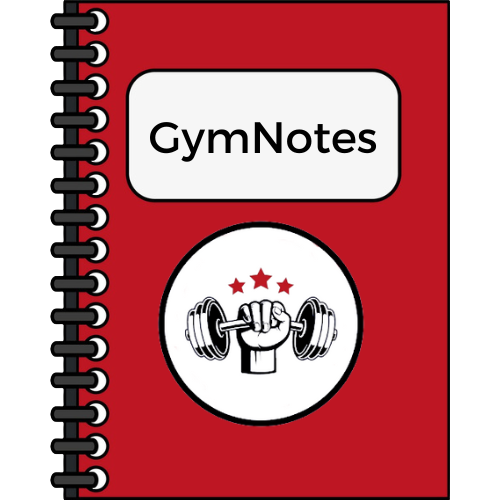
Proyecto de Fin de Grado  
GymNotes

2ºDAM



ÍNDICE

1.Portada ✔️

2.Índice ✔️

[**3. Introducción, expectativas/objetivos y antecedentes: Desarrollo de la idea inicial, de las tecnologías a utilizar y de antecedentes (aplicaciones similares).**](#_xhizqybjiqx)✔️ **4**

[**4. Descripción: Descripción del resultado obtenido: funcionalidades, organización, etc.**](#_pv0cm7vcw28j)✔️ **5**

[Crear Ejercicios:](#_ys47xdme6pg) 5

[EjercicioController.php:](#_i5tjya2m7ewi) 6

[admin\ejercicio.blade.php](#_s4hpjj7fgeii) 7

[gestionpeso.blade.php](#_ptti718ilh7h) 9

[Web.php](#_kqgbdpz1dbhb) 13

[Gestión de Peso:](#_4rqyckg7kdx4) 14

[GestionPesoController.php](#_z1aslnva68v) 15

[web.php](#_ht2dbhimjs28) 16

[Calculadora RM:](#_wix3paiqe181) 17

[web.php](#_3c7b5abauhvo) 17

[RmMaxController.php](#_hp0d49xexhiu) 18

[rm\_max.index.blade.php](#_z3pn2ygl5hb2) 19

[Músculos](#_hrt8fuy753q7) 23

[MusculosController.php](#_ykhre44jov3t) 23

[musculos.blade.php](#_bebnnxvxob72) 24

[web.php](#_v6fudelmonpj) 31

[Ejercicio](#_izgqkogd4d98) 32

[EjercicioController.php](#_60eblf2lddjp) 32

[ejercicio.blade.php](#_in7qrxjl4zxa) 33

[web.php](#_e8e8h7vb3lv1) 34

[Alimentos](#_iq23zxhyk9p4) 35

[AlimentosController.php](#_lca8mk4okkvx) 35

[alimentos.blade.php](#_i0zd3bmk6eza) 36

[web.php](#_suw9c2yzfixc) 37

[Valor Alimentos](#_quqlvbvq4lc6) 38

[ValorAlimentosController](#_msyicmc19rq) 39

[web.php](#_cr87x286sir2) 39

[valor\_alimentos\index.blade.php](#_7ur7wrwsc2f6) 40

[Ejercicios](#_6k4d72u3m984) 46

[SeriesController.php](#_dd8ren55xyai) 46

[web.php](#_g50fpzcele2) 48

[ejercicios\index.blade.php](#_artl965x8q0a) 49

[Calendario](#_buec77uq52su) 59

[EjercicioController.php](#_toxe6fq9407d) 59

[calendario.blade.php](#_r83xv3fnt03x) 61

[web.php](#_yzbdqz90i61k) 63

[**5. Instalación y preparación: Todo lo necesario para hacer funcionar el proyecto y cómo se instala. Edición de los archivos de configuración, creación y puesta en funcionamiento del control de versiones en github, etc.**](#_rzgaz27krv2a) **64**

[**6. Prototipado. Se incluye un prototipo wireframe de la aplicación.**](#_1h2jc6zbu40y)✔️ **65**

[Ejercicios](#_vdvn75hc4a3y) 65

[Macros](#_p0wjjl61a6sr) 66

[Alimentos](#_7hruqujavygf) 67

[Valores nutricionales](#_98ubzmn6gajh) 68

[**7. Diseño funcional: Descripción y diagramas de diseño funcional: entidad relación, casos de uso, diagramas de flujo...etc. Es preferible usar el estándar UML.**](#_x82rcht993rk)✔️ **69**

[**8. Desarrollo: Secuencia de desarrollo, dificultades encontradas y decisiones afrontadas. Descripción de las herramientas de control de versiones y revisión del código utilizadas.**](#_lws2rags2e8u)**👌**✔️ [**70**](#_lws2rags2e8u)

[**9. Pruebas: Descripción de las pruebas realizadas y automatización de las pruebas unitarias.**](#_ppdpufd09ep) **71**

[**10. Distribución: Tecnología de distribución y descripción del proceso.**](#_hw7kapdnqaxi)✔️ **72**

[Tecnología de distribución](#_oqmlun99ba84) 72

[Descripción del proceso](#_qydaflwpcnxc) 72

[**11. Manual: Manual de instalación y uso de la aplicación.**](#_vl9lcx3yq9xq) **73**

[**12. Conclusiones: Comparación del resultado con la idea inicial y mejoras futuras.**](#_jy4ud12rt26u)✔️ **74**

[Comparación del resultado con la idea inicial](#_289viqypft8w) 74

[Mejoras futuras](#_a1gnuxczz4g2) 75

[**13. Índice de tablas e imágenes.**](#_tk8y113b976m) **76**

[**14. Bibliografía y referencias.**](#_dyef983n6d2) **77**

# 

# 3. Introducción, expectativas/objetivos y antecedentes: Desarrollo de la idea inicial, de las tecnologías a utilizar y de antecedentes (aplicaciones similares).

Esta idea o al menos la idea abstracta que la inició surge a partir de muchos factores, dos hobbies que en este caso fueron ir al **gimnasio** y **programar**, y una necesidad, la necesidad de apuntar los resultados del **entrenamiento** (para progresar adecuadamente en el gimnasio es necesario hacerte un planning e ir apuntando todo lo que vas levantando, los tiempos, repeticiones, etc) ya que en un principio íbamos un grupito al garaje de un amigo a entrenar y para apuntarlo todo utilizamos una libreta y un boli, en el momento que perdimos el boli dejamos de apuntar el progreso porque decíamos que lo recordaremos si íbamos todas las semanas. En parte fue cierto, pero los detalles era imposible recordarlos exactamente.

En ese momento yo aún estaba a mitad de curso y con las tareas y demás se me ocurrió que podría hacer **una app para el gimnasio** para ir apuntando los avances en vez de dejarlos simplemente anotados en el smartphone y verlos sin formato y sin estilos.

Al día siguiente de la idea les comente a mis amigos la idea de la aplicación y les pareció buena idea así que pensé en cómo la iba a hacer, lo primero que se me ocurrió fue hacerla con **Android Studio** y **Kotlin** ya que sería lo más óptimo para apuntar los avances al momento y además era lo que estaba estudiando en el momento, así que no se me haría pesado empezar a aprender un nuevo lenguaje e incluso podría mejorar algo en el lenguaje.

Por desgracia tuve que ir aplazando la idea debido a que fui acumulando tareas y tenía que recuperar exámenes, aún así la idea no desapareció. Después de aprobar las asignaturas antes del verano podría haberla hecho en este mismo verano pero pensé que ya que la aplicación iba a ser más sería la podría dejar para la idea del proyecto.

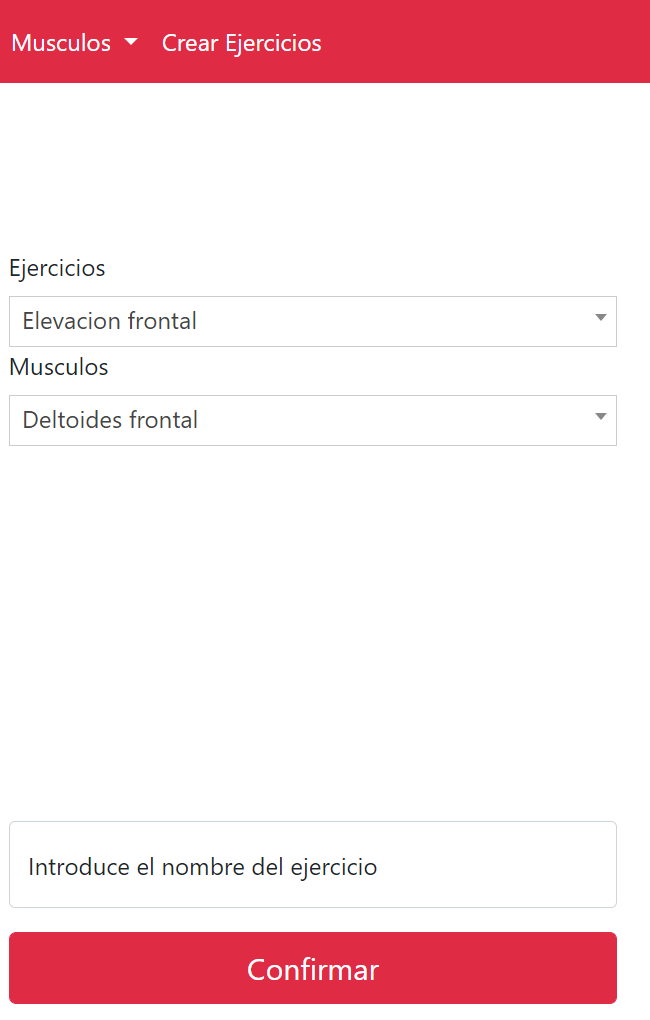
En un principio iba a ser **Python** con **Qt** para una versión de escritorio y luego utilizar un programa para crear una apk para android y si no me gustaba hacerla en **Android Studio** con **Kotlin**, pero la idea cambió en el último momento debido a que en las prácticas se utilizaba **PHP** y mi tutor (David Romero) me recomendó hacer el proyecto con las tecnologías de las prácticas ya que me vendría bien usar el conocimiento de las prácticas en el proyecto y viceversa. En el primer momento pensé en que podría ser muy complicado aprender un nuevo lenguaje y un framework en tres meses o al menos lo suficiente para hacer un trabajo en condiciones pero al final acepté porque me quise retar y además era lo más lógico para el proyecto, ya que si hacia una página web tendría las versiones de escritorio y smartphone y además sin la necesidad de descargar nada para el usuario.

Al final decidí hacerlo con **PHP**, el framework **Laravel** y para los estilos utilicé **Bootstrap**, aparte de eso quise utilizar páginas y apps que utilizo para guiarme de cómo podría hacer la página y sobretodo, que necesita. Mis dos bases son la web <https://musclewiki.com/> para los ejercicios y para la alimentación la app de Macros de google play <https://play.google.com/store/apps/details?id=com.josmantek.macros&hl=es&gl=US>.

# 4. Descripción: Descripción del resultado obtenido: funcionalidades, organización, etc.

El resultado obtenido no ha sido el esperado, por una parte no he conseguido hacer el lector de códigos de barra (cosa que quiero hacer con el tiempo en JavaScript), pero por otro lado he conseguido añadirle más funcionalidades de las que creía en un principio, como por ejemplo:

### Crear Ejercicios:



Implementa un modo administrador (la pestaña “Crear Ejercicios” solo está accesible para los usuarios que se seleccionen, en este caso será el primero que se crea desde el seeder aunque se puede modificar desde la base de datos), el apartado de ejercicios sirve para ver los ejercicios actuales (confirmar que se ha introducido un nuevo ejercicio), luego el apartado de los músculos sirve junto al label de abajo para crear un ejercicio llamado como el nombre escrito y el músculo (principal del ejercicio) seleccionado.

#### EjercicioController.php:

La pestaña de “Crear Ejercicios” se divide en dos partes desde el controlador, la función de getEjercicios

| public function getEjercicios()  {  $datos = [];  $ejercicios = Ejercicio::all();  $musculos = Musculo::all();  return view('admin.ejercicio', ['ejercicios' => $ejercicios, 'musculos' => $musculos, 'datos' => $datos]);  } |
| --- |

Esta función se encarga de mostrar todos los ejercicios y músculos que se encuentran en la base de datos, donde se muestran en un select box para seleccionar y buscar los que quieras, en la página que se encuentra en resources\views\admin\ejercicio.blade.php.

| public function postEjercicio(Request $request)  {  $ejercicio = new Ejercicio();  $ejercicio->musculo\_id = $request->musculo\_id;  $ejercicio->nombre = $request->ejercicio;  $ejercicio->descripcion = $request->descripcion;  $ejercicio->foto = $request->foto;  $ejercicio->save();    $datos = [];  $ejercicios = Ejercicio::all();  $musculos = Musculo::all();  return view('admin.ejercicio', ['ejercicios' => $ejercicios, 'musculos' => $musculos, 'datos' => $datos]);  } |
| --- |

Esta función se activa cuando se aprieta el botón de confirmar en la página, lo que hace es crear un nuevo ejercicio con los datos que se han introducido (que en este momento solo es el nombre) y retorna de nuevo la página.

#### admin\ejercicio.blade.php

| <!DOCTYPE html> <html>  <head>  <link href="//maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css" rel="stylesheet" id="bootstrap-css">  <script src="//maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"></script>  <script src="//cdnjs.cloudflare.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>  <title>Crear Ejercicios</title>  <style>  .select2-container .select2-selection--single {  height: 34px !important;  }   .select2-container--default .select2-selection--single {  border: 1px solid #ccc !important;  border-radius: 0px !important;  }   \* {  webkit-appearance: none;  }  .es{  margin-right:20px;  }  </style> </head>  <body>  @extends('layouts.app-master')   @section('content')   <link href="https://cdnjs.cloudflare.com/ajax/libs/select2/4.0.6-rc.0/css/select2.min.css" rel="stylesheet" />  <script src="https://cdnjs.cloudflare.com/ajax/libs/select2/4.0.6-rc.0/js/select2.min.js"></script>  <div class="container">  <div class="row">   <form action=crear-ejercicios method="POST" class="col-md-4">  @csrf  <div>  <div>  <label class="es">Ejercicios</label>  <select name="ejercicio\_id" class="form-control select2">  @foreach ($ejercicios as $ejercicio)  <option value="{{$ejercicio->id}}">{{$ejercicio->nombre}}</option>  @endforeach  </select>  </div>   <div>  <label class="es">Musculos</label>  <select name="musculo\_id" class="form-control select2">  @foreach ($musculos as $musculo)  <option value="{{$musculo->id}}">{{$musculo->nombre}}</option>  @endforeach  </select>  </div>  </div>  <div class="form-group form-floating mb-3" style="margin-top: 250px">  <input id='ejercicio' type="text" class="form-control" name="ejercicio"  value="{{ old('ejercicio') }}" placeholder="ejercicio" required="required" min="2">  <label for="floatingPassword">Introduce el nombre del ejercicio</label>  @if ($errors->has('ejercicio'))  <span class="text-danger text-left">{{ $errors->first('ejercicio') }}</span>  @endif  </div>  <button class="w-100 btn btn-lg btn-danger" type="submit">Confirmar</button>  </form>   </div>  </div>  <script>  $('.select2').select2();  </script>  @endsection </body>  </html> |
| --- |

Esta es la página que carga cuando accedes a crear ejercicios.

Todas las páginas extienden de layouts.app-master, donde se encuentra el navbar y el footer (actualmente no está incluido), para así no tener que estar copiando y pegando estos elementos todo el tiempo.

Una cosa que trae php es que puedes utilizar funciones dentro de los html para cargar los datos, yo los cargo mediante los foreach.

