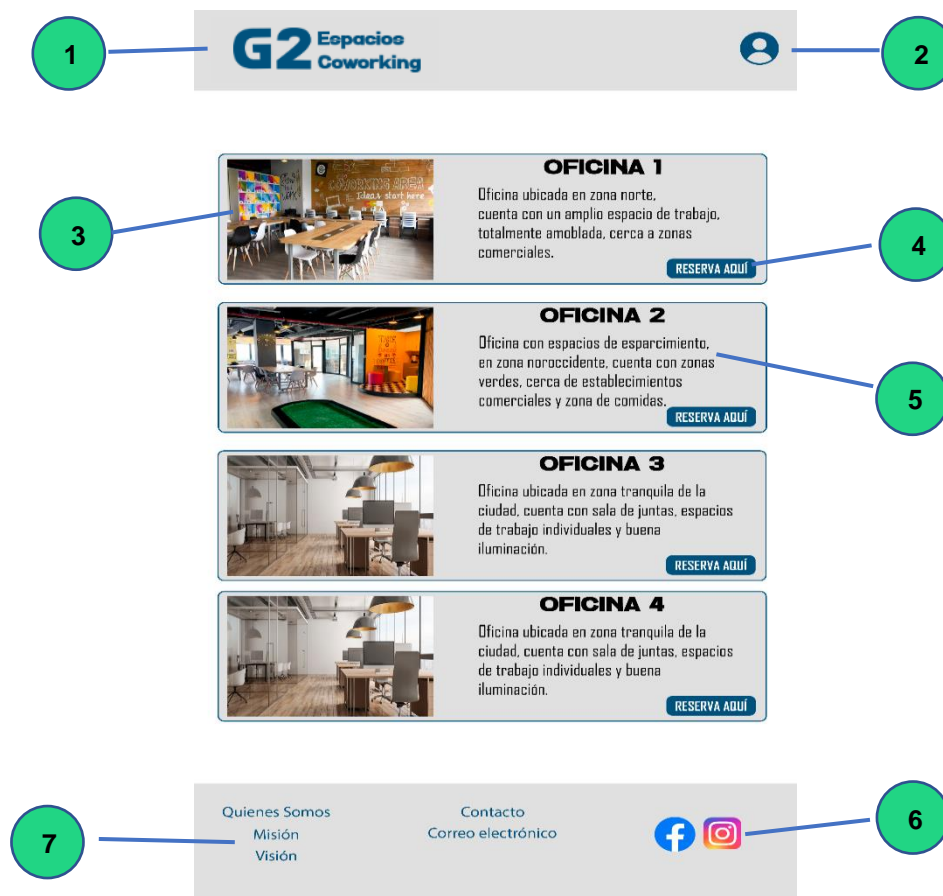


SPRINT 3

1. Descripción del Front: Página principal.




IDENTIFICADORES

1. Logo del proyecto.
2. Icono para dirigirse al inicio de sesión.
3. Imagen representativa del espacio coworking.
4. Botón para reservar un espacio coworking.
5. Descripción del espacio coworking.
6. Iconos redes sociales.
7. Información de relevante.



- Front registro de Usuario.

