# C# Web Development Basics – Regular Exam

# Musaca

**MUSACA** (**M**ulti-**U**ser **S**elling and **A**ccounting **C**ashier **A**pplication) is a web platform application for cashiers at selling points. You know how a cashier at a supermarket checks in your products and then tells you how much your cost is, well this one is the same, but kind of like a web app. Anyway, you’ve been employed by the **SB** (**S**lavic **B**abushka) group to implement the **MUSACA** platform for a client of theirs, which is a really big hypermarket company.

However, there are specific requirements that must be followed.

## Technological Requirements

* Use the SIS.WebServer
* Use the SIS.Framework or the Niki.Framework
* Use Entity Framework Core

The Technological Requirements are **ABSOLUTE**. If you **do not follow** them, you will **NOT** be scored for other Requirements.

Now that you know the Technological Requirements, let us see what are the Business Requirements.

## Database Requirements

The **Database** of the MUSACA application needs to support **3 entities**:

### User – DONE!!!

* Has an Id – a GUID String or an **Integer**.
* Has an Username
* Has a Password
* Has an Email
* Has an Role – can be one of the following values (“User”, “Admin”)

### Product

* Has an Id – a GUID String or an **Integer**.
* Has a Name – a string.
* Has a Price – a decimal.
* Has a Barcode – a 12-digit long integer.
* Has a Picture – a string.

### Order

* Has an Id – a GUID String or an **Integer**.
* Has a Status – can be one of the following values ("Active", "Completed").
* Has a Product – an Product object.
* Has a Quantity – an integer.
* Has a Cashier – a User object.

### Receipt

* Has an Id – a GUID String.
* Has a Issued On – a DateTime object.
* Has a Orders – a collection of Order objects.
* Has a Cashier – a User object.

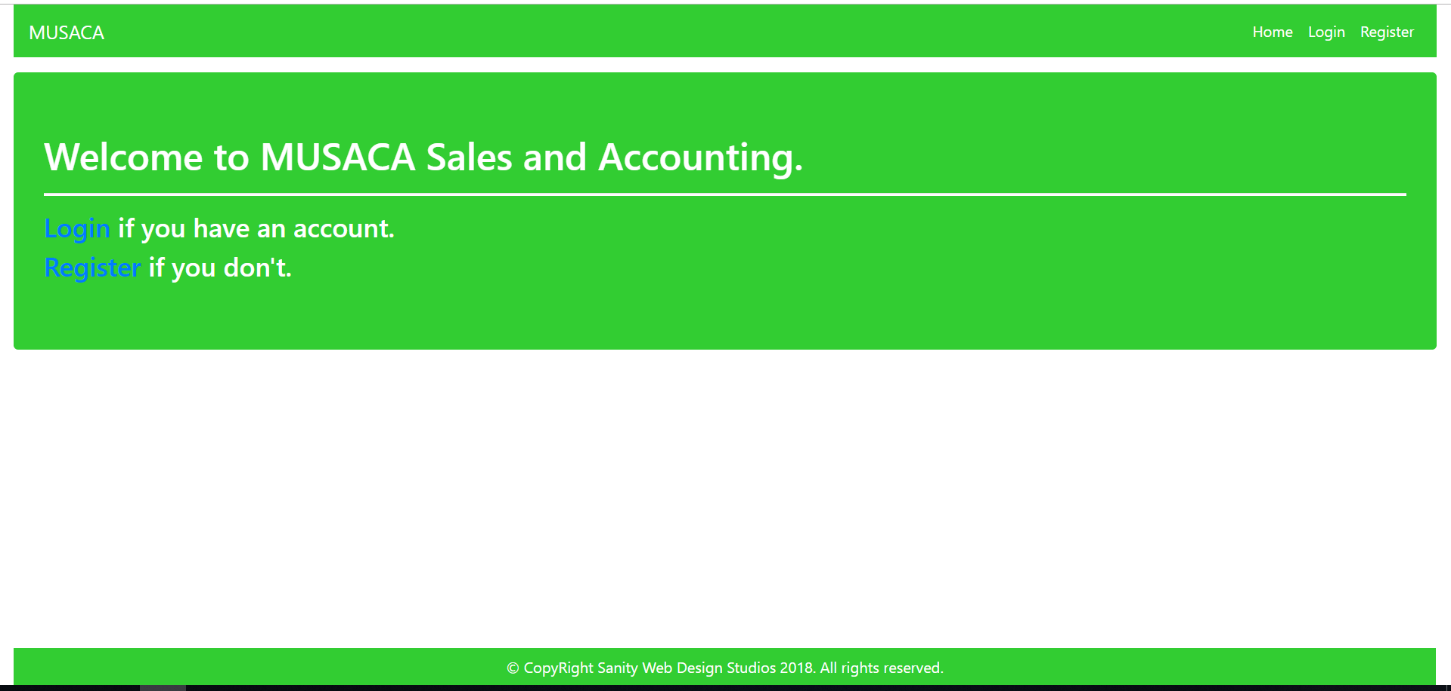
Implement the entities with the **correct datatypes**.