#### Web.php

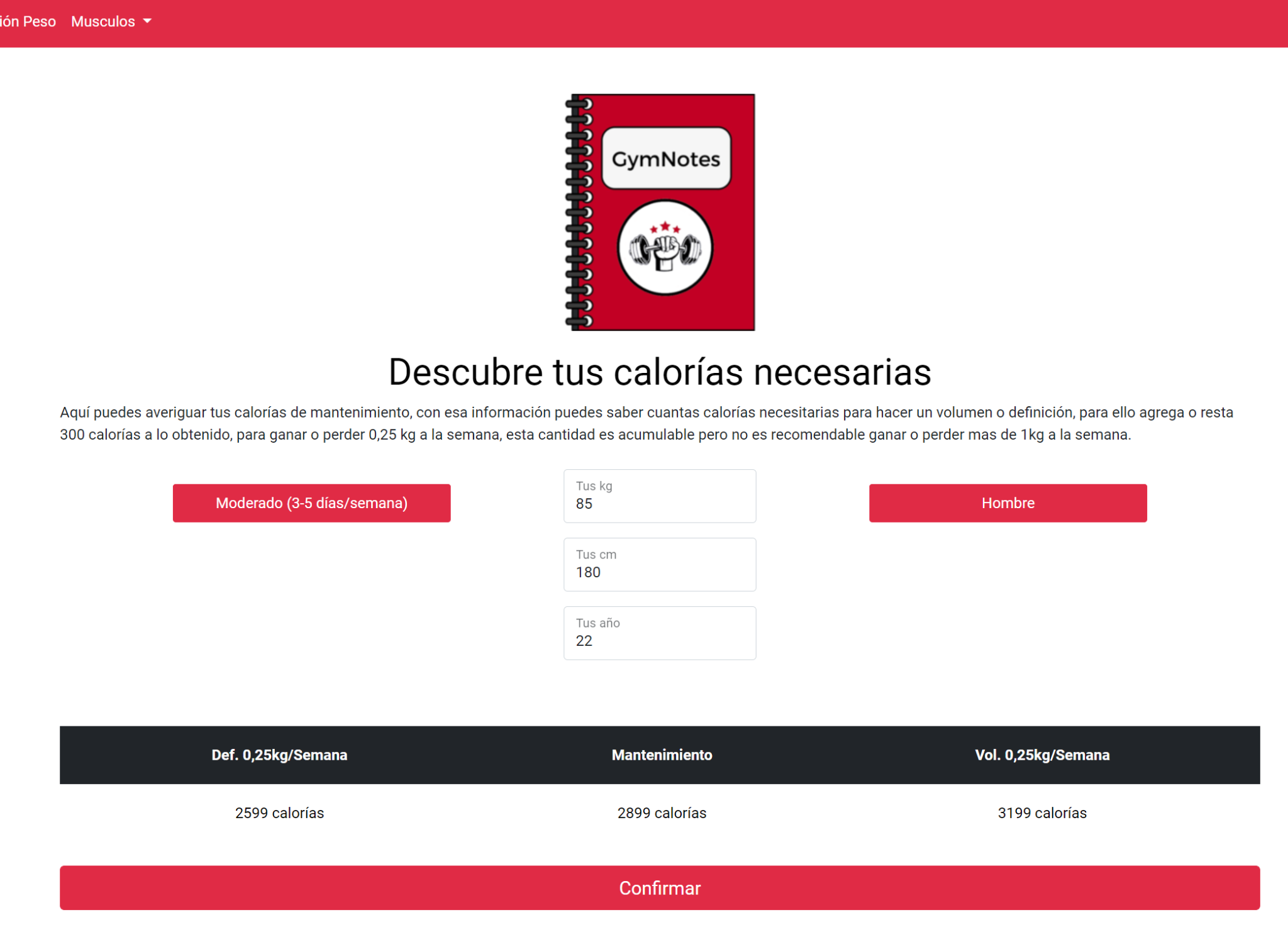
Para que la página pueda acceder a los métodos y asignarle una url hay que asignarla en el archivo web.php de la siguiente manera:

| Route::get('/crear-ejercicios', [EjercicioController::class, 'getEjercicios']); Route::post("/crear-ejercicios", [EjercicioController::class, 'postEjercicio']); |
| --- |

De esta manera crea una ruta (todas parten de la ruta base en mi caso localhost:8000) con la url asignada por el método y su clase.

### 

### Gestión de Peso:



Este gestionador de peso sirve para hacerte una idea de las kcal que consume tu cuerpo al día, de esta manera puedes hacerte una idea de cuántas kcal deberías consumir al día si quieres adelgazar, aumentar de peso, o mantenerte.

Rellenando los datos obtienes las kcal de mantenimiento, a partir de ahí se suma o resta 300 kcal para adelgazar o engordar, esta cantidad es acumulable pero no es recomendable pasar de las 1000 kcal de diferencia.

#### GestionPesoController.php

La pestaña de Gestión de peso está controlada por una sola función en este archivo.

| <?php  namespace App\Http\Controllers;  use Illuminate\Http\Request;  class GestionPesoController extends Controller {   // https://exafitstore.myshopify.com/blogs/news/cuantas-calorias-necesito-para-volumen-definicion-o-perdida-de-peso#:~:text=HOMBRES%3A%20TMB%20%3D%20(10%20X,edad%20en%20a%C3%B1os)%20%2D%20161.   public function getGestionPeso(Request $request)  {  $datos = [];  if (isset($request->altura) && isset($request->peso) && isset($request->edad) && ($request->tipoTMB) && ($request->genero)) {  $operacionTMB = (10 \* $request->peso) + (6.25 \* $request->altura) - (5 \* $request->edad);   $genero = [  'hombre' => +5,  'mujer' => -161,  'calorias\_generales' => -70  ];    $operacionGenero = $operacionTMB + $genero[$request->genero];    $tipoTMB = [  'Nada' => 1.2,  'Ligero' => 1.375, // 1-3 dias/semana  'Moderado' => 1.55, // 3-5 dias/semana  'Fuerte' => 1.735, // 6-7 dias/semana  'Muy Fuerte' => 1.9, // dos veces al día, entrenamientos duros  ];   $caloriasTMB = round($operacionGenero \* $tipoTMB[$request->tipoTMB]);   $volumen = $caloriasTMB + 300;  $definicion = $caloriasTMB - 300;   $datos = [  'volumen' => $volumen,  'definicion' => $definicion,  'caloriasTMB' => $caloriasTMB,  ];  }  return view('gestionpeso', ['datos' => $datos]);  } } |
| --- |

El controlador se recibe las request del usuario, en el caso de que entre a la página por primera vez no llegará las request por lo tanto solo mandará la página html de gestionpeso.blade.php y cuando rellene los campos y los envíe se actualizarán los resultados con la request del usuario.

#### web.php

Su ruta es la siguiente:

| Route::get('/gestion-peso', [GestionPesoController::class, 'getGestionPeso'])->name('gestionPeso.calculadora'); |
| --- |

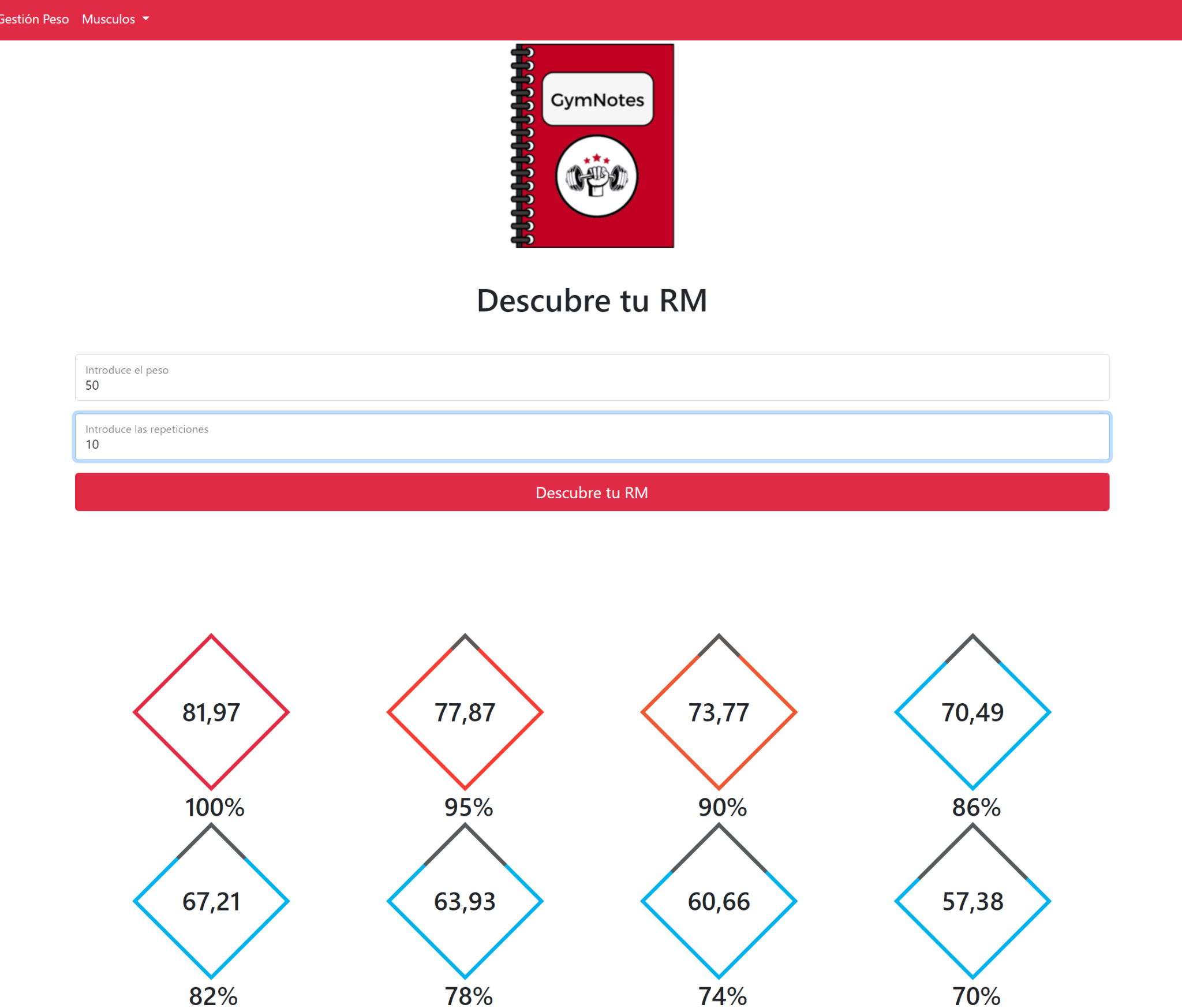
#### gestionpeso.blade.php

| <!doctype html> <html lang="es">  <head>  <meta charset="UTF-8">  <meta name="viewport" content="initial-scale=1,width=device-width">  <meta name="format-detection" content="telephone=no">  <title>Gestión de Peso</title>  <link href='https://fonts.googleapis.com/css?family=Roboto:400,100,300,700' rel='stylesheet' type='text/css'>  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/lato-font/3.0.0/css/lato-font.min.css">  <script defer src="js/calendario.js"></script>  <link href="css/gestionpeso.css" rel="stylesheet">   <style>  .toggle-checkbox:checked {  @apply: right-0 border-green-400;  right: 0;  border-color: #68D391;  }   .toggle-checkbox:checked+.toggle-label {  @apply: bg-green-400;  background-color: #68D391;  }  </style> </head>  <body>  @extends('layouts.app-master')   @section('content')  <form method="get" action="{{ route('gestionPeso.calculadora') }}">  <div class="centrado">  <a href="/"><img class="centrado" src="{!! url('gymnotes.ico') !!}" alt="GymNotes logo"></a>  </div>   <h1 style="margin-top: 20px;">Descubre tus kcal necesarias</h1>  <p style="margin-bottom: 25px;">Aquí puedes averiguar tus kcal de mantenimiento, con esa información puedes  saber cuantas kcal necesitarias  para hacer un volumen o definición, para ello agrega o resta 300 kcal a lo obtenido, para ganar o perder  0,25 kg  a la semana, esta cantidad es acumulable pero no es recomendable ganar o perder mas de 1kg a la semana.  </p>   <div style="display: flex">   <div class="list-choice" style="margin-bottom: 220px">  <div class="list-choice-title">Actividad:</div>  <div class="list-choice-objects">  <label>  <input type="radio" name="tipoTMB" value="Nada" required="required"/> <span>Nada</span>  </label>  <label>  <input type="radio" name="tipoTMB" value="Ligero" /> <span>Ligero (1-3 días/semana)</span>  </label>  <label>  <input type="radio" name="tipoTMB" value="Moderado" /> <span>Moderado (3-5 días/semana)</span>  </label>  <label>  <input type="radio" name="tipoTMB" value="Fuerte" /> <span>Fuerte (6-7 días/semana)</span>  </label>  <label>  <input type="radio" name="tipoTMB" value="Muy Fuerte" /> <span>Muy fuerte (Dos veces al  día)</span>  </label>  </div>  </div>   <div style="display: inline-block">   <div class="form-group form-floating mb-3">  <input id='peso' type="number" class="form-control" name="peso" value="{{ old('peso') }}"  placeholder="peso" required="required|numeric|min:2" min="1" autofocus>  <label for="floatingName">Tus kg</label>  @if ($errors->has('peso'))  <span class="text-danger text-left">{{ $errors->first('peso') }}</span>  @endif  </div>   <div class="form-group form-floating mb-3">  <input id='altura' type="number" class="form-control" name="altura" value="{{ old('altura') }}"  placeholder="altura" required="required" min="2">  <label for="floatingPassword">Tu altura</label>  @if ($errors->has('altura'))  <span class="text-danger text-left">{{ $errors->first('altura') }}</span>  @endif  </div>   <div class="form-group form-floating mb-3">  <input id='edad' type="number" class="form-control" name="edad" value="{{ old('edad') }}"  placeholder="edad" required="required" min="2">  <label for="floatingPassword">Tus años</label>  @if ($errors->has('edad'))  <span class="text-danger text-left">{{ $errors->first('edad') }}</span>  @endif  </div>   </div>   <div class="list-choice" style="margin-bottom: 20px">  <div class="list-choice-title">Kcal para:</div>  <div class="list-choice-objects">  <label>  <input type="radio" name="genero" value="hombre" required="required"/> <span>Hombre</span>  </label>  <label>  <input type="radio" name="genero" value="mujer" required="required"/> <span>Mujer</span>  </label>  <label>  <input type="radio" name="genero" value="calorias\_generales" required="required"/> <span>Kcal. generales</span>  </label>  </div>  </div>  </div>   @if ($datos != null)  <table>  <tr class="bg-dark" style="padding: 25px">  <th>Def. 0,25kg/Semana</th>  <th>Mantenimiento</th>  <th>Vol. 0,25kg/Semana</th>  </tr>  <tr class="bg-success">  <td>{{ $datos['caloriasTMB'] - 300 }} kcal</td>  <td>{{ $datos['caloriasTMB'] }} kcal</td>  <td>{{ $datos['caloriasTMB'] + 300 }} kcal</td>  </tr>  </table>  @endif   <button href="{{ route('gestionPeso.calculadora') }}" class="w-100 btn btn-lg btn-danger" onclick="enviarRM"  type="submit">Confirmar</button>  </form>      @endsection </body>  </html> |
| --- |

Lo importante de aquí es que me traigo la variable caloriasTMB y la sumo y resto para obtener más tablas.

### 

### Calculadora RM:



Esta calculadora sirve para saber cual es el máximo peso que puedes levantar, con tus datos, sirve sobre todo para ejercicios compuestos que requieran de mucho peso y es recomendable no pasar del 90%, y menos aún si no tienes a alguien ayudándote.

#### web.php

| Route::get('/rm-max', [RmMaxController::class, 'rmCalculator'])->name('rmmax.rmcalculator'); |
| --- |

El atributo name se utiliza para llamar más rápido a la ruta.

#### RmMaxController.php

Como en la anterior pestaña esta se controla con una función y es esta:

| <?php  namespace App\Http\Controllers;  use Illuminate\Http\Request;  class RmMaxController extends Controller {  const REPS = [1, 0.95, 0.90, 0.86, 0.82, 0.78, 0.74, 0.70, 0.65, 0.61, 0.57, 0.53 ];   public function rmCalculator(Request $request)  {  //https://www.vitonica.com/entrenamiento/como-calcular-los-porcentajes-de-cargas-submaximas  // https://echarts.apache.org/  // https://www.youtube.com/watch?v=wkui8RT5mFQ   $resultados = [];  if (isset($request->rep) && isset($request->peso)){  $reps = RmMaxController::REPS;    $valor = $reps[$request->rep - 1];  $rm = $request->peso;  $rmMax = $rm/$valor;       for ($i=0; $i < count($reps); $i++){  $calculadora = number\_format((float)$rmMax \* $reps[$i], 2, ',', '');  array\_push($resultados, $calculadora);  }  }    return view('rm\_max.index', ['resultados' => $resultados]);  } } |
| --- |

La constante de reps indican el porcentaje utilizado y luego se utiliza la fórmula que se indica en la página comentada que es la que utilizo tomo de referencia.

