

Simple Gym System

Roles:-

1- Admin

- Admin will have access to everything in the system, **he can see any links or make any action Gym Manager and City Manager can do** with these extra functionalities
- We will have **only one admin** in the system and by **default will be seeded through laravel seeders** , with email (admin@admin.com) & pw (123456) **but we can create other admins through console command , for example i can make this**

```
php artisan create:admin --email=admin2@admin.com  
--password=123456
```

- In the left menu, we will have these links (**Gym Managers, City Managers , Users, Cities, Gyms, Training Packages, Coaches, Attendance, Buy Package For User, Revenue**) these links will allow us to do CRUD operations
- **Gym Manager & City Manager** will have (email, name , password , national_id, avatar_image)
- national_id & email must be unique | password must be at least 6 chars | avatar_image must be an image with extensions jpg,jpeg and **if there's no avatar_image then we will provide a default one**
- **Admin** will crud on **training packages**(name, price, sessions number)
Price Must Be Stored In Database in Cents And Shown In Dollars

2- City Manager

- **City Manager** can do what **Gym Manager** do with extra functionalities ... like he can see all gyms in his city and make **CRUD** on any gym or gym manager in his city

Gyms Tab

- **City Manager** will have a menu called **Gyms** where he can make **CRUD** on **Gyms**
- In **Gyms** page, we will need to show these **columns** (name, created at **date only**, cover_image) **but if the current logged in user is Admin** then we will need to **show an extra column called City Manager Name**, which is the name of the city manager who created the gym
- When **Deleting a gym** make sure it doesn't have **training sessions**

Gym Managers Tab

- **City Manager** will have a menu called **Gyms Managers** where he can make **CRUD** on **Gym Managers** and **Ban Or UnBan Gym Manager**
- When creating a gym manager we will show a drop-down of gyms and choose which gym the gym manager belongs to, **it's one gym has many gym managers relation**
- (Actions column will have **Edit , Delete**) and **if banned then there will be an action called unban**, and **if unbanned then will have an action called ban**.

3- Gym Manager

- **Gym Manager can CRUD training sessions and assign coaches to these sessions, also he can buy training package for a user through stripe**
- **training_sessions Table** will have these attributes (name, starts_at, finishes_at) where starts_at,finishes_at are dateTime , **when deleting or editing a training session make sure there are no users attended that session and when editing session we can edit date & time only if there's no user attended the session.**
- when creating a new training session **make sure it doesn't overlap with any other training session in that gym** , for example when gym manager creates a session from 03/07/2019 (01:00 pm) to 03/07/2019 (02:00 pm) , then when he tries to create a session for example from 12:00 pm to 03:00 pm it will be considered as overlapping
- **The form of creating training session** will consist of (name, day, start time, finish time, coaches ([it will be multiple select](#))) **this means it's a many to many relation** between the training session and coach

Buy Package For User Tab

- When **Gym Manager clicks on Buy Package For User Tab** it will show a form, where he chooses a user from drop-down list , and choose training package from drop down , and then enters **the visa card number through stripe, be careful when the admin changes package price or package sessions number it won't affect on bought packages**

- **Any Training Package must be bought from a gym**, so if the admin is going to buy a package for a user he must choose a gym from drop-down list , and the same will be for city manager

Revenue Tab

- We need to show the **total revenue as a card** at the top of page, **if i am logged in as gym manager, then will show the total revenue of my gym**, and if i am logged in as a **city manager** it will show the **total revenue of my city gyms**
- we need to show the **purchases history in dataTables** , we will need to show **user email, name, package name, amount the user bought with (Not The Package Price But The Amount user Paid When He Bought The Package, Cause Admin Can Edit The Package Price)**
If the logged-in user is **City Manager** then we will need to show **which gym this package is bought from**, and if the logged in user is **Admin** then we will need to show **which city this package is bought from**

Attendance Tab

- We need to show **attendance in dataTables** , the columns will be (user name, email , training session name, attendance time, attendance date), **and if i am logged in as admin i will show to which gym & city the user attended**, and if i am logged in as a **city manager** then i will need to show **which gym the user attended**

4- User (it will be API only)

- User will have an **endpoint to register**, he needs to enter his (name ,email, gender ,password, password confirmation, date of birth, profile_image) all of these fields are required, after he registers [you must send him an email verification link to his email](#)
- After User **is verified** we will need to greet him by sending an **email using laravel notifications** and it **must be Queued again Must Be Queued - Must Be Queued - Must Be Queued**

So read this page very very very carefully

<https://laravel.com/docs/master/notifications#queueing-notifications>

- User will have an **endpoint to log in**, using email & password, the response will be object that contains the user info with access token (**same response that we did in labs like sanctum package**)
- User will have an **endpoint to update his profile info. And when updating email make sure it's still unique . (and watch out of the case we did in lab 3 with uniqueness ... like if the user didn't change any info and made request to update endpoint we shouldn't throw email unique validation error)**
- User will have an **endpoint to see his remaining training sessions** for example the response will be json object {total_training_sessions:1000 , remaining__training_sessions:300}