## Template Requirements

### Guest Templates – DONE!!!

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

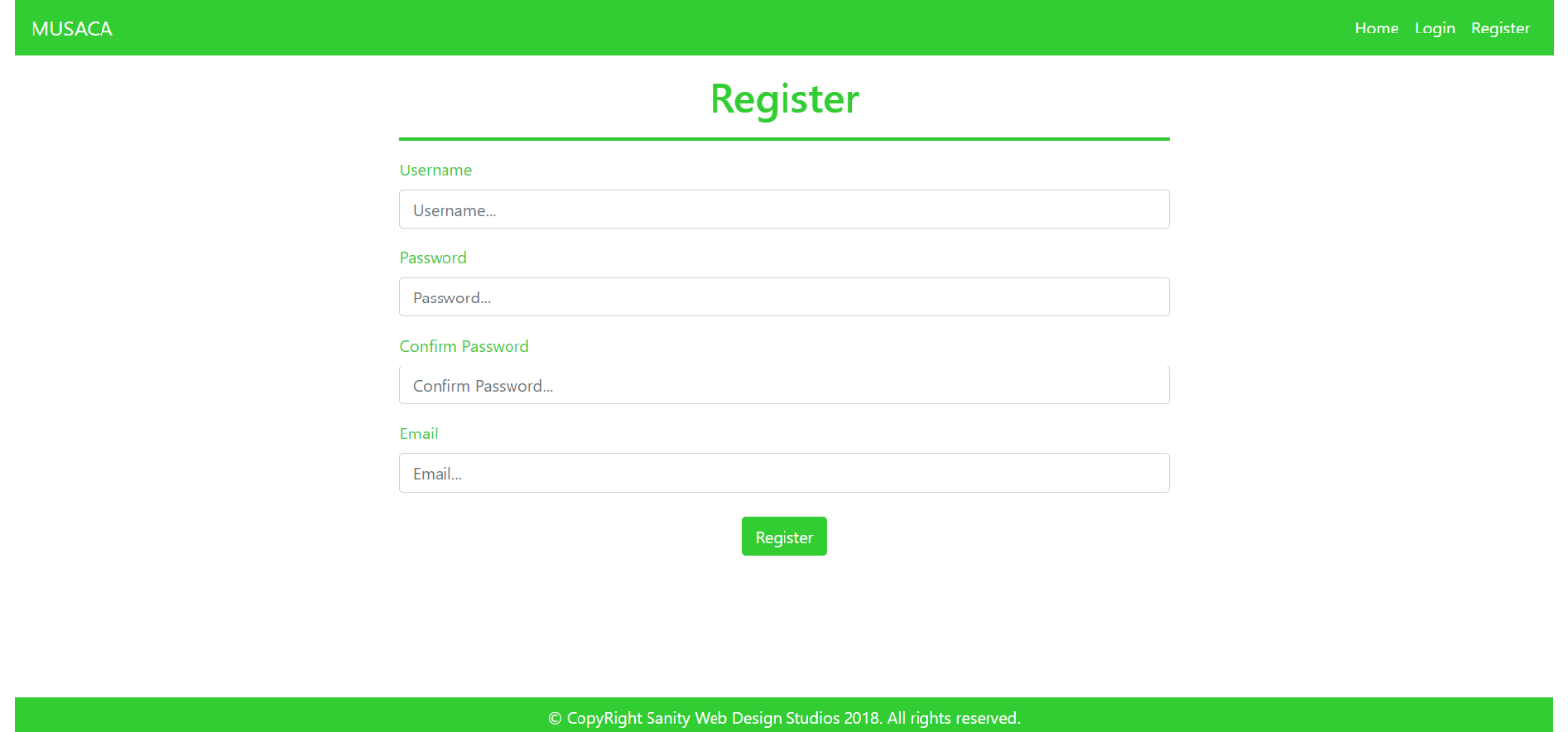
#### Index Template (route = “/Home/Index”) (logged out user)



