

Sistema web – gestión de entrenamiento



Descripción breve

La página web XXL será desarrollada y pensada para aquellas personas que tengan la necesidad de entrenarse con el asesoramiento de un profesional a través del sistema web. El usuario podrá ver su entrenamiento a seguir, junto con el profesional, ver sus avances o limitaciones. Además, el sistema gestionará la utilización de aparatos o maquinarias disponible tenga el Gimnasio.

Autores: Marcelo Peysa - Gustavo Godoy - Alfredo Nuñez - Gabriel Peysa



Grupo 24



(55) 1234-5678



correo@gymsistem.com.ar

Contenido:

Paginas	Nombre	Paginas2	Nombre3
1	Introduccion	13	Pagina en angular
2	Contenidos	14	Pagina en angular
3	Product Backlog - Sprint 0	15	Pagina en angular
4	Esquemas	16	Pagina en angular
5	Sprint: 1	17	Pagina en angular
6	Sprint: 1	18	Restropectiva
7	Breve explicacion backend y frontend 2021	19	The scrum framework / El marco de Scrum
8	Breve explicacion backend y frontend - 2	20	Evaluación Ingresos de Proceso 2 (EvP2) Ambas aulas
9	Breve explicacion backend y frontend - 3	21	Evaluación Ingresos de Proceso 2 (EvP2) Ambas aulas
10	Breve explicacion backend y frontend - 4	22	Evaluación Ingresos de Proceso 2 (EvP2) Ambas aulas
11	Breve explicacion backend y frontend - 10	23	Referencia - Restricciones
12	Pagina en angular	24	Personal - Ficha del documento

Introducción

Este documento tiene la finalidad de documentar un sistema web, realizado con HTML.

JavaScript, Angular, CSS, etc. intentando explicar el funcionamiento del sistema y procesos de construcción. De esta manera facilitar a un usuario y al ISPC la navegación.

El presente documento tiene como propósito definir las especificaciones funcionales, para el desarrollo de un sistema de información web que permitirá a los gimnasios llevar un control de la rutina de ejercicios de sus clientes como así también programar turnos asegurando acceso a las diferentes maquinas.

El sistema va a permitir a los dueños de gimnasios generar diferentes usuarios y planes de entrenamientos para sus clientes organizando así los tiempos de rutinas de ejercicios.

El sistema XLL va a ser desarrollado en una plataforma Web, esta aplicación va a permitir a los dueños de gimnasios administrar los planes de ejercicios de sus clientes y de las maquinas que van a usar en dichos planes, permitiendo una mejora en el tiempo de uso de cada máquina en los planes, teniendo el cliente siempre una maquina asegurada para su uso.

Además, el usuario (Cliente) podrá realizar su entrenamiento con un plan de ejercicio previamente cargado por un profesional, dependiendo solo del sistema y gestionando sus avance e historial desde la web.



Product Backlog 2021 - 2022

Sprint 0

- Documento formato ieee-830...
- Control de paginas, menu y contenido
- Menú de navegación
- Formulario de Alta de Usuario.
- Diseño de logo de la aplicación XXL
- Base de datos
- Pagina Nosotros
- Página de contacto

Sprint: Actividad 1 - Sprint 0

Se les solicitará un proyecto aplicando todos los contenidos dados en los módulos de la cursada (HTML5, CSS3, BOOTSTRAP, JAVASCRIPT), pueden recurrir a la web de W3School para extraer recursos necesarios.

Al proyecto desarrollado en 2021, la “consultora ISPC”, solicita en esta oportunidad, incorporar un módulo de e-commerce para comercializar sus productos y/o servicios online

Antes de comenzar

Definir Scrum Master inicial (No se registró en planilla) y registrarse todos dentro del nuevo repositorio. VER PLANILLA
Importar el repositorio anterior al nuevo brindado en el foro de cada grupo.
Revisar documentación IEEE830 si está completa y retomar desde allí para esta nueva etapa seguir documentando los avances.
Revisar la estructura web Semántica y responsive (RWD)

Pasos a seguir de acuerdo a las Fases del Ciclo de Vida de un Software:

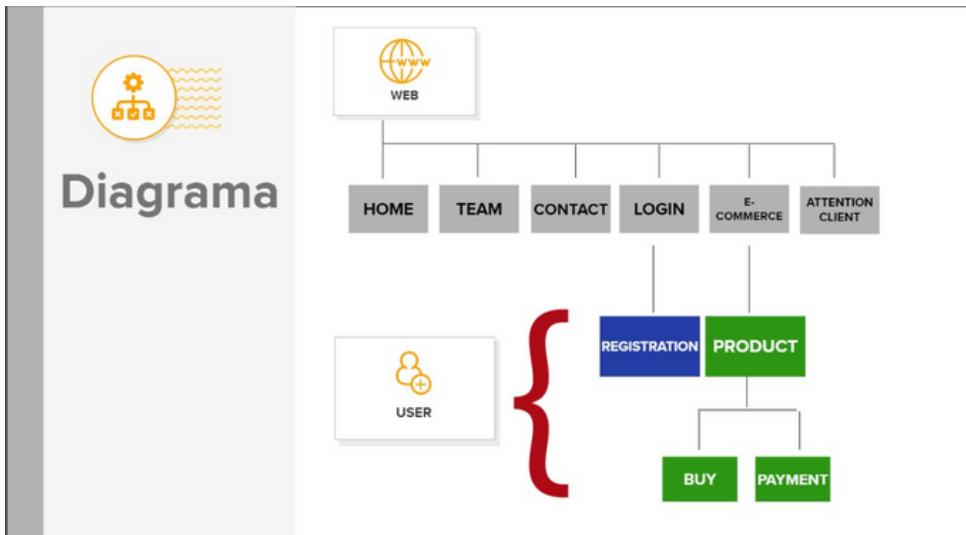
ANÁLISIS

Definir requerimientos para el nuevo módulo a desarrollar e-commerce (colocarlos en el Product Backlog del Project), a su vez revisar si han cumplimentado todos los requerimientos previos, realizar mejoras

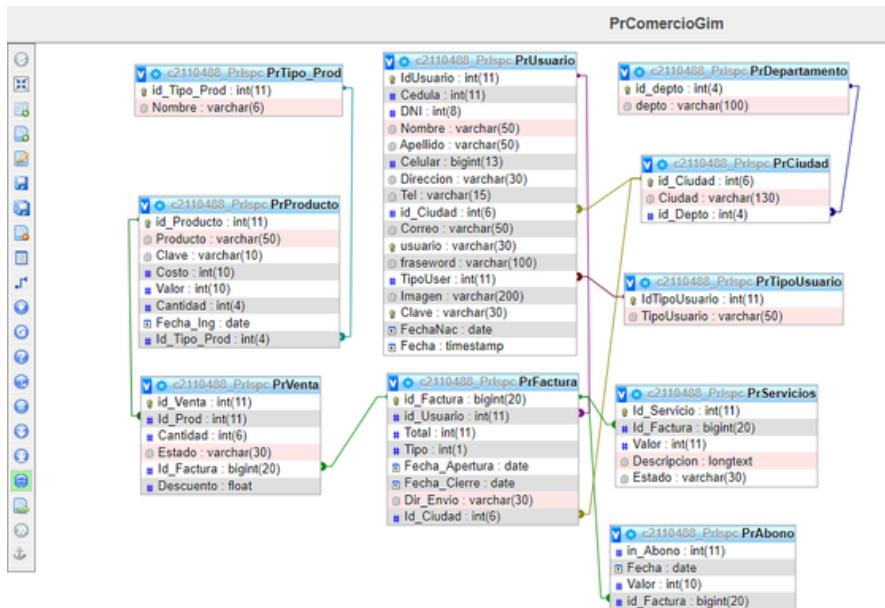
Plantear Historias de Usuarios y Tareas dependientes de las US para incorporarlas en el repositorio remoto gitHub. (Issues y Milestones) - Tener en cuenta la redaccion adecuada para las US y nomenclatura, ej “#US01 Como usuario quiero ingresar al carrito para poder comprar” .



Esquema



Modelo Relacional



Definir tareas dentro de las Historias de Usuario (GITHUB) ej dentro de las ISSUES #TK01 importar repositorio.

DISEÑO

Revisar la base de datos previa en MySQL si es funcional, e incorporar nuevas tablas necesarias para el funcionamiento del módulo e-commerce. (Ver el Modelo Relacional publicado en el libro Desafío E-commerce - Proyecto de trabajo Integrador
Crear su propio DER y Modelo relacional para documentar las tablas en la DB.
Crear un Diagrama de Clases y Casos de Uso para facilitar el modelado en POO.

