# Exercises: Introduction to ASP.NET Core MVC

## Fluffy Duffy Munchkin Cats (FDMC)

You have been tasked to create a very simple, very non-quality code web application. The offer sounds pretty absurd and disturbing to you since you are a good (non-Indian) developer and such low-quality applications are not in your style, but hey… they pay well.

The application you must implement is called FDMC, an online Cat Exhibition website which should support very basic functionality.

Use ASP.NET Core. It’s up to you to **decide whether to create a Razor page or to use the MVC pattern**. I recommend that you try out both, just to get a little taste of them.

### Implementing the Data Layer

Your employers wanted you to store the database on a **.txt file** but you deeply **DISAGREED** with that idea. And they said you can use your black magic databases. You decided to use **Entity Framework Core**.

You have a single model – Cat, which has a name, age, breed and imageUrl.

### Implementing the Index page

On the home route (“/”), you should see a heading (h1) with the name of the project (Fluffy Duffy Munchkin Cats).

Below it you should see a very simple unordered list of items – each item being a cat. The items should be links, which should hold the cat’s name and should lead to a route “/cats/{catId}”.

Below all the links, there should be a button “Add Cat”, which leads to route “/cats/add”, where a simple form is rendered to **add** a **cat** to the application’s databae.

### Implementing the Cat Add Page

The cat add page should hold a simple form, with 4 input fields, for the:

* Cat Name – text
* Cat Age – number
* Cat Breed – text
* Cat Image Url – text

And it should submit a POST request to route “/cat/add”.

The cat add page should hold a heading (h1) – “Add Cat”.

### Implementing the Cat page

Each cat should have a page for itself, which is accessed through the link on the Index page.

It has a heading (h1), holding the **cat’s name**.

It has an image with an alt attribute – the **cat’s name** and a **max width** of 300px.

It has a paragraph, holding the **cat’s age**, and a paragraph, holding the **cat’s breed**. Both should be **bold**, and should have a prefix with the **property’s name**.

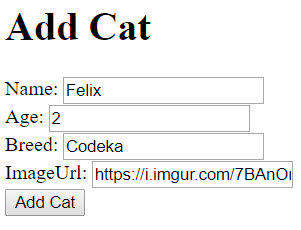
### Visual Looks

The application pages should look like this:

#### Index page

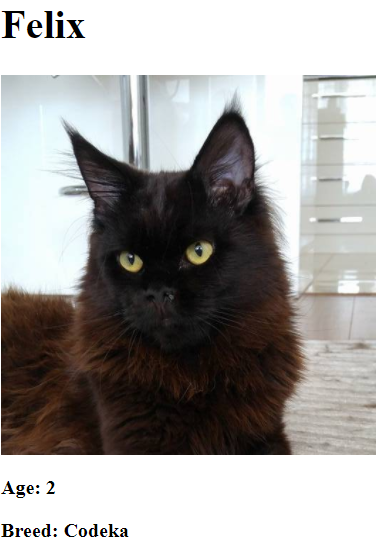


#### Add Cat Page



You can think about displaying “ImageUrl” in a better way. There are a lot of ways to do that.

#### Cat page



And now you can proudly say you’ve implemented your first basic application with ASP.NET Core. Now let’s get to the real thing.

# PANDA

**PANDA** (**P**ackage **A**cceptance and **N**ational **D**elivery **A**pplication) is a platform for package deliveries, which is a fast-rising Start-Up, which lacks a web application. You have been employed by the **KFC** (**K**ung-**F**u-**C**hicken) Corporation to implement a web platform for **PANDA**. However, there are specific requirements that must be followed.

## Database Requirements

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

### User

* 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”)

### Package

* Has an Id – a GUID String or an **Integer**.
* Has a Description – a string.
* Has a Weight – a floating-point number.
* Has a Shipping Address – a string.
* Has a Status – can be one of the following values (“Pending”, “Shipped”, “Delivered”, “Acquired”)
* Has an Estimated Delivery Date – A DateTime object.
* Has a Recipient – a User object.

### Receipt

* Has an Id – a GUID String or an **Integer**.
* Has a Fee – a decimal number.
* Has an Issued On – a DateTime object.
* Has a Recipient – a User object.
* Has a Package – a Package object.

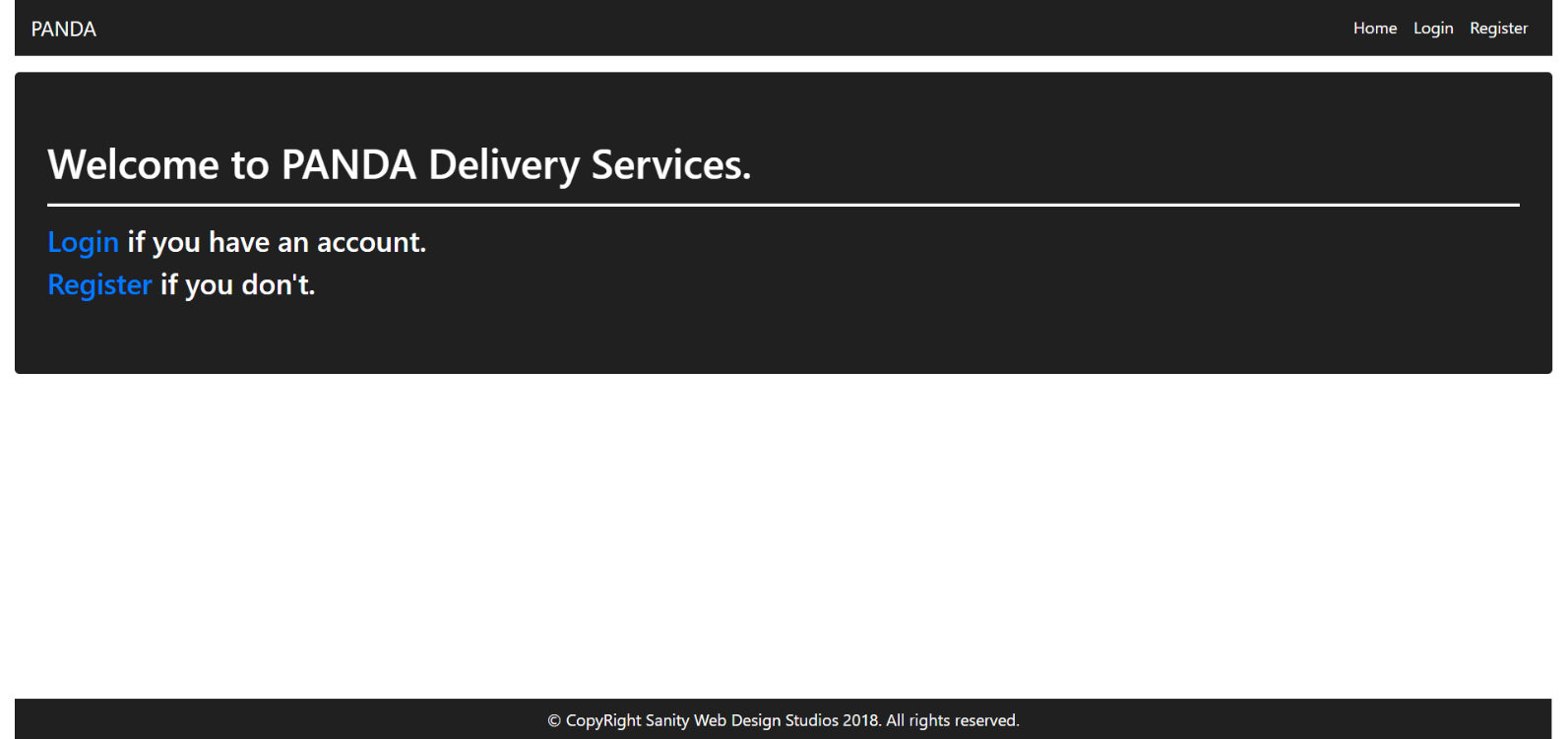
Implement the entities with the **correct datatypes**.