#### Login Template (route = “/Users/Login”) (logged out user)

#### 

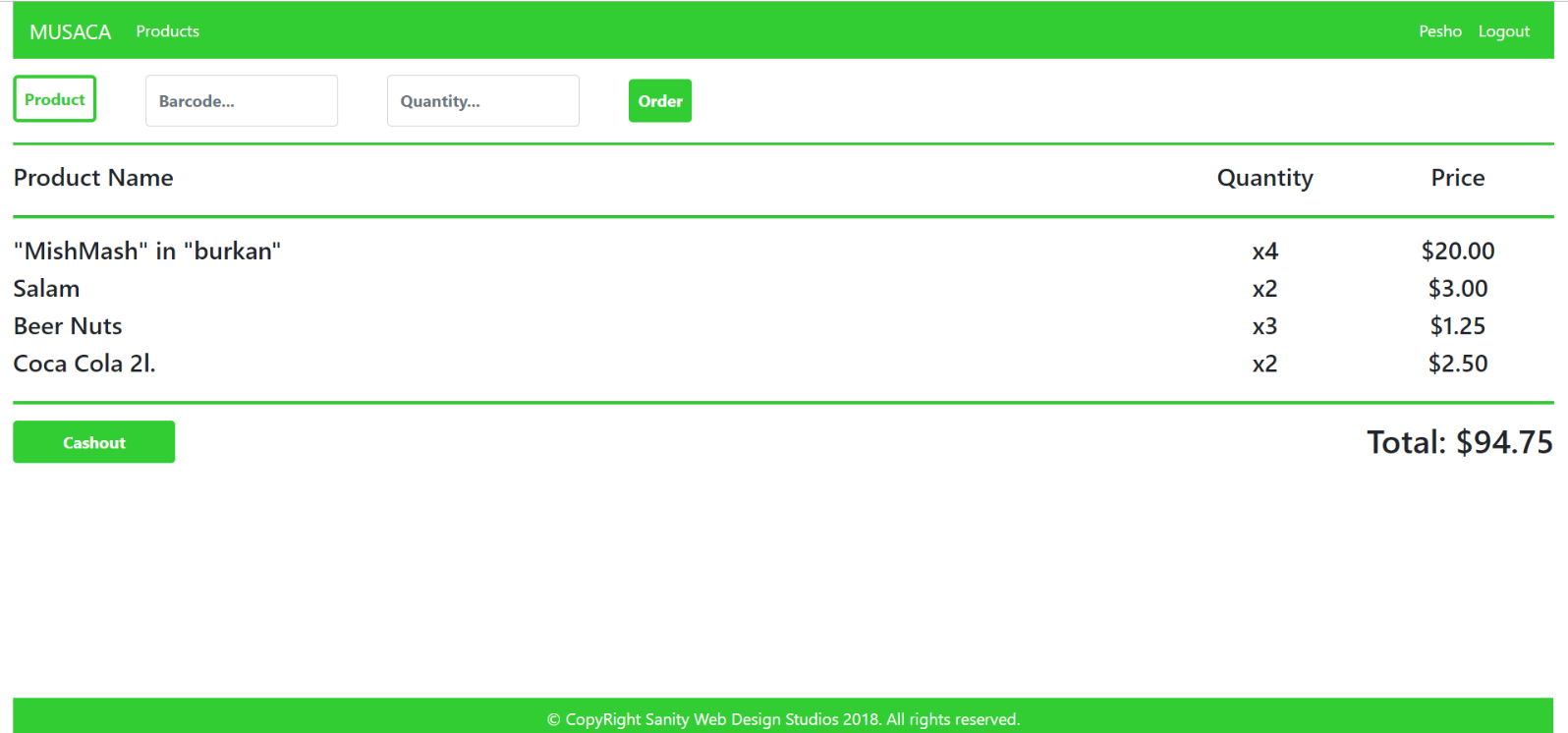
#### Register Template (route = “/Users/Register”) (logged out user)