IMPLEMENTACIÓN

Convertir el index.html (su home) en una SPA de Angular con módulos y components.



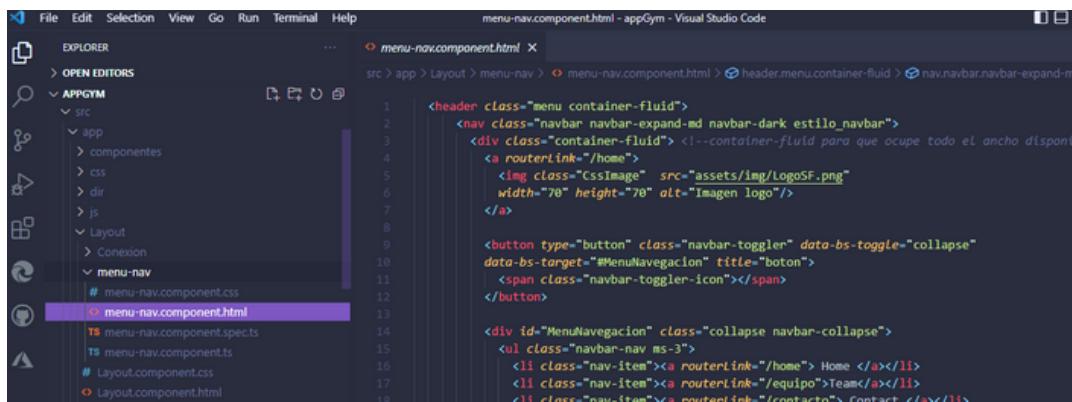
Sprint: Sprint 1

El sprint 1 es la implantación del FronEnd, para ver el app creada llamada appGim, lo puede hacer en el siguiente links:

<https://github.com/PPROF2-2022ProgWeb/g24-aula2-gimnasio-g24/tree/main/appGym>

IMPLEMENTACIÓN FRONTEND

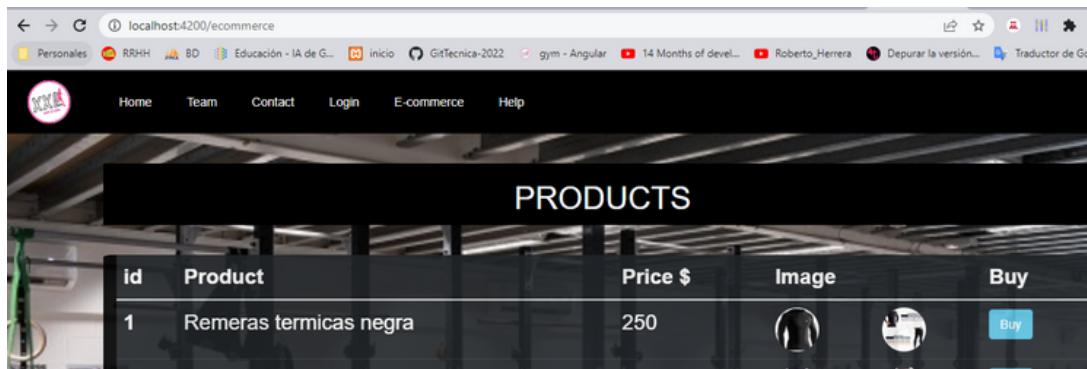
- Convertir los archivos .html del proyecto previo, en una SPA de Angular con módulos (Layouts y Pages por ejemplo) con sus components correspondientes. (HOME, REGISTRO, LOGIN, DASHBOARD)



```
<header class="menu container-fluid">
  <nav class="navbar navbar-expand-md navbar-dark estilo-navbar">
    <div class="container-fluid"> <!--container-fluid para que ocupe todo el ancho disponible-->
      <a routerLink="/">
        
      </a>

      <button type="button" class="navbar-toggler" data-bs-toggle="collapse"
        data-bs-target="#MenuNavegacion" title="boton">
        <span class="navbar-toggler-icon"></span>
      </button>

      <div id="MenuNavegacion" class="collapse navbar-collapse">
        <ul class="navbar-nav ms-3">
          <li class="nav-item"><a routerLink="/"> Home </a></li>
          <li class="nav-item"><a routerLink="/equipo"> Team</a></li>
          <li class="nav-item"><a routerLink="/contacto"> Contact </a></li>
        </ul>
      </div>
    </div>
  </nav>
</header>
```



Crear módulos y componentes para la tienda virtual o ecommerce. (PRODUCTO o SERVICIOS, PRODUCTO INDIVIDUAL, COMPRA).



Crear módulos y componentes para la tienda virtual o ecommerce. (PRODUCTO o SERVICIOS, PRODUCTO INDIVIDUAL, COMPRA).

```

File Edit Selection View Go Run Terminal Help
menu-nav.component.ts - appGym - Visual Studio Code
EXPLORER OPEN EDITORS APPGYM
# menu-nav.component.css
menu-nav.component.html
TS menu-nav.component.specs
TS menu-nav.component.ts
# Layout.component.css
Layout.component.html
TS Layout.component.ts
TS Layout.module.ts
TS LayoutComponent.ts
Page
> ayuda
> contacto
ecommerce
src > app > Layout > menu-nav > TS menu-nav.component.ts
1 import { Component, OnInit } from '@angular/core';
2
3
4 @Component({
5   selector: 'app-menu-nav',
6   templateUrl: './menu-nav.component.html',
7   styleUrls: ['./menu-nav.component.css']
8 })
9 export class MenuNavComponent implements OnInit {
10
11   constructor() { }
12
13   ngOnInit(): void {
14   }
15 }
16

```

Incorporar la navegabilidad de la aplicación mediante Routing con Angular.

```

src > app > TS app-routing.module.ts
1 import { NgModule } from '@angular/core';
2 import { RouterModule, Routes } from '@angular/router';
3 import { HomeComponent } from '../app/Page/home/home.component';
4 import { ContactoComponent } from '../app/Page/contacto/contacto.c
5 import { EquipoComponent } from '../app/Page/equipo/equipo.compon
6 import { LoginComponent } from '../app/Page/login/login.component'
7 import { AyudaComponent } from '../app/Page/ayuda/ayuda.component'
8 import { PageNotFoundComponent } from '../app/Page/PageNotFound/PageN
9 import { EcommerceComponent } from '../app/Page/eCommerce/eCommerce,
10 import { RegisterComponent } from '../app/Page/register/register.co
11
12
13 const routes: Routes = [
14
15 { path: 'home', component: HomeComponent },
16 { path: 'contacto', component: ContactoComponent },
17 { path: 'equipo', component: EquipoComponent },
18 { path: 'login', component: LoginComponent },
19 { path: 'ayuda', component: AyudaComponent },
20 { path: 'ecommerce', component: EcommerceComponent },
21 { path: 'register', component: RegisterComponent },
22
23 /* re direcciona al inicio */
24 {path: '', redirectTo: '/home', pathMatch: 'full'},
25
26 /* Para configurar la página 404 (not found) */
27 {path: '**', component: PageNotFoundComponent}
28

```

```

src > app > Layout > menu-nav > menu-nav.component.html > router-outlet
<div id="MenuNavegacion" class="collapse navbar-collapse">
<ul class="navbar-nav ms-3">
<li class="nav-item"><a routerLink="/home"> Home </a></li>
<li class="nav-item"><a routerLink="/equipo">Team</a></li>
<li class="nav-item"><a routerLink="/contacto">Contacto</a></li>

```

- Subirla al repo grupal, con GIT , en una branch por desarrollador para que cada uno tenga una copia. Luego crear una branch feature, para realizar nuestros cambios que no se encuentran aún en condiciones de incorporarse a la branch develop.

<https://github.com/PPROF2-2022ProgWeb/g24-aula2-gimnasio-g24>

Commit	Message	Date
gustavogidearg1 act appGym	actualizacion ieee	00ec1c8 hace 1 hora
2021	Act IEE830 y Evaluacion de ingles	
Conexion	Act IEE830 y Evaluacion de ingles	
_notes	Act IEE830 y Evaluacion de ingles	
appGym	act appGym	
css	Act IEE830 y Evaluacion de ingles	
dir	Act IEE830 y Evaluacion de ingles	
font	Act IEE830 y Evaluacion de ingles	



Grupo 24



(55) 1234-5678



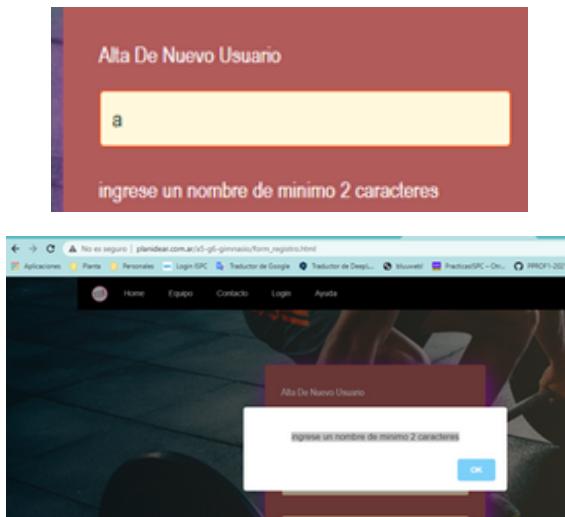
correo@gymsistem.com.ar

Breve explicacion backend y frontend 2021

1. Validación de campos vacíos mediante Bootstrap (de acuerdo al template de la web oficial), luego con Javascript la longitud de los campos, tipo de datos en los input. Los campos nombre y apellido deberán validarse con más de 2 caracteres.