G2 Espacios Coworking


 1
 2
 3
 4
 5
 6
 7

Quiénes Somos
Misión
Visión

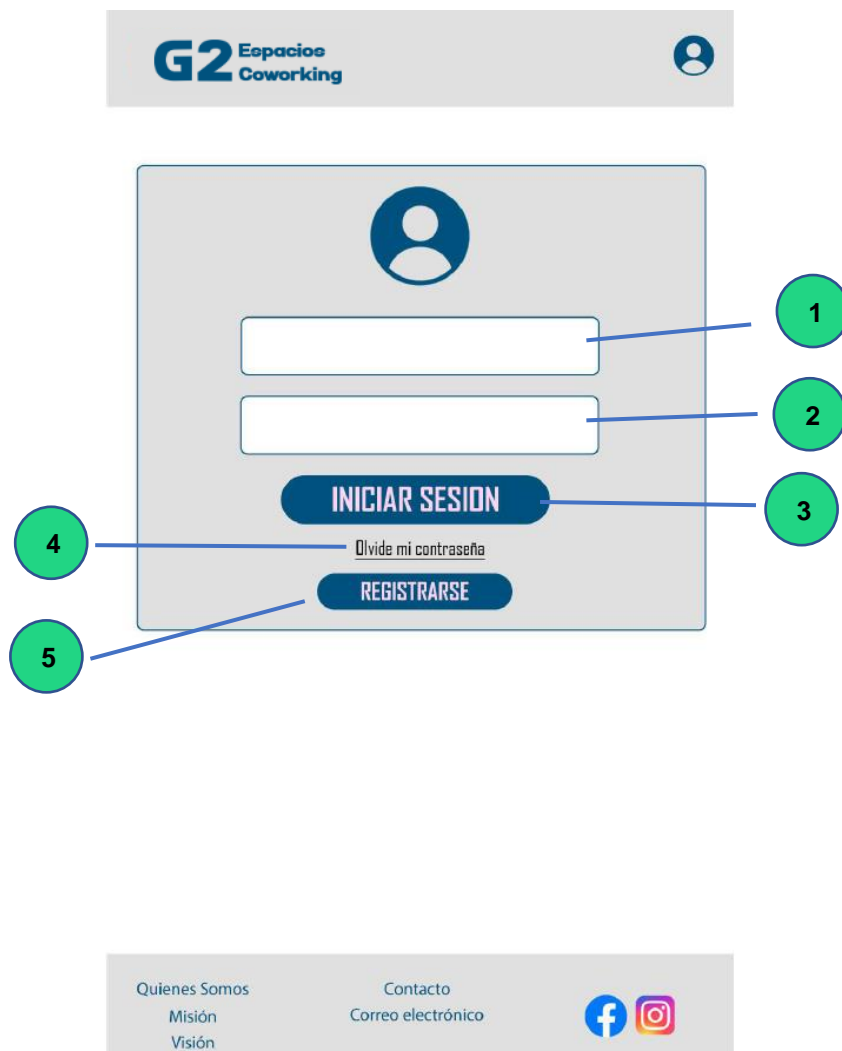
Contacto
Correo electrónico

IDENTIFICADORES

1. Nombre del usuario.
2. Correo electrónico del usuario.
3. Crear una contraseña.
4. Confirmar la contraseña creada.
5. Agregar un teléfono de contacto.
6. Elegir el perfil (Usuario o Administrador).
7. Botón para crear el registro.

- Front inicio de Sesión.



