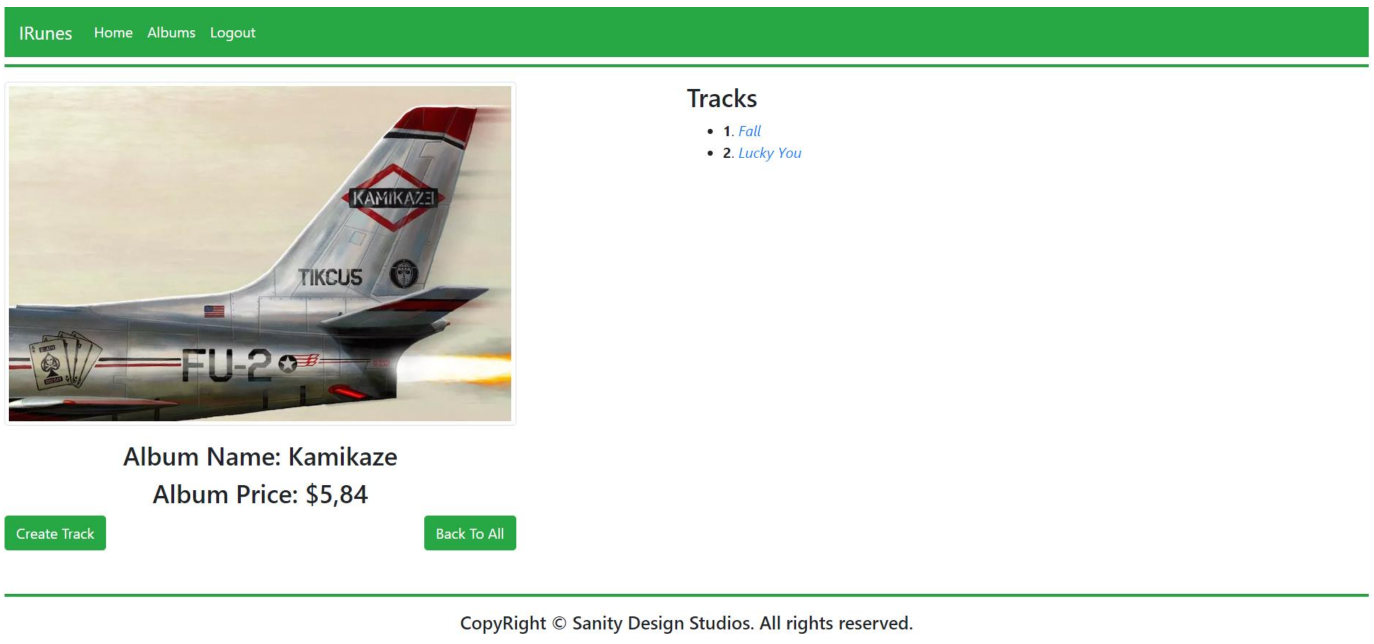
Tracks

Create Track

- 1. *Fall*
- 2. *Lucky+You*

Back To All

CopyRight © Sanity Design Studios. All rights reserved.


IRunes Home Albums Logout

Album Name: Kamikaze

Album Price: $5,84

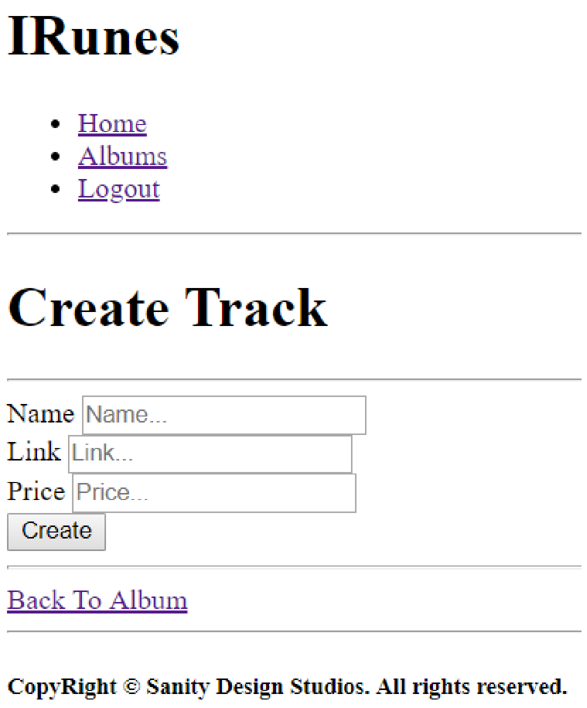
Create Track Back To All

Tracks

- 1. *Fall*
- 2. *Lucky You*

CopyRight © Sanity Design Studios. All rights reserved.