Una de las validaciones están en el formulario registro de usuario: (http://planidear.com.ar/a5-g6-gimnasio/form_registro.html)

El nombre y apellido luego de hacer un cambio en elemento con onChange="validaNombre()"



El código en el archivo funcionesGrupo6.js

```
function validaNombre(){
    if(document.getElementById("txtNombre").value.length <= 2 ){
        document.getElementById ("AvNombre").innerHTML = "ingrese un nombre de minimo 2 caracteres" ;
        swal ( "ingrese un nombre de minimo 2 caracteres" );
        return false;
    }else{
        document.getElementById ("AvApellido").innerHTML = "ok con el ingreso nombre" ;
    }
}

function validaApellido(){
    if(document.getElementById("txtApellido").value.length <= 2){
        document.getElementById ("AvApellido").innerHTML = "ingrese un apellido de minimo 2 caracteres" ;
        swal ( "ingrese un Apellido de minimo 2 caracteres" );
        return false;
    }else{
        document.getElementById ("AvApellido").innerHTML = "ok con el ingreso apellido" ;
    }
}
```

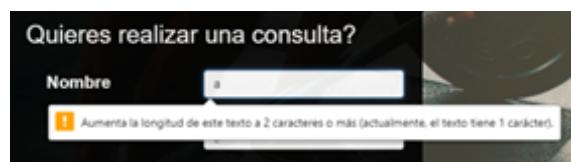


En el formulario:

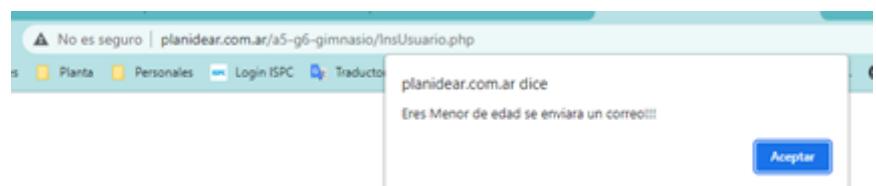
```
<input class="Controls" type="text" name="txtNombre" id="txtNombre" onChange="validaNombre()" maxLength="20"
placeholder="Ingrese su nombre">
<h4 id="AvNombre"></h4>

<input class="Controls" type="text" name="txtApellido" id="txtApellido" placeholder="Ingrese su Apellido"
onChange="validaApellido()" maxLength="20" required>
<h4 id="AvApellido"></h4>
```

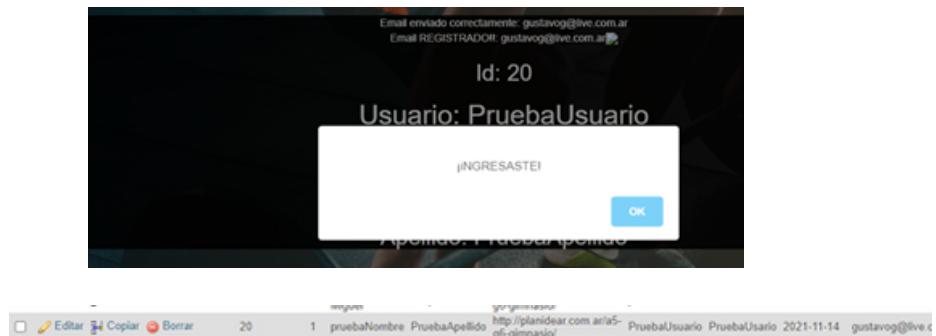
En un pequeño formulario en el index (<http://planidear.com.ar/g24-aula2-gimnasio-g24/index.html>) Se agrego restricciones con el html5:



- 2.Crear una función en JavaScript para mostrar un cálculo de fechas (edad, día de turno, u otro pertinente al proyecto en desarrollo).
- En el formulario registro de usuario: (http://planidear.com.ar/g24-aula2-gimnasio-g24/form_registro.html)
- Al ingresas la fecha de nacimiento además de crear el registro y cargarlo a la base de datos, valida si el usuario es menor de edad. En caso de que sea menor de edad además de aparecer una alerta, se enviara un correo a su correo cargado:
-

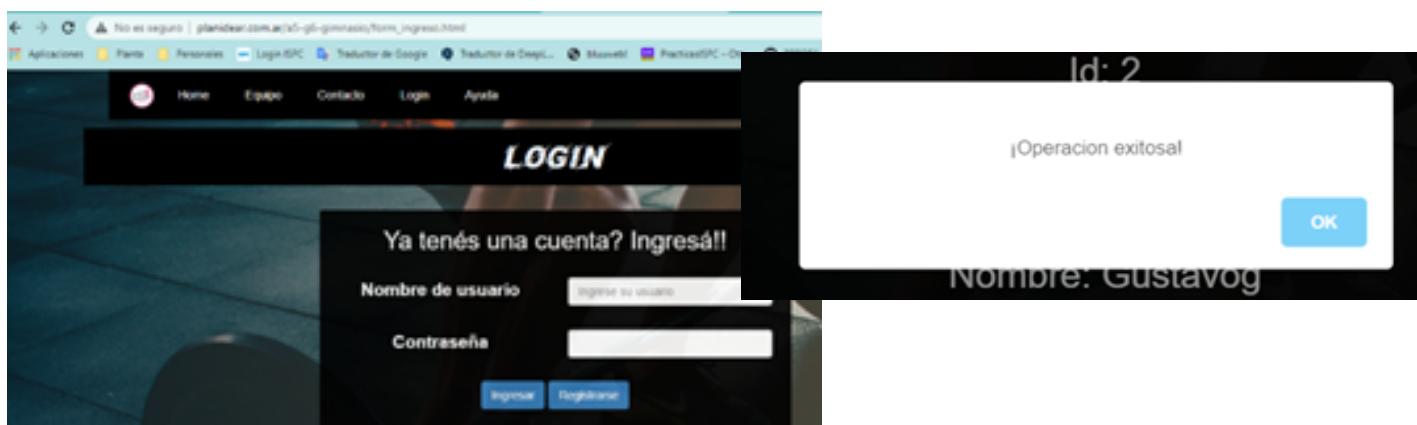


En el correo:



- 3.Uno de los formularios debe tener funcionalidad en el botón Enviar, mostrando un Alert de operación exitosa.

- El formulario de login (http://planidear.com.ar/g24-aula2-gimnasio-g24/form_ingreso.html) después de hacer el post y validar los datos además de que no este vacío da la alerta ingresado con éxito:



- 4.Una vez procesado el formulario, mostrar en una pantalla siguiente, los datos procesados, la cual debe mantener la estética del sitio, luego de unos segundos, redireccionar a una página de sitio (por ejemplo al index).