G2 Espacios Coworking

1. Campo para ingresar el nombre de usuario.

2. Campo para ingresar el correo electrónico de usuario.

3. Botón INICIAR SESION

4. Enlace [Olvide mi contraseña](#)

5. Botón REGISTRARSE

Quienes Somos
Misión
Visión

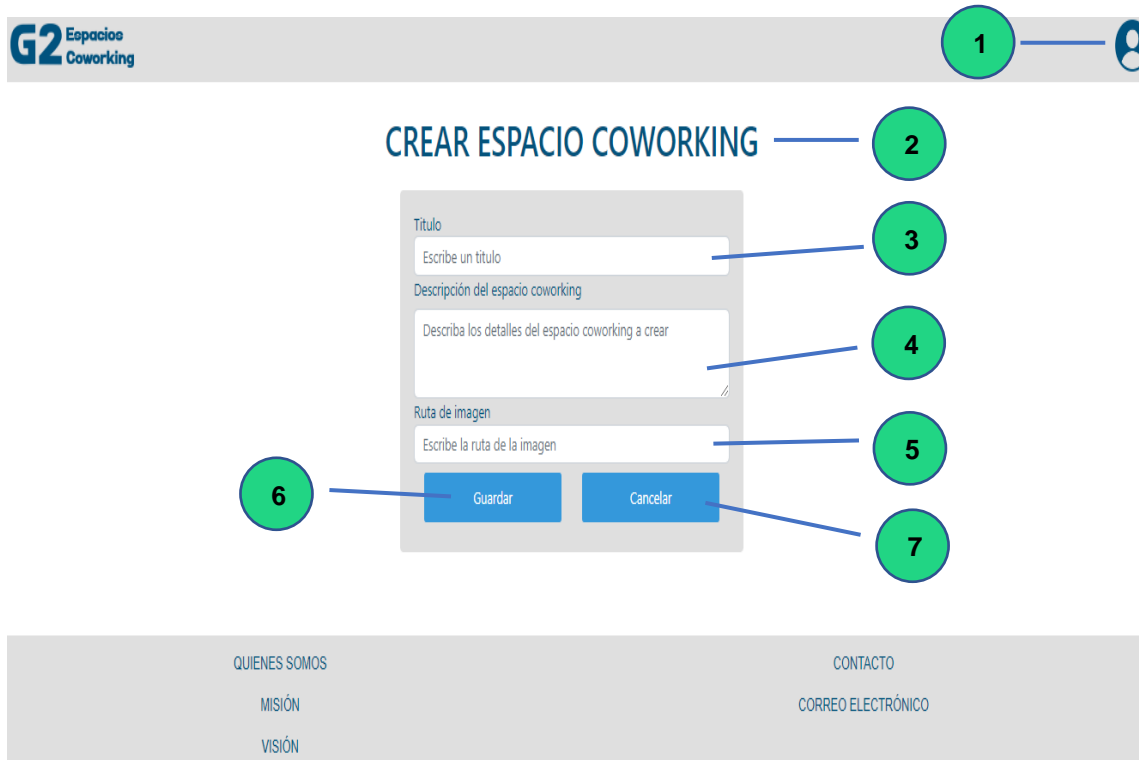
Contacto
Correo electrónico

f i

IDENTIFICADORES

1. Espacio para ingresar el nombre de usuario.
2. Espacio para ingresar el correo electrónico de usuario.
3. Botón para iniciar sesión.
4. Enlace para recordar contraseña.
5. Botón para registrar un usuario.

- Front creación de espacios coworking.



La imagen muestra la interfaz de usuario para crear un espacio coworking. El formulario está titulado 'CREAR ESPACIO COWORKING' y contiene los siguientes campos y botones:

- 1:** Perfil del usuario en la barra superior.
- 2:** Título del formulario.
- 3:** Campo de texto para el título ('Escribe un título').
- 4:** Campo de texto para la descripción ('Describe los detalles del espacio coworking a crear').
- 5:** Campo de texto para la ruta de la imagen ('Escribe la ruta de la imagen').
- 6:** Botón 'Guardar'.
- 7:** Botón 'Cancelar'.

En la parte inferior de la página, se encuentran los enlaces: QUIENES SOMOS, MISIÓN, VISIÓN, CONTACTO y CORREO ELECTRÓNICO.

IDENTIFICADORES

1. Perfil del usuario.
2. Título del front.
3. Campo para agregar el nombre a un espacio coworking.
4. Campo para diligenciar los detalles de un espacio a crear.
5. Campo para url-imagen del espacio coworking.
6. Botón para aceptar y crear un espacio coworking.
7. Botón para cancelar.

- Front administración de espacios coworking desde el perfil administrador.

G2 Espacios Coworking

ADMINISTRACIÓN DE ESPACIOS COWORKING

CREAR ESPACIO

CÓDIGO	TÍTULO	DESCRIPCIÓN	IMAGEN	ACCIONES
29	Espacio 1	Oficina para 10 personas, cuenta con sala de juntas.		

QUIENES SOMOS
MISIÓN
VISIÓN

CONTACTO
CORREO ELECTRÓNICO

IDENTIFICADORES

1. Título del front.
2. Identificador del espacio coworking.
3. Nombre del espacio coworking.
4. Descripción del espacio coworking.
5. Imagen del espacio coworking.
6. Botón para editar el espacio coworking.
7. Botón para eliminar el espacio coworking.
8. Botón para crear un espacio coworking.

- Front actualización y edición de espacios coworking.



EDITAR ESPACIO COWORKING

1. Título

2. Nombre del espacio coworking

3. Descripción del espacio coworking

4. Ruta de imagen

5. Botón para editar y actualizar el espacio

6. Botón para cancelar la edición del espacio

Formulario de edición de espacio coworking:

Título: Espacio 1

Descripción del espacio coworking: Oficina para 10 personas, cuenta con sala de juntas, con salón de entretenimiento.

Ruta de imagen: <https://amqueretaro.com/wp-content/uploads/2020/11/2-4.jpg>

Botones: Editar, Cancelar

QUIENES SOMOS

MISIÓN

VISIÓN

CONTACTO

CORREO ELECTRÓNICO

IDENTIFICADORES

1. Título del front.
2. Nombre del espacio coworking.
3. Descripción del espacio coworking.
4. Url de la imagen del espacio coworking.
5. Botón para editar y actualizar el espacio
6. Botón para cancelar la edición del espacio.

- Front formulario de reserva.

The image shows a web form for reserving a coworking space. At the top, there is a header with the 'G2 Espacios Coworking' logo and a user profile icon. Below this is the title 'FORMULARIO DE RESERVA'. The form itself contains several input fields: 'Nombre Completo', 'Correo electrónico', 'Teléfono de Contacto', 'Espacio a Reservar' (a dropdown menu), 'Cantidad de Personas' (a dropdown menu), 'Tiempo a Reservar' (a dropdown menu), and a date field showing '09/09/2022'. A blue 'RESERVAR' button is at the bottom of the form. At the very bottom of the page, there is a footer with links for 'Quienes Somos', 'Misión', 'Visión', 'Contacto', and 'Correo electrónico', along with Facebook and Instagram social media icons. Ten green circles with numbers 1 through 10 are placed around the form, with lines pointing to specific elements: 1 points to the title, 2 to the name field, 3 to the email field, 4 to the phone field, 5 to the space dropdown, 6 to the quantity dropdown, 7 to the time dropdown, 8 to the date field, 9 to the 'RESERVAR' button, and 10 to the footer area.