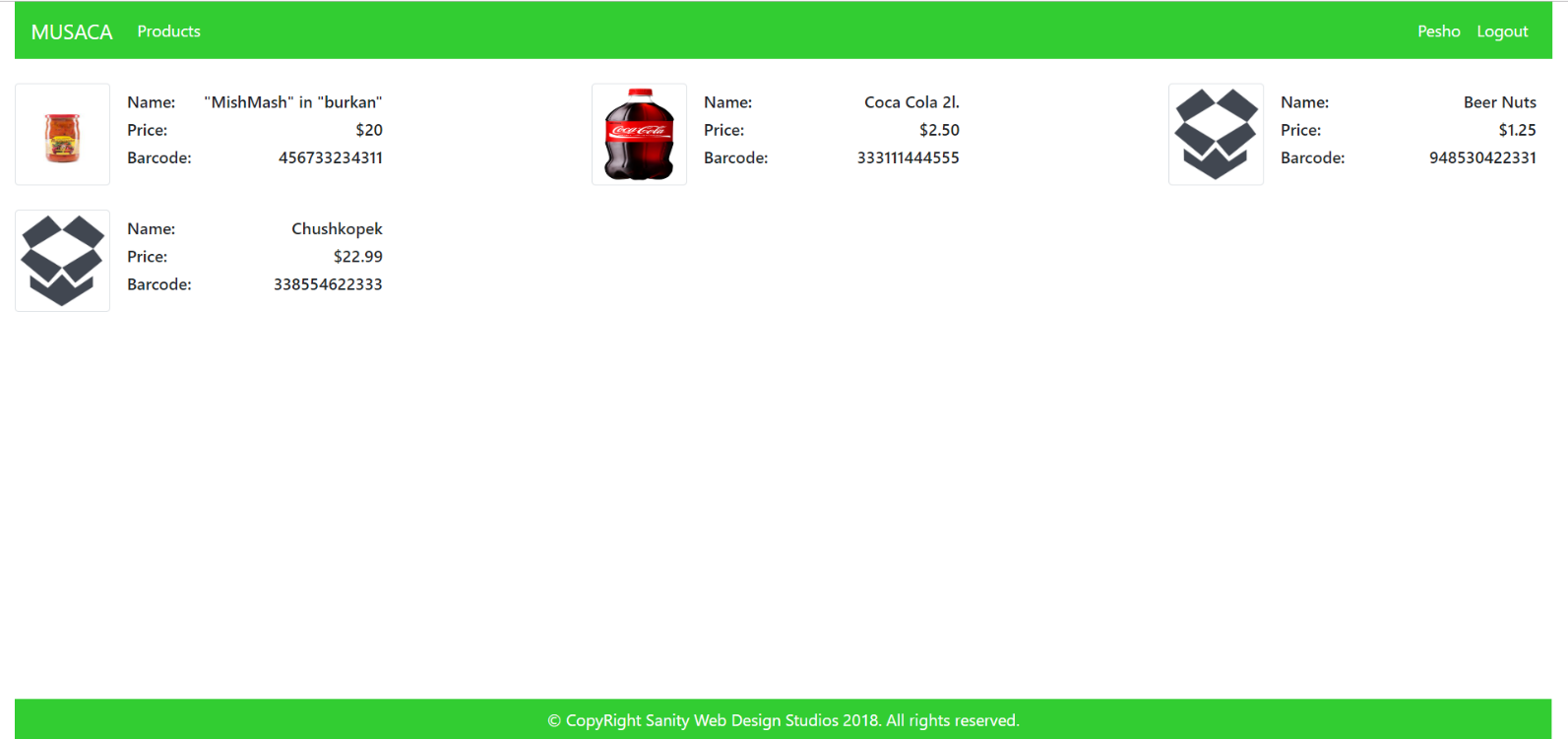
### User Templates

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

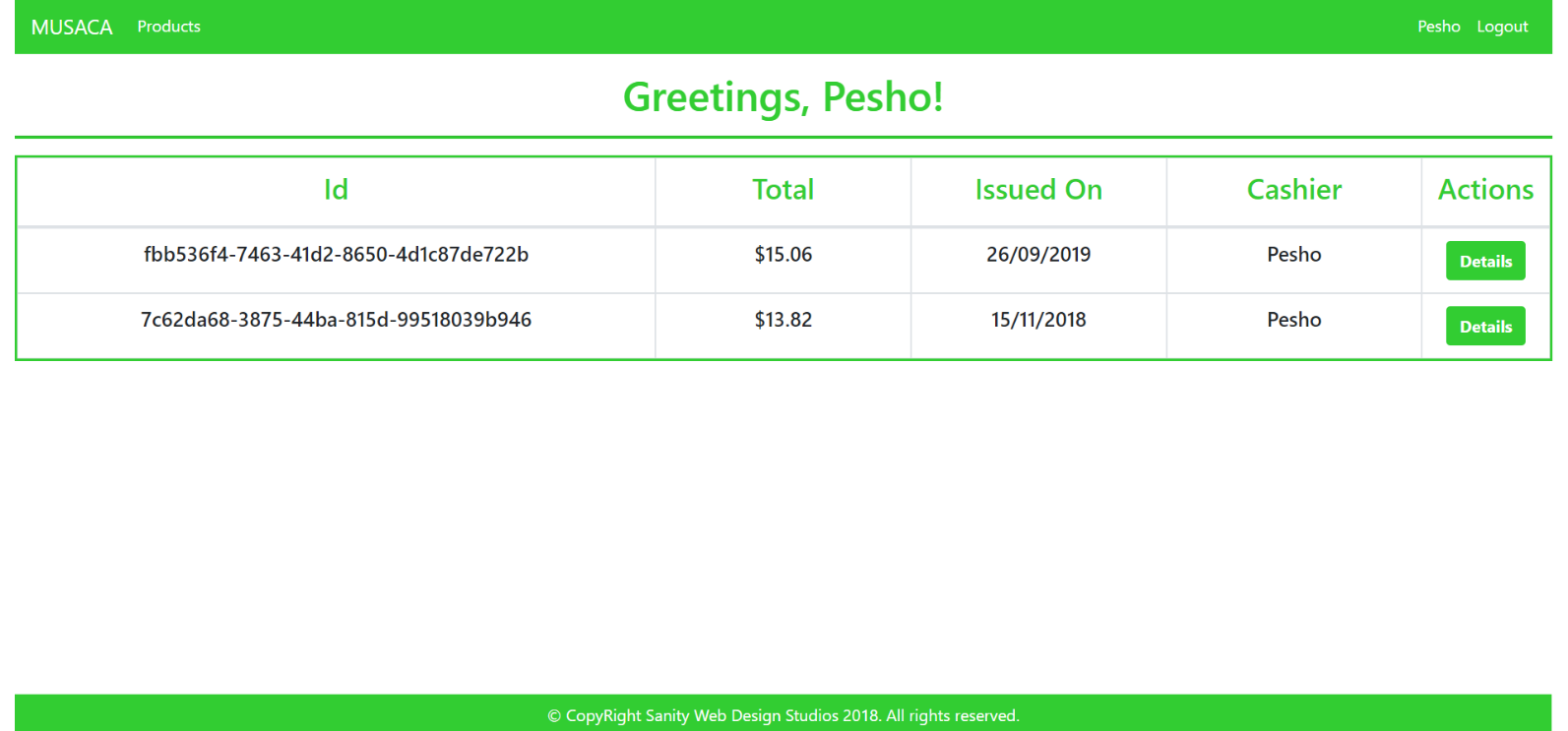
#### LoggedIn Index Template (route=”/Home/Index”) (logged in user)