En el ejemplo anterior, si el usuario no existe o esta vacío lo direcciona a la página login nuevamente:



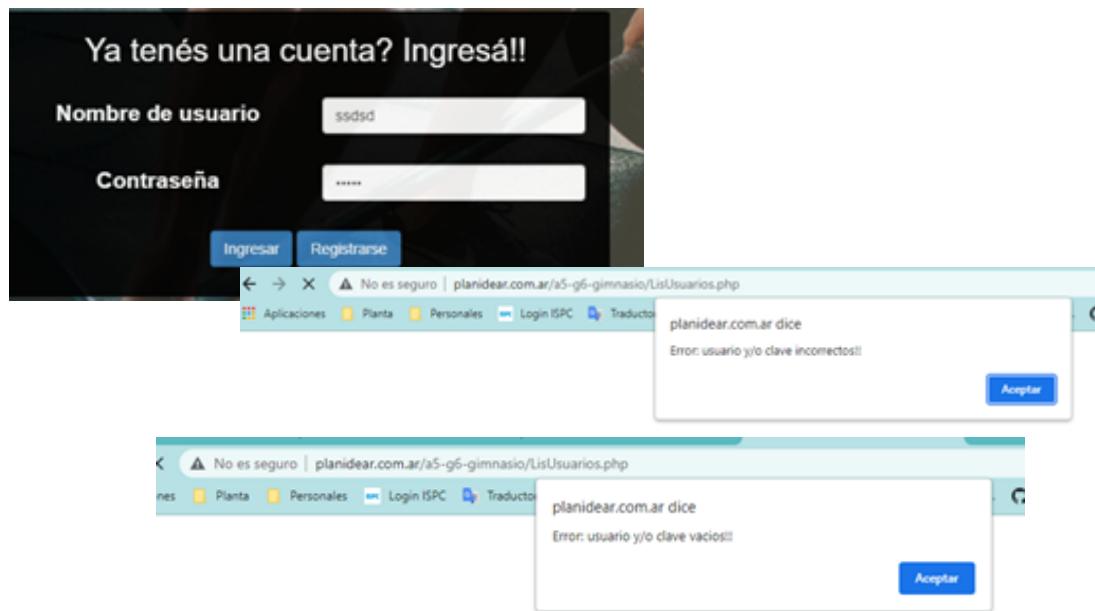
Grupo 24



(55) 1234-5678



correo@gymsistem.com.ar



5.Las funciones en JavaScript deberán estar en un archivo llamado funcionesGrupo99.js .

- Agregar al menos 2 eventos de Javascript para que el usuario interactúe con el DOM.

Este equipo > Windows 10 (C) >xampp >htdocs > planidear > a5-g6-gimnasio > js				
Nombre	Fecha de modificación	Tipo	Tamaño	
.notes	14/11/2021 12:57	Carpeta de archivos		
Archivo.js	10/11/2021 13:56	Archivo JavaScript	1 KB	
funcionesGrupo6.js	14/11/2021 12:57	Archivo JavaScript	2 KB	
scriptContacto.js	10/11/2021 13:56	Archivo JavaScript	2 KB	
scriptForm.js	10/11/2021 13:56	Archivo JavaScript	1 KB	

```
//funciones.js
//valida la edad después de enviar los datos desde
//http://planidear.com.ar/a5-g6-gimnasio/form_registro.html
function EresMayoEdad(){
    alert("Eres mayor de edad");
    //swal ( "Eres mayor de edad" );
}

function EresMenorEdad(){
    alert("Eres Menor de edad se enviara un correo!!!!");
    //swal ( "Eres Menor de edad se enviara un correo!!!!" );
}

//Esta función está en la página index.html
//cuando se pasa el mouse por encima del logo en el cuero de la pagina
//esta cambia de tamaño
function CambiarImagen(){
    document.getElementById("logo").style.height="200px"
    document.getElementById("logo").style.width="200px"
}
```



```
//las validaciones validar nombre y apellido lo hace cuando existe un cambio  
//en el formulario registro, si tiene menos de los caracteres da un aviso, ademas cambia el DOM h4  
  
function validaNombre(){  
    if(document.getElementById("txtNombre").value.length <= 2 ){  
        document.getElementById ("AvNombre"). innerHTML = "ingrese un nombre de minimo 2 caracteres" ;  
        swal ( "ingrese un nombre de minimo 2 caracteres" );  
        return false;  
    }else{  
        document.getElementById ("AvNombre"). innerHTML = "ok con el ingreso nombre" ;  
    }  
}  
  
function validaApellido(){  
    if(document.getElementById("txtApellido").value.length <= 2){  
        document.getElementById ("AvApellido"). innerHTML = "ingrese un apellido de minimo 2 caracteres" ;  
        swal ( "ingrese un Apellido de minimo 2 caracteres" );  
        return false;  
    }else{  
        document.getElementById ("AvApellido"). innerHTML = "ok con el ingreso apellido" ;  
    }  
}
```



Nuestra página fue realizada con HTML, JS y PHP, ahora deberemos pasar la misma página en Angular, para luego hacer un sistema venta de productos.

Lo primero que debemos ver es que versión tenemos de node, luego si tenemos instalado angular

Ej:

Para ver la version de node:

```
PS C:\xampp\htdocs\planidear\app-lspc> node -v
```

```
v16.15.0
```

Para ver la version de angular:

```
npm -v
```

```
PS C:\xampp\htdocs\planidear\app-lspc> npm -v
```

```
8.12.2
```

En caso de no tener instalado angular, realizar la instalación con el siguiente comando en la terminal de Sistema:

Instalar angular cli

```
npm install -g @angular/cli
```

Crear una nueva aplicación:

```
ng new my-app, en este caso appgym
```



Cambiar de carpeta:

```
cd my-app
```

Abrir la aplicación:

```
ng serve --open
```

Para crear un nuevo componente, como por ejemplo el menu:

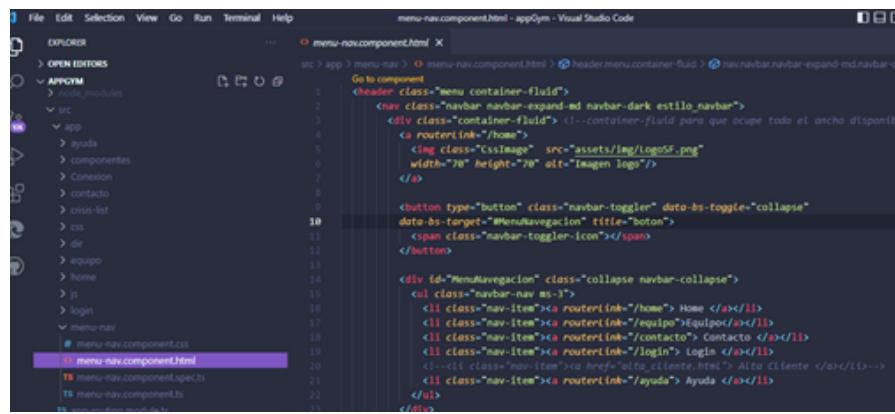
Debemos agregar en la terminal (dentro de nuestro app):

```
ng new menu-nav
```

```
menu-nav.component.ts - appGym - Visual Studio Code
src > app > menu-nav > TS menu-nav.component.ts > MenuNavComponent
1 import { Component, OnInit } from '@angular/core';
2
3
4 @Component({
5   selector: 'app-menu-nav',
6   templateUrl: './menu-nav.component.html',
7   styleUrls: ['./menu-nav.component.css']
8 })
9 export class MenuNavComponent implements OnInit {
10
11   constructor() { }
12
13   ngOnInit(): void {
14   }
15
16 }
17
```



En el archivo menu-nav.component.html, agregar el contenido html a mostrar



```
<header class="menu container-fluid">
  <nav class="navbar navbar-expand-md navbar-dark estilo-navbar">
    <div class="container-fluid" ><!--container-fluid para que ocupe todo el ancho disponible-->
      <a routerLink="/home">
        
      </a>

      <button type="button" class="navbar-toggler" data-bs-toggle="collapse"
        data-bs-target="#MenuNavegacion" title="boton">
        <span class="navbar-toggler-icon"></span>
      </button>

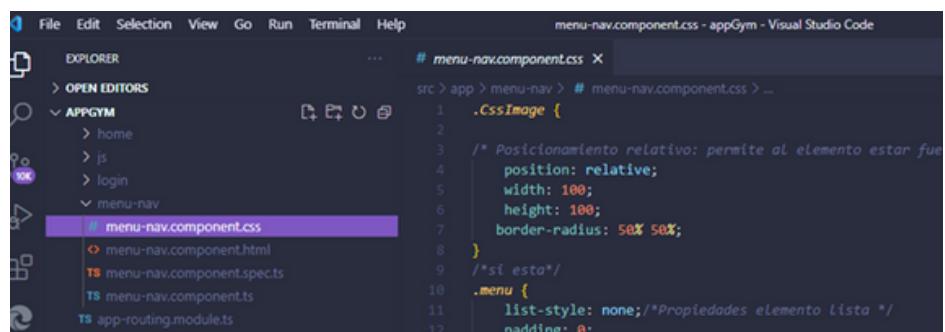
      <div id="MenuNavegacion" class="collapse navbar-collapse">
        <ul class="navbar-nav ms-3">
          <li class="nav-item"><a routerLink="/home"> Home </a></li>
          <li class="nav-item"><a routerLink="/equipo">Equipo</a></li>
          <li class="nav-item"><a routerLink="/contacto"> Contacto </a></li>
          <li class="nav-item"><a routerLink="/login"> Login </a></li>
          <li class="nav-item"><a href="#">Alta Cliente</a></li>
          <li class="nav-item"><a routerLink="/ayuda"> Ayuda </a></li>
        </ul>
      </div>
    </div>
```