1 FORMULARIO DE RESERVA

2 Nombre Completo

3 Correo electrónico

4 Teléfono de Contacto

5 Espacio a Reservar

6 Cantidad de Personas

7 Tiempo a Reservar

8 09/09/2022

9 RESERVAR

10

Quienes Somos
Misión
Visión

Contacto
Correo electrónico

f i

IDENTIFICADORES

1. Título del front.
2. Nombre del usuario que reserva.
3. Correo electrónico del usuario que reserva.
4. Teléfono de contacto del usuario.
5. Nombre de espacio a reservar.
6. Cantidad de personas a usar el espacio.
7. Tiempo que será usado el espacio coworking.
8. Fecha en la cual quiere reservar el espacio coworking.
9. Botón para desplegar un menú.
10. Botón para crear la reserva.

- Front gestión de reservas desde el perfil de usuario.

Juan Camilo Salcedo

GESTIÓN DE RESERVAS

CÓDIGO	DESCRIPCIÓN DE LA RESERVA	FECHA RESERVA	FECHA CREACIÓN	ACCIONES
22455	Oficina 1 reservada para 10 personas de 08:00 - 05:00.	12/09/2022	01/09/2022	 
250240	Oficina 4 reservada para 5 personas de 08:00 - 12:00.	20/09/2022	02/09/2022	 
276854	Oficina 2 reservada para 7 personas de 10:00 - 5:00.	15/09/2022	31/08/2022	 
250456	Oficina 3 reservada para 8 personas de 08:00 - 3:00.	15/09/2022	31/08/2022	 

IDENTIFICADORES

1. Título del front.
2. Identificador del espacio coworking.
3. Descripción de la reserva del espacio coworking.
4. Fecha para la cual se crea la reserva.
5. Fecha en la cual se creó la reserva del espacio.
6. Botón para editar la reserva seleccionada.
7. Botón para eliminar la reserva seleccionada.

- Front modificación de reservas.

G2 Espacios Coworking Juan Camilo Salcedo

MODIFICACIÓN DE RESERVA

2 Juan Camilo Salcedo Rodríguez

3 kamilorod@gmail.com

4 3154224068

5 5

6 1 Día

7 12/09/2022

8 ACTUALIZAR

9 CANCELAR

IDENTIFICADORES

1. Título del front.
2. Nombre del usuario que reservó un espacio.
3. Correo electrónico del usuario.
4. Teléfono del usuario.
5. Cantidad de personas a usar el espacio.
6. Tiempo que será usado el espacio coworking.
7. Fecha en la cual quiere reservar el espacio coworking.
8. Botón para actualizar la reserva.
9. Botón para cancelar la modificación.

2. Pantallazos del proyecto (Visual Studio Code).

- CoworkingDao.java (folder acces Dao):

```
1 package com.example.demo.access.dao;
2
3 import org.springframework.data.jpa.repository.Query;
4 import org.springframework.data.repository.CrudRepository;
5
6 import com.example.demo.domain.entity.Coworking;
7
8 public interface ICoworkingDao extends CrudRepository<Coworking, Long>{
9
10     // @Query("select c from Coworking c where c.usuario.id = ?1")
11     // Coworking[] findById(Long idUser);
12
13     @Query("select c from Coworking c where c.id = ?1")
14     Coworking[] findById(Long idCoworking);
15 }
16
```

- IUserdao.java (folder acces Dao):

```
1 package com.example.demo.access.dao;
2
3 import org.springframework.data.repository.CrudRepository;
4 import org.springframework.data.jpa.repository.Query;
5
6 import com.example.demo.domain.entity.Usuario;
7
8
9 public interface IUserdao extends CrudRepository<Usuario, Long>{
10
11     @Query("select c from Usuario c where c.usuario = ?1 and c.clave = ?2")
12     Usuario findByUserPassword(String usuario, String clave);
13 }
14
```



- Coworking.java (folder entity):

```

1  package com.example.demo.domain.entity;
2
3  import javax.persistence.Column;
4  import javax.persistence.Entity;
5  import javax.persistence.GeneratedValue;
6  import javax.persistence.GenerationType;
7  import javax.persistence.Id;
8  import javax.persistence.Table;
9
10
11  @Entity
12  @Table(name = "Coworking")
13  public class Coworking {
14
15      @Id
16      @Column(unique = true, nullable = false)
17      @GeneratedValue(strategy = GenerationType.IDENTITY)
18      private Long id;
19      private String titulo;
20      private String descripcion;
21      private String media;
22
23      public Coworking() {
24      }
25
26      public Coworking(Long id, String titulo, String descripcion, String media) {
27          this.id = id;
28          this.titulo = titulo;
29          this.descripcion = descripcion;
30          this.media = media;
31      }
32
33      public Long getId() {
34          return id;
35      }
36
37      public String getTitulo() {
38          return titulo;
39      }
40
41      public String getDescripcion() {
42          return descripcion;
43      }
44