#### rm\_max.index.blade.php

Este es el html que utilizo para la pestaña Calculadora RM.

| <!DOCTYPE html> <html>  <head>  <link href='https://fonts.googleapis.com/css?family=Roboto:400,100,300,700' rel='stylesheet' type='text/css'>   <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css">  <link href="css/bootstrap.min.css" rel="stylesheet">   <link href="{{ asset('css/rmmax.css') }}" rel="stylesheet" type="text/css">   <meta name="viewport" content="width=device-width, initial-scale=1">   <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>   <title>Calculadora-RM</title> </head>  <body>  @extends('layouts.app-master')   @section('content')  <div class="inicio">  <form method="get" action="{{ route('rmmax.rmcalculator') }}">  <div class="centrado">   <a href="/"><img src="{!! url('gymnotes.ico') !!}" alt="GymNotes logo"></a>  </div>   <h1>Descubre tu RM</h1>   <div class="form-group form-floating mb-3">  <input id='peso' type="number" class="form-control" name="peso" value="{{ old('peso') }}" placeholder="peso" required="required|numeric|min:2" min="1" autofocus>  <label for="floatingName">Introduce el peso</label>  @if ($errors->has('peso'))  <span class="text-danger text-left">{{ $errors->first('peso') }}</span>  @endif  </div>   <div class="form-group form-floating mb-3">  <input id='rep'type="number" class="form-control" name="rep" value="{{ old('rep') }}" placeholder="rep"  required="required" max="12" min="2">  <label for="floatingPassword">Introduce las repeticiones</label>  @if ($errors->has('rep'))  <span class="text-danger text-left">{{ $errors->first('rep') }}</span>  @endif  </div>   <button href="{{ route('rmmax.rmcalculator') }}" class="w-100 btn btn-lg btn-danger" onclick="enviarRM"  type="submit">Descubre  tu RM</button>   </form>   @if ($resultados != null)   <section class="progress-section" >  <div class="container">  <div class="row">  </div>    {{-- Cirulares --}}  <section class="progress-section" >  <div class="container">  <div class="row">    <div class="col-md-3">  <div class="progress-bars-100">  <h2>{{ $resultados[0]}}</h2>  </div>  <h2 style="margin-top: 40px;margin-left: 10px">100%</h2>  <!--progress-bars-->  </div>   <div class="col-md-3">  <div class="progress-bars-95">  <h2>{{ $resultados[1]}}</h2>  </div>  <h2 style="margin-top: 40px;margin-left: 10px">95%</h2>  <!--progress-bars-->  </div>   <div class="col-md-3">  <div class="progress-bars-90">  <h2>{{ $resultados[2]}}</h2>  </div>  <h2 style="margin-top: 40px;margin-left: 10px">90%</h2>  <!--progress-bars-->  </div>    <div class="col-md-3">  <div class="progress-bars-86">  <h2>{{ $resultados[3]}}</h2>  </div>  <h2 style="margin-top: 40px;margin-left: 10px">86%</h2>  <!--progress-bars-->  </div>  </div>  </div>   <div class="container">  <div class="row">    <div class="col-md-3">  <div class="progress-bars-82">  <h2>{{ $resultados[4]}}</h2>  </div>  <h2 style="margin-top: 40px;margin-left:2%">82%</h2>  <!--progress-bars-->  </div>   <div class="col-md-3">  <div class="progress-bars-78">  <h2>{{ $resultados[5]}}</h2>  </div>  <h2 style="margin-top: 40px;margin-left: 10px">78%</h2>  <!--progress-bars-->  </div>   <div class="col-md-3">  <div class="progress-bars-74">  <h2>{{ $resultados[6]}}</h2>  </div>  <h2 style="margin-top: 40px;margin-left: 10px">74%</h2>  <!--progress-bars-->  </div>    <div class="col-md-3">  <div class="progress-bars-70">  <h2>{{ $resultados[7]}}</h2>  </div>  <h2 style="margin-top: 40px;margin-left: 10px">70%</h2>  <!--progress-bars-->  </div>  </div>  </div>   <div class="container" ">  <div class="row">    <div class="col-md-3">  <div class="progress-bars-65">  <h2>{{ $resultados[8]}}</h2>  </div>  <h2 style="margin-top: 40px;margin-left:2%">65%</h2>  <!--progress-bars-->  </div>   <div class="col-md-3">  <div class="progress-bars-61">  <h2>{{ $resultados[9]}}</h2>  </div>  <h2 style="margin-top: 40px;margin-left: 10px">61%</h2>  <!--progress-bars-->  </div>   <div class="col-md-3">  <div class="progress-bars-57">  <h2>{{ $resultados[10]}}</h2>  </div>  <h2 style="margin-top: 40px;margin-left: 10px">57%</h2>  <!--progress-bars-->  </div>    <div class="col-md-3">  <div class="progress-bars-53">  <h2>{{ $resultados[11]}}</h2>  </div>  <h2 style="margin-top: 40px;margin-left: 10px">53%</h2>  <!--progress-bars-->  </div>  </div>  </div>  </section>  @endif  </div>  @endsection </body>  </html> |
| --- |

### Músculos



La pestaña de músculos es solo una guía para ver los músculos que trabajamos con los ejercicios y para qué sirven, también puede servir para ver la dirección de las fibras musculares y así trabajar en un rango de movimiento que lo favorezca.

#### MusculosController.php

| public function musculos()  {  setlocale(LC\_ALL, 'es\_ES');  return view('musculos');  } |
| --- |

Simplemente muestra la página de los músculos, como son datos de prueba no carga nada de la base de datos.

Para ver un ejemplo cargando los datos de la bd puedes ver musculos\index.php que es el antiguo archivo html de músculos pero al final opté por este ya que está más orientado a dispositivo móvil.

#### musculos.blade.php

| <!DOCTYPE html> <html>  <head>  <link href='https://fonts.googleapis.com/css?family=Roboto:400,100,300,700' rel='stylesheet' type='text/css'>   <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css">  <link href="css/bootstrap.min.css" rel="stylesheet">   {{--  <link href="{{ asset('css/rmmax.css') }}" rel="stylesheet" type="text/css"> --}}   <meta name="viewport" content="width=device-width, initial-scale=1">   <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>   <link href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" rel="stylesheet" />  <script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.3.1/jquery.min.js">  </script>  <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js">  </script>  <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js">  </script>  <title>Músculos</title>    <style>  .card-img-top {  width: 700px !important;  height: 650px !important;  align-self: center !important;  }   @media (max-width: 800px) {  .card-img-top {  width: 100% !important;  height: 100% !important;  }  }   a.reset-color-enlace {  color: white;  text-decoration: none;  }   a.reset-color-enlace:hover {  color: white;  }   h1 {  /\* font-size-adjust: initial;  text-align: center;  white-space: nowrap;  overflow: hidden;  text-overflow: ellipsis;  padding-bottom: 5%;  padding-top: 5%; \*/   border: black solid 5px;  border-radius: 22px;  box-shadow: 6px 20px 10px black;  background-color: #0b5ed7;  color: white;  text-align: center;  margin-bottom: 25px !important;  margin-top: 45px !important;   }  </style> </head>  <body>  @extends('layouts.app-master')   @section('content')   <h1><a class="reset-color-enlace" href="/ejercicios#pecho">Pecho</a></h1>  <div id="carouselExampleIndicators" class="carousel slide" data-ride="carousel">  <ol class="carousel-indicators">  <li data-target="#carouselExampleIndicators" data-slide-to="0" class="active">  </li>  <li data-target="#carouselExampleIndicators" data-slide-to="1">  </li>  <li data-target="#carouselExampleIndicators" data-slide-to="2">  </li>  </ol>  <div class="carousel-inner">  <div class="carousel-item active">  <div class="card carta">  <img class="card-img-top carta"  src="https://media.istockphoto.com/photos/chest-pectoralis-major-minor-anatomy-muscles-picture-id179538762?k=20&m=179538762&s=612x612&w=0&h=mg49byyQ8tmU2Afiz\_g\_00cgrdOJpS4X6Heu595AwDQ="  alt="Card image cap">  <div class="card-body">  <h5 class="card-title">Pectoral Mayor</h5>  <p class="card-text">El pectoral mayor trabaja directamente en el hombro, específicamente  realizando  rotación interna y aducción, es decir dirige el brazo hacia la parte medial del cuerpo.</p>  <a href="https://www.fisioterapia-online.com/glosario/musculos-pectoral-mayor-y-menor">info:</a>  </div>  </div>  </div>  <div class="carousel-item">  <div class="card carta">  <img class="card-img-top carta"  src="https://www.shutterstock.com/image-illustration/3d-medical-illustration-explanation-smaller-260nw-1930719302.jpg"  alt="Card image cap">  <div class="card-body">  <h5 class="card-title">Pectoral Menor</h5>  <p class="card-text">Este musculo actúa en diversos movimientos realizados por el hombro, la  escapula, interviniendo incluso durante la respiración. Teniendo como principal función: la  de  tirar del hombro hacia abajo  y hacia adelante, rotarla escapula y elevar el borde medial de la misma alejándola de las  costillas.  También actúa simultáneamente en: La respiración profunda, depresión del hombro, rotación  hacia  abajo con el romboide y como elevador de la escápula.</p>  <a  href="https://www.cuidate-murcia.com/el-pectoral-menor-anatomia-funcion-y-sus-patologias/#:~:text=El%20pectoral%20menor%20es%20un,como%20el%20de%20hombros%20adelantados">info:</a>  </div>  </div>  </div>  <div class="carousel-item" id="image2">  <div class="card carta">  <img class="card-img-top carta" src="https://i.ytimg.com/vi/fUS7A0usLrA/maxresdefault.jpg"  alt="Card image cap">  <div class="card-body">  <h5 class="card-title">Serrato Anterior</h5>  <p class="card-text">Este musculo actúa en diversos movimientos realizados por el hombro, la  escapula, interviniendo incluso durante la respiración. Teniendo como principal función: la  de  tirar del hombro hacia abajo  y hacia adelante, rotarla escapula y elevar el borde medial de la misma alejándola de las  costillas.  También actúa simultáneamente en: La respiración profunda, depresión del hombro, rotación  hacia  abajo con el romboide y como elevador de la escápula.</p>  <a href="https://www.fisioterapia-online.com/glosario/musculos-pectoral-mayor-y-menor">info:</a>  </div>  </div>  </div>  </div>  <a class="carousel-control-prev" href="#carouselExampleIndicators" role="button" data-slide="prev">  <span class="carousel-control-prev-icon" aria-hidden="true">  </span>  <span class="sr-only">Previous</span>  </a>  <a class="carousel-control-next" href="#carouselExampleIndicators" role="button" data-slide="next">  <span class="carousel-control-next-icon" aria-hidden="true"></span>  <span class="sr-only">Next</span>  </a>  </div>  <script>  //handle Bootstrap carousel slide event  $(".carousel").attr("data-interval", "15000");  </script>    <h1><a class="reset-color-enlace" href="/ejercicios#espalda">Espalda</a></h1>  <div id="carouselExampleIndicators2" class="carousel slide" data-ride="carousel">  <ol class="carousel-indicators">  <li data-target="#carouselExampleIndicators2" data-slide-to="0" class="active">  </li>  <li data-target="#carouselExampleIndicators2" data-slide-to="1">  </li>  <li data-target="#carouselExampleIndicators2" data-slide-to="2">  </li>  <li data-target="#carouselExampleIndicators2" data-slide-to="3">  </li>  </ol>  <div class="carousel-inner">  <div class="carousel-item active">  <div class="card carta">  <img class="card-img-top carta"  src="https://media.istockphoto.com/id/1313854778/de/foto/menschliches-muskelsystem-torso-muskeln-trapezius-muskel-anatomie.jpg?s=612x612&w=0&k=20&c=rd89GLIih4S0np4WPZxRz46xHgVhQ2rVppGHgNslLu8="  alt="Card image cap">  <div class="card-body">  <h5 class="card-title">Trapecios</h5>  <p class="card-text">El pectoral mayor trabaja directamente en el hombro, específicamente  realizando  rotación interna y aducción, es decir dirige el brazo hacia la parte medial del cuerpo.</p>  <a href="https://www.fisioterapia-online.com/glosario/musculos-pectoral-mayor-y-menor">info:</a>  </div>  </div>  </div>  <div class="carousel-item">  <div class="card carta">  <img class="card-img-top carta"  src="https://c8.alamy.com/zoomses/9/ea225f8d69314184a1470a4d4593f6c1/e8cjbg.jpg"  alt="Card image cap">  <div class="card-body">  <h5 class="card-title">Dorsales</h5>  <p class="card-text">Este musculo actúa en diversos movimientos realizados por el hombro, la  escapula, interviniendo incluso durante la respiración. Teniendo como principal función: la  de  tirar del hombro hacia abajo  y hacia adelante, rotarla escapula y elevar el borde medial de la misma alejándola de las  costillas.  También actúa simultáneamente en: La respiración profunda, depresión del hombro, rotación  hacia  abajo con el romboide y como elevador de la escápula.</p>  <a  href="https://www.cuidate-murcia.com/el-pectoral-menor-anatomia-funcion-y-sus-patologias/#:~:text=El%20pectoral%20menor%20es%20un,como%20el%20de%20hombros%20adelantados">info:</a>  </div>  </div>  </div>  <div class="carousel-item">  <div class="card carta">  <img class="card-img-top carta" src="https://image.shutterstock.com/image-illustration/this-3d-illustration-shows-view-260nw-1849652074.jpg"  alt="Card image cap">  <div class="card-body">  <h5 class="card-title">Romboide</h5>  <p class="card-text">Este musculo actúa en diversos movimientos realizados por el hombro, la  escapula, interviniendo incluso durante la respiración. Teniendo como principal función: la  de  tirar del hombro hacia abajo  y hacia adelante, rotarla escapula y elevar el borde medial de la misma alejándola de las  costillas.  También actúa simultáneamente en: La respiración profunda, depresión del hombro, rotación  hacia  abajo con el romboide y como elevador de la escápula.</p>  <a href="https://www.fisioterapia-online.com/glosario/musculos-pectoral-mayor-y-menor">info:</a>  </div>  </div>  </div>  <div class="carousel-item">  <div class="card carta">  <img class="card-img-top carta" src="https://media.istockphoto.com/id/1289573584/es/foto/quadratus-lumborum-anatom%C3%ADa-muscular-para-concepto-m%C3%A9dico-ilustraci%C3%B3n-3d.jpg?s=612x612&w=0&k=20&c=lwNUyAqg23VLsx3mUiv4skXdjgHaMtwI7lG-LMXwbN0="  alt="Card image cap">  <div class="card-body">  <h5 class="card-title">Cuadrado Lumbar</h5>  <p class="card-text">Este musculo actúa en diversos movimientos realizados por el hombro, la  escapula, interviniendo incluso durante la respiración. Teniendo como principal función: la  de  tirar del hombro hacia abajo  y hacia adelante, rotarla escapula y elevar el borde medial de la misma alejándola de las  costillas.  También actúa simultáneamente en: La respiración profunda, depresión del hombro, rotación  hacia  abajo con el romboide y como elevador de la escápula.</p>  <a href="https://www.fisioterapia-online.com/glosario/musculos-pectoral-mayor-y-menor">info:</a>  </div>  </div>  </div>  </div>  <a class="carousel-control-prev" href="#carouselExampleIndicators2" role="button" data-slide="prev">  <span class="carousel-control-prev-icon" aria-hidden="true">  </span>  <span class="sr-only">Previous</span>  </a>  <a class="carousel-control-next" href="#carouselExampleIndicators2" role="button" data-slide="next">  <span class="carousel-control-next-icon" aria-hidden="true"></span>  <span class="sr-only">Next</span>  </a>  </div>   @endsection </body>  </html> |
| --- |

Básicamente la idea es meter a todos los músculos del mismo grupo muscular en el mismo carousel y que vayan pasando automáticamente, los datos ya se cargarán desde la base de datos cuando tenga las imágenes (me gustaría aprender un poquito de blender e intentar hacer yo el modelo o que me ayude un amigo con ese tema).

#### web.php

| Route::get('/musculos', [MusculosController::class, 'musculos']); |
| --- |

### Ejercicio



No confundir con la pestaña de ejercicios, aquí se guardan todos los ejercicios guardados, por ahora las imágenes estarán de prueba.

Los datos los carga la base de datos y solo se puede acceder a esta zona clicando el nombre de un ejercicio de la pestaña ejercicios.

#### EjercicioController.php

| public function ejercicio()  {  $ejercicios = Ejercicio::all();  return view('ejercicio', ['ejercicios' => $ejercicios]);  } |
| --- |

Obtiene todos los ejercicios de la base de datos y los manda al html.