Para mostrar la imagen, primero debemos guardar la carpeta de imágenes dentro de assets:

```

```

En menu-nav.component.css agregaremos el estilo de este componente menu-nav:



```
.CssImage {
  /* Posicionamiento relativo: permite al elemento estar fuera del flujo normal de los otros elementos */
  position: relative;
  width: 100px;
  height: 100px;
  border-radius: 50% 50%;

}

/* si esta */
.menu {
  list-style: none; /* Propiedades elemento Lista */
  padding: 0;
```



Para trabajar con los enlaces de botones, ya no trabajamos con ``, sino que nos direccionamos a través de las rutas:

En el archivo `app-routing.module.ts`, dentro `src`, importar y agregar dentro de la constante `routes` las pistas de cada componentes. Los datos tomarlo del componentes terminado en `.ts`, así para poder navegar entre ellos:

```
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { HomeComponent } from './home/home.component';
import { ContactoComponent } from './contacto/contacto.component';
import { EquipoComponent } from './equipo/equipo.component';
import { LoginComponent } from './login/login.component';
import { AyudaComponent } from './ayuda/ayuda.component';
import { MenuNavComponent } from './menu-nav/menu-nav.component';

const routes: Routes = [
  { path: 'home', component: HomeComponent },
  { path: 'contacto', component: ContactoComponent },
  { path: 'equipo', component: EquipoComponent },
  { path: 'login', component: LoginComponent },
  { path: 'ayuda', component: AyudaComponent },
];

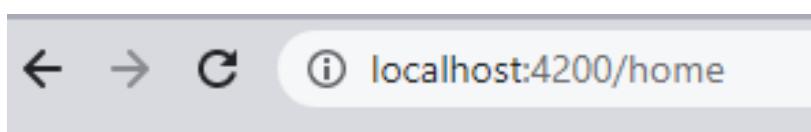
@NgModule({
  imports: [RouterModule.forRoot(routes)],
  exports: [RouterModule]
})
export class AppRoutingModule { }
```

- Dentro de la etiqueta de hipertexto agregar `routerLink="/comando"`:

```
<li class="nav-item"><a routerLink="/home"> Home </a></li>
```

```
<li class="nav-item"><a routerLink="/equipo">Equipo</a></li>
<li class="nav-item"><a routerLink="/contacto"> Contacto </a></li>
<li class="nav-item"><a routerLink="/login"> Login </a></li>
<li class="nav-item"><a routerLink="/ayuda"> Ayuda </a></li>
```

De esta forma funcionaran los botones de navegación:

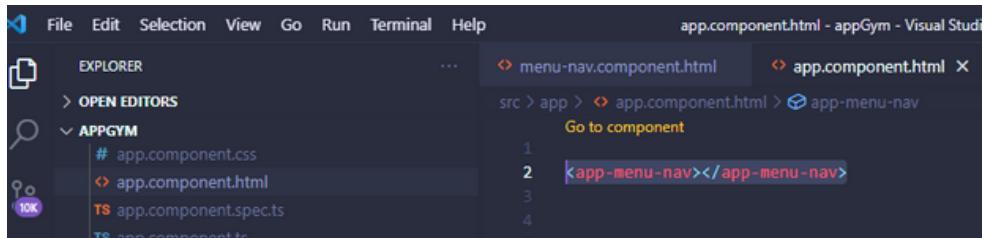


Para que se muestren los componentes/paginas, debemos primero importar y agregarla en el archivo `app.module.ts`:

```
File Edit Selection View Go Run Terminal Help
EXPLORER
OPEN EDITORS
APPSYM
menu-nav.component.html TS app-routing.module.ts TS app.routing.ts TS app.module.ts
src > app > TS app.module.ts - AppModule
9 import { AppComponent } from './app.component';
10 import { MenuNavComponent } from './menu-nav/menu-nav.component';
11 import { HomeComponent } from './home/home.component';
12 import { EquipoComponent } from './equipo/equipo.component';
13 import { ContactoComponent } from './contacto/contacto.component';
14
15
16
17 @NgModule({
18   declarations: [
19     AppComponent,
20     MenuNavComponent,
21     HomeComponent,
22     EquipoComponent,
23     ContactoComponent,
```

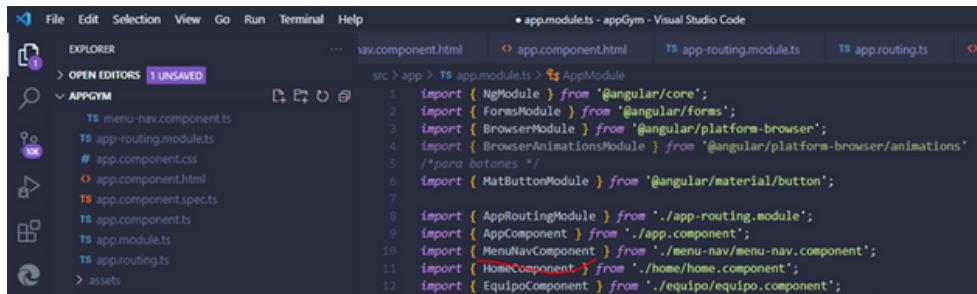


Luego en el componente app.component.html, agregar app de menú-var:



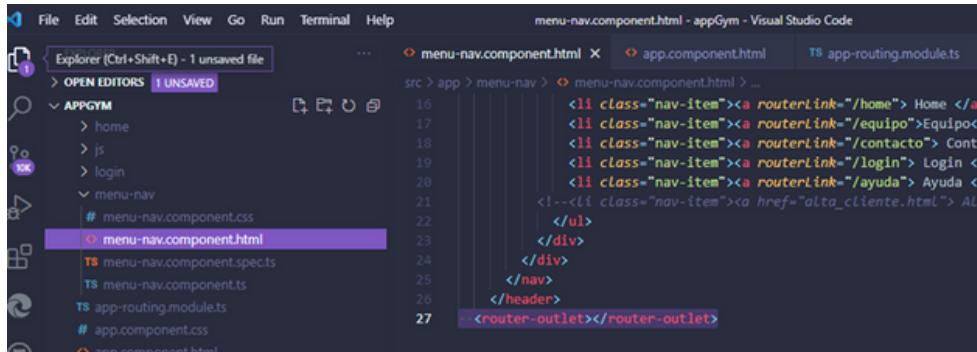
```
<app>
```

Antes declarada en app.module.ts:



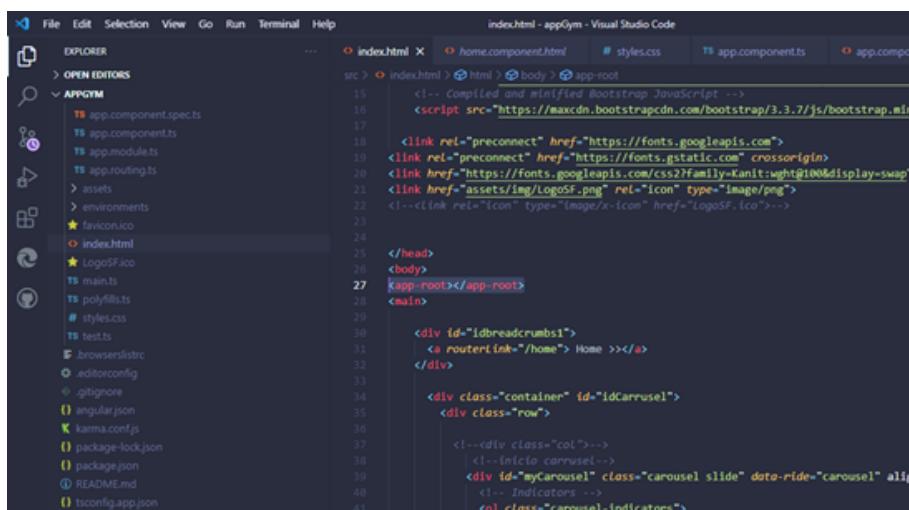
```
import { MenuNavComponent } from './menu-nav/menu-nav.component';
```

En menu-nav.component.html agregar <router-outlet></router-outlet> para que funcionen las rutas:



```
<router-outlet></router-outlet>
```