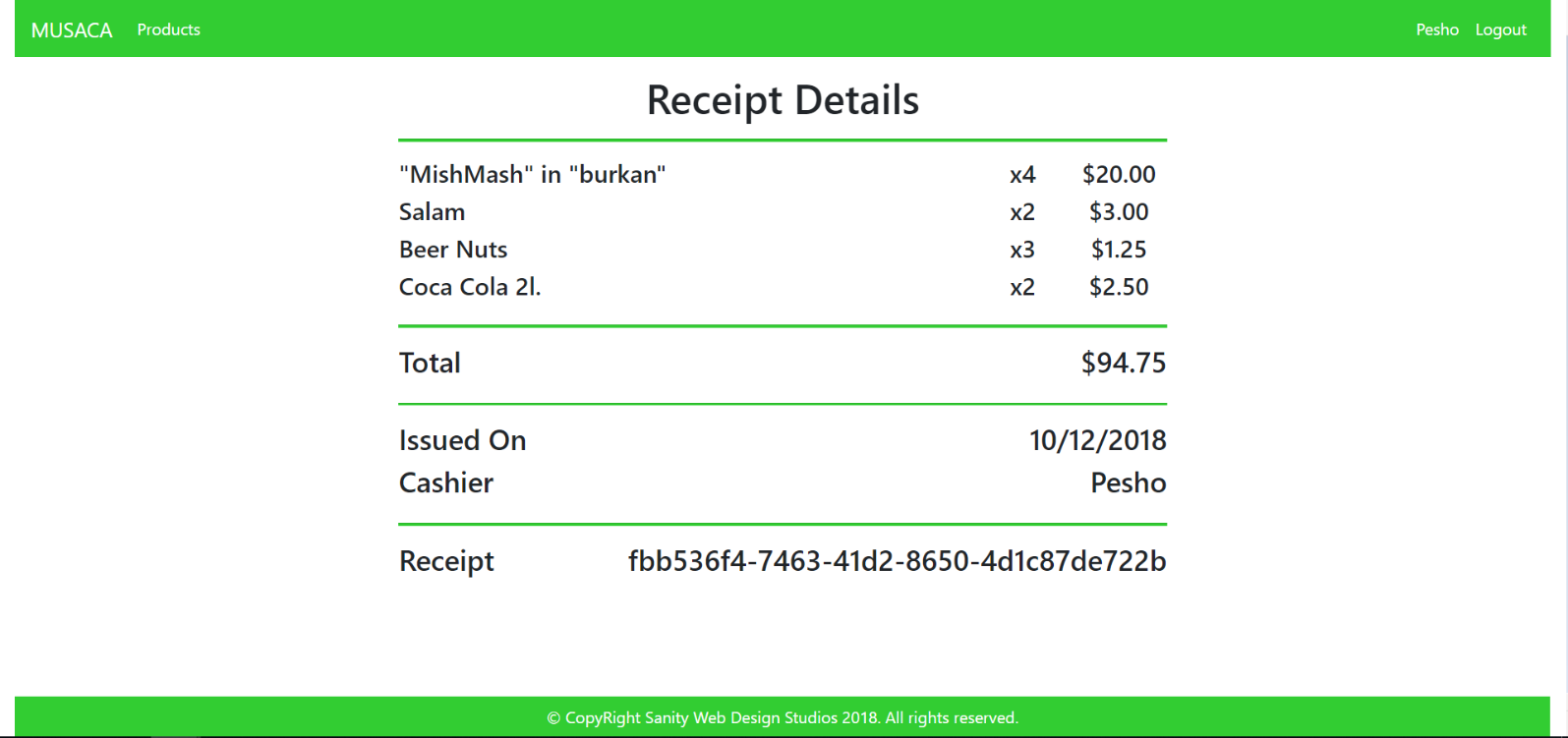
#### Products All Template (route=”/Products/All”) (logged in user)