## Template Requirements

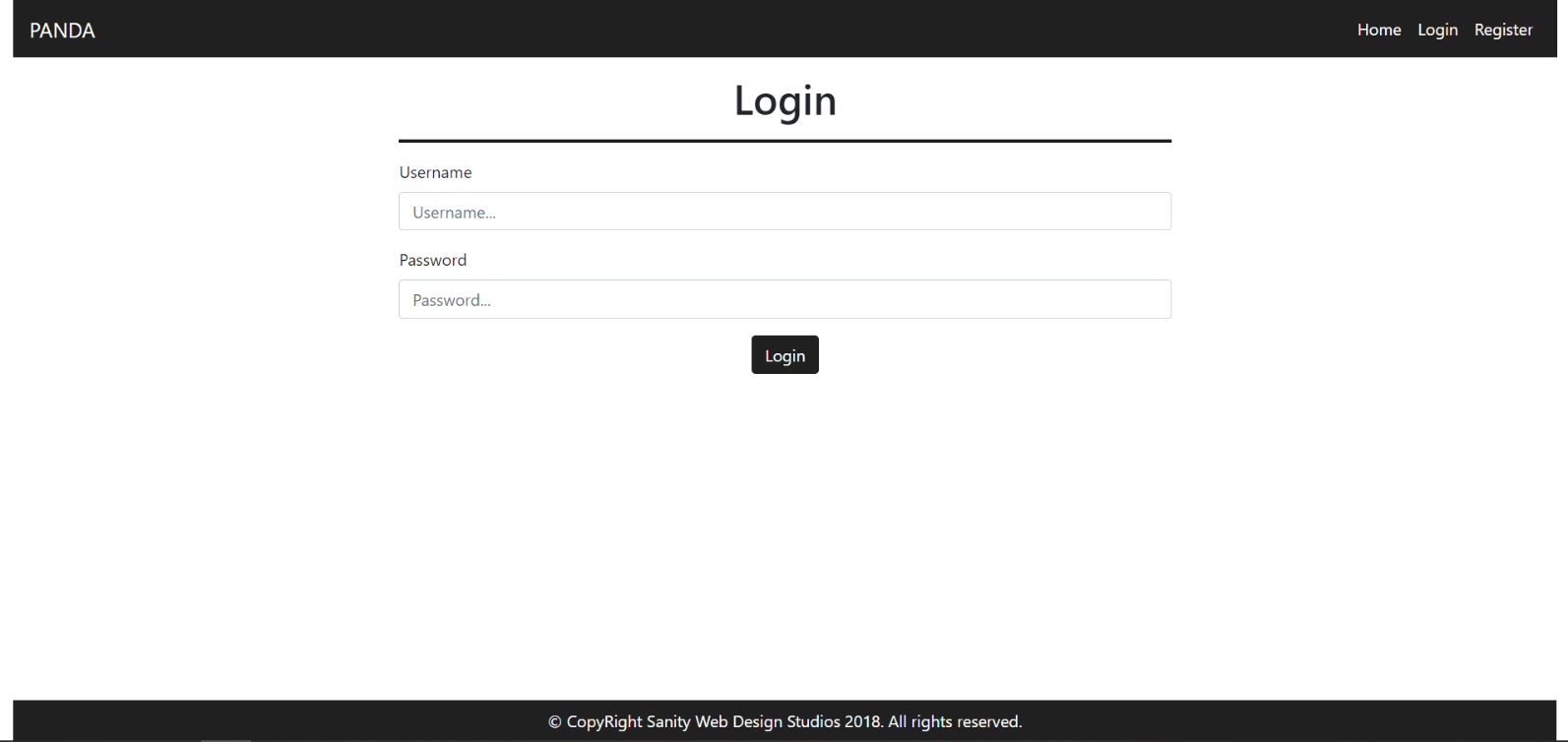
### Guest Templates

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

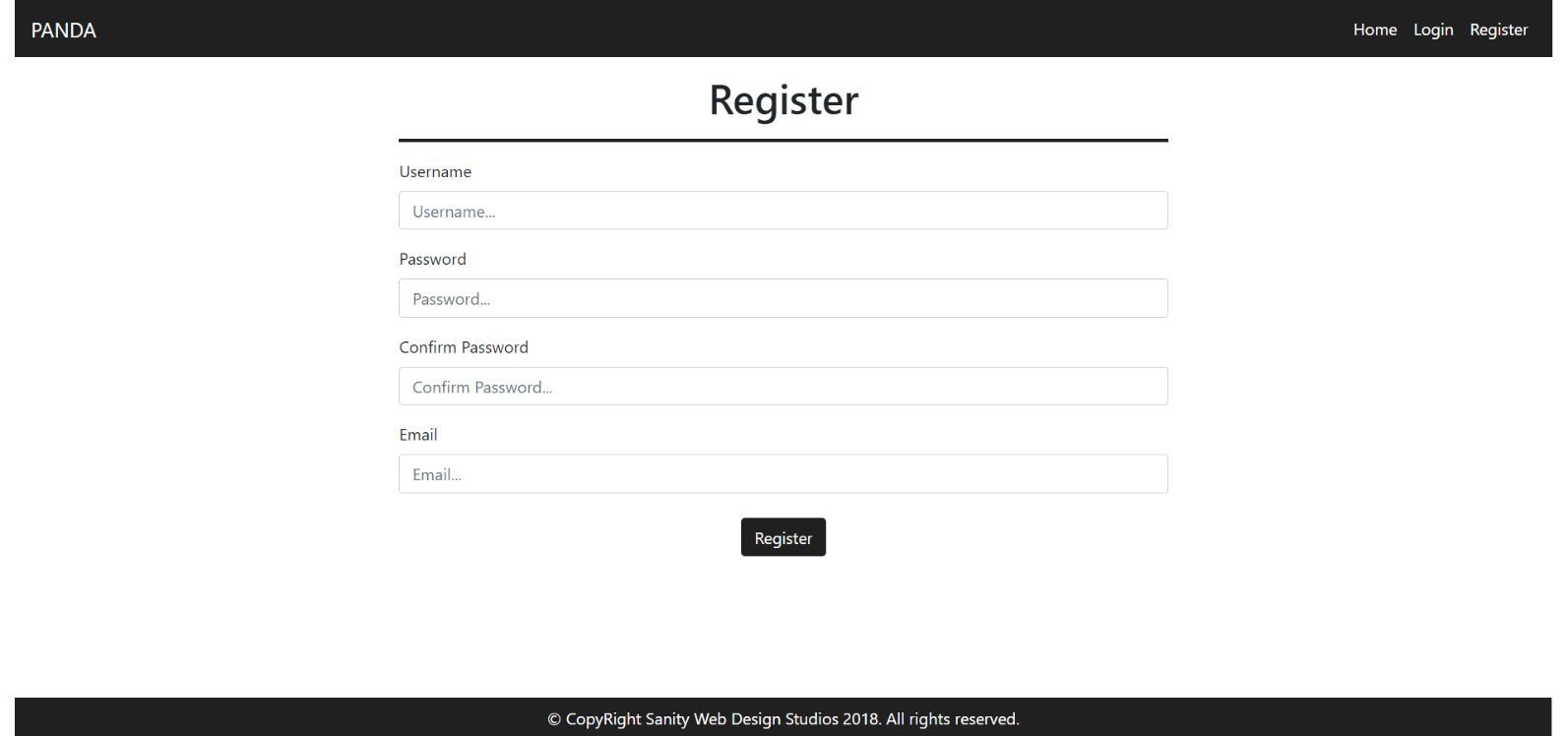
#### Index Template (route = “/Home/Index”) (logged out user)



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



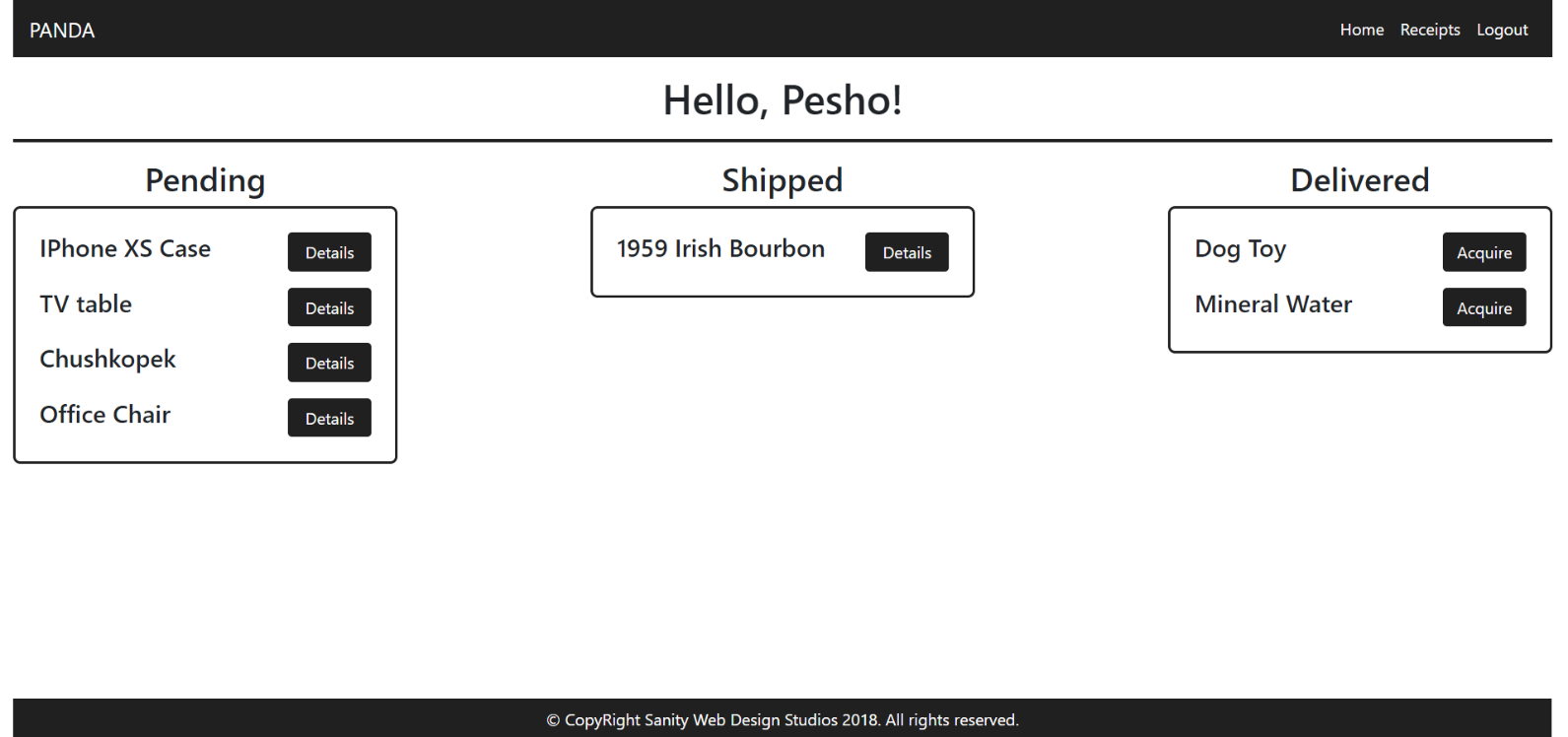
#### Register Template (route = “/Users/Register”) (logged out user)