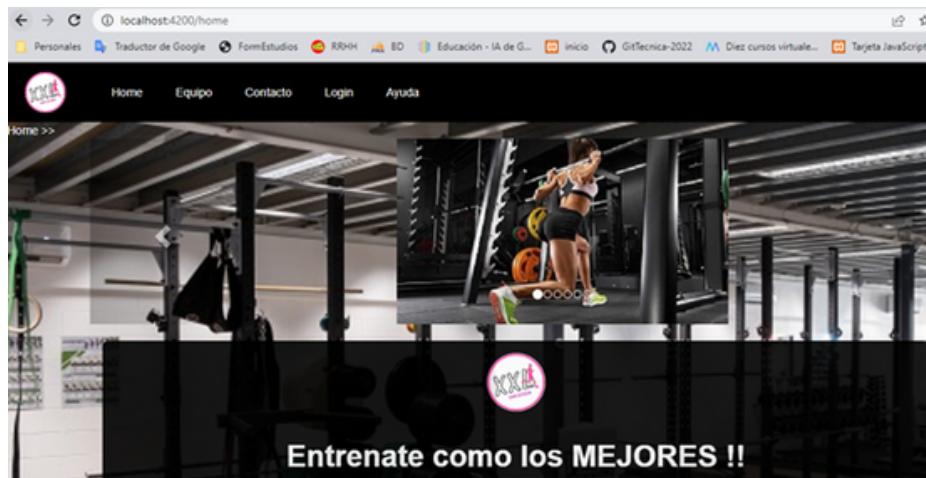
Para que la app ingrese con contenido, se agrego en el body <app-root></app-root> y código html:



```
<app-root></app-root>
```

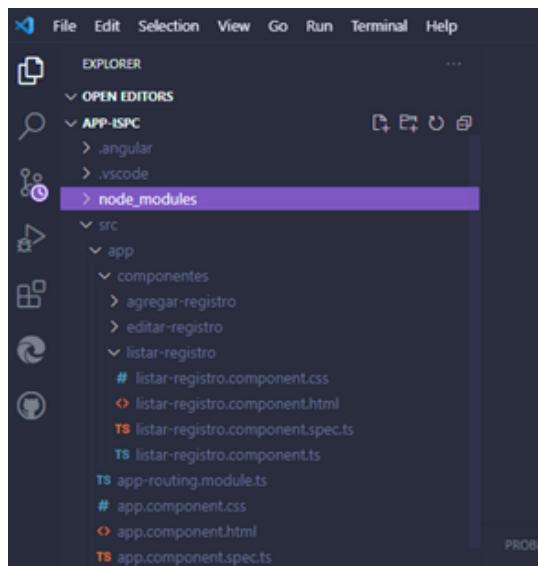


La aplicación se ve de este modo:



Consejos y atajos:

- Para crear la carpeta de componente y un componente nuevo:
ng g c componentes/nuevoComponente



Para instalar bootstrap

Detener el servidor antes

npm install bootstrap

Para crear un menú rápido:

bs5-nav-minimal-a

```
<nav class="navbar navbar-expand navbar-light bg-light">
  <div class="nav navbar-nav">
    <a class="nav-item nav-link active" href="#">Home <span class="visually-hidden">(current)</span></a>
    <a class="nav-item nav-link" href="#">Home</a>
  </div>
</nav>
```



Grupo 24



(55) 1234-5678



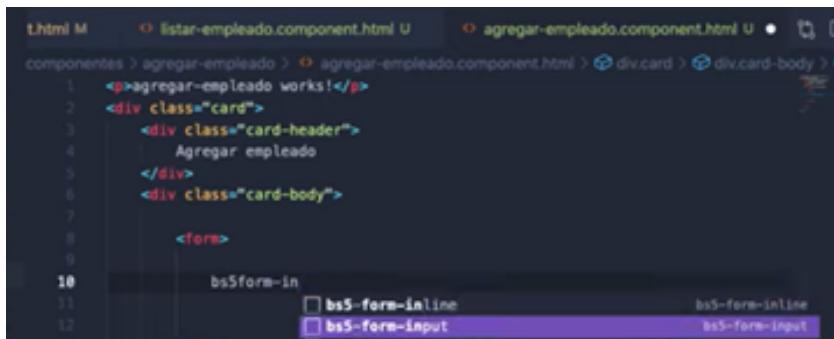
correo@gymsistem.com.ar

Modificar el scprit de Archivo angular.json

```
    ],
    "styles": [
        "src/styles.css",
        "node_modules/bootstrap/dist/css/bootstrapnode.min.css"
    ],
```

Crear una tabla rápido con bootstrap bs5-table-default

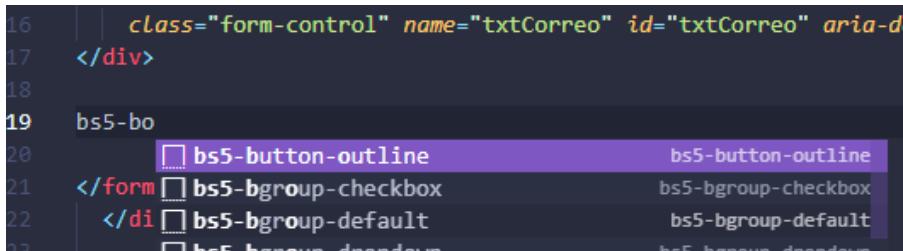
Crear un formulario rápido con bootstrap Bs5-form-input



A screenshot of a code editor showing a dropdown menu for 'bs5-form-input'. The menu items include 'bs5-form-inline' and 'bs5-form-input'. The 'bs5-form-input' item is highlighted with a purple background.

```
1. <p>agregar-empleado works!</p>
2. <div class="card">
3.   <div class="card-header">
4.     Agregar empleado
5.   </div>
6.   <div class="card-body">
7.
8.     <form>
9.
10.       bs5Form-in
11.         □ bs5-form-inline
12.         □ bs5-form-input
13.         bs5-form-input
```

Botton con bootstrap bs5-button-ouline



A screenshot of a code editor showing a dropdown menu for 'bs5-button-outline'. The menu items include 'bs5-button-outline' and 'bs5-button-outline'. The 'bs5-button-outline' item is highlighted with a purple background.

```
16.   <div class="form-control" name="txtCorreo" id="txtCorreo" aria-d
17.   </div>
18.
19.   bs5-bo
20.     □ bs5-button-outline
21.     </form> □ bs5-bgroup-checkbox
22.     </di> □ bs5-bgroup-default
23.     □ bs5-bgroup-dropdown
```



Restropectiva

<https://easyretro.io/dashboard>

Nombre de usuario: gymsistem@planidear.com.ar // Contraseña: Practicaslspc2022

EasyRetro RETROSPECTIVAS-practicas-2022 ★ Primera directiva practicaslspc

Establece el contexto del tablero aquí... Buscar Ordenar por orden Agregar Cuota Ajustes

PREPARAR ELESCENARIO

Como se sintieron con el trabajo en equipo?? Cuestionario: ggodoy1984@gmail.com.ar
practiceslspc 0 1 0