#### ejercicio.blade.php

| <!DOCTYPE html> <html lang="en">  <head>  <meta charset="UTF-8">  <meta name="viewport" content="width=device-width, initial-scale=1.0">  <meta http-equiv="X-UA-Compatible" content="ie=edge">  <title>Ejercicios</title>  <style>  .ejercicio {  display: flex ;  flex-direction: row;  flex-wrap: wrap;  align-content: center;  justify-content: space-between;  align-items: center;  }  h1.centrado{  text-align: center;  }  </style> </head>  <body>  @extends('layouts.app-master')   @section('content')  <h1 class="centrado" style="margin-bottom: 50px">Ejercicios</h1>  <div class="ejercicio">  @foreach ($ejercicios as $ejercicio)  <div id="{{ $ejercicio->id }}" style="width: 30%">  <img src="https://image.shutterstock.com/image-illustration/this-3d-illustration-shows-view-260nw-1849652074.jpg"  alt="">   <h5 class="centrado">{{ $ejercicio->nombre }}</h5>  <p style="">{{ $ejercicio->descripcion }}</p>   </div>  @endforeach  </div>  @endsection </body>  </html> |
| --- |

Con el foreach trato la collection y voy ordenándolos.

#### web.php

| Route::get('/ejercicio', [EjercicioController::class, 'ejercicio']); |
| --- |

Por último el home, el login y el register lo hice con este tutorial: <https://codeanddeploy.com/blog/laravel/laravel-8-authentication-login-and-registration-with-username-or-email> .

Dejando los proyectos que han ido surgiendo durante el desarrollo de la aplicación aquí están los que tenía planeado desde el principio:

### 

### Alimentos



La idea era mostrar una lista de múltiples alimentos, con descripciones e imágenes, pero no conseguí encontrar una base de datos gratuita de alimentos así que he puesto tres ejemplos para ver cómo funciona.

#### AlimentosController.php

| <?php  namespace App\Http\Controllers;  use App\Models\Alimento; use Illuminate\Http\Request;  class AlimentosController extends Controller {  public function getAlimentos(){  $alimentos = Alimento::all();  $fotos = [  'https://sgfm.elcorteingles.es/SGFM/dctm/MEDIA03/202009/18/00118611301427\_\_\_\_3\_\_600x600.jpg',  'https://www3.gobiernodecanarias.org/medusa/wiki/images/e/ed/Platano-canario.png',  'https://static.condisline.com/resize\_1280x1024/images/catalog/large/210544.jpg'  ];   return view('alimentos', ['alimentos' => $alimentos, 'fotos' => $fotos]);  } } |
| --- |

Su funcionamiento es bastante simple, con la variable $alimentos obtengo todos los alimentos de la base de datos y como aún no tengo añadida la parte de las fotos las meto en un array y trabajo con ellas como si fuese una parte más de alimentos.

El resto es pasarle los datos al html y mandar los datos.

#### alimentos.blade.php

| <!DOCTYPE html> <html lang="es">  <head>  <meta charset="UTF-8">  <meta name="viewport" content="width=device-width, initial-scale=1.0">  <meta http-equiv="X-UA-Compatible" content="ie=edge">  <title>Alimentos</title>  <style>  .centrado {  text-align: center;  }   .oculto {  display: none;  }   .alimentos {  display: flex;  justify-content: space-between;  align-items: center;  }  h1{  text-align: center;  margin-bottom: 75px !important;  }  </style> </head>  <body>  @extends('layouts.app-master')   @section('content') <h1>Alimentos y sus valores nutricionales</h1>  @php  $contador = 0;  @endphp  <div class="alimentos">  @foreach ($alimentos as $alimento)  <div style="width: 30%">  <form action="valor-alimentos" method="post">  @csrf  <p class="oculto">id: <input type="text" name="alimento\_id" value="{{$alimento->id}}">  <p>  <input type="image" src="{{$fotos[$contador]}}" alt="{{$alimento->nombre}}" width="100%"  height="350px" value="{{$alimento->id}}">  </form>  <div class="centrado"> Producto: <u>{{$alimento->nombre}}</u></div>  <div class="centrado">{{$alimento->descripcion}}</div>  <div class="centrado">Creación de la marca: <u>{{$alimento->fecha\_creacion}}</u></div>  </div>  @php  $contador++;  @endphp  @endforeach  </div>  @endsection </body>  </html> |
| --- |

Esta es una de las páginas que utilizo el id o el nombre para crear enlaces a otra parte de la página, más adelante la enseño.

#### web.php

| Route::get('/alimentos', [AlimentosController::class, 'getAlimentos']); |
| --- |

### Valor Alimentos



A esta sección solo se puede acceder clickeando una imagen de la pestaña “Alimentos”, que te llevará a la pestaña valor alimentos donde encontrarás una imagen del alimento seleccionado con los valores nutricionales personalizados.

#### ValorAlimentosController

| <?php  namespace App\Http\Controllers;  use App\Models\ValorAlimento; use Illuminate\Http\Request; use Illuminate\Support\Facades\DB;  class ValorAlimentosController extends Controller {  public function getValorAlimentos(Request $request)  {  $valor\_alimento = ValorAlimento::where('alimento\_id', $request->alimento\_id)->first();  $fotos = [  'https://sgfm.elcorteingles.es/SGFM/dctm/MEDIA03/202009/18/00118611301427\_\_\_\_3\_\_600x600.jpg',  'https://www3.gobiernodecanarias.org/medusa/wiki/images/e/ed/Platano-canario.png',  'https://static.condisline.com/resize\_1280x1024/images/catalog/large/210544.jpg'  ];  return view('valor\_alimentos.index', ['valor\_alimento' => $valor\_alimento, 'foto' => $fotos[$request->alimento\_id-1]]);    } } |
| --- |

Desde la pestaña de alimentos se envía una request con el id del alimento para que haga la búsqueda desde la base de datos, encuentra los valores nutricionales del alimento y los carga en la página.

#### web.php

| Route::post('/valor-alimentos', [ValorAlimentosController::class, 'getValorAlimentos']); |
| --- |

#### valor\_alimentos\index.blade.php

| <!DOCTYPE html> <html> <head>  <style>  #nutritionfacts {  background-color: white;  border: 1px solid black;  padding: 3px;  width: 244px;  }  #nutritionfacts td {  color: black;  font-family: 'Arial Black', 'Helvetica Bold', sans-serif;  font-size: 8pt;  padding: 0;  }  #nutritionfacts td.header {  font-family: 'Arial Black', 'Helvetica Bold', sans-serif;  font-size: 28px;  white-space: nowrap;  }  #nutritionfacts div.label {  float: left;  font-family: 'Arial Black', 'Helvetica Bold', sans-serif;  }  #nutritionfacts div.serving {  font-family: Arial, Helvetica, sans-serif;  font-size: 8pt;  text-align: center;  }  #nutritionfacts div.weight {  display: inline;  font-family: Arial, Helvetica, sans-serif;  padding-left: 1px;  }   #nutritionfacts div.dv {  display: inline;  float: right;  font-family: 'Arial Black', 'Helvetica Bold', sans-serif;  margin-right: 5px;  }   #nutritionfacts table.vitamins td {  font-family: Arial, Helvetica, sans-serif;  white-space: nowrap;  width: 33%;  }   #nutritionfacts div.line {  border-top: 1px solid black;  }   #nutritionfacts div.labellight {  float: left;  font-family: Arial, Helvetica, sans-serif;  }   #nutritionfacts .highlighted {  border: 1px dotted grey;  padding: 2px;  }   .contenedor {  display: flex;  flex-direction: column;  align-items: center;  }  </style> </head> <body>  @extends('layouts.app-master')  @section('content')  <div class=" p-5 rounded" style="display: flex">  <div class="contenedor">  <div style="margin-bottom: 50px">  <img src="{{$foto}}" alt="" height="200px">  </div>  <div id="nutritionfacts">  <table width="242" cellspacing="0" cellpadding="0">  <tbody>  <tr>  <td align="center" class="header" style="font-size: 20px; ">Valores Nutricionales </td>  </tr>  <tr>  <td>  <div class="serving">Por cada <span class="highlighted">100.0g</span></div>  </td>  </tr>  <tr style="height: 7px">  <td bgcolor="#000000"></td>  </tr>  <tr>  <td style="font-size: 7pt">  <div class="line">Cantidad por cada 100g</div>  </td>  </tr>  <tr>  <td>  <div class="line">  <div class="label">Calorias <div class="weight"></div>  </div>  <div class="dv">{{$valor\_alimento->energia}}cal</div>  {{-- <div style="padding-top: 1px; float: right;" class="labellight">Calories from  Fat <div class="weight">56</div> --}}  </div>  </div>  </td>  </tr>  <tr>  <td>  <div class="line">  <div class="label">Grasas <div class="weight"></div>  </div>  <div class="dv">{{$valor\_alimento->grasas}}g</div>  </div>  </td>  </tr>  <tr>  <td class="indent">  <div class="line">  <div class="labellight">De las cuales saturadas: <div class="weight"></div>  </div>  <div class="dv">{{$valor\_alimento->g\_saturadas}}g</div>  </div>  </td>  </tr>  <tr>  <td class="indent">  <div class="line">  <div class="labellight">De las cuales monosaturadas: <div class="weight"></div>  </div>  <div class="dv">{{$valor\_alimento->g\_monosaturadas}}g</div>  </div>  </td>  </tr>  <tr>  <td>  <div class="line">  <div class="label">De las cuales trans: <div class="weight"></div>  </div>  <div class="dv">{{$valor\_alimento->g\_trans}}g</div>  </div>  </td>  </tr>  <tr>  <td>  <div class="line">  <div class="label">Carbohidratos: <div class="weight"></div>  </div>  <div class="dv">{{$valor\_alimento->carbohidratos}}g</div>  </div>  </td>  </tr>  <tr>  <td class="indent">  <div class="line">  <div class="labellight">De los cuales azucares:  <div class="weight"></div>  </div>  <div class="dv">{{$valor\_alimento->azucares}}g</div>  </div>  </td>  </tr>  <tr>  <td class="indent">  <div class="line">  <div class="labellight">Fibra:  <div class="weight"></div>  </div>  <div class="dv">{{$valor\_alimento->fibra\_alimentaria}}g</div>  </div>  </td>  </tr>   <tr>  <td>  <div class="line">  <div class="label">Proteinas: <div class="weight"></div>  </div>  <div class="dv">{{$valor\_alimento->proteinas}}g</div>  </div>  </td>  </tr>   <tr style="height: 7px">  <td bgcolor="#000000"></td>  </tr>  <tr>  <td>  <table cellspacing="0" cellpadding="0" border="0" class="vitamins">  <tbody>  <tr>  <td>Sal: {{$valor\_alimento->sal}}g</td>  <td style="margin-left:79px;float:left;">Calcio {{$valor\_alimento->calcio}}g  </td>  </tr>  <tr>  <td>Almidon {{$valor\_alimento->almidon}}g</td>  <td style="margin-left:80px;float:left;">Sodio {{$valor\_alimento->sodio}}g  </td>  </tr>  <tr>  <td>Vitaminas: {{$valor\_alimento->vitaminas}}</td>  </tr>  </tbody>  </table>  </td>  </tbody>  </table>  </div>  </div>   </div> </body> @endsection  </html> |
| --- |

Desde la plantilla <https://codepen.io/chriscoyier/pen/ApavyZ> voy cargando los datos del valor del alimento acompañado de su imagen.

### Ejercicios



Esta sección junto a la de calendario son las dos únicas que requieren de iniciar sesión para que se pueda utilizar, sirve para poner las series que has hecho en un día, al seleccionar un ejercicio ya se te añade el músculo principal que actúa en la tabla.

También puede utilizarse como recordatorio, es decir, puedes ir dejando los ejercicios preparados para ese día y cuando llegue los puedes ir rellenando.

#### SeriesController.php

| <?php  namespace App\Http\Controllers;  use App\Models\AgrupacionesEjercicios; use App\Models\Ejercicio; use App\Models\EjercicioMaquina; use App\Models\EjercicioMusculo; use App\Models\Serie; use App\Models\User; use Illuminate\Http\Request; use Illuminate\Support\Facades\Auth;  use function PHPSTORM\_META\map;  class SeriesController extends Controller {   public function postSeries(Request $request)  {  $usuario = Auth::id();  if ($request->tipo == "new") {   $agrupacion\_ejercicio = AgrupacionesEjercicios::where('usuario\_id', $usuario)->where('fecha', $request->fecha)->first();  $ejercicio = Ejercicio::where('nombre', $request->ejercicio)->first();   $ejercicio\_maquina = new EjercicioMaquina();  $ejercicio\_maquina->agrupacion\_ejercicio\_id = $agrupacion\_ejercicio->id;  $ejercicio\_maquina->ejercicio\_id = $ejercicio->id;  $ejercicio\_maquina->save();   $serie = new Serie();  $serie->ejercicio\_maquina\_id = $ejercicio\_maquina->id;  $serie->peso = $request->peso;  $serie->repeticiones = $request->repeticiones;  $serie->tiempo\_descanso = $request->tiempo\_descanso;  $serie->descripcion = $request->descripcion;  $serie->save();  }   if ($request->tipo == "edit") {  $serie = Serie::where('id', $request->serie\_id)->first();  $serie->descripcion = $request->descripcion;  $serie->peso = $request->peso;  $serie->repeticiones = $request->repeticiones;  $serie->tiempo\_descanso = $request->tiempo\_descanso;  $serie->save();  }   if ($request->tipo == "delete"){  $serie = Serie::where('id', $request->serie\_id)->first();  $serie->delete();  }    $usuarios = User::where('id', $usuario)->with('agrupacionesEjercicios.ejercicioMaquina.serie')->get();  $ejercicios = Ejercicio::all();   return view(  'ejercicio.index',  ['usuarios' => $usuarios],  ['ejercicios' => $ejercicios]  );  }    public function getSeries()  {  $usuario = Auth::id();  $usuarios = User::where('id', $usuario)  ->with('agrupacionesEjercicios.ejercicioMaquina.serie')  ->get();  $ejercicios = Ejercicio::all();   return view(  'ejercicio.index',  ['usuarios' => $usuarios],  ['ejercicios' => $ejercicios]  );  } } |
| --- |

Este controlador se divide en dos funciones, la primera con getSeries, la cual se encarga de obtener el id del usuario logueado en este momento que está haciendo la petición y luego con ese id carga la relación de la base de datos del usuario con las series que tiene y las muestra.

La segunda, postSeries, se encarga de controlar cuando se añade, edita o elimina una serie de la tabla, lo hace a través del id que se le añade en el html y se le asocia.