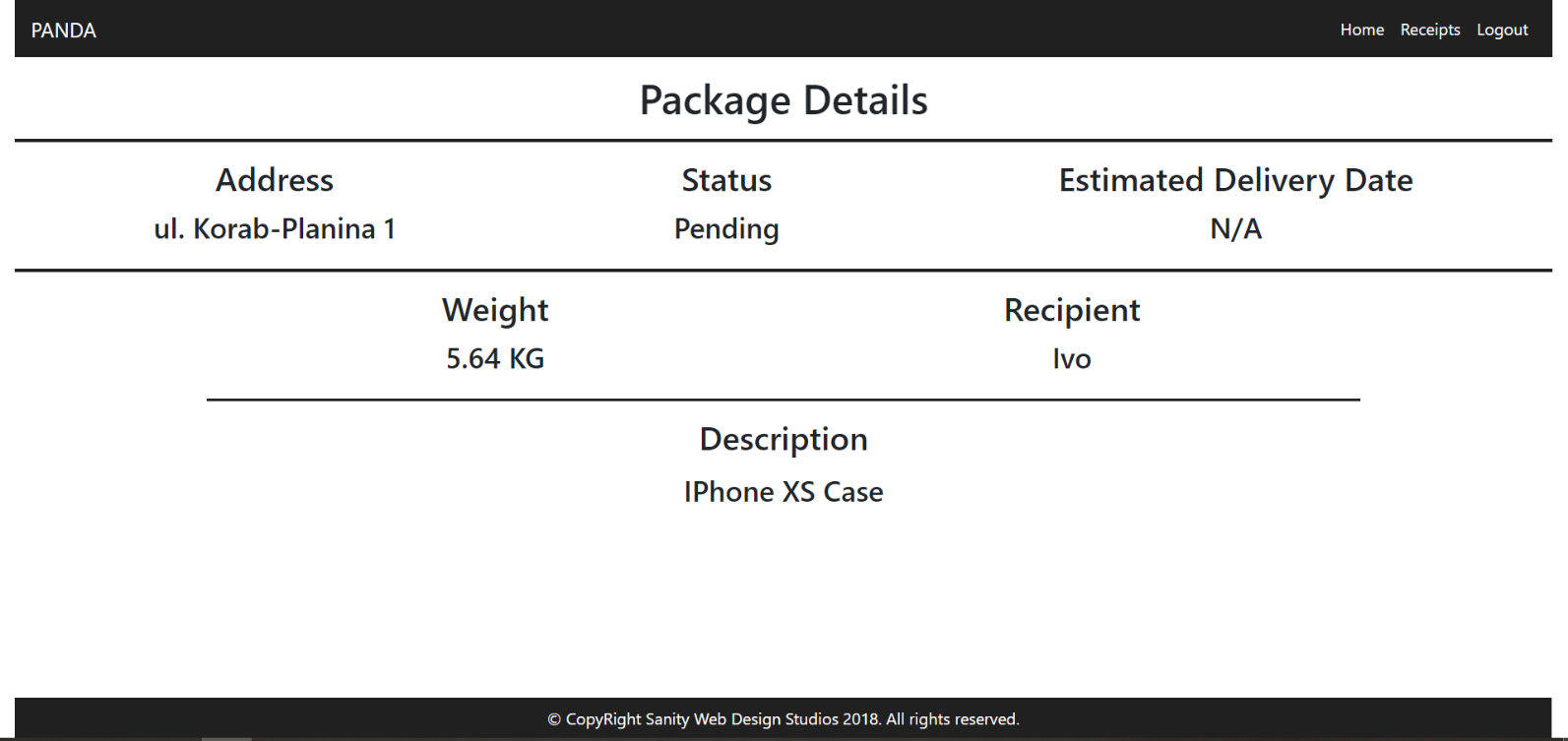
### User Templates

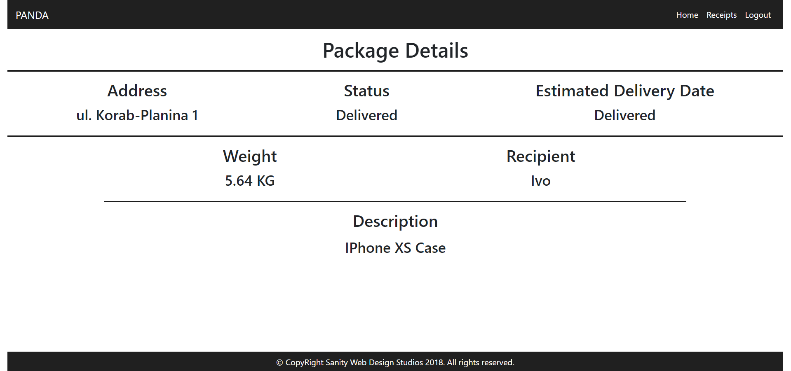
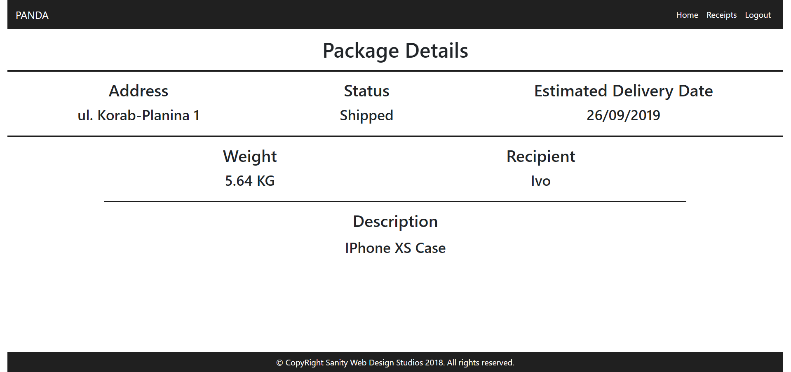
These are the **templates** and **functionalities**, accessible by Users (**logged in** users with Role - User).