```

```

45      public String getMedia() {
46          return media;
47      }
48
49      public void setId(Long id) {
50          this.id = id;
51      }
52
53      public void setTitulo(String titulo) {
54          this.titulo = titulo;
55      }
56
57      public void setDescripcion(String descripcion) {
58          this.descripcion = descripcion;
59      }
60
61      public void setMedia(String media) {
62          this.media = media;
63      }
64

```



- Usuario.java (folder entity):

```
1 package com.example.demo.domain.entity;
2
3 import javax.persistence.Entity;
4 import javax.persistence.GeneratedValue;
5 import javax.persistence.GenerationType;
6 import javax.persistence.Id;
7 import javax.persistence.Table;
8 import javax.persistence.Column;
9
10 @Entity
11 @Table(name = "Usuario")
12 public class Usuario {
13
14     @Id
15     @Column(unique = true, nullable = false)
16     @GeneratedValue(strategy = GenerationType.IDENTITY)
17     private Long id;
18     private String usuario;
19     private String clave;
20
21     public Usuario() {
22     }
23
24     public Usuario(Long id, String usuario, String clave) {
25         this.id = id;
26         this.usuario = usuario;
27         this.clave = clave;
28     }
29
30     public Long getId() {
31         return id;
32     }
33
34     public String getUsuario() {
35         return usuario;
36     }
37
38     public String getClave() {
39         return clave;
40     }
41
42     public void setId(Long id) {
43         this.id = id;
44     }
45
46     public void setUsuario(String usuario) {
47         this.usuario = usuario;
48     }
49
50     public void setClave(String clave) {
51         this.clave = clave;
52     }
53
54 }
```

- CoworkingService.java (folder service):

```
1 package com.example.demo.domain.service;
2
3 import org.springframework.beans.factory.annotation.Autowired;
4 import org.springframework.stereotype.Service;
5
6 import com.example.demo.access.dao.ICoworkingDao;
7 import com.example.demo.domain.entity.Coworking;
8
9 @Service
10 public class CoworkingService implements ICoworkingService {
11
12     @Autowired
13     private ICoworkingDao coworkingDao;
14
15     @Override
16     public Coworking[] getCoworkingByUser(Long id) {
17         return coworkingDao.findByUserId(id);
18     }
19
20     // @Override
21     // public Coworking getCoworkingById(Long id) {
22     //     return coworkingDao.getCoworkingById(id);
23     // }
24
25     @Override
26     public Iterable<Coworking> getAllCoworking() {
27         return coworkingDao.findAll();
28     }
29
30     @Override
31     public Coworking getById(Long id) {
32         Coworking coworking = coworkingDao.findById(id).orElse(null);
33         if (coworking == null) {
34             return new Coworking(id, "", titulo: "", descripcion: "", media: "");
35         }
36         return coworking;
37     }
38
39     //crear
40     @Override
41     public Coworking saveCoworkingById(Coworking coworking) {
42         return coworkingDao.save(coworking);
43     }
44
45     @Override
46     public Coworking saveCoworking(Coworking coworking) {
47         return coworkingDao.save(coworking);
48     }
49
50     //eliminar
51     @Override
52     public void deleteCoworkingById(Long id) {
53         coworkingDao.deleteById(id);
54     }
55
56     //actualizar
57     @Override
58     public Coworking updateCoworking(Coworking coworking) {
59         return coworkingDao.save(coworking);
60     }
61 }
```

- ICoworkingService (folder service)

```
1 package com.example.demo.domain.service;
2
3 //import java.util.List;
4
5 import com.example.demo.domain.entity.Coworking;
6
7 public interface ICoworkingService {
8
9     public Coworking[] getCoworkingByUser(Long id);
10    //public List<Coworking> getCoworkingById(Long id);
11    public Iterable<Coworking>getAllCoworking();
12    public Coworking getById(Long id);
13    public Coworking saveCoworkingById(Coworking coworking);
14    public Coworking saveCoworking(Coworking coworking);
15    public void deleteCoworkingById(Long id);
16    public Coworking updateCoworking(Coworking coworking);
17
18 }
19
```

- IUserarioService (folder service)

```
1 package com.example.demo.domain.service;
2
3 import com.example.demo.domain.entity.Usuario;
4
5 public interface IUserarioService {
6     public Usuario findByUserPassword(String user, String password);
7
8     public Usuario findById(Long id);
9
10 }
11
```