#### Profile Template (route=”/Users/Profile”) (logged in user)



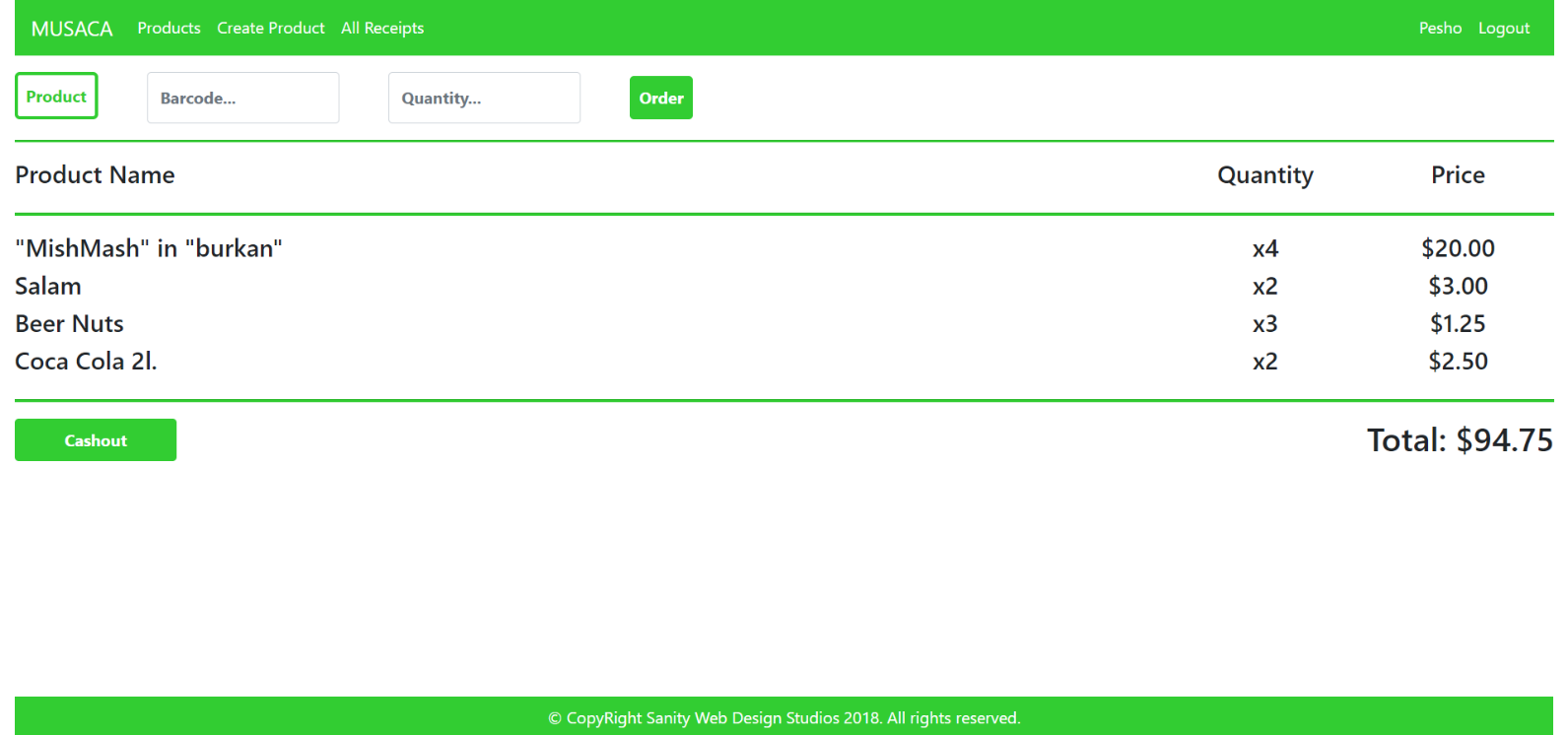
#### Receipt Details Template (route=”/Receipts/Details?id={id}”) (logged in user)