11. Track Create (user, logged-in) (route="/Tracks/Create?albumId={albumId}")



IRunes

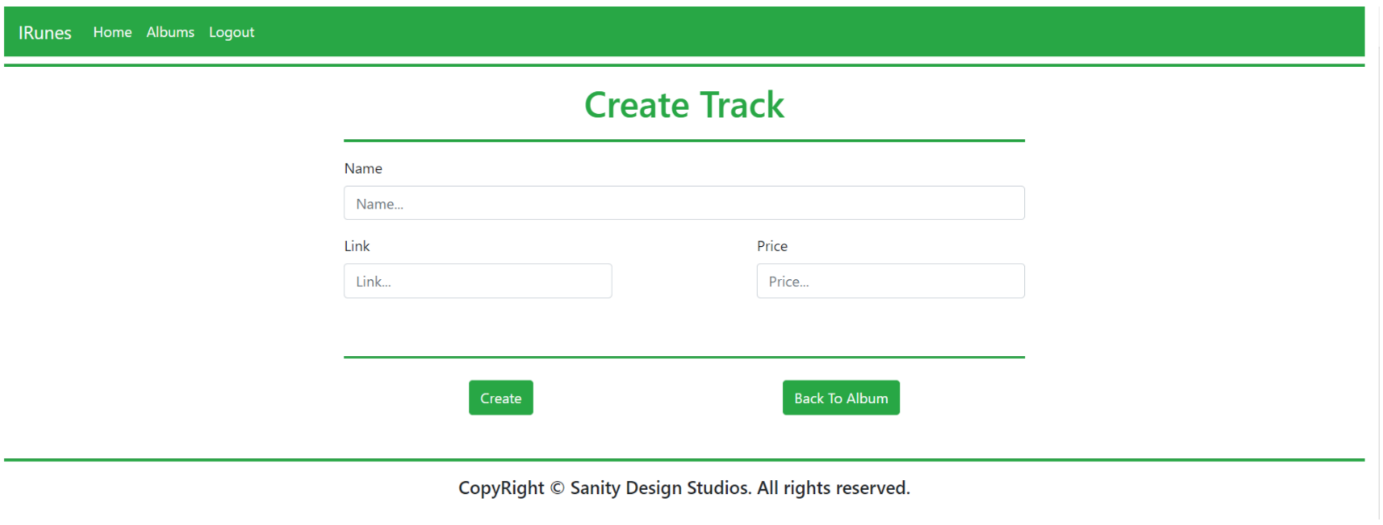
- Home
- Albums
- Logout

Create Track

Name Name...
Link Link...
Price Price...
Create

Back To Album

CopyRight © Sanity Design Studios. All rights reserved.



IRunes Home Albums Logout

Create Track

Name
Name...

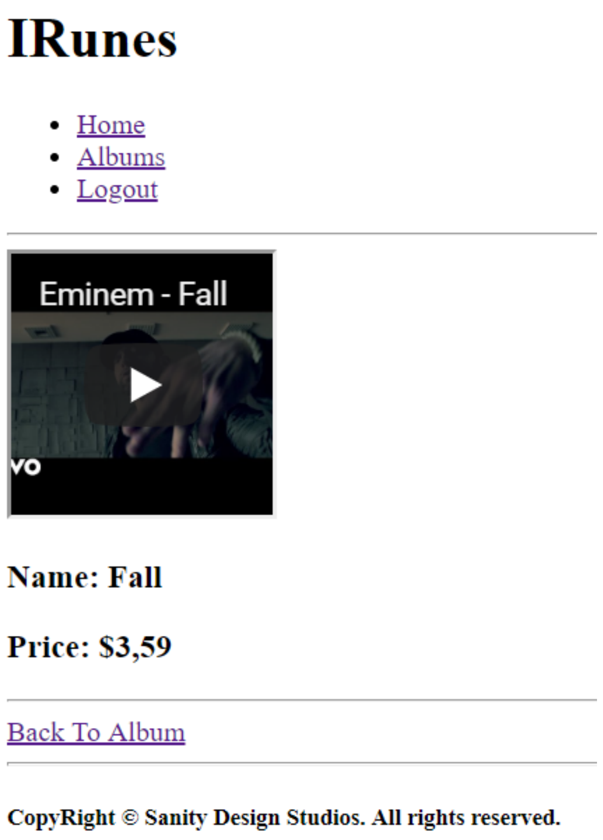
Link
Link...

Price
Price...

Create
Back To Album

CopyRight © Sanity Design Studios. All rights reserved.