- UsuarioImplService (folder service)

```
1 package com.example.demo.domain.service;
2
3 import org.springframework.beans.factory.annotation.Autowired;
4 import org.springframework.stereotype.Service;
5
6
7 import com.example.demo.access.dao.IUsuarioDao;
8 import com.example.demo.domain.entity.Usuario;
9
10 @Service
11 public class UsuarioImplService implements IUsuarioService{
12
13     @Autowired
14     private IUsuarioDao usuarioDao;
15
16     @Override
17     public Usuario findByUserPassword(String usuario, String clave) {
18         Usuario varUser = usuarioDao.findByUserPassword(usuario, clave);
19         if (varUser == null) {
20             return new Usuario(id: 0L, usuario: "", clave: "");
21         }
22         return varUser;
23     }
24
25     @Override
26     public Usuario findById(Long id) {
27         Usuario user = usuarioDao.findById(id).orElse(other: null);
28         if (user == null) {
29             return new Usuario(id: 0L, usuario: "", clave: "");
30         }
31         return user;
32     }
33
34 }
```

- UsuarioController (folder controller)

```
package com.example.demo.presentation;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import com.example.demo.domain.entity.Coworking;
import com.example.demo.domain.service.ICoworkingService;

@Controller
public class UsuarioController {

    // @Autowired
    // private IUserService usuarioService;
    @Autowired
    private ICoworkingService coworkingService;

    Long idLogueado;

    // @GetMapping({ "/", "HOME", "index" })
    // public String home(Model model) {
    //     model.addAttribute("AllCoworking", coworkingService.getAllCoworking());
    //     return "home";
    // }

    // listar
    @GetMapping("/espaciosCoworking")
    public String espaciosCoworking(Model model) {
        model.addAttribute("AllCoworking", coworkingService.getAllCoworking());
        return "espaciosCoworking";
    }

    // Front Página principal
    @GetMapping("/PáginaPrincipal")
    public String PáginaPrincipal(Model model) {
        Coworking coworking = new Coworking();
        model.addAttribute("coworking", coworking);
        return "PáginaPrincipal";
    }

    // Front Inicio de Sesión
    @GetMapping("/InicioSesion")
    public String InicioSesion(Model model) {
        Coworking coworking = new Coworking();
        model.addAttribute("coworking", coworking);
        return "InicioSesion";
    }

    // Front Recuperar Contraseña
    @GetMapping("/RecuperarContraseña")
    public String RecuperarContraseña(Model model) {
        Coworking coworking = new Coworking();
        model.addAttribute("coworking", coworking);
        return "RecuperarContraseña";
    }
}
```

```
// Front Registro Usuario
@GetMapping("/RegistroUsuario")
public String RegistroUsuario(Model model) {
    Coworking coworking = new Coworking();
    model.addAttribute("coworking", coworking);
    return "RegistroUsuario";
}

// Front Formulario de reserva
@GetMapping("/FormularioReserva")
public String FormularioReserva(Model model) {
    Coworking coworking = new Coworking();
    model.addAttribute("coworking", coworking);
    return "FormularioReserva";
}

// crear
@GetMapping("/crearCoworking")
public String crearCoworking(Model model) {
    Coworking coworking = new Coworking();
    model.addAttribute("coworking", coworking);
    return "crearCoworking";
}

@PostMapping("/guardar")
public String saveCoworking(@ModelAttribute("coworking") Coworking coworking) {
    coworkingService.saveCoworking(coworking);

    return "redirect:/espaciosCoworking";
}

// editar
@GetMapping("/editarCoworking/{id}")
public String actualizarCoworking(@PathVariable Long id, Model model) {
    Coworking coworking = coworkingService.getById(id);
    if (coworking.getId() != 0) {
        model.addAttribute("coworking", coworking);
        return "editarCoworking";
    } else {
        return "editarCoworking";
    }
}

@PostMapping("/editar/{id}")
public String updateCoworking(@PathVariable Long id,
    @ModelAttribute("coworking") Coworking coworking,
    Model model) {
    Coworking coworkingExistente = coworkingService.getById(id);

    coworkingExistente.setId(coworking.getId());
    coworkingExistente.setTitulo(coworking.getTitulo());
    coworkingExistente.setDescripcion(coworking.getDescripcion());

    coworkingService.saveCoworking(coworking);
    return "redirect:/espaciosCoworking";
}
```

```
// eliminar
@GetMapping("/eliminar/{id}")
public String eliminar(@PathVariable Long id, Model model) {
    coworkingService.deleteCoworkingById(id);

    return "redirect:/espaciosCoworking";
}

@PostMapping("/eliminar/{id}")
public String deleteCoworking(@PathVariable Long id, Model model) {
    coworkingService.deleteCoworkingById(id);

    return "redirect:/espaciosCoworking";
}
```