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



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

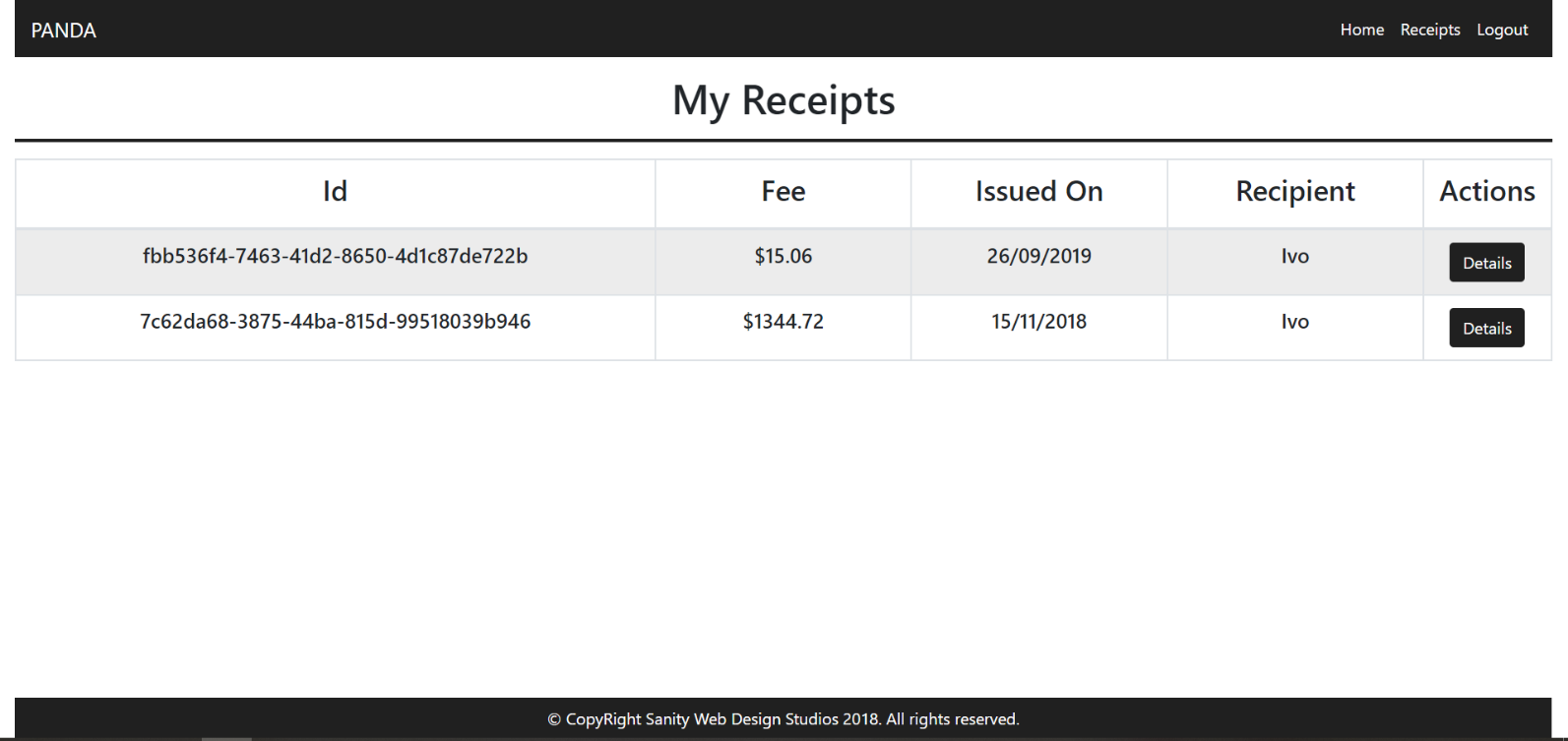




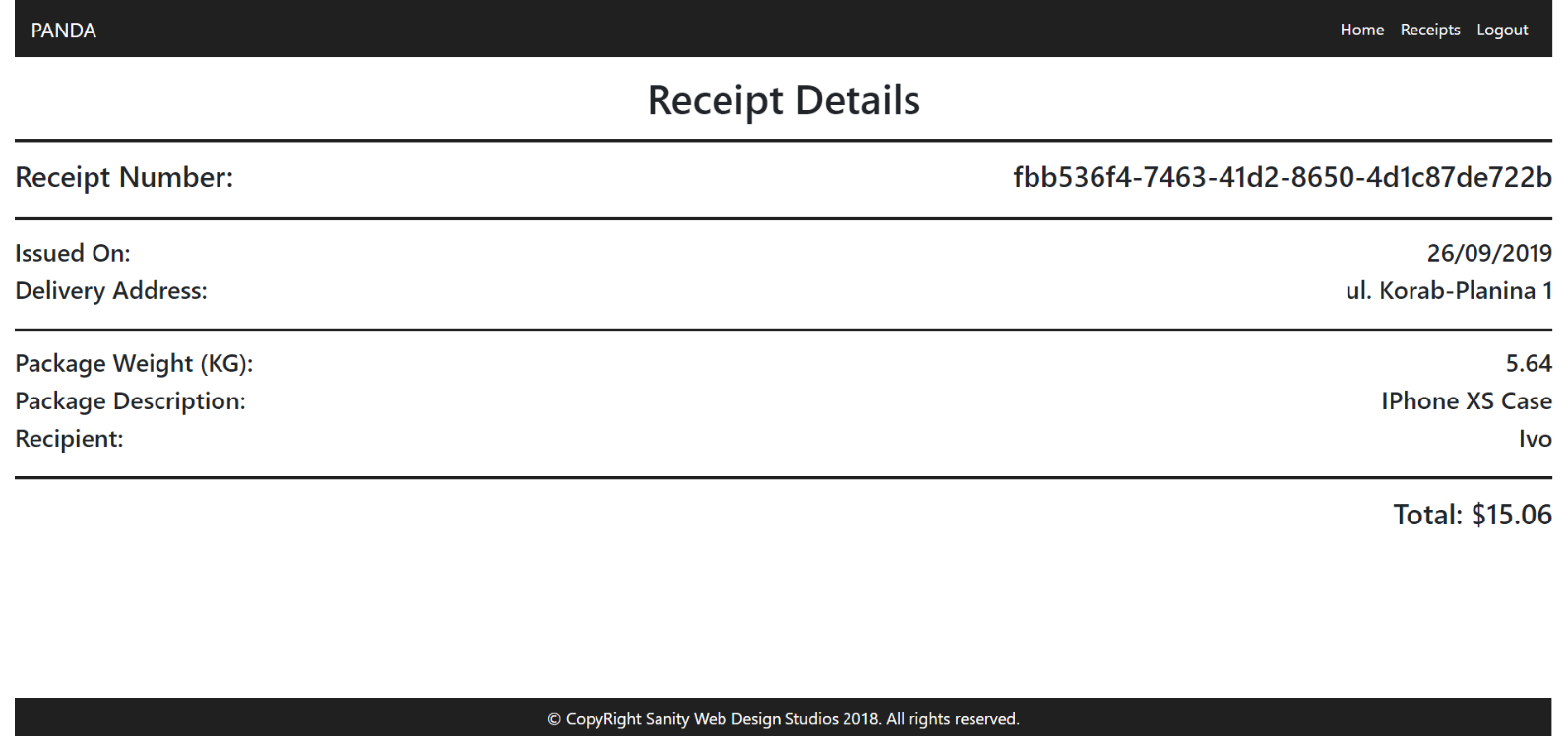
**NOTE**:

* If the **package’s status** is Pending, it should **NOT** have an Estimated Delivery Date, and “N/A” should be presented.
* If the **package’s status** is Shipped, it will have an Estimated Delivery Date. In that case the **date** should presented instead of “N/A”. (Example: “29/02/2016”).
* If the **package’s status** is Delivered or Acquired, instead of the **date**, “Delivered” should be presented.

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



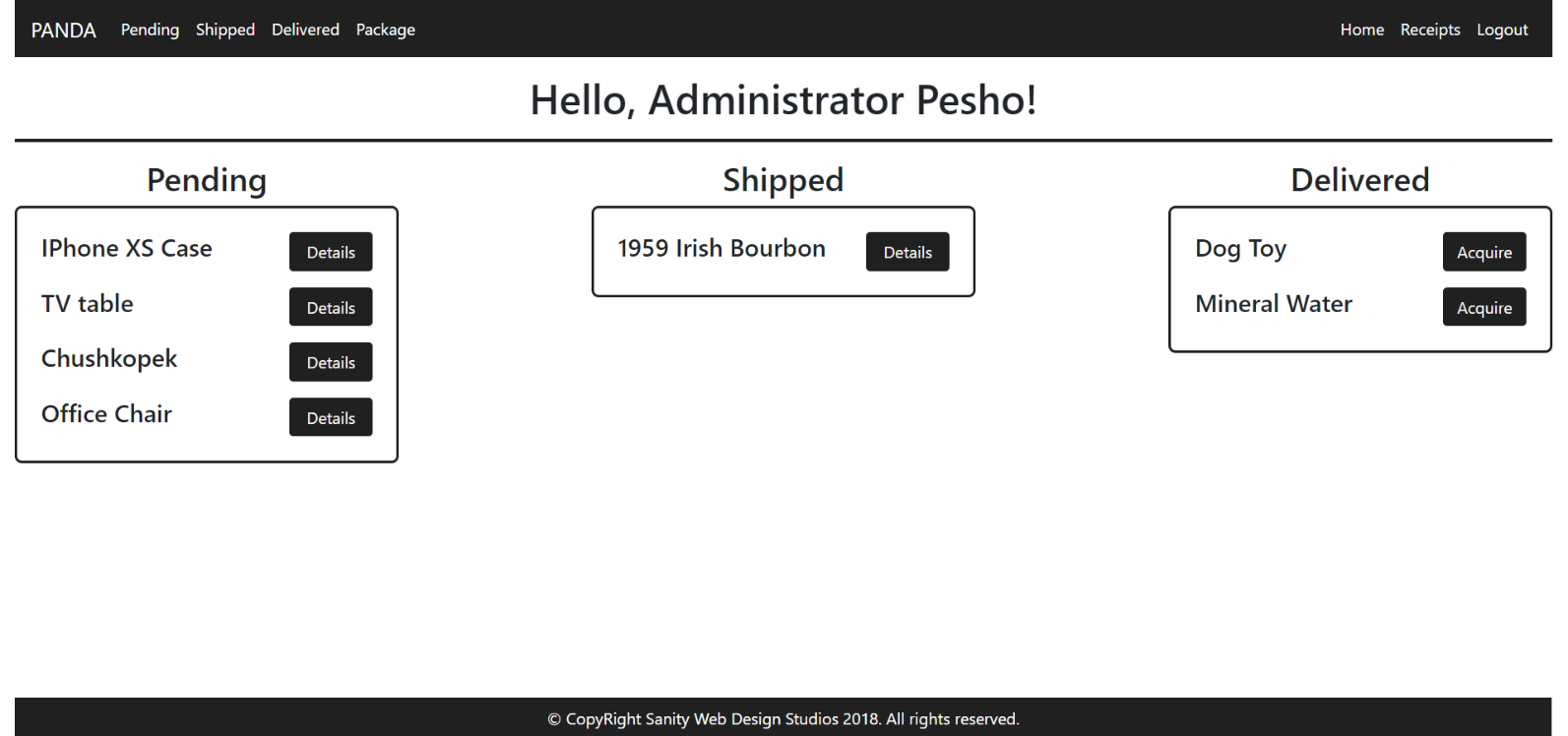
#### Receipt Details Template (route=”/Receipts/Details?id={id}”) (logged in user)