12. Track Details (user, logged-in) (route="/Tracks/Details?albumId={albumId}&trackId={trackId}")



IRunes

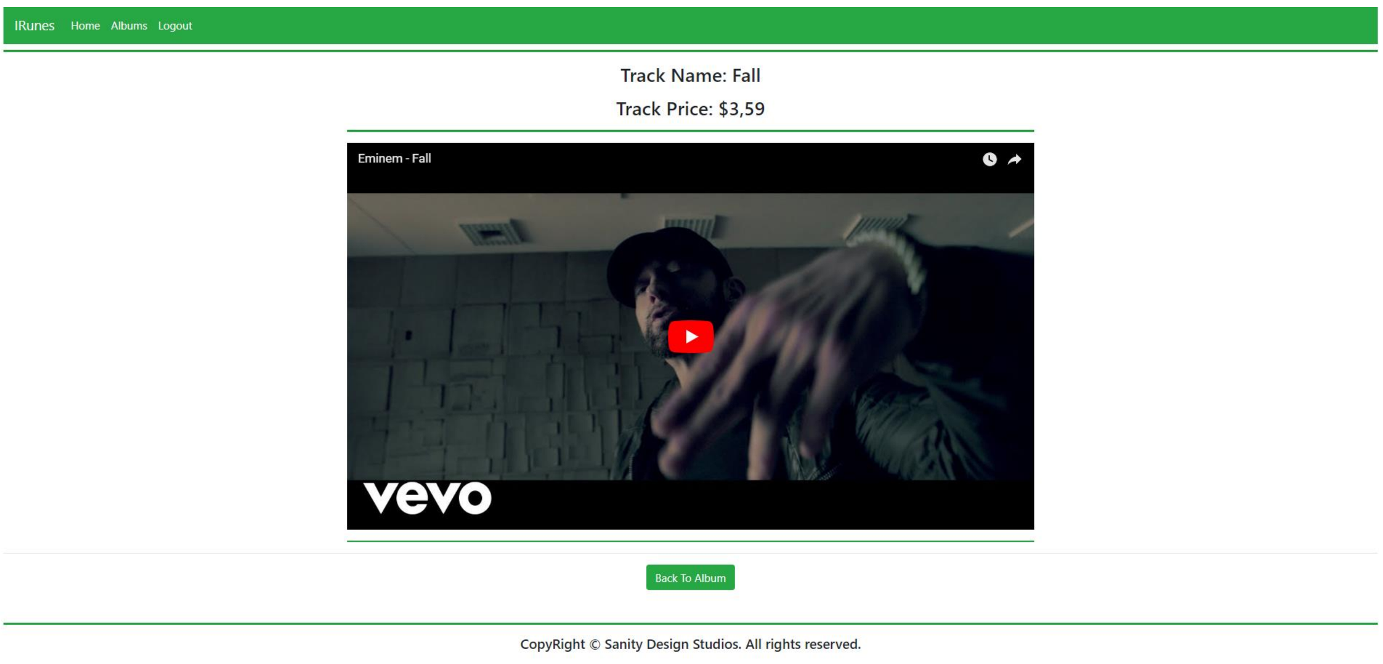
- Home
- Albums
- Logout

Name: Fall

Price: $3,59

Back To Album

CopyRight © Sanity Design Studios. All rights reserved.



Track Name: Fall

Track Price: $3,59

CopyRight © Sanity Design Studios. All rights reserved.

13. Hints

If you don't know how to do something, just search for it on the internet. The **Bootstrap Documentation** and **Community** are quite helpful.

Bootstrap does not provide style for Horizontal lines (**<hr />**). To implement the lines shown in the screenshots above with the appropriate height. Just use the following way (**<hr style="height: 2px"**)

Functional Requirements

The functional requirements are quite simple.

Users

The application should provide **`guests`** (logged-out) with the **functionality** to access:

- The **`Guest Index Page`**
- The **`Login Page`** and **`Functionality`**
- The **`Register Page`** and **`Functionality`**

The application should provide **`users`** (logged-in) with the **functionality** to access:

- The **`User Index Page`**
- The **`All Albums Page`** and **`Functionality`**
- The **`Album Create Page`** and **`Functionality`**
- The **`Album Details Page`** and **`Functionality`**
- The **`Track Create Page`** and **`Functionality`**
- The **`Track Details Page`** and **`Functionality`**

Albums

The **`Albums`** are **created** and **presented** on the **`All Albums Page`**, in a **listed format** with only their **`names`** as elements. Each **album name** should be a **link** which **leads** to the corresponding **`Album`**'s **`Details Page`**.

If there **are no Albums currently** in the Database, a message "**`There are currently no albums.`**" should be printed.

On the **`Album`**'s **`Details Page`**, its tracks should be **listed**, in an **indexed list**, **starting** from **1**. The **order** of **data** is **not mandatory**.

Each **track name** should be a link which leads to the corresponding **`Track`**'s **`Details Page`**.

Tracks

The **`Tracks`** are **created** and **presented** on their **`Album's Details Page`**. **`Tracks`** are created using the **`Album`**'s **`id`** which is passed through the **query parameters**.

When you **create** a **`Track`**, you can pass it the **iframe-ready url**, in order to make your work easier.