- UsuarioRepository (folder repositories)

```
1 package com.example.demo.repositories;
2
3 public class UsuarioRepository {
4
5 }
6
```

- DemoApplication

```
1 package com.example.demo;
2
3 import org.springframework.boot.SpringApplication;
4 import org.springframework.boot.autoconfigure.SpringBootApplication;
5
6 @SpringBootApplication
7 public class DemoApplication {
8
9     Run | Debug
10     public static void main(String[] args) {
11         SpringApplication.run(DemoApplication.class, args);
12     }
13 }
14
```



- ServletInitializer

```
1 package com.example.demo;
2
3 import org.springframework.boot.builder.SpringApplicationBuilder;
4 import org.springframework.boot.web.servlet.support.SpringBootServletInitializer;
5
6 public class ServletInitializer extends SpringBootServletInitializer {
7
8     @Override
9     protected SpringApplicationBuilder configure(SpringApplicationBuilder application) {
10         return application.sources(DemoApplication.class);
11     }
12
13 }
14
```

- Estilos (folder resources – static - css)

```
1 /*Estilos página principal*/
2
3 header {
4     margin-bottom: 2rem;
5 }
6
7 .Logo {
8     margin-left: 1.5rem;
9 }
10
11 .titulos {
12     text-align: center;
13     margin-bottom: 1.5rem;
14     color: #04537c;
15 }
16
17 .cards {
18     border-color: #04537c;
19     height: 230px;
20     width: 1100px;
21     margin: auto;
22 }
23
24 .cards:hover {
25     box-shadow: 0px 0px 11px 0px #04537c;
26     cursor: auto;
27 }
28
29 .cards img {
30     height: 210px;
31     width: 340px;
32     margin-top: 0.5rem;
33 }
34
35 /*Códigos para footer página principal*/
36 > .footer { ...
42 }
```

- HTML Página principal (folder resource - templates)

```

1  <!DOCTYPE html>
2  <html xmlns:th="http://www.thymeleaf.org">
3      <head>
4          <meta charset="UTF-8" />
5          <meta http-equiv="X-UA-Compatible" content="IE=edge" />
6          <meta
7              name="viewport"
8              content="width=device-width, initial-scale=1.0"
9          />
10         <title>G2 Reserva Coworking</title>
11         <link
12             href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0/dist/css/bootstrap.min.css"
13             rel="stylesheet"
14             integrity="sha384-gH2yIJqKdNHPEq0n4Mqqa/HGKIhSkIHeL5AyhkYV8i59U5AR6csBvApHHNl/vI1Bx"
15             crossorigin="anonymous"
16         />
17         <link th:href="@{css/G2styles.css}" rel="stylesheet" />
18     </head>
19     <body>
20         <!--Etiquetas para navbar-->
21         <header>
22             <nav class="navbar" style="background-color: #dfdfdf">
23                 <div class="container-fluid">
24                     <a class="navbar-brand" href="#">
25                         
30                     </a>
31                     <a class="nav-link" th:href="@{/InicioSesion}">
32                         </a>
33                 </div>

```

- Application.properties

```
1 spring.application.name=DEMO
2 server.port=8001
3
4 #Data source
5 #Indica el driver/lib para conectar java a mysql
6 spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
7
8 #Url donde esta el servicio de tu mysql y el nombre de la base de datos
9 spring.datasource.url=jdbc:mysql://localhost:3306/coworking
10
11 #Usuario y contraseña para tu base de datos descrita en la linea anterior
12 spring.datasource.username=root
13 spring.datasource.password=
14
15 #[opcional]Imprime en tu consola las instrucciones hechas en tu base de datos.
16 spring.jpa.show-sql = true
17
18 spring.jpa.hibernate.ddl-auto=update
```

- Base de datos phpMyAdmin

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
<input type="checkbox"/> 1	id	bigint(20)			No	Ninguna		AUTO_INCREMENT	Cambiar Eliminar Más
<input type="checkbox"/> 2	descripcion	varchar(255)	utf8mb4_general_ci		Sí	NULL			Cambiar Eliminar Más
<input type="checkbox"/> 3	media	varchar(255)	utf8mb4_general_ci		Sí	NULL			Cambiar Eliminar Más
<input type="checkbox"/> 4	titulo	varchar(255)	utf8mb4_general_ci		Sí	NULL			Cambiar Eliminar Más