#### 

#### web.php

| Route::get('/ejercicios', [SeriesController::class, 'getSeries']); Route::post('/ejercicios', [SeriesController::class, 'postSeries']); |
| --- |

#### ejercicios\index.blade.php

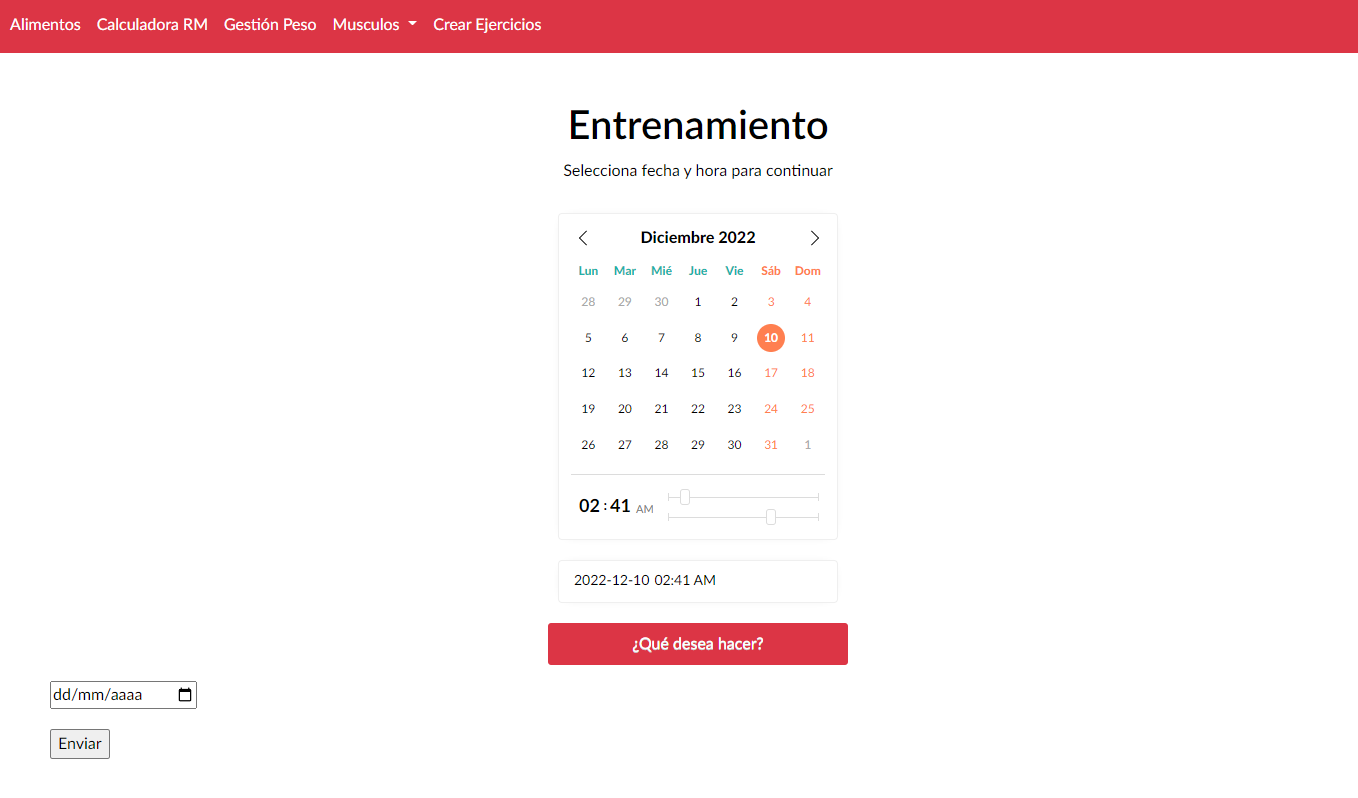
| <!DOCTYPE html> <html>  <head>  <link href='https://fonts.googleapis.com/css?family=Roboto:400,100,300,700' rel='stylesheet' type='text/css'>   <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css">   <link href="//maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css" rel="stylesheet" id="bootstrap-css">  <script src="//maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"></script>  <script src="//cdnjs.cloudflare.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>  <style>  h1 {  font-size-adjust: initial;  text-align: center;  white-space: nowrap;  overflow: hidden;  text-overflow: ellipsis;  padding-top: 2%;   }   table.ejercicio {  text-align: center;  width: 100%;  font-family: 'Roboto', sans-serif;  color: white;  padding: 2%;  }   tr,  td,  th {  padding: 1.5%;  }   td {  /\* https://www.svgbackgrounds.com/ \*/  background-color: #110933;  background-image: url("data:image/svg+xml,%3Csvg xmlns='http://www.w3.org/2000/svg' width='200' height='200' viewBox='0 0 800 800'%3E%3Cg fill='none' stroke='%230C0944' stroke-width='4.7'%3E%3Cpath d='M769 229L1037 260.9M927 880L731 737 520 660 309 538 40 599 295 764 126.5 879.5 40 599-197 493 102 382-31 229 126.5 79.5-69-63'/%3E%3Cpath d='M-31 229L237 261 390 382 603 493 308.5 537.5 101.5 381.5M370 905L295 764'/%3E%3Cpath d='M520 660L578 842 731 737 840 599 603 493 520 660 295 764 309 538 390 382 539 269 769 229 577.5 41.5 370 105 295 -36 126.5 79.5 237 261 102 382 40 599 -69 737 127 880'/%3E%3Cpath d='M520-140L578.5 42.5 731-63M603 493L539 269 237 261 370 105M902 382L539 269M390 382L102 382'/%3E%3Cpath d='M-222 42L126.5 79.5 370 105 539 269 577.5 41.5 927 80 769 229 902 382 603 493 731 737M295-36L577.5 41.5M578 842L295 764M40-201L127 80M102 382L-261 269'/%3E%3C/g%3E%3Cg fill='%230A5553'%3E%3Ccircle cx='769' cy='229' r='5'/%3E%3Ccircle cx='539' cy='269' r='5'/%3E%3Ccircle cx='603' cy='493' r='5'/%3E%3Ccircle cx='731' cy='737' r='5'/%3E%3Ccircle cx='520' cy='660' r='5'/%3E%3Ccircle cx='309' cy='538' r='5'/%3E%3Ccircle cx='295' cy='764' r='5'/%3E%3Ccircle cx='40' cy='599' r='5'/%3E%3Ccircle cx='102' cy='382' r='5'/%3E%3Ccircle cx='127' cy='80' r='5'/%3E%3Ccircle cx='370' cy='105' r='5'/%3E%3Ccircle cx='578' cy='42' r='5'/%3E%3Ccircle cx='237' cy='261' r='5'/%3E%3Ccircle cx='390' cy='382' r='5'/%3E%3C/g%3E%3C/svg%3E");  }   .prueba {  text-decoration: none;  color: white;  }   .prueba:hover {  text-decoration: none;  color: white;  }   .b {  color: white;  }   input[type=number]::-webkit-inner-spin-button,  input[type=number]::-webkit-outer-spin-button {  -webkit-appearance: none;  margin: 0;  }   .list-choice {  width: 300px;  margin: 1em auto;  position: relative;  cursor: pointer;  }   .list-choice input[type="radio"] {  position: absolute;  left: -9999px;  }   .list-choice-title {  width: 100%;  display: block;  background: #dc3545;  text-align: center;  padding: 0.55em 1em;  box-sizing: border-box;  color: #FFF;  text-shadow: 0 1px 0 #CACACA;  border-radius: 0.2em;  }   .list-choice:hover .list-choice-title {  border-radius: 0.2em 0.2em 0 0;  }   .list-choice-objects label:nth-last-of-type(1) span {  border-radius: 0 0 0.2em 0.2em;  }   .list-choice input[type="radio"]+span {  color: #FFF;  background: #dc3545;  padding: 0.55em 1em;  display: block;  text-align: center;  box-sizing: border-box;  cursor: pointer;  width: 100%;  }   .list-choice-objects {  position: absolute;  top: 0;  padding-top: 2.1em;  box-sizing: border-box;  width: 100%;  overflow: hidden;  max-height: 0;  transition: all 250ms ease;  }   .list-choice:hover .list-choice-objects input[type="radio"]+span {  position: relative;  top: 0;  transition: all 250ms ease-in-out;  }   .list-choice:hover input[type="radio"]+span:hover {  background: #0b5ed7;  }   .list-choice:hover input[type="radio"]:checked+span:hover {  background: #74D68E;  }   .list-choice input[type="radio"]:checked+span {  background: #dc3545;  position: absolute;  top: 0em;  border-radius: 0.2em;  }   .list-choice:hover input[type="radio"]:checked+span {  border-radius: 0;  }   .list-choice:hover .list-choice-objects label:nth-last-of-type(1) input[type="radio"]:checked+span {  border-radius: 0 0 0.2em 0.2em;  }   .list-choice:hover .list-choice-objects {  max-height: 540px;  }   .list-choice-objects>label {  display: block;  }   \* {  font-family: 'Roboto', sans-serif;  }   .select2-container .select2-selection--single {  height: 34px !important;  }   .select2-container--default .select2-selection--single {  border: 1px solid #ccc !important;  border-radius: 0px !important;  }  </style> </head>  <body style="width: 100%">   @extends('layouts.app-master')   @section('content')  {{ setlocale(LC\_ALL,"es\_ES") }}   {{-- @php  $idUsuario = Auth::user()->id;  $agruEj= App\Models\Serie::where('id',$idUsuario)->get();  @endphp --}}    @guest  <h1>Para poder acceder a esta sección debes iniciar sesión</h1>  @endguest   @auth  <div>  @php  $contador = 0;  @endphp  @foreach ($usuarios as $usuario)  @foreach ($usuario->agrupacionesEjercicios as $agrupacionesEj)  <div>   <div style="color: black">  <!-- Button trigger modal -->  <button type="button" class="btn btn-primary" data-bs-toggle="modal"  data-bs-target="#modalNew{{ $contador }}" style="float: right">  Nuevo entrenamiento  </button>   <!-- Modal -->  <div class="modal fade" id="modalNew{{ $contador }}" tabindex="-1" aria-hidden="true">  <div class="modal-dialog">  <div class="modal-content">  <div class="modal-header bg-primary b">  <h1 class="modal-title fs-5">Escribe el ejercicio que va a hacer:</h1>  <button type="button" class="btn-close" data-bs-dismiss="modal"  aria-label="Close"></button>  </div>  <div class="modal-body" style="text-align: start;">  <form action="/ejercicios" method="post">  @csrf  <input type="hidden" name="fecha" value="{{$agrupacionesEj->fecha}}"  id="{{$agrupacionesEj->fecha}}">  <input type="hidden" name="tipo" value="new">  <input type="text" placeholder="Ejercicio" name="ejercicio">  <input type="text" name="descripcion" placeholder="Descripción">  <input type="number" name="peso" placeholder="Peso">  <input type="number" name="repeticiones" placeholder="Repeticiones">  <input type="number" name="tiempo\_descanso" placeholder="Descanso">  {{-- <div style="z-index: 999">  <link  href="https://cdnjs.cloudflare.com/ajax/libs/select2/4.0.6-rc.0/css/select2.min.css"  rel="stylesheet" />  <script  src="https://cdnjs.cloudflare.com/ajax/libs/select2/4.0.6-rc.0/js/select2.min.js">  </script>  <div class="container" style="z-index: 999">  <div class="row" style="z-index: 999">  <form class="col-md-4" style="z-index: 999">  <label style="z-index: 999">Ejercicios</label>  <select class="form-control select2" style="z-index: 999">  @foreach ($ejercicios as $ejercicio)  <option value="{{$ejercicio->id}}" style="z-index: 999">  {{$ejercicio->nombre}}</option>  @endforeach  </select>  </form>  </div>  </div>  </div>  <script>  $('.select2').select2();  </script> --}}  </div>  <div class="modal-footer">  <button type="button" class="btn btn-danger" data-bs-dismiss="modal">Cerrar</button>  <input type="submit" class="btn btn-primary"></button>  </form>  </div>  </div>  </div>  </div>  </div>     <h1>{{ ucwords(\Carbon\Carbon::parse(strftime($agrupacionesEj->fecha))->formatLocalized('%A %d de %B del  %Y')) }}</h1>  {{ ucwords(\Carbon\Carbon::parse(strftime($agrupacionesEj->fecha))->diffForHumans(null,null, 1, 1)) }}  </div>  {{-- \*Añadir Modales https://makitweb.com/dynamically-load-content-in-bootstrap-modal-laravel-9/ --}}  {{-- \* Modales Boostrap https://getbootstrap.com/docs/5.2/components/modal/ --}}  {{-- Todo https://www.itsolutionstuff.com/post/laravel-9-fullcalendar-ajax-tutorial-exampleexample.html --}}   {{-- @if ($usuario->nombre\_musculo == "Espalda") --}}  <table class="ejercicio" style="margin-bottom: 50px;">  <tr class="bg-dark">  <th>Ejercicio</th>  <th>Músculo</th>  <th>Descripción</th>  <th>Peso</th>  <th>Repeticiones</th>  <th>Descanso</th>  <th>Editar</th>  <th>Borrar</th>  {{-- <th>Hora</th> --}}  </tr>  @foreach ($agrupacionesEj->ejercicioMaquina as $ejMaquina)  @foreach ($ejMaquina->serie as $serie)   <tr class="bg-success">  <td> <a class="prueba" href="ejercicio#{{$ejMaquina->ejercicio->id}}">  {{ $ejMaquina->ejercicio->nombre }} </a>  </td>   <td> {{ $ejMaquina->ejercicio->musculo->nombre }} </td>  <td> {{ $serie->descripcion}} </td>  <td> {{ $serie->peso}} </td>  <td> {{ $serie->repeticiones}} </td>  <td> {{ $serie->tiempo\_descanso}} </td>  <td>  <div style="color: black">  <!-- Button trigger modal -->  <button type="button" class="btn btn-success" data-bs-toggle="modal"  data-bs-target="#modalEdit{{ $contador }}">  +  </button>   <!-- Modal -->  <div class="modal fade" id="modalEdit{{ $contador }}" tabindex="-1" aria-hidden="true">  <div class="modal-dialog">  <div class="modal-content">  <div class="modal-header bg-success b">  <h1 class="modal-title fs-5">Edita los campos</h1>  <button type="button" class="btn-close" data-bs-dismiss="modal"  aria-label="Close"></button>  </div>  <div class="modal-body" style="text-align: start;">  <p>Ejercicio: {{ $ejMaquina->ejercicio->nombre }}</p>  <p>Peso: {{ $serie->peso}} kg</p>  <p>Repeticiones: {{ $serie->repeticiones}}</p>  <p>Descanso: {{ $serie->tiempo\_descanso}} seg</p>  <p>Descripción: {{ $serie->descripcion}}</p>  <p>Fecha: {{ $agrupacionesEj->fecha }}</p>  <form action="/ejercicios" method="post">  @csrf  <input type="hidden" name="serie\_id" value="{{$serie->id}}" id="{{$serie->id}}">  <input type="hidden" name="tipo" value="edit">  <input type="text" name="descripcion" placeholder="Descripción">  <input type="number" name="peso" placeholder="Peso">  <input type="number" name="repeticiones" placeholder="Repeticiones">  <input type="number" name="tiempo\_descanso" placeholder="Descanso">  </div>  <div class="modal-footer">  <button type="button" class="btn btn-danger"  data-bs-dismiss="modal">Cerrar</button>  <input type="submit" class="btn btn-success"></button>  </form>  </div>  </div>  </div>  </div>  </div>  </td>     <td>  <div style="color: black">  <!-- Button trigger modal -->  <button type="button" class="btn btn-danger" data-bs-toggle="modal"  data-bs-target="#modalClose{{ $contador }}">  -  </button>   <!-- Modal -->  <div class="modal fade" id="modalClose{{ $contador }}" tabindex="-1" aria-hidden="true">  <div class="modal-dialog">  <div class="modal-content">  <div class="modal-header bg-danger b">  <h1 class="modal-title fs-5">¿Está seguro que quiere borrar estos campos?</h1>  <button type="button" class="btn-close" data-bs-dismiss="modal"  aria-label="Close"></button>  </div>  <div class="modal-body" style="text-align: start;">  <p>Ejercicio: {{ $ejMaquina->ejercicio->nombre }}</p>  <p>Peso: {{ $serie->peso}} kg</p>  <p>Repeticiones: {{ $serie->repeticiones}}</p>  <p>Descanso: {{ $serie->tiempo\_descanso}} seg</p>  <p>Descripción: {{ $serie->descripcion}}</p>  <form action="/ejercicios" method="post">  @csrf  <input type="hidden" name="serie\_id" value="{{$serie->id}}" id="{{$serie->id}}">  <input type="hidden" name="tipo" value="delete">   </div>  <div class="modal-footer">  <button type="button" class="btn btn-primary" data-bs-dismiss="modal">Cerrar</button>  <input type="submit" class="btn btn-danger"></button>  </form>  </div>  </div>  </div>  </div>  </div>  </td>  </tr>  @php  $contador++;  @endphp  @endforeach  @endforeach  </table>  @endforeach  @endforeach  </div>    @endauth   @endsection </body>  </html> |
| --- |

A base de los foreach voy recorriendo toda la collection que llega desde el controlador, y voy cargando los datos, también con la ayuda de un contador puedo poner muchas etiquetas con diferentes id.

Lo que está comentado es un select box (como el de crear ejercicios) que al ponerlo dentro de un modal se bugea, al seleccionar las opciones aparecen por detrás del modal haciendo que fuese imposible su selección.

También está la opción de clicar en un ejercicio y que te mande a la página de los ejercicios.

### Calendario



Este es el calendario que te guiará en el futuro, la idea es hacer un calendario como este para que te ponga de colores los diferentes días que has entrenado y los que tienes planeado entrenar.

Desgraciadamente están completamente en JavaScript (lenguaje que no domino mucho) y por ahora está de “adorno” hasta que pueda aprender JavaScript.

La fecha se selecciona desde el calendario de abajo a la izquierda y en el cuadro rojo se selecciona si se quiere crear un día o eliminarlo, después se le da a enviar y luego al dirigirse a ejercicios puedes observar que se ha actualizado.