### Admin Templates

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

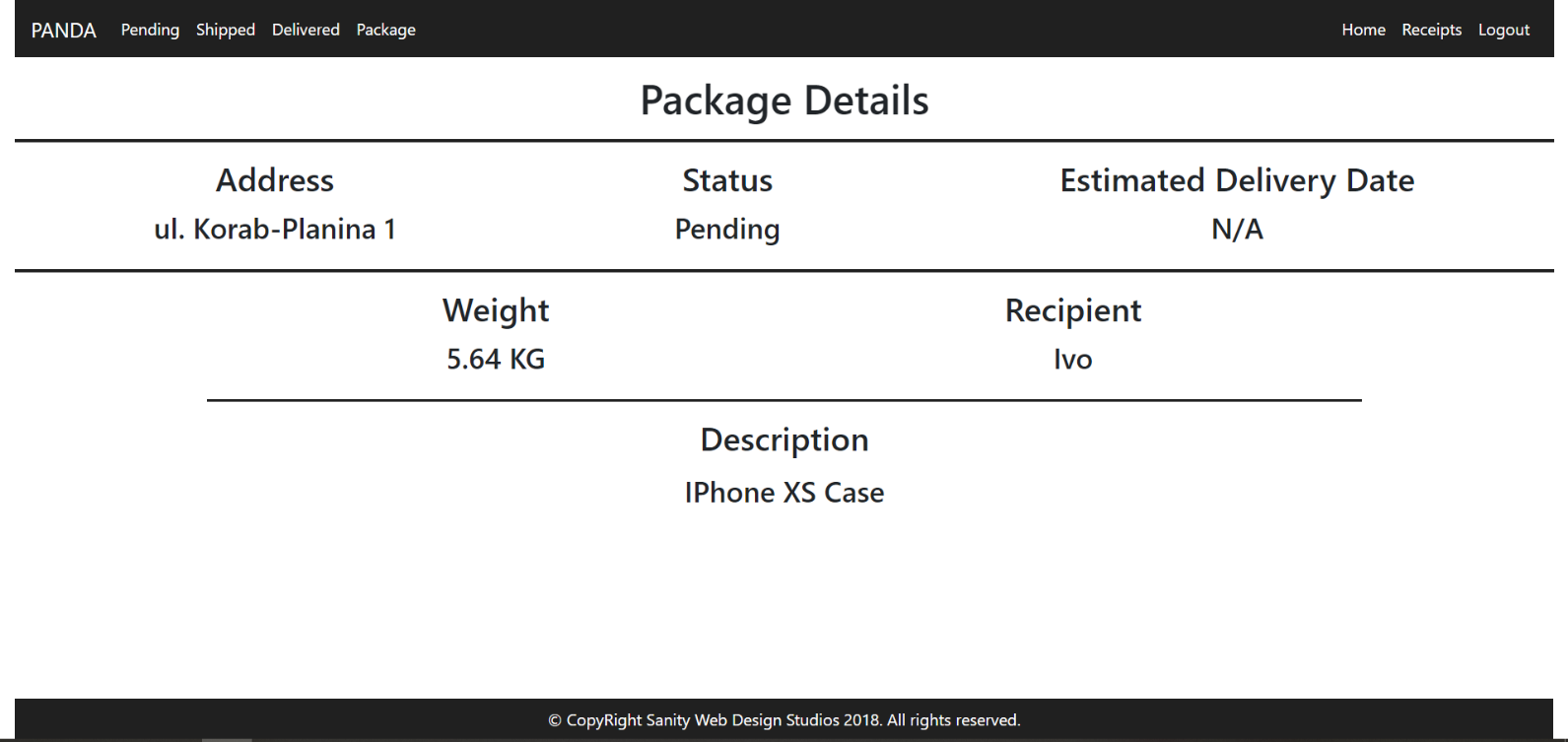
#### Admin Index Template (route=”/Home/Index”) (logged in admin)

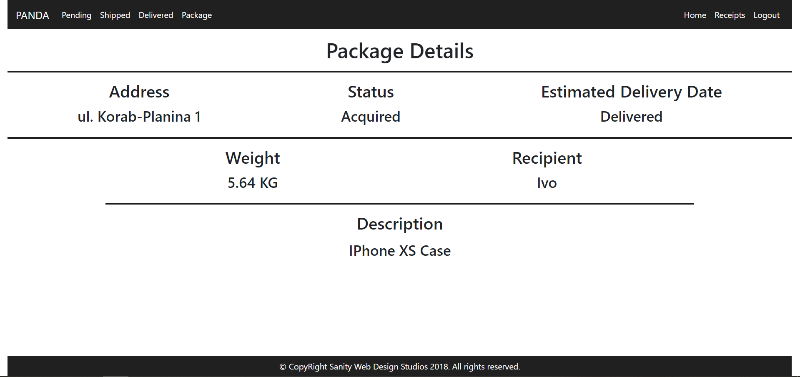
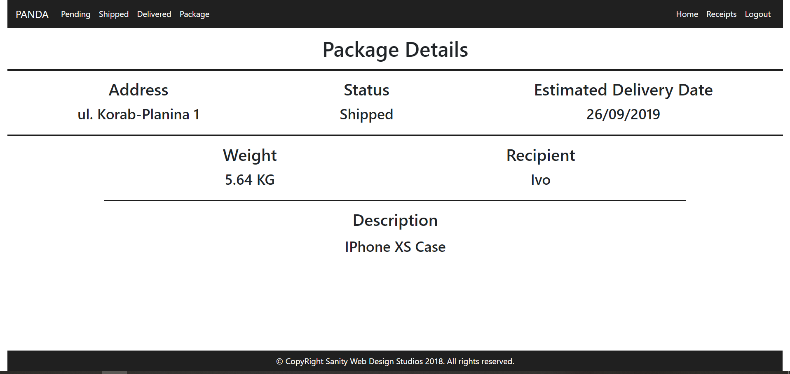


The navigation links above should lead to:

* [**Pending**] – **Pending Packages**
* [**Shipped**] – **Shipped Packages**
* [**Delivered**] – **Delivered Packages**
* [**Package**] – **Create Package**