- User will have an **endpoint to attend a training session** for example training-sessions/{id}/attend POST request and make a validation that the user can attend a session that it's **date not before today or after today** , also make sure the user has **bought enough sessions** , for example if user tries to attend a training session and have 0 bought sessions or 0 remaining sessions then a **validation error message** must tell him that **he needs to buy training sessions in order to attend**
- User will have **endpoint to show his attendance history** , and the json object will be array that contains (training session name , gym name , attendance date , attendance time)
- We need a **schedule command that runs daily** using laravel , that will send an email notification to users **who didn't log in from the past month**, and tell them that we miss you

A Code Example that will be in **console/Kernel.php**:-

```
$schedule->command('notify:users-not-logged-in-for-month')  
->daily();
```

So read Very Veryyyyyyyyy Carefully

<https://laravel.com/docs/master/scheduling#scheduling-artisan-commands>

- **Authentication** will be using [Laravel Sanctum](#) , **all routes are allowed only for authenticated** users except for register & log in, also any other routes like attend or edit profile [user must be verified](#)

System General Rules

- All tables in the system must use **DataTables Plugin** and all logical **Fields Must Be Searchable And Orderable** like (name, price , email, session number, and so on)
- All charts in the system must use **Chart Js** and **Data Retrieved By Ajax**
- All **Delete Buttons** must show a **warning** before deleting and also must be done through **ajax requests**, and if the ajax request succeeded you can then [refresh the Datatables](#)
- **Price Must Be Stored In Database in Cents And Shown In Dollars**
- All of the users **can change their profile info** like name or email, etc
- Use default laravel ui package , and make **sure the forgot password functionality works**

Bonus (**Bonus Is Approved If The Old Code Is Clean**):-

- **Try To Reduce Or Eliminate The Duplication In Code, And Make Your Code More Readable And Pleasing For Eye, If The Code Isn't Clean Then Any Bonus Won't Be Counted As Finished**

So Please Clean Your Code First Before Implementing The Bonus

Statistics

- We will have a menu Tab called **Statistics** , it will open a page that will have four charts

(Male - Female Attendance) pie chart that shows how much percent males to females attendance

(Revenue) basic line chart which will show how much money we got in 12 months for this year, this chart **by default will show current year**

(Attendance Cities) pie chart to show each city name and how many attendances have been made in this city

(Top Purchased Users) pie chart to show the **top 10 users** with there bough training sessions number, for example will show (ahmed : 20 , mohamed : 3 , ali :4 , ...)

- Deploy Your Project On Heroku
<https://mikateach.com/setting-up-laravel-5-6-on-heroku/>

- Use laravel cache to cache the drop-down list of cities
- **All Charts support filtering by year** , so i can choose a specific year on each chart and it will show its statistics
- **Gym Managers Can Export** all the users attendance as **excel** sheet

Project General Rules

- All classes must follow **StudlyCase** (pascal case) naming for example (PostsController , UsersController , Post , StorePostRequest)
- All **routes must follow resource** naming convention
<https://laravel.com/docs/master/controllers#resource-controllers>
- All **views must be divided into sub directories** like (posts/index.blade.php - users/index.blade.php - clients/index.blade.php)
- All **Middlewares should be applied on routes** (check Route::group)
- All **Models** must be Singular like (User, Gym, User, City, ... etc)
- Don't Push Any **Deadly Comments on the Git Repo** like:-
//dd(\$request); // \$user = User::first();

Project Resources:-

- For AdminLTE (**Mandatory**) after installing open starter.html file and remove unused bower_components
<https://adminlte.io/>
- DataTables have two plugins.... one will be used on the frontend , and another one will be used on backend (**Mandatory**) , also all the filtering and searching and ordering **must be done from backend**
(<https://datatables.net/manual/server-side>)
<https://datatables.net/> (Frontend)
<https://github.com/yajra/laravel-datatables> (Backend)
- For roles & permissions (**Mandatory**)
<https://github.com/spatie/laravel-permission>
- For Ban Or Unban you can use
<https://github.com/cybercog/laravel-ban>
- For Api Authentication (**Mandatory**) we will use [laravel sanctum](#)
- For Charts use Chart Js (**Mandatory If You Will Work On Bonus**)
<https://www.chartjs.org/>
- For Stripe, follow the quick start guide (**Mandatory**)
<https://stripe.com/docs/quickstart>
- For Excel, you can use(**f You Will Work On Bonus**)
<https://github.com/Maatwebsite/Laravel-Excel>