#### 

#### 

#### 

#### EjercicioController.php

| public function calendario()  {  return view('calendario');  }   public function postAgEj(Request $request)  {  $usuario = Auth::id();   if ($request->fecha) {   $agrupacion\_ejercicio\_buscar = AgrupacionesEjercicios::where('fecha', $request->fecha)  ->where('usuario\_id', $usuario)->first();  if ($agrupacion\_ejercicio\_buscar == null && $request->crud=="new") {  $agrupacion\_ejercicio = new AgrupacionesEjercicios();  $agrupacion\_ejercicio->usuario\_id = $usuario;  $agrupacion\_ejercicio->rutina\_id = 1;  $agrupacion\_ejercicio->fecha = $request->fecha;  $agrupacion\_ejercicio->save();   $usuarios = User::where('id', $usuario)->with('agrupacionesEjercicios.ejercicioMaquina.serie')->get();  $ejercicios = Ejercicio::all();   return view(  'ejercicio.index',  ['usuarios' => $usuarios],  ['ejercicios' => $ejercicios]  );     }   if($agrupacion\_ejercicio\_buscar != null && $request->crud=="delete"){  $agrupacion\_ejercicio\_buscar->delete();  return view('calendario');  }  }  } |
| --- |

Para el calendario actúan dos funciones, calendario, que simplemente hace que sea vea el html de calendario y postAgEj, que según lo que llegue de la request creará o eliminará un agrupación ejercicio de la base de datos (un dia de ejercicio).

#### calendario.blade.php

| <!doctype html> <html lang="es">  <head>  <meta charset="UTF-8">  <meta name="viewport" content="initial-scale=1,width=device-width">  <meta name="format-detection" content="telephone=no">  <title>Entrenamiento</title>  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/lato-font/3.0.0/css/lato-font.min.css">  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>  <script defer src="js/calendario.js"></script>  <link href="css/calendario.css" rel="stylesheet">   </head>  <body>    @extends('layouts.app-master')   @section('content')   @guest  <h1>Para acceder a esta sección debes de iniciar sesión.</h1>  @endguest   @auth   <form action="/calendario" method="post">  @csrf  <h1 class="inicio">Entrenamiento</h1>  <p>Selecciona fecha y hora para continuar</p>  <div class="vanilla-calendar" data-provide="datepicker" name="data"></div>  <div class="vanilla-calendar-info">  <span id="vanilla-calendar-info-date" data-provide="datepicker" name="date"> </span>  <span id="vanilla-calendar-info-time" data-provide="datepicker" name="time"></span>  </div>  <script>  document.addEventListener('DOMContentLoaded', () => {  const generateDate = (date) => {  const year = date.getUTCFullYear();  let month = date.getUTCMonth() + 1;  let day = date.getUTCDate();   month = month < 10 ? `0${month}` : month;  day = day < 10 ? `0${day}` : day;   return `${year}-${month}-${day}`;    };   const calendarInfoDate = document.querySelector('#vanilla-calendar-info-date');  const calendarInfoTime = document.querySelector('#vanilla-calendar-info-time');  const calendar = new VanillaCalendar('.vanilla-calendar', {  settings: {  lang: 'es',  selection: {  time: 12,  },  selected: {  dates: [generateDate(new Date())],  },  },  actions: {  clickDay(e, dates) {  calendarInfoDate.innerText = dates;  },  changeTime(e, time) {  calendarInfoTime.innerText = time;  },  },  });  calendar.init();  calendarInfoDate.innerText = generateDate(new Date());  calendarInfoTime.innerText = calendar.selectedTime;  // $.post('users',calendar.selectedTime, function(){  // return calendar.selectedTime.json()  // })  });  </script>   <div class="list-choice">  <div class="list-choice-title">¿Qué desea hacer?</div>  <div class="list-choice-objects">  <label>  <input type="radio" name="crud" value="new" /> <span class="dd">Crear</span>  </label>  <label>  <input type="radio" name="crud" value="delete" /> <span class="dd">Borrar</span>  </label>  </div>  </div>  <input type="date" name="fecha">  <br>  <input type="submit" style="margin-top: 20px">  </form>  @endauth  @endsection  </body>  </html> |
| --- |

El script es el vanilla calendar <https://vanilla-calendar.frontend.uvarov.tech/> y lo demás ya se ha usado anteriormente.

#### web.php

| Route::get('/calendario', [EjercicioController::class, 'calendario']); Route::post('/calendario', [EjercicioController::class, 'postAgEj']); |
| --- |

# 5. Instalación y preparación: Todo lo necesario para hacer funcionar el proyecto y cómo se instala. Edición de los archivos de configuración, creación y puesta en funcionamiento del control de versiones en github, etc.

Se necesita PHP 8 y laravel 9, laragon y HeidiSQL

Al instalar PHP 8 hay que habilitar un plugin.

Una vez instalado todo abrimos laragon y luego metemos la carpeta de GitHub.

Explicar Archivo env.

Abrimos la carpeta y ponemos los siguiente comandos

php artisan migrate, nos preguntara si queremos crear una nueva tabla en la base de datos con el nombre del .env, le damos que si,luego ejecutamos un php artisan db:seed para meter todos los datos de prueba y por último escribimos

php artisan serve se creará por defecto en el localhost (.env) y puerto 8000 con el –port le cambiamos el puerto y podemos escribir el que queramos.

Ya solo sería buscar esa ruta en nuestro navegador y ya lo tendríamos.

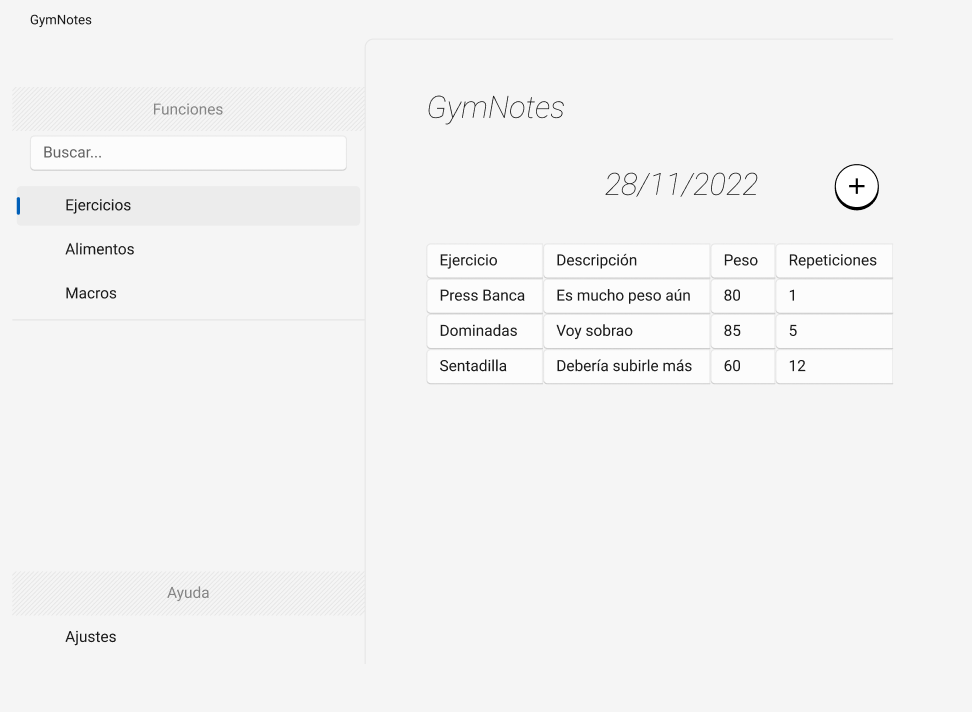
# 

# 6. Prototipado. Se incluye un prototipo wireframe de la aplicación.

El prototipo lo he realizado con figma y se puede ver desde aquí: <https://www.figma.com/file/X3l8te30TjqQ83QRYVNNzl/Prototype?node-id=0%3A1&t=gKOeNArrEoQKuT5C-3>

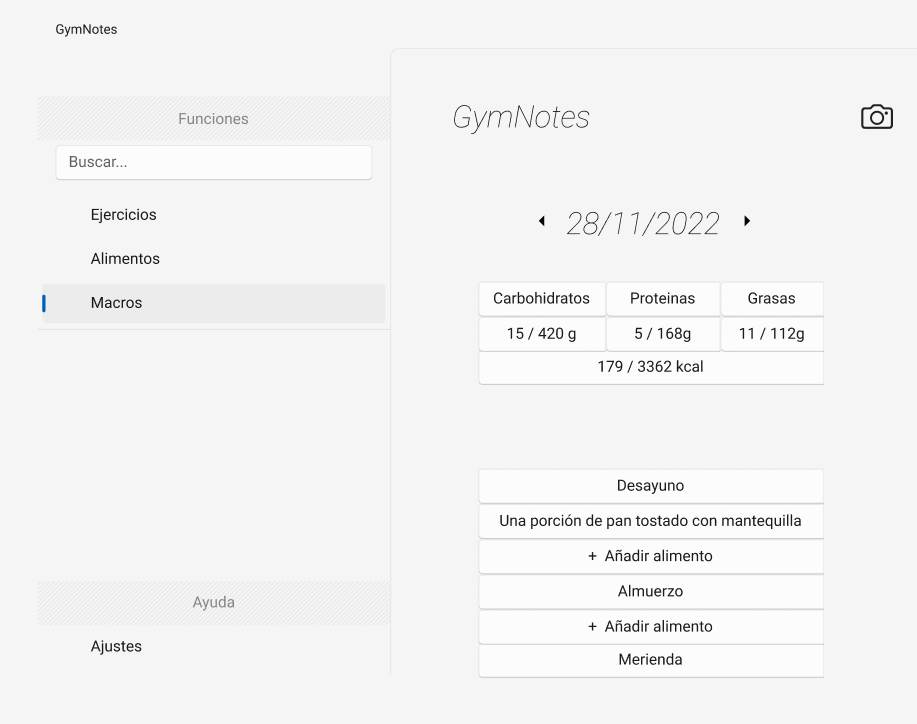
Al principio iba a ser una aplicación ligera con tres funciones.

### Ejercicios



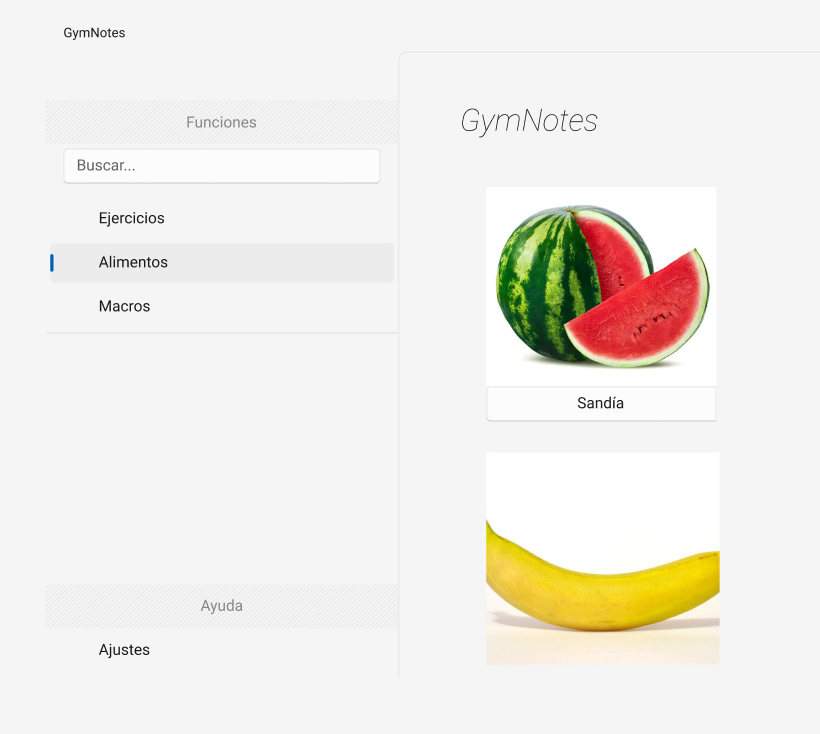
En la primera versión la idea era añadir todo manualmente en ejercicios e ir creando más días cuando hiciesen falta.

### Macros



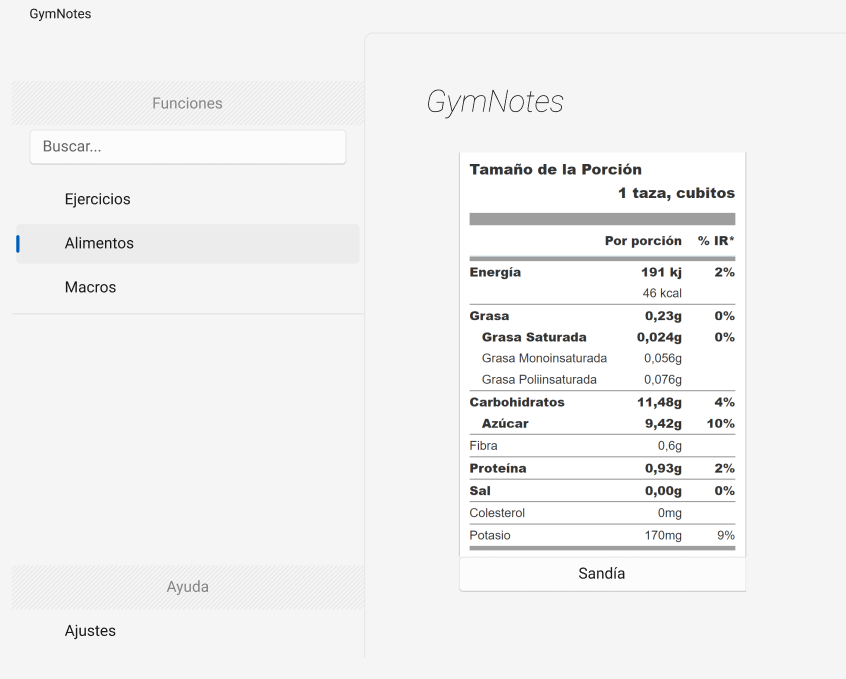
También en el prototipo tenía pensado hacer una pestaña para llevar tus kcal al día y tener una ligera idea de cómo ibas.

### Alimentos



La parte de alimentos sigue siendo la misma idea que actualmente, al clicar en una imagen que te lleve a la parte de los valores nutricionales.

### Valores nutricionales

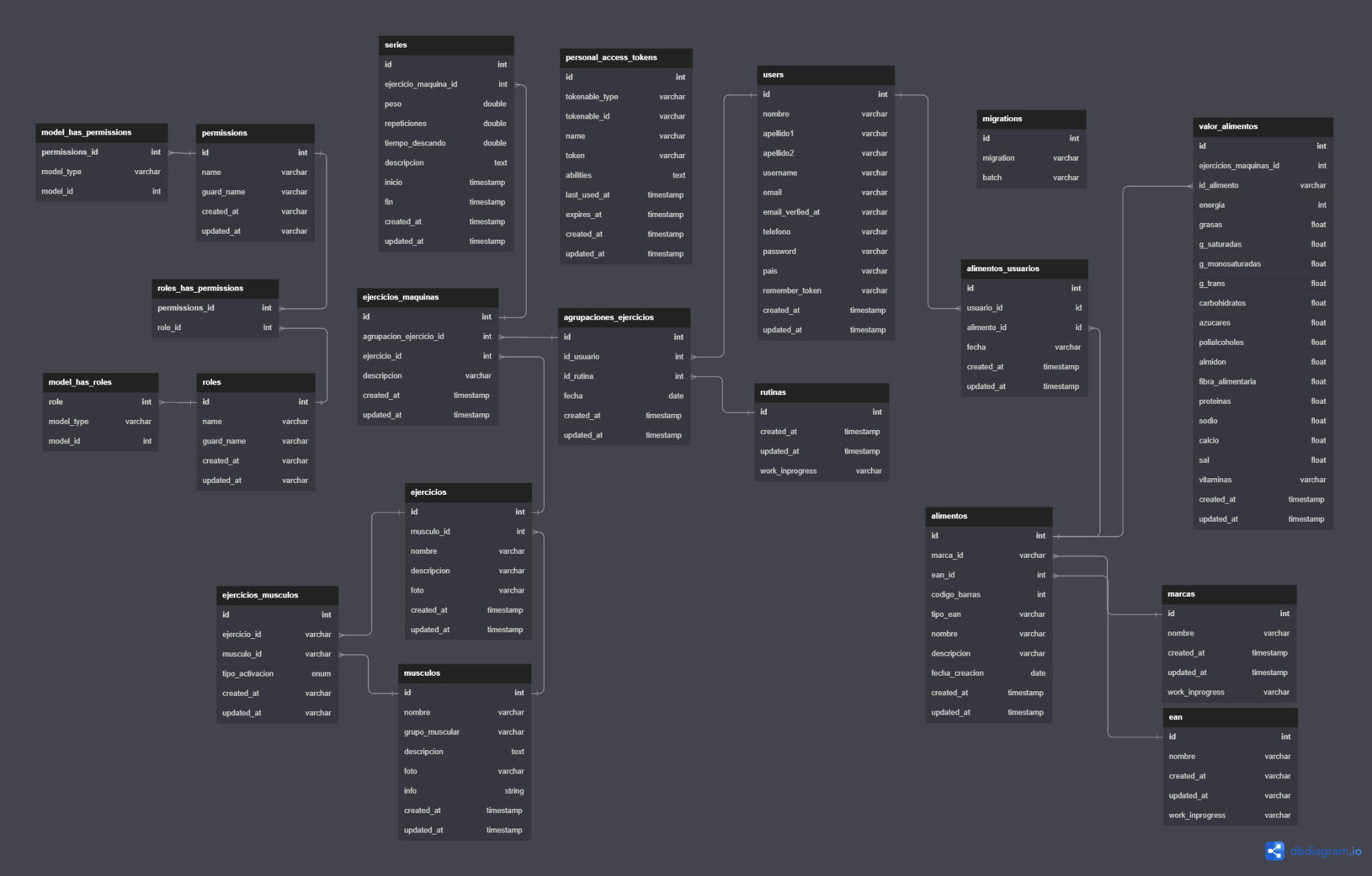


Una vez accedido desde el alimento la ventana de valores nutricionales sirve como ayuda para decantarse por alimentos según lo que más necesites.