#### Package Details Admin Template (route=”/Packages/Details?id={id}”) (logged in admin)

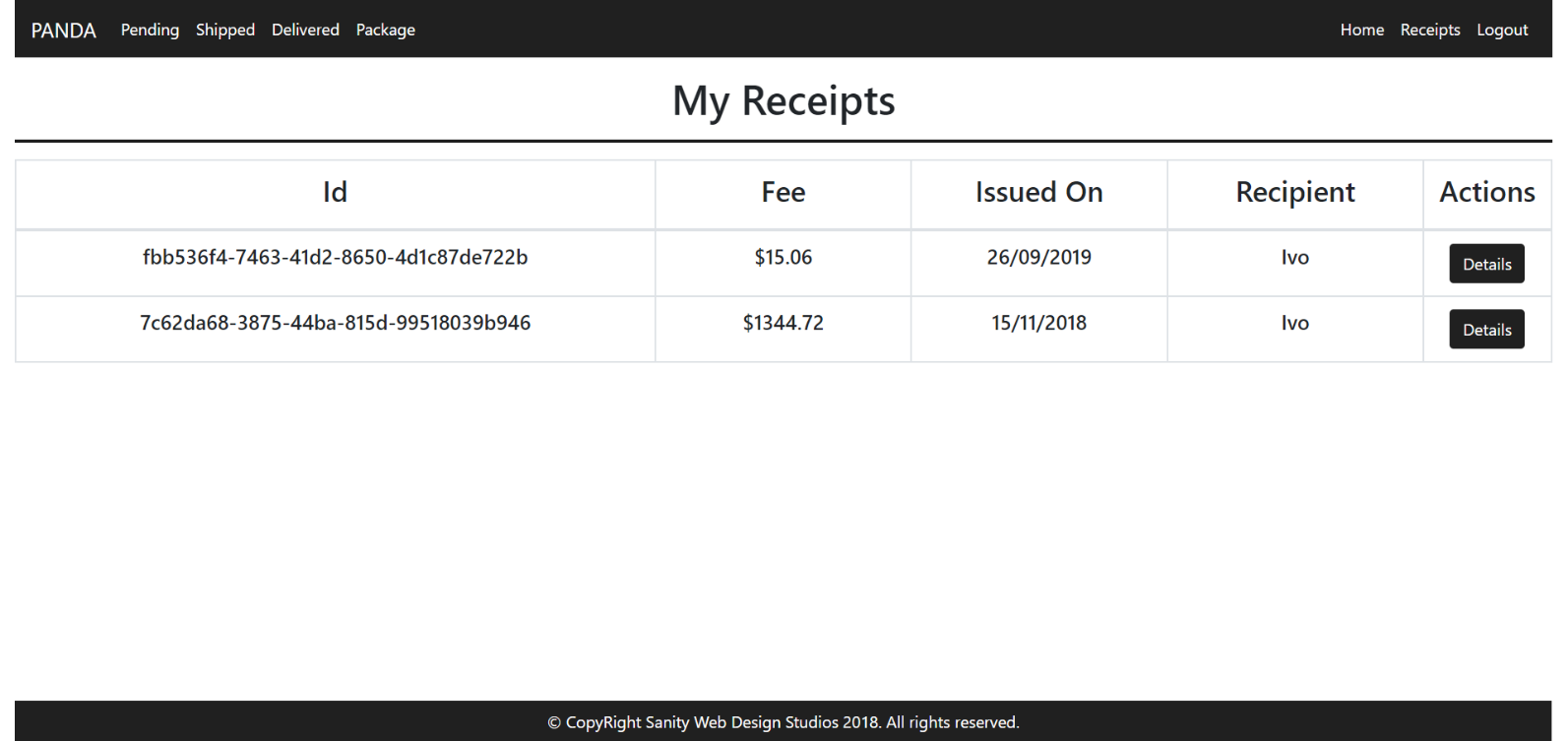




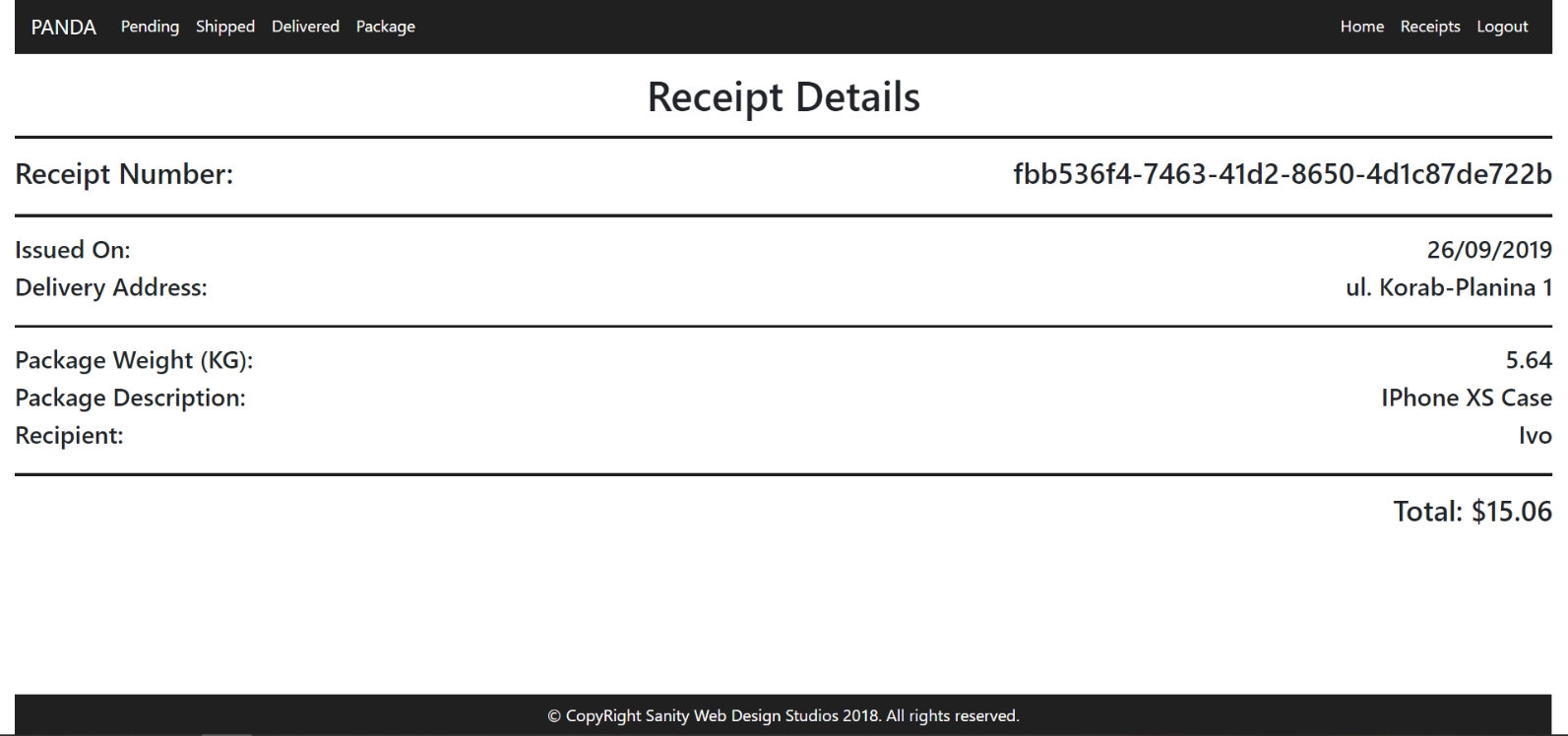
**NOTE**:

* If the **package’s status** is Pending, it should **NOT** have an Estimated Delivery Date, and “N/A” should be presented.
* If the **package’s status** is Shipped, it will have an Estimated Delivery Date. In that case the **date** should presented instead of “N/A”. (Example: “29/02/2016”).
* If the **package’s status** is Delivered or Acquired, instead of the **date**, “Delivered” should be presented.

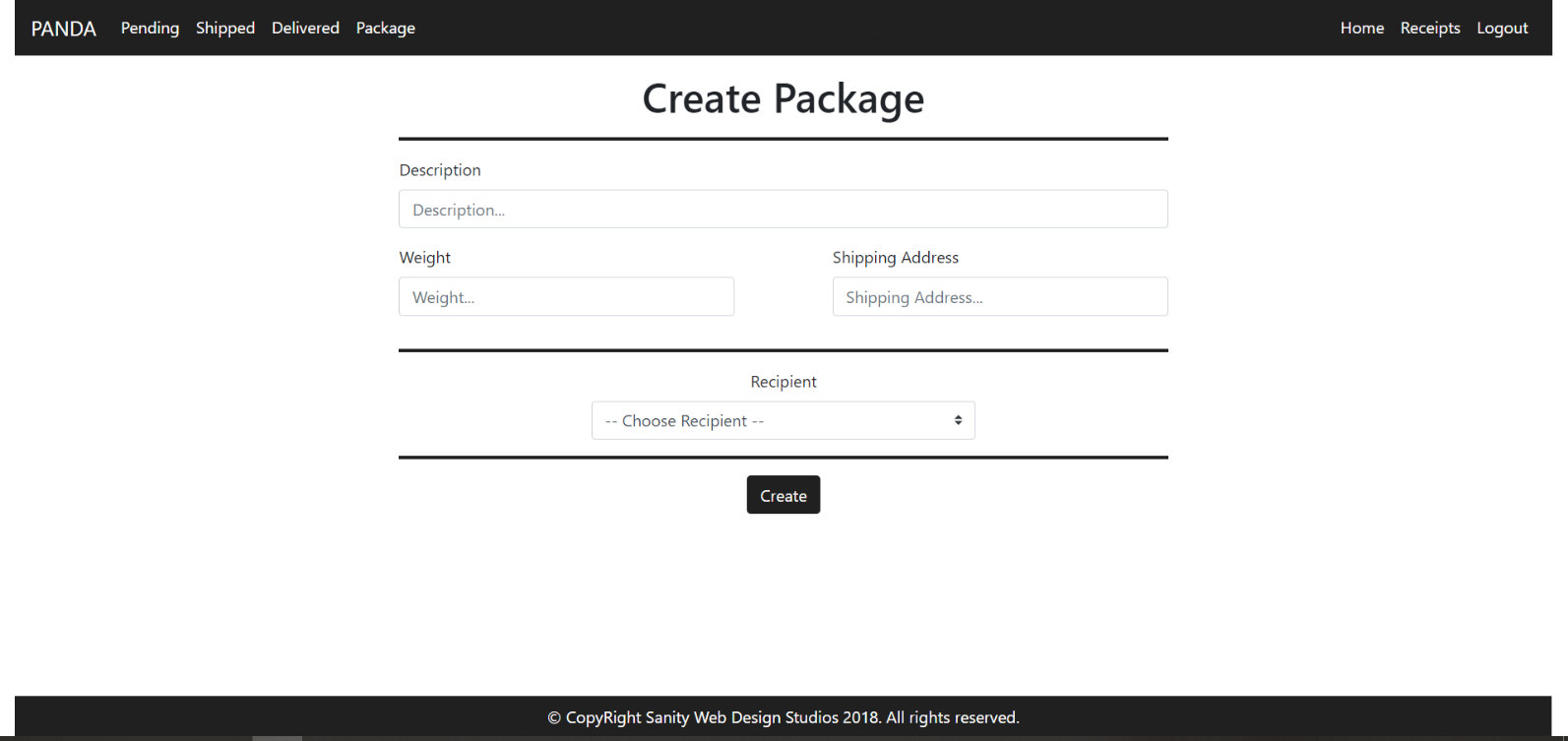
#### Receipts Admin Template (route=”/Receipts/Index”) (logged in admin)