Que siente que se puede mejorar?? practiceslspc 0 0 0

Como se sintieron con lo que pido la tecnicatura?? practiceslspc 0 0 1

RECOGER DATOS

¿Deberíamos tener un tiempo de reunión y respetarla? practiceslspc 0 0 0

¿Cómo deberíamos hacerlo? practiceslspc 0 0 0

Que deberíamos mejorar para cumplir con el sprint? practiceslspc 0 0 0

¿Por qué costo terminar el sprint? practiceslspc 0 0 0

¿Quién se involucró? practiceslspc 0 0 0

Que tema debemos discutir para mejorar? practiceslspc 0 0 0

GENERAR APRENDIZAJES

Que anduve mal? practiceslspc 0 0 0

Que funcionó bien? practiceslspc 0 0 0

Porque pasó? practiceslspc 0 0 0

ACCIONES

¿Podemos mejorarlo? De que forma?? practiceslspc 0 0 0

Podemos cambiar para llegar al 100% de la actividad?? practiceslspc 0 0 0

CERRAR EL RETRO

La página funciona, pero debíramos subirla a la web practiceslspc 0 0 0

El trabajo se cumplió, pero debería mejorar los tiempos practiceslspc 0 0 0



Grupo 24

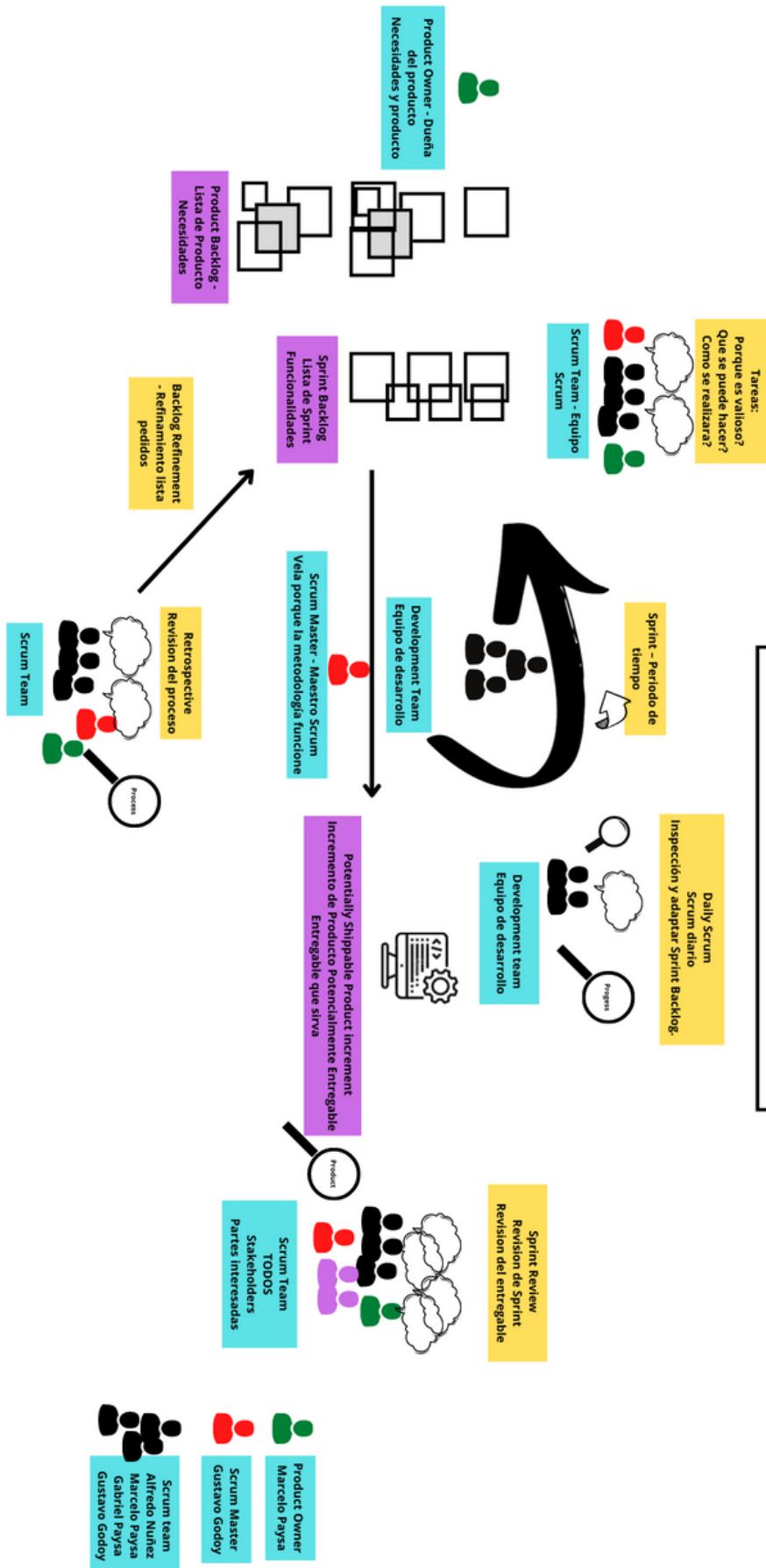


(55) 1234-5678



correo@gymsistem.com.ar

The scrum framework / El marco de Scrum



Evaluación Ingresos de Proceso 2 (EvP2) Ambas aulas

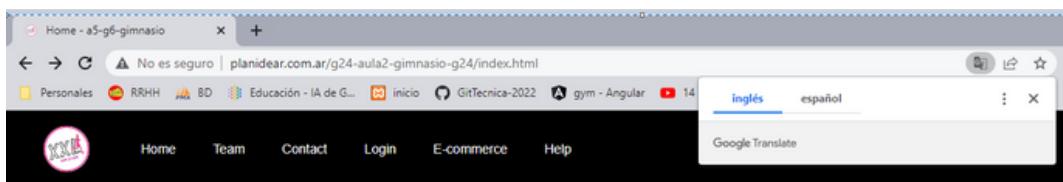
The system will allow gym owners to generate different users and training plans for their clients, thus organizing the times of exercise routines.

The XLL system will be developed on a Web platform, this application will allow gym owners to manage the exercise plans of their clients and the machines that they will use in said plans, allowing an improvement in the time of use. of each machine in the plans, with the client always having an insured machine for their use.

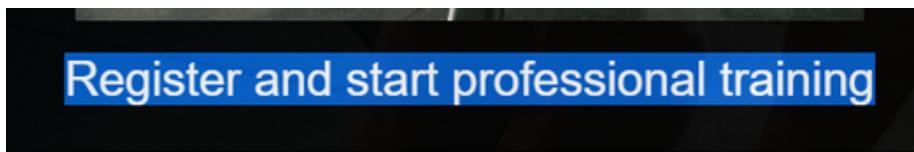
In addition, the user (Client) will be able to carry out their training with an exercise plan previously loaded by a professional, depending only on the system and managing their progress and history from the web.

Initial page coded in English:

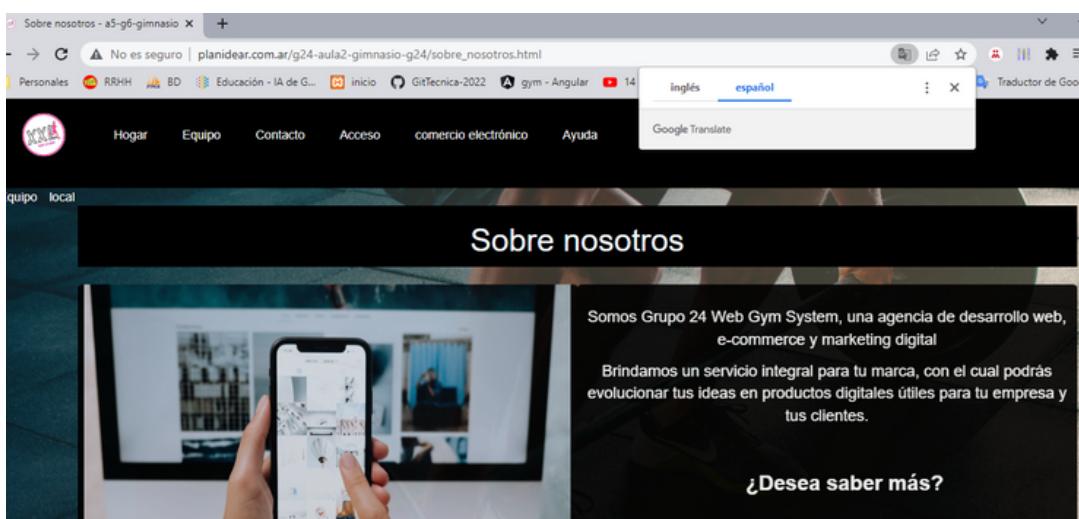
<http://planidear.com.ar/g24-aula2-gimnasio-g24/index.html>