# 

# 7. Diseño funcional: Descripción y diagramas de diseño funcional: entidad relación, casos de uso, diagramas de flujo...etc. Es preferible usar el estándar UML.

### Entidad relación

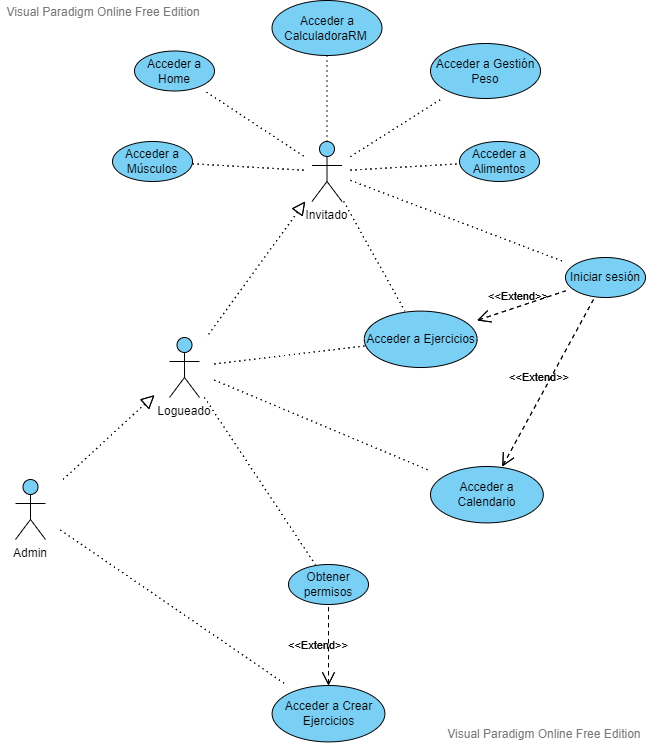


<https://github.com/Casta626/GymNotes/blob/main/GymNotes.png>

### 

### 

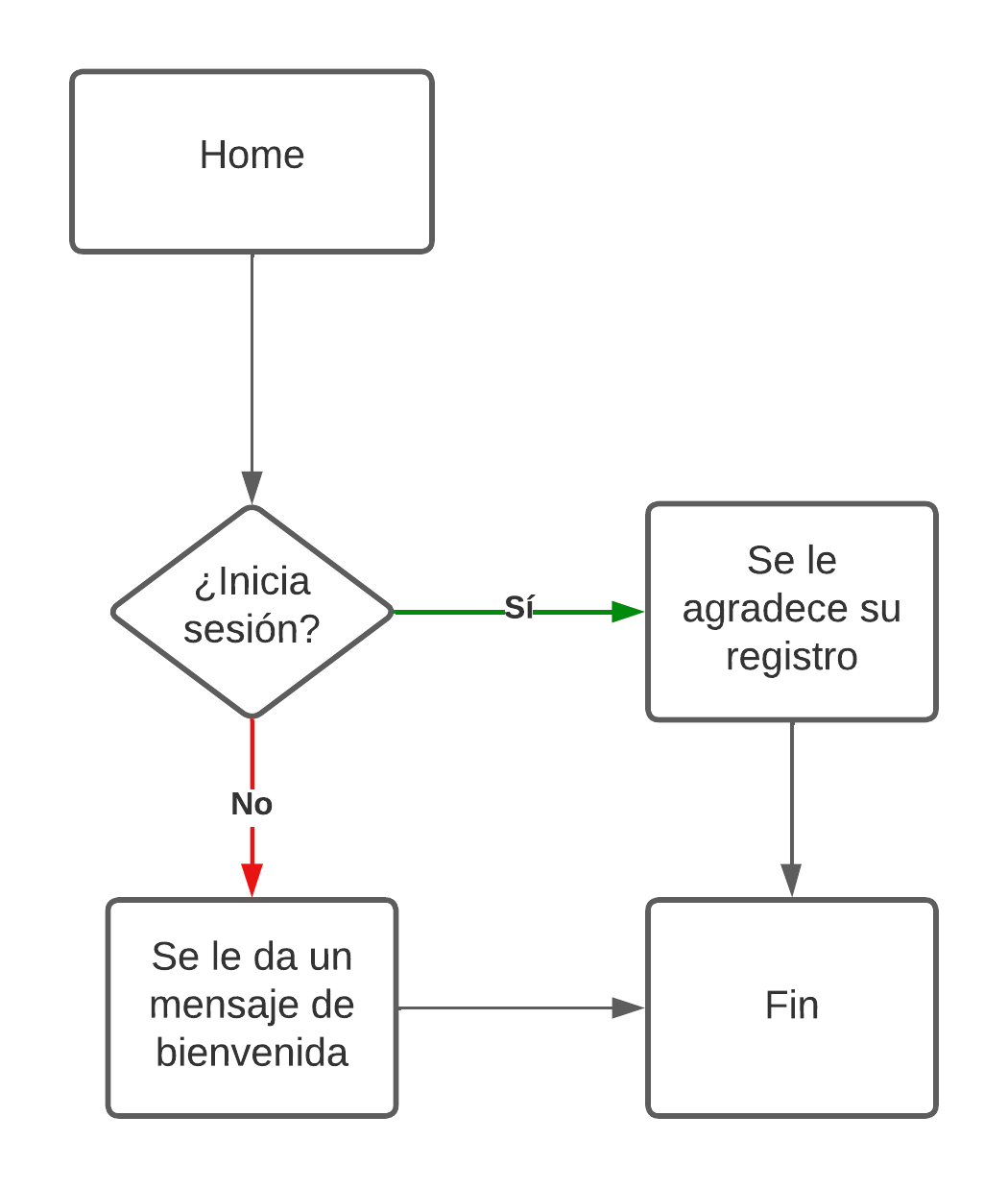
### Casos de uso



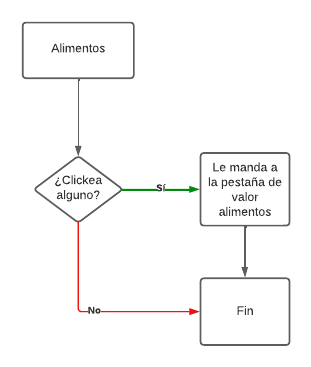
### 

### Diagramas de Flujo

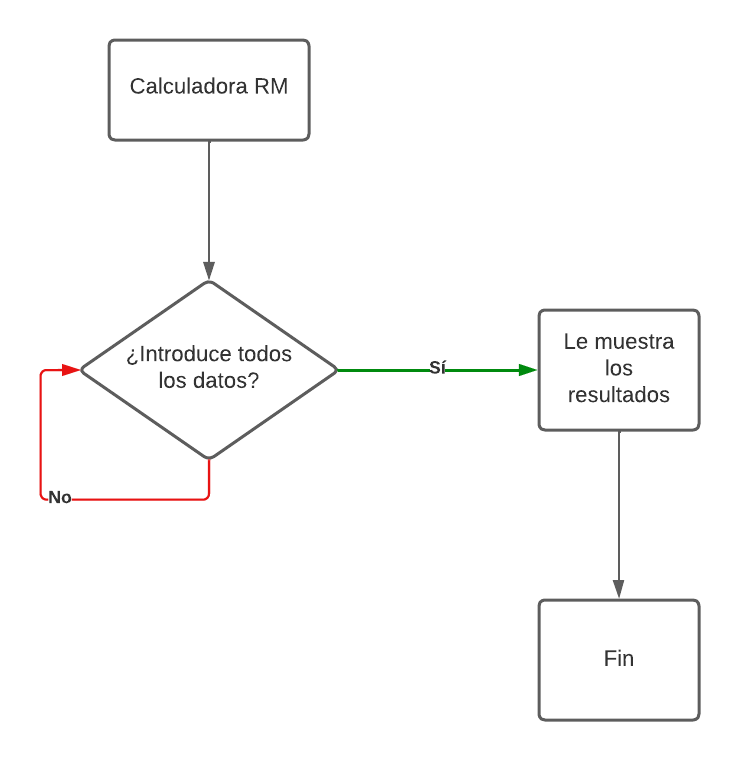
#### Home



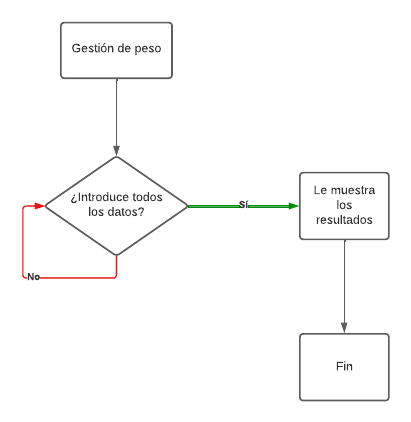
#### Alimentos



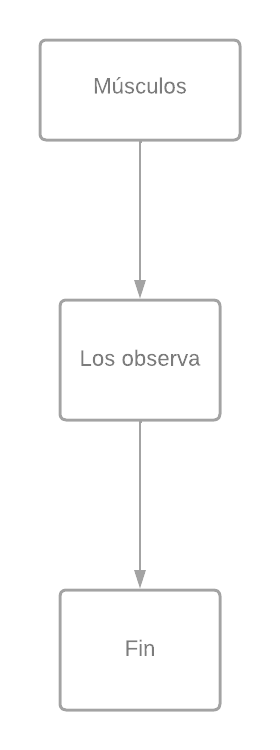
#### Calculadora RM



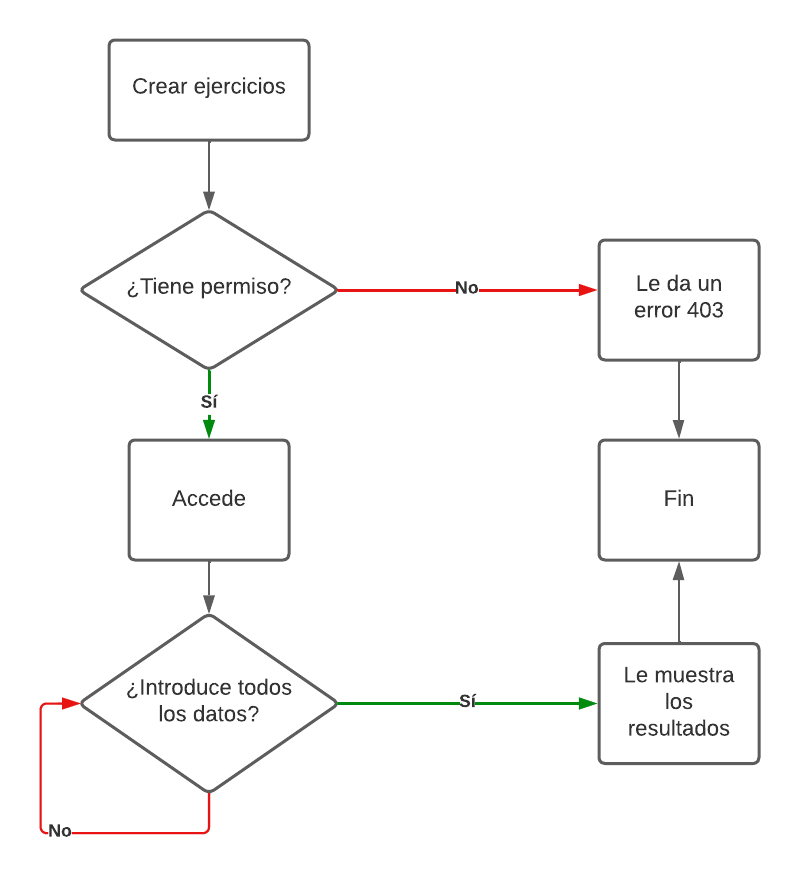
#### Gestión de peso



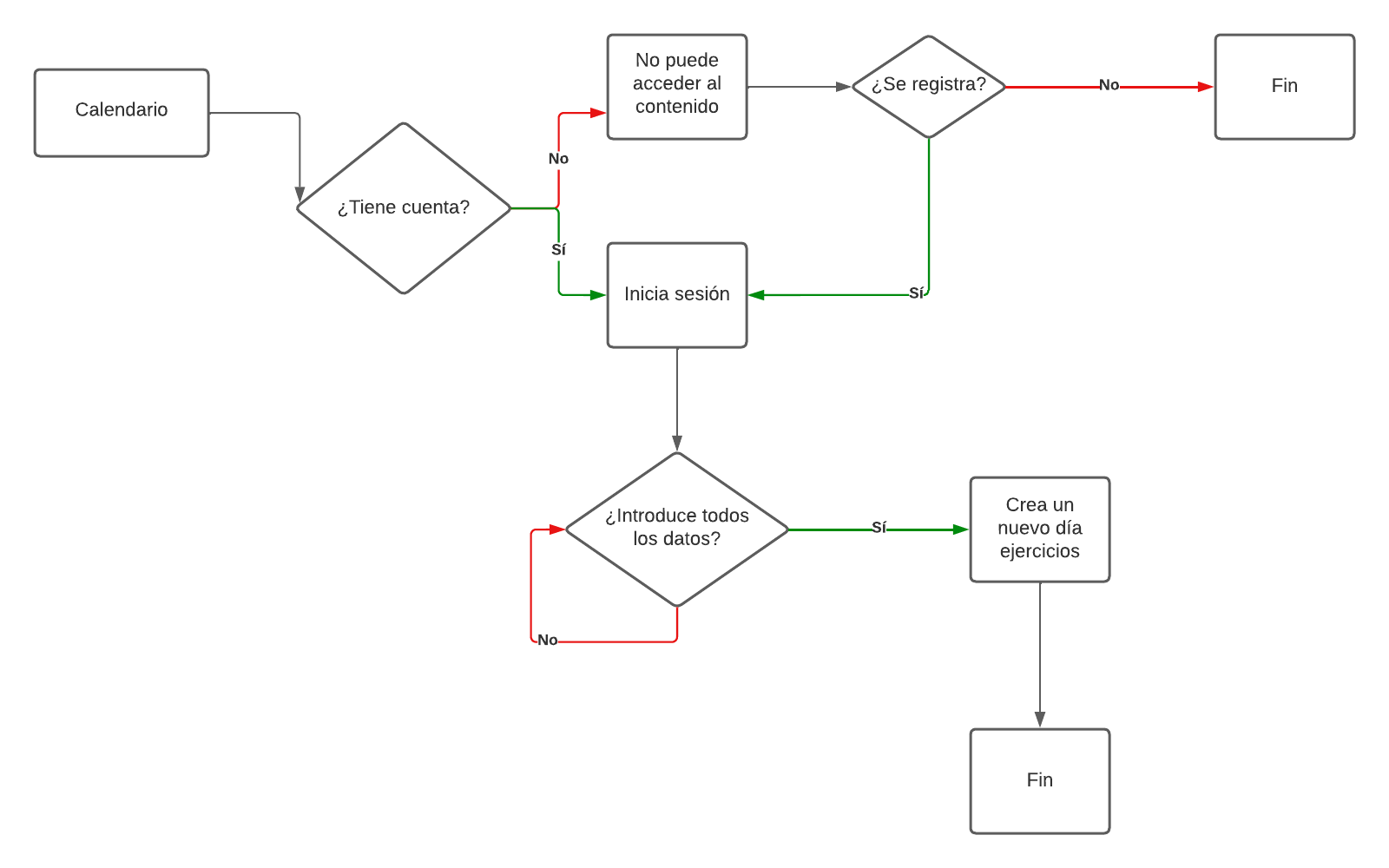
#### Músculos



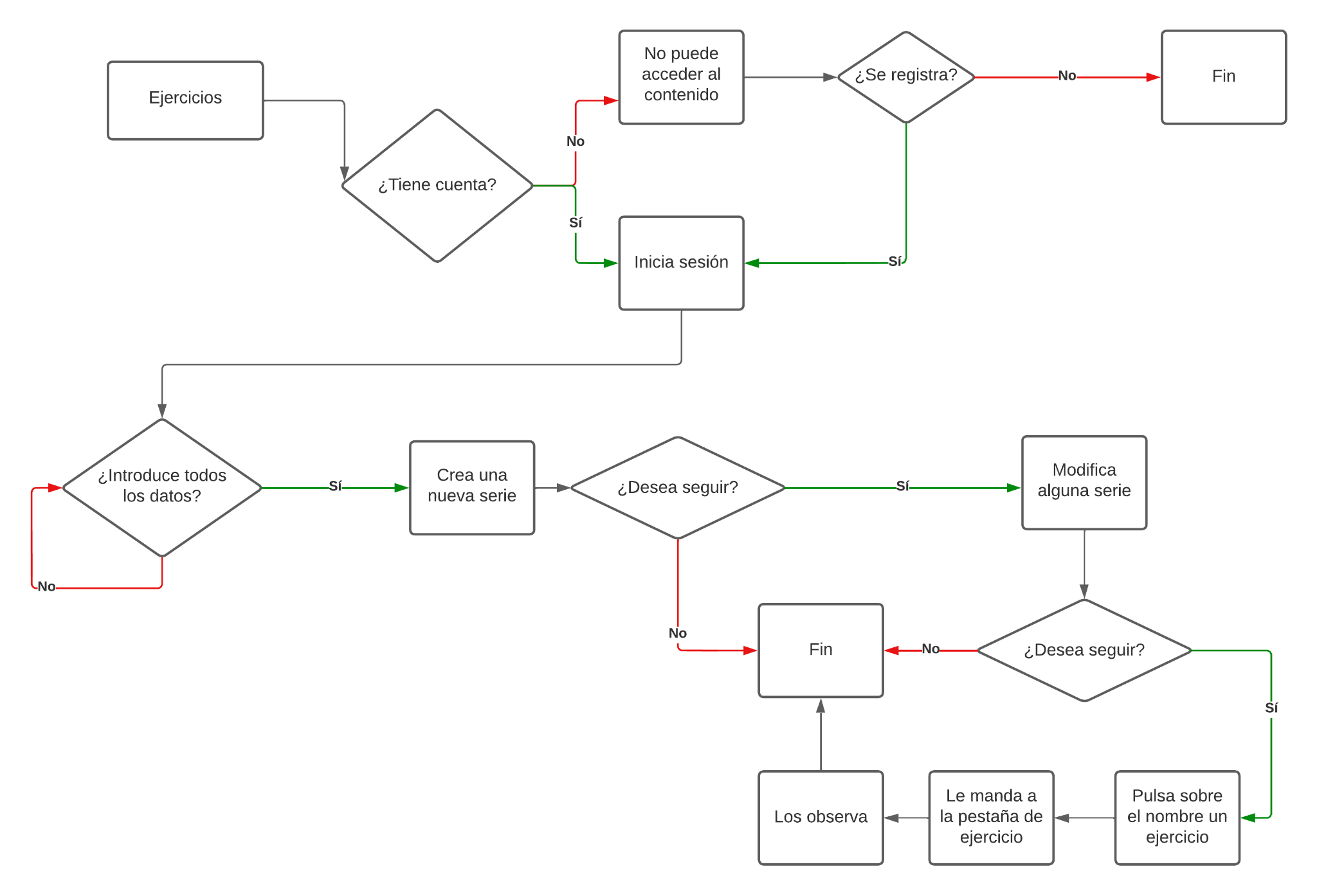
#### Crear ejercicios



#### Calendario



#### Ejercicios



# 

# 8. Desarrollo: Secuencia de desarrollo, dificultades encontradas y decisiones afrontadas. Descripción de las herramientas de control de versiones y revisión del código utilizadas.

# 

Secuencia de desarrollo

La lluvia de ideas me la salté ya que ya la tenía, luego procedí a pensar en las funciones importantes que debería de tener la aplicación, hice un prototipo a papel y con eso me puse manos a la obra.

Dificultades encontradas y decisiones afrontadas:

La primera decisión difícil fue cambiar el lenguaje de Python a PHP, esto fue debido a que en las prácticas se usaba PHP y mi tutor me recomendó hacer el proyecto con la misma tecnología para complementar lo aprendido en las prácticas con lo aprendido en casa y al final fue una buena decisión ya que me ha gustado bastante trabajar con PHP/Laravel.

Otra decisión dura fue dejar de lado el lector de códigos de barra, lo tuve que hacer por qué por mucho que buscaba solo encontraba de pago o a medio hacer y eso conlleva a no poder bien la parte de macros y alimentos, ya que si las metía sería por poner algún ejemplo.

Hubo ocasiones en las que me plantee usar livewire, que es una biblioteca de Laravel que te ayuda a hacer las páginas pero al principio costo utilizarlo y en las prácticas lo estaban utilizando al principio y luego se cambiaron a vue decidí que ya aprovecharía cuando aprendiera JavaScript para cambiar yo también a vue.

Descripción de las herramientas de control:

En mi caso yo solo he utilizado GitHub Desktop me parece una muy buena opción para controlar las versiones de tu proyecto ya que se pueden crear fácilmente nuevas ramas, hacer commits, rollbacks y de todo de una manera muy cómoda y con una interfaz bonita.

Herramientas de revisión del código:

Hasta el momento lo único que he utilizado son los formateadores de texto, ya que Laravel se centra en un MVC todo está dividido muy bien asi que formateaba el código y comprobaba que quedase lo más limpio posible.

# 

# 

# 

# 9. Pruebas: Descripción de las pruebas realizadas y automatización de las pruebas unitarias.

Persona 1:

Tiempo en hacer una consulta de Calculadora RM: 30 seg pensando

Tiempo en hacer una consulta de Gestión Peso: 1min 26

Tiempo en iniciar sesión: 31 seg

Tiempo en hacer una consulta de Calendario + Ejercicios: 1 min 38

Tiempo en salir de la sesión y llegar a home: 15 seg

Calificación de web: 8

### 

### Automatización de las pruebas unitarias

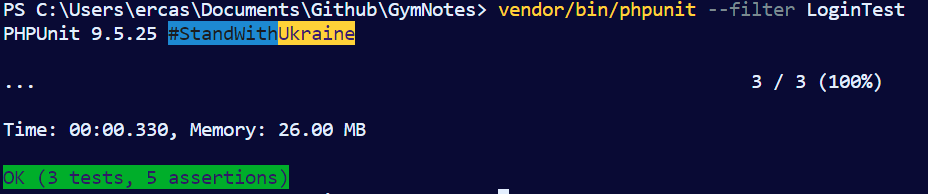
Para los test he utilizado dos archivos, LoginTest.php y UserTest.php.

#### LoginTest.php

Aquí hago tres pruebas diferentes, la primera trata sobre intentar acceder a rutas que no existen, la segunda accede a login y la tercera es un inicio de sesión.

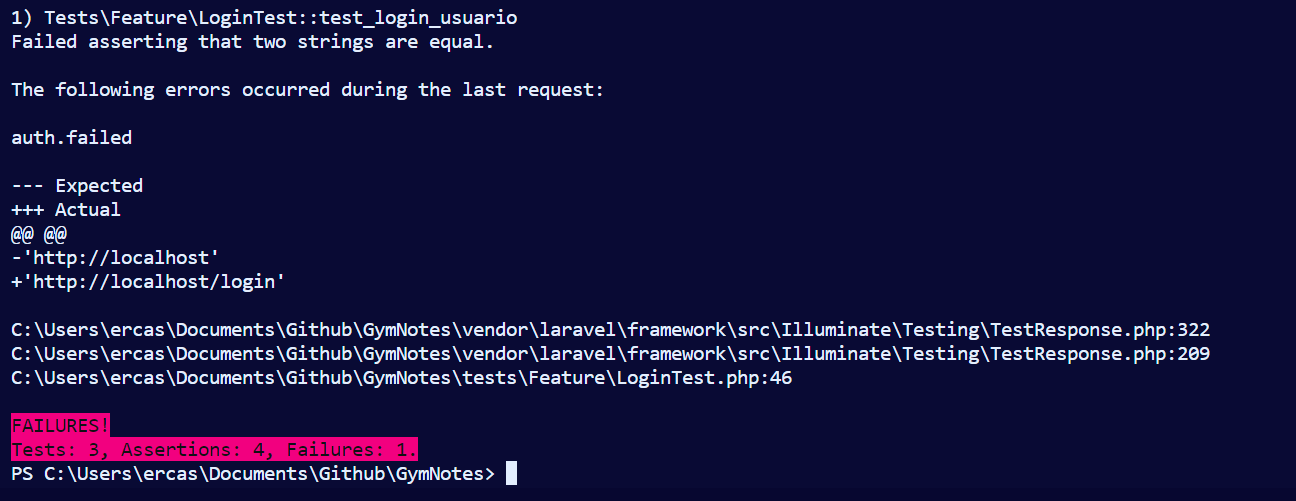
| <?php namespace Tests\Feature; use App\Models\User; use Illuminate\Foundation\Testing\RefreshDatabase; use Illuminate\Foundation\Testing\WithFaker; use Illuminate\Support\Facades\Hash; use Tests\TestCase;  class LoginTest extends TestCase {  public function test\_visitar\_web\_inexistente()  {  $this->get('/sala-estudio')  ->assertStatus(404);  }  public function test\_visitar\_web\_existente()  {  $this->get('/login')  ->assertStatus(200);  }   public function test\_login\_usuario()  {  // $usuario = new User();  // $usuario->username = 'SrPruebas';  // $usuario->email = "usuarioprueba@gmail.com";  // $usuario->password = Hash::make('usuarioprueba');  // $usuario->save();   $this->get('/login');  $credentials = [  "username" => 'SrPruebas',  "email" => "usuarioprueba@gmail.com",  "password" => 'usuarioprueba',  ];   $response = $this->post('/login', $credentials);  $response->assertRedirect('');  $this->assertCredentials($credentials);  }  } |
| --- |

Lo que está comentado es la creación del usuario y para demostrarlo voy a correr los test utilizando este comando en la terminal de Visual Studio vendor/bin/phpunit --filter LoginTest



y si cambio por ejemplo la contraseña:

| $credentials = [  "username" => 'SrPruebas',  "email" => "usuarioprueba@gmail.com",  "password" => 'esto es una contraseña diferente',  ]; |