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

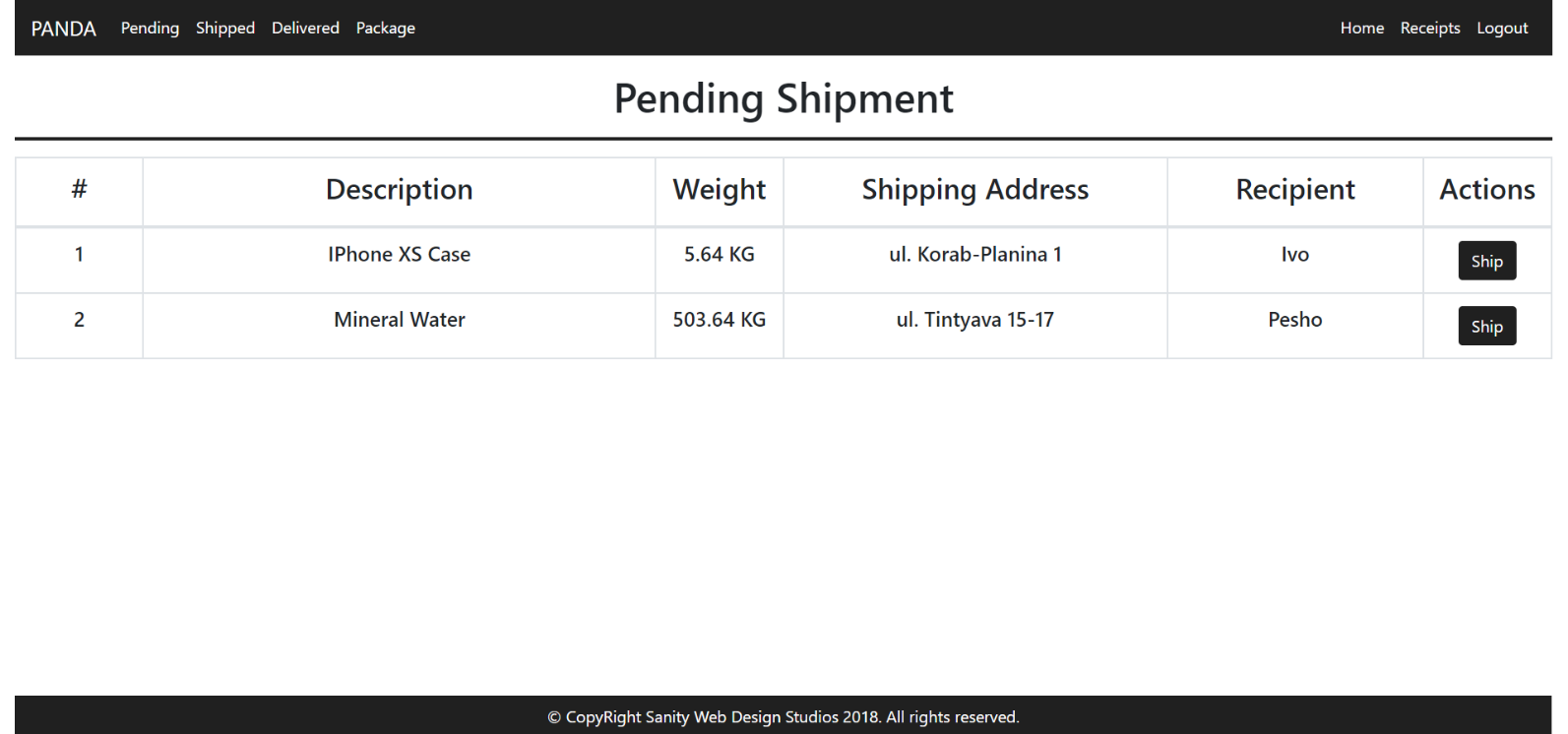


#### Package Create Admin Template (route=”/Packages/Create”) (logged in admin)

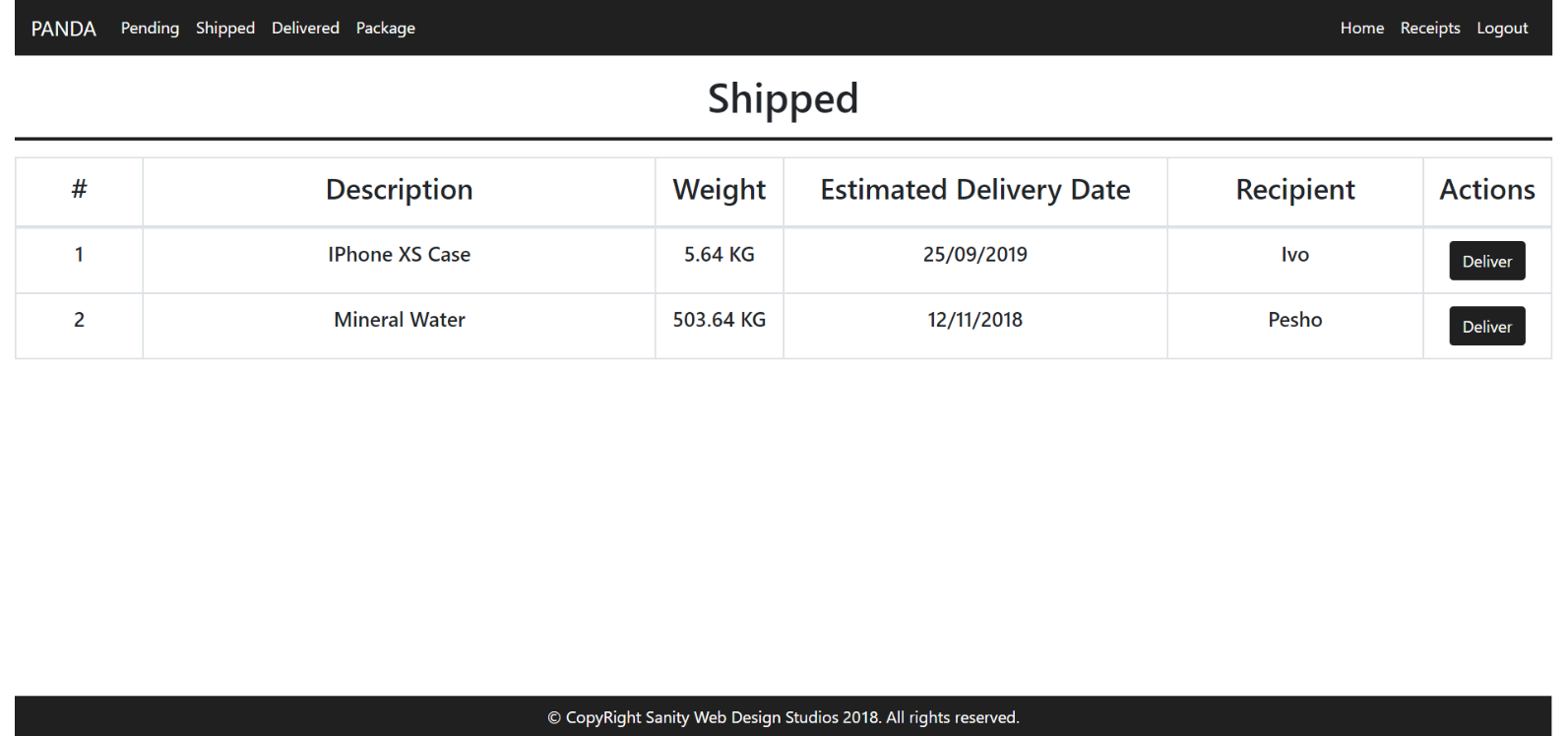


**NOTE**: The **Recipients** field is a **dropdown** **menu**, with **all the registered users** in the **application** (including **yourself**). The **primarily selected** **option** should be “-- Choose Recipient --" and it should be **disabled**, so that it **cannot** be **selected**.