The text is in the present continuous



The Team page speaks in the first person plural



Grupo 24

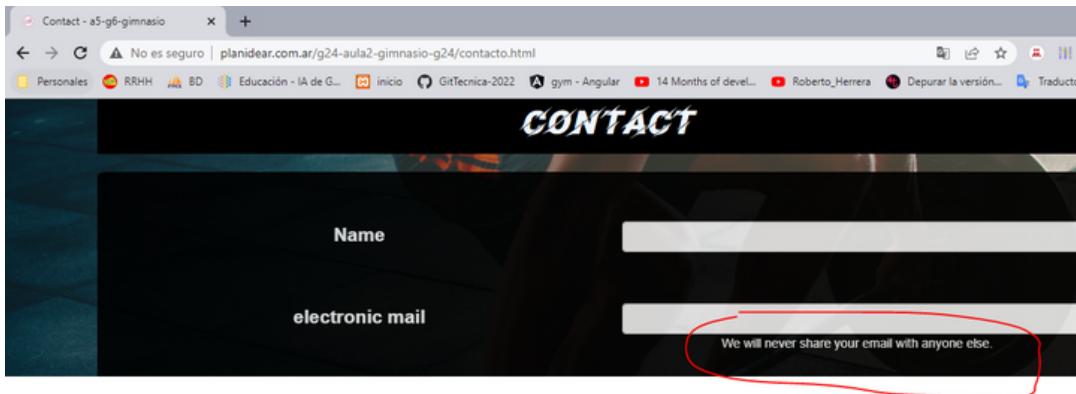


(55) 1234-5678

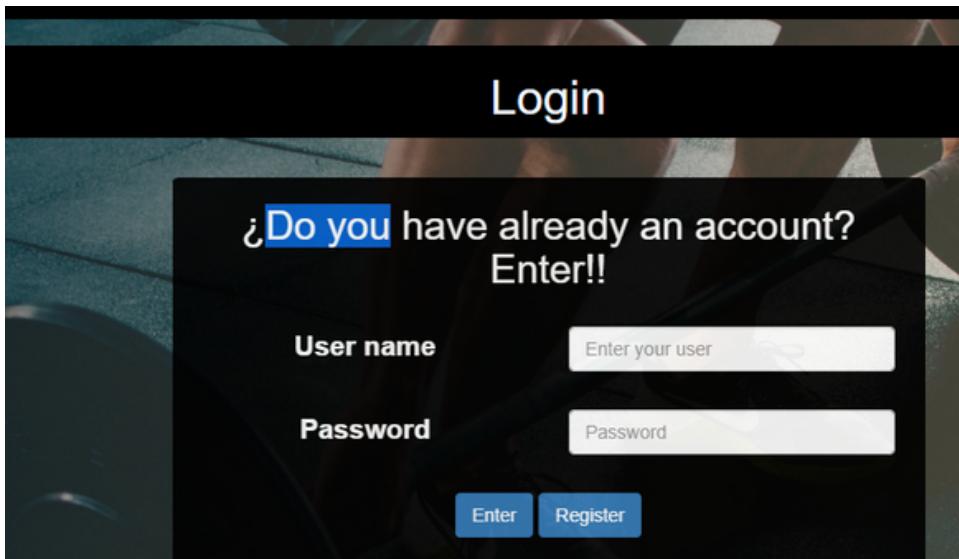


correo@gymsystem.com.ar

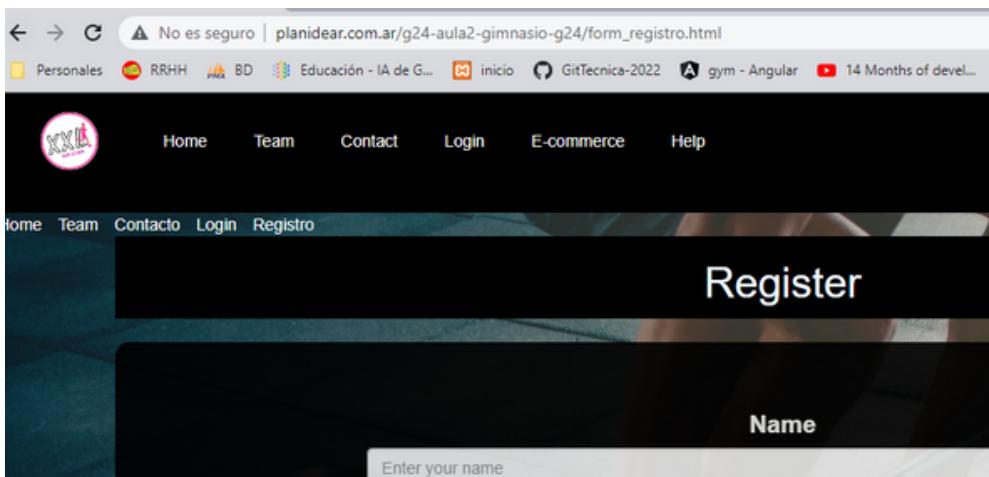
The Contact page, in the text We will never share your email with anyone else., is in Simple Future.



The text "Do you already have an account?" has a possessive pronoun



To register you must press the Register button



Grupo 24

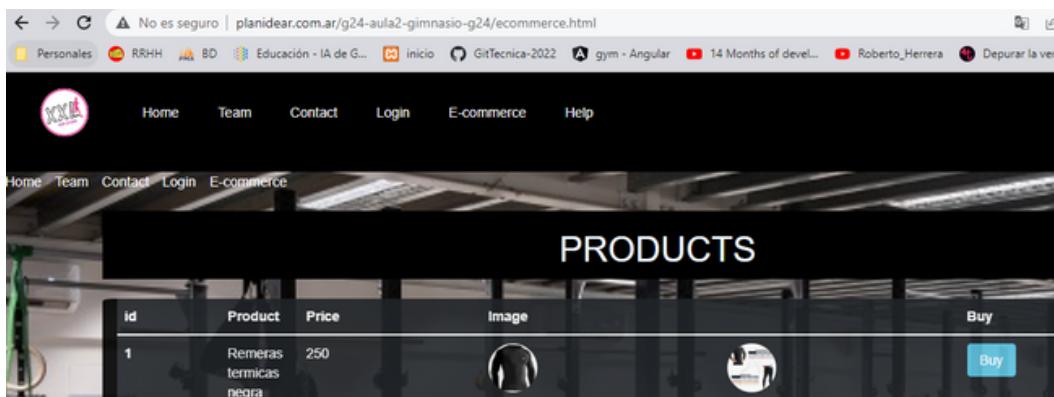


(55) 1234-5678

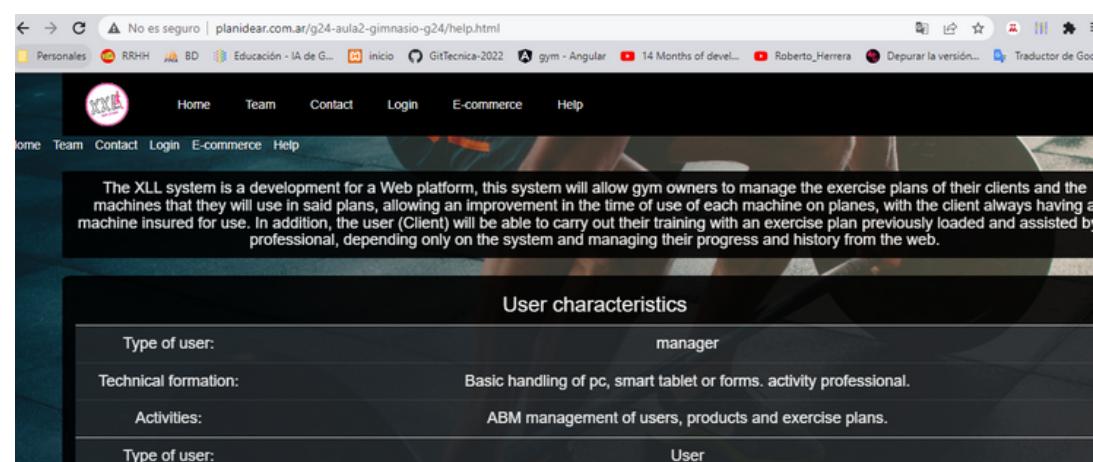


correo@gymsistem.com.ar

The e-commerce page is in English, although the products can be loaded according to the user's language



The Help page is for the person visiting the page, to understand its use and its limitations. It is in process.



Translation of pages	Work done
Alfredo Nuñez	Page team, contact, login, e-commerce.
Gustavo Godoy	Page Home, registrar, help. Update IEE830.



Referencia:

Titulo del Documento	Referencia
Standard IEEE 830 - 1998	IEEE
Fuentes de Google	https://fonts.google.com/
Estilo Bootstrap	https://getbootstrap.com/
Página de Ayuda	https://www.w3schools.com/
Guía Git	https://rogerdudler.github.io/git-guide/index.es.html
Canva	https://www.canva.com/

Links	Referencia
GitHub	https://github.com/PPROF2-2022ProgWeb/g24-aula2-gimnasio-g24
Pagina prueba	http://planidear.com.ar/g24-aula2-gimnasio-g24/index.html
Explicacion usuario	http://planidear.com.ar/g24-aula2-gimnasio-g24/help.html

Restricciones:

- Interfaz para ser usada con internet
- Lenguajes y tecnologías en uso: HTML, JavaScript, angular, typescript, php, ect.
- Base de datos Mysql
- Bootstrap
- Navegadores Web.
- Alojamiento web y base de datos.



Personal Involucrado:

Nombre	Responsabilidad - Rol	e-mail
Marcelo Peysa (Abandona 07-2022)	Desarrollo y actualizacion GitHub	mpeysa@gmail.com.ar
Gustavo Godoy	Desarrollo, base de datos y informe IEEE	grgodoy1984@gmail.com
Alfredo Nuñez	Diseño y Desarrollo.	alfredonunez0688@gmail.com
Gabriel Peysa	Scrum Master - Diseño y Desarrollo.	gabriel.peysa@gmail.com

Ficha del documento

Año	Revisión	Autores
2021	Version 1.0	Marcelo Peysa - Gabriel Rodenas - Cristian Martin Herrera - Gustavo Godoy - Alfredo Nuñez - Gabriel Peysa - Roger Ferreyra
2022	Version 2.0	Marcelo Peysa (Abandona 07-2022) - Gustavo Godoy - Alfredo Nuñez - Gabriel Peysa - Roger Ferreyra (Abandona 05-2022)