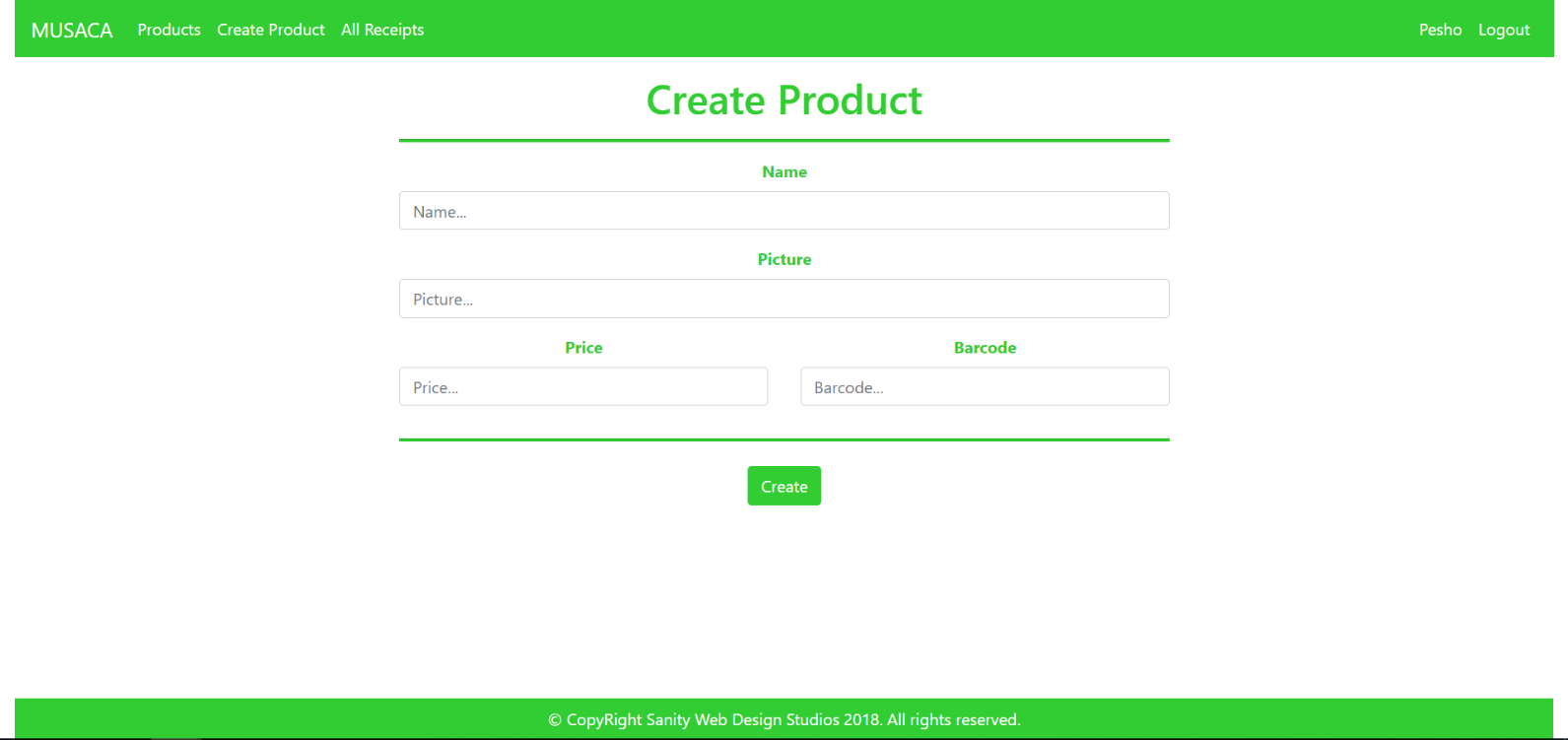
### Admin Templates

These are the **templates** and **functionalities**, accessible by Admins (**logged in** users with Role - Admin).

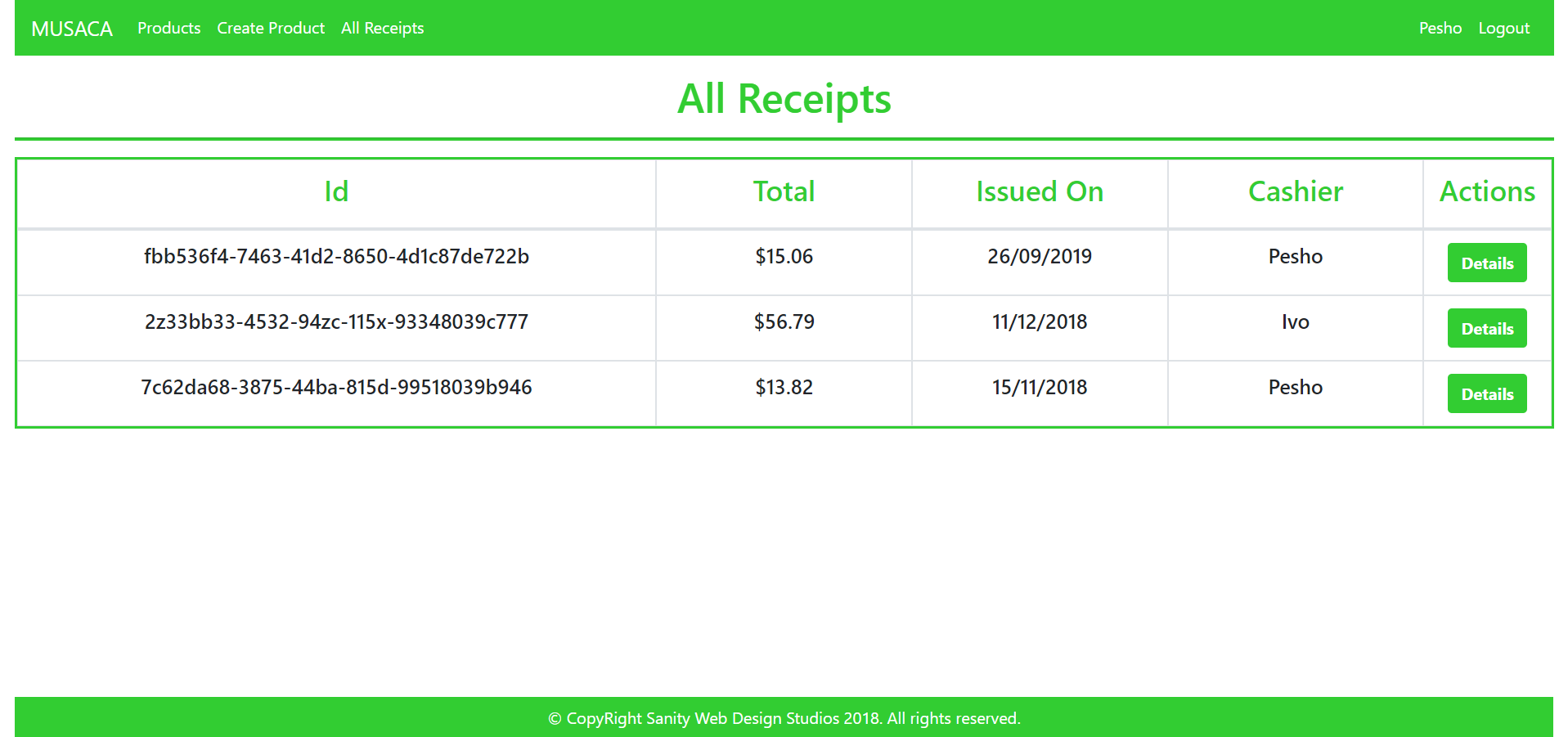
#### Admin Index Template (route=”/Home/Index”) (logged in admin)