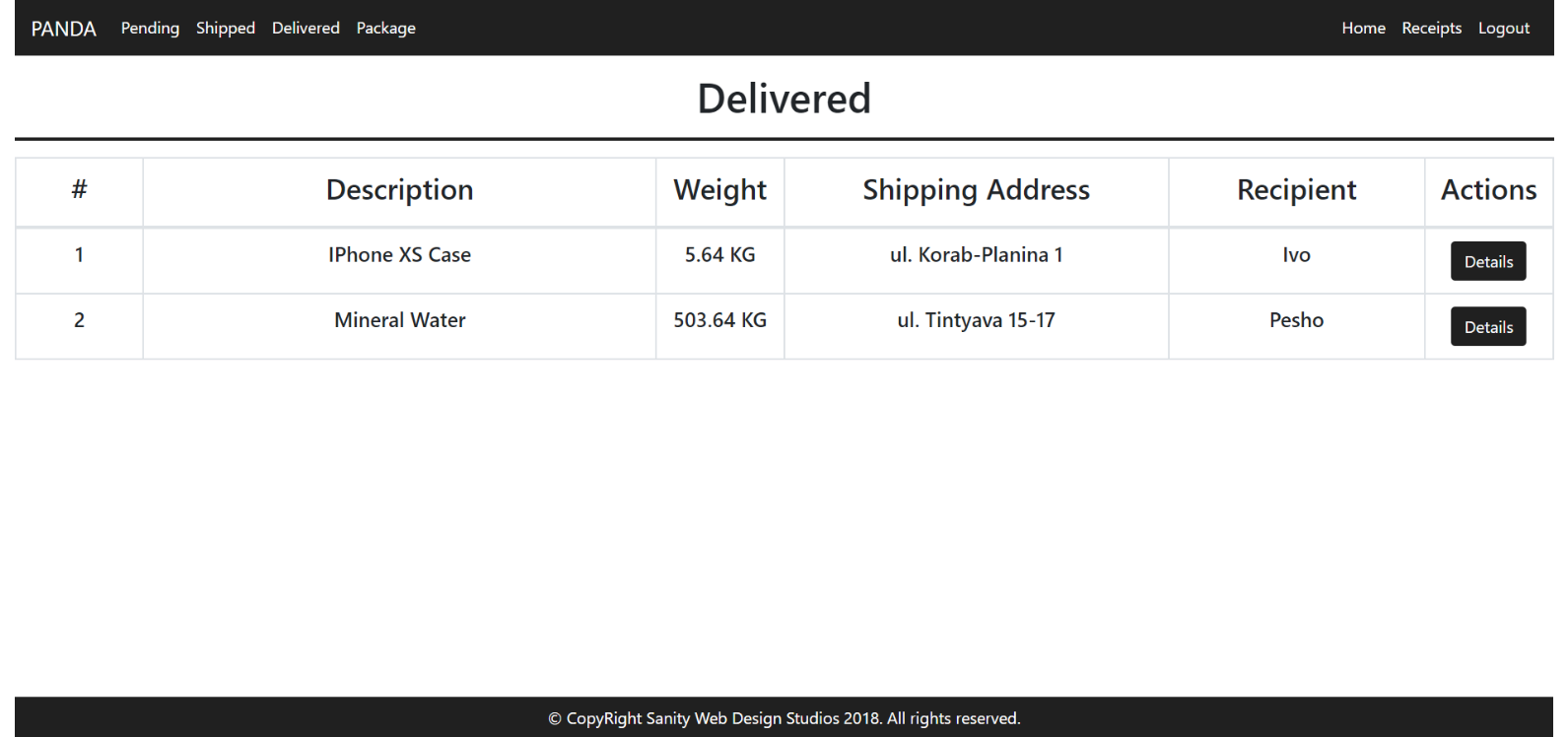
#### Pending Packages Admin Template (route=”/Packages/Pending”) (logged in admin)



#### Shipped Packages Admin Template (route=”/Packages/Shipped”) (logged in admin)



#### Delivered Packages Admin Template (route=”/Packages/Delivered”) (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-panda”, “text-panda”, “border-panda”. “border-panda” will help you with the **border** of the **blocks** on the **Index Pages**.

## Functional Requirements

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

The **application** should provide Guest (not logged in) users with the functionality to:

* Login
* Register
* View the Guest Index page

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

* Logout
* View their Packages
* View details about a Package
* View their Receipts
* View details about a Receipt

The **application** should provide Admins (logged in, with role - Admin) with the functionality to:

* Logout
* View their Packages
* View details about a Package
* View their Receipts
* View details about a Receipt
* View all Pending Packages
* View all Shipped Packages
* View all Delivered Packages
* View details about all Delivered Packages
* Ship Packages
* Deliver Packages

### Users

The **first registered** User, should be **assigned** a **role** – “Admin”. Every User after that, should have a **role** – “User”.

Users have Packages, which are **created** and **controlled** for them, by an Administrator. Users can **view** **Details** about their **own** Packages. When a Package is **delivered**, a User can **acquire** it, at which point a Receipt is **created** with **that** Package and **that** User. Users can **view** their Receipts, and **details** about each Receipt.

Administrators (role = “Admin”) are essentially like normal Users. They can also have Packages, which are **delivered**, **acquired** and they also have Receipts. Administrators can also **create** Packages for a specific User.

* They can also **view** all Pending Packages, and they can Ship them.
* They can also **view** all Shipped Packages, and they can Deliver them.
* They can also **view** all Delivered Packages, and they can view Details about them.

### Packages

When Packages are **created**, they are **created** with a Description, a Weight, a Shipping Address and a Recipient User.

* Upon creation, the Status of a Package should be set to Pending.
* Upon creation, the Estimated Delivery Date of a Package should be set to NULL.

#### Functionality

##### Pending Packages

A Pending Package, can be Shipped by an Administrator, by clicking on the [Ship] button from the Pending Packages Page. At that moment the Package Status becomes “Shipped” and the Estimated Delivery Date is to be set to a **random** of **20**-**40** days from then.

* All Pending Packages are presented on the Pending Packages Page.
* A User can view his Pending Packages on his Index Page in the Pending rectangular block.
* A User can **view details** about each one of his Pending Packages from his Index Page, by clicking on the [Details] button.

##### Shipped Packages

A Shipped Package, can be Delivered by an Administrator, by clicking on the [Deliver] button from the Shipped Packages Page. At that moment the Package Status becomes “Delivered”.

* All Shipped Packages are presented on the Shipped Packages Page.
* A User can view his Shipped Packages on his Index Page in the Shipped rectangular block.
* A User can **view details** about each one of his Shipped Packages from his Index Page, by clicking on the [Details] button.

##### Delivered Packages

A Delivered Package, can be Acquired by the Package’s Recipient, by clicking on the [Acquire] button from his Index Page. At that moment the Package Status becomes “Acquired” and a Receipt is **generated** to the User for that Package.

* All Delivered Packages are presented on the Delivered Packages Page.
* A User can view his Delivered Packages on his Index Page in the Delivered rectangular block.
* A User can Acquire each one of his Delivered Packages from his Index Page, by clicking on the [Acquire] button.

**NOTE**: The **INDEX PAGE** visualizes **ONLY** the **CURRENTLY LOGGED IN USER / ADMIN’s** **PACKAGES**.

**NOTE**: Acquired Packages are viewable only by Administrators on the Delivered Packages Page.

**NOTE**: Administrators can **view** **details** about **ALL** Delivered / Acquired Packages from the  
Delivered Packages Page, by clicking on the [Details] button.

### Receipts

Receipts are just data entities. They are **created** when a Package is Acquired by its Recipient User.   
A Receipt should be **created** with a Package and a Recipient User.

Upon creation, a Receipt’s Fee should be **set** to the Package’s Weight **multiplied** (\*) by 2.67.

Upon creation, a Receipt’s IssuedOn should be set to the **current moment**.

## 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 User Package Details page and functionality.
* Users (logged in) can access User Receipts page and functionality.
* Users (logged in) can access User Receipt Details page and functionality.
* Users (logged in) can access User Package Acquire 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 Package Create page and functionality.
* Admins (logged in) can access the Admin Pending Packages page and functionality.
* Admins (logged in) can access the Admin Shipped Packages page and functionality.
* Admins (logged in) can access the Admin Delivered Packages page and functionality.
* Admins (logged in) can access the Admin Package Ship functionality.
* Admins (logged in) can access the Admin Package Deliver functionality.