| --- |

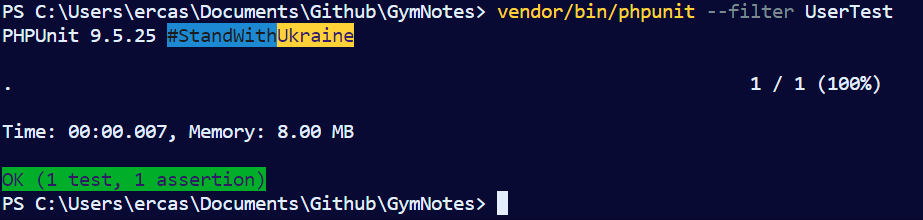


Muestra este error.

#### UserTest.php

| namespace Tests\Unit;  use App\Models\User; use PHPUnit\Framework\TestCase;  class UserTest extends TestCase {   public function test\_comprobar\_permiso()  {  $permiso = 'crear-ejercicios';   $usuario = [    "nombre" => "pruebas",  "permiso" => $permiso    ];   $response = $usuario["permiso"];   $this->assertEquals($response, $permiso);  } } |
| --- |

Este test unitario recoge el valor permiso de un usuario y comprueba que tenga ese permiso.



https://www.laraveltip.com/haciendo-pruebas-automatizadas-en-laravel-unit-test/

# 10. Distribución: Tecnología de distribución y descripción del proceso.

### Tecnología de distribución

Creación de contenido en redes sociales (Tik Tok, Instagram, Facebook).

Publicidad en redes sociales.

Publicidad mediante el uso de páginas.

### Descripción del proceso

Para la distribución de la aplicación la manera más sencilla que veo es utilizando las redes sociales, una vez terminada la aplicación por completo, me crearía una cuenta donde me presentaría junto a la aplicación y me dedicaría a hacer muchos videos utilizando y promocionando GymNotes por Tik Tok resubiendo todos los videos a Instagram y resubiendo solo los más virales a Facebook.

En el caso de que se hiciera mínimamente famosa me plantearía pagar por la publicidad en las redes sociales y algunas páginas web para darle un empujoncito y que llegase a más gente.

# 

# 11. Manual: Manual de instalación y uso de la aplicación.

# 

# 12. Conclusiones: Comparación del resultado con la idea inicial y mejoras futuras.

### Comparación del resultado con la idea inicial

Durante el transcurso del proyecto han ocurrido muchos cambios, nuevas ideas, ideas que han seguido e ideas que se han quedado a mitad de camino.

La idea principal era hacer una aplicación para apuntarse el día a día del gimnasio con el plus de poder añadir tus comidas para contar las calorías y tener una idea si con lo que estás comiendo vas encaminado con tu objetivo.

Desgraciadamente no he sido capaz de encontrar una forma de crear un lector de códigos de barra para PHP/Laravel así que el apartado de los alimentos se ha quedado a medias y por tanto la pestaña de Macros no ha podido seguir adelante.

Al no poder seguir con la idea de los macros me he centrado en seguir otras ideas que me han ido surgiendo, por ejemplo la calculadora de RM (Repetición Máxima), el gestionador de peso e incluso las páginas de músculos y ejercicio.

Lo bueno es que a medida que me iba atascando con los problemas se me iban ocurriendo más ideas y es gracias a eso que puedo seguir trabajando con la aplicación durante más tiempo e ir mejorando cada vez un poco más.

### 

### Mejoras futuras

En calendario quiero hacer que el calendario que actualmente está de adorno se pueda utilizar o al menos mejorar el datepicker para que tenga una mejor interfaz.

En ejercicios me gustaría que se ordenen por la fecha ya que actualmente van apareciendo según se van creando, tengo que seguir mirando como hacer un orderby si le acompaño a la función con un with, también me gustaría seguir mirando más fondos para poner en la tabla de ejercicios.

En músculos y ejercicio me gustaría aprender algo de blender e intentar hacer yo las imágenes con los modelos, o al menos encontrar buenas imágenes para así dejar ya todo preparado para cargar los datos.

También relacionado con las fotos quiero poner en marcha el helper que está de mientras en AgrupacionesEjercicioController.php que sirve para guardar las imágenes que se manden por request en base 64.

En el apartado de administración quiero hacer un crud de toda clase de datos, por ahora está solo para crear ejercicios pero la idea final es que se pueda crear, editar y eliminar músculos, ejercicio, alimentos, valores alimentos y roles para que se pueda hacer todo desde la aplicación.

Me encantaría añadir el apartado de macros cuando todo esté más automatizado.

Y por último me gustaría implementar las rutinas que fue una idea que se quedó en el aire pero está bastante bien, se trata de planificar la rutina de una semana y de esta manera puedes seleccionar la rutina y crear automáticamente los ejercicios de la semana sin tener que estar haciéndolo continuamente.

Aparte de todo esto aún tengo más ideas que están en modo espera, como por ejemplo cambiar la forma en la que aparece el músculo en ejercicio, ya que hay ejercicios que trabajan más de uno y en más o menos cantidad, una idea es que aparecieran los músculos que trabaja en un modal aparte para que se pueda leer bien.

Otra idea es poner a la derecha de crear ejercicios una imagen svg con los músculos para seleccionarlos directamente desde ahí y de paso utilizarlo para acceder a la vista de músculos, tal y como se hace en <https://musclewiki.com/> .

Y así muchas más ideas.

# 

# 13. Índice de tablas e imágenes.

# 

# 14. Bibliografía y referencias.