#### Admin Product Create Template (route=”/Products/Create”) (logged in admin)



#### Admin All Receipts Template (route=”/Receipts/All”) (logged in admin)



Some of the templates have been given to you in the application skeleton, but the others will be for you to implement, so make sure you implement them correctly. You can use the given ones as helpers.

**NOTE**: The templates should look **EXACTLY** as shown above.

**NOTE**: The templates do **NOT** **require** **additional** **CSS**. Only **bootstrap** and the given **style.css** are enough.

**NOTE**: In the given **style.css** you’ll see some helpful classes “bg-musaca”, “text-musaca”, “border-musaca”. “border-musaca” will help you with the **border** of the **Label** of the **Form** on the **Index Pages**.

## Functional Requirements

The functionality of the MUSACA application is quite simple. The Users are the Cashiers, and they are “checking out products” literally.

### Main functionality

When a User logs in, he sees several things on his Index page.

* A form for **ordering** a Product by barcode and quantity.
* A list of the active Orders for the current User.
  + With product name, product price (initial product price) and quantity
* A [Cashout] button which completes the Orders and generates a Receipt.

### Products

The Products are just data entities. They are only created and persisted. The Orders do not affect them in any way. The Products are like templates, so that the User knows what he is ordering.

### Orders

By entering a barcode of an existent Product and a quantity, in the **form** on the **top** of the Index page, a User creates an Order. Upon completing this functionality, the application should **redirect** back to the Index page, where the User (Cashier) can see his Order.

An Order is created with status – "Active" by **default**. Active Orders are visualized on the Index page as a list, with their Product’s Name, their quantity and their Product’s price. A Total Price is also visualized, below the list, which is a sum of every Order’s **total price** (**quantity** \* **product price**).

**NOTE**: Only Orders with status – "Active" are visualized on the Index page.

### Receipts

Receipts are naturally just data entities. They have no functionality around them – they just store data.

Clicking the [Cashout] button on the bottom of the Index page should create a Receipt with the Orders – currently visualized on the Index page, and the corresponding (currently logged-in) User (Cashier).   
All the Orders of the corresponding Receipt should be **completed** (status set to "Completed").

Upon completing this functionality, the application should **redirect** to the newly created Receipt’s Details page, where the User (Cashier) can see the generated Receipt.

### Users

All casual Users (role – "User") can perform the functionalities specified above. Check Products, Create Orders, Cashout (Create Receipts), Check personal Receipts, Check personal Receipt Details.

Each User can check data about all the Products, by clicking on the [Products] button from the **navbar**. This should **redirect** him to the Products All page.

Each User can check data about all **HIS** Receipts, by clicking on the [{Username}] button from the **navbar**. This should redirect him to the Profile page. On that page, the User can check a list of only **HIS** Receipts, and he can view details about each one of them, by clicking on the [Details] button, which should **redirect** him to the corresponding Receipt’s Details page.

### Admins

All administrator Users (role – "Admin") can perform the functionalities of a casual User.

Each Admin can also create new Products, by clicking on the [Create Product] button from the **navbar**. This should redirect him to the Product Create page. NOTE: Creating a Product, should **redirect** to the Products All.

Each Admin can also check all the Receipts (including his own), by clicking on the [All Receipts] button from the **navbar**. This should redirect him to the Receipts All page. NOTE: If an Admin goes to his Profile page, he can see only **HIS** Receipts there.

**NOTE**: All decimal data is visualized up to the second digit after the decimal point.

## Security Requirements

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

* Guest (not logged in) users can access Index page and functionality.
* 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 User LoggedIn Index page and functionality.
* Users (logged in) can access Products page and functionality.
* Users (logged in) can access Order functionality.
* Users (logged in) can access User Profile page and functionality.
* Users (logged in) can access Receipt Details page and functionality.
* Users (logged in) can access Logout functionality.
* Admins (logged in) can access **every functionality** a **normal** logged in User can.
* Admins (logged in) can access Admin LoggedIn Index page and functionality.
* Admins (logged in) can access the Admin Product Create page and functionality.
* Admins (logged in) can access the Admin All Receipts page and functionality.

## Scoring

This section describes how the scoring of the Exam will be made.

### Database Requirements – 10 points.

### Template Requirements – 25 points.

### Functional Requirements – 55 points.

### Security Requirements – 10 points.