- Pantallazo proyecto inicializado con springboot

```

import.sql - demo - Visual Studio Code

EXPLORADOR
  EDITORES ABIERTOS
    import.sql src/main/resources
  DEMO
    .mvn
    .vscode
    src
      main
        java\com\example\demo
          access\dao
            ICoworkingDao.java
            IUsuarioDao.java
          domain
            entity
              Coworking.java
              Usuario.java
          service
            CoworkingService.java
            ICoworkingService.java
            IUsuarioService.java
            UsuarioImplService.java
          presentation
            UsuarioController.java
          repositories
            UsuarioRepository.java
            DemoApplication.java
            ServletInitializer.java
          resources
            static
            css
      ESQUEMA
      LÍNEA DE TIEMPO
      JAVA PROJECTS
      MAVEN

PROBLEMAS
SALIDA
TERMINAL
JUPYTER
COMENTARIOS
CONSOLA DE DEPURACIÓN

:: Spring Boot :: (v2.7.3)

2022-09-28 12:25:42.815 INFO 17840 --- [ restartedMain] com.example.demo.DemoApplication : Starting DemoApplication using Java 11.0.15 on JCamilosR
with PID 17840 (D:\demo\demo\target\classes started by USUARIO in D:\demo\demo)
2022-09-28 12:25:42.817 INFO 17840 --- [ restartedMain] com.example.demo.DemoApplication : No active profile set, falling back to 1 default profile:
"default"
2022-09-28 12:25:42.894 INFO 17840 --- [ restartedMain] .e.DevToolsPropertyDefaultsPostProcessor : Devtools property defaults active! Set 'spring.devtools.a
dd-properties' to 'false' to disable
2022-09-28 12:25:42.895 INFO 17840 --- [ restartedMain] .e.DevToolsPropertyDefaultsPostProcessor : For additional web related logging consider setting the '
logging.level.web' property to 'DEBUG'
2022-09-28 12:25:43.491 INFO 17840 --- [ restartedMain] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in DEFAULT mod
e.
2022-09-28 12:25:43.552 INFO 17840 --- [ restartedMain] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 53 ms. Found
2 JPA repository interfaces.
2022-09-28 12:25:44.248 INFO 17840 --- [ restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port(s): 8001 (http)
2022-09-28 12:25:44.265 INFO 17840 --- [ restartedMain] o.apache.catalina.core.StandardService : Starting service [Tomcat]
2022-09-28 12:25:44.266 INFO 17840 --- [ restartedMain] org.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat/9.0.65]
2022-09-28 12:25:44.383 INFO 17840 --- [ restartedMain] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationContext
2022-09-28 12:25:44.383 INFO 17840 --- [ restartedMain] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 1
485 ms
2022-09-28 12:25:44.416 INFO 17840 --- [ restartedMain] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Starting...
2022-09-28 12:25:44.570 INFO 17840 --- [ restartedMain] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Start completed.
2022-09-28 12:25:44.579 INFO 17840 --- [ restartedMain] o.s.b.a.h2.H2ConsoleAutoConfiguration : H2 console available at '/h2-console'. Database available
at 'jdbc:mysql://localhost:3306/coworking'
2022-09-28 12:25:44.781 INFO 17840 --- [ restartedMain] o.hibernate.jpa.internal.util.LogHelper : HHH000204: Processing PersistenceUnitInfo [name: default]
2022-09-28 12:25:44.853 INFO 17840 --- [ restartedMain] org.hibernate.Version : HHH000412: Hibernate ORM core version 5.6.10.Final
2022-09-28 12:25:45.003 INFO 17840 --- [ restartedMain] o.hibernate.annotations.common.Version : HCANN000001: Hibernate Commons Annotations (5.1.2.Final)
2022-09-28 12:25:45.105 INFO 17840 --- [ restartedMain] org.hibernate.dialect.Dialect : HHH000400: Using dialect: org.hibernate.dialect.MySQL55Di
alect
2022-09-28 12:25:45.608 INFO 17840 --- [ restartedMain] o.h.e.t.j.p.i.JtaPlatformInitiator : HHH000490: Using JtaPlatform implementation: [org.hiberna
te.engine.transaction.jta.platform.internal.NoJtaPlatform]
2022-09-28 12:25:45.616 INFO 17840 --- [ restartedMain] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit
"default"
2022-09-28 12:25:45.966 WARN 17840 --- [ restartedMain] JpaBaseConfiguration$JpaWebConfiguration : spring.jpa.open-in-view is enabled by default. Therefore,
database queries may be performed during
view rendering. Explicitly configure spring.jpa.open-in-view to disable this warning
2022-09-28 12:25:46.244 INFO 17840 --- [ restartedMain] o.s.b.d.a.OptionalLiveReloadServer : LiveReload server is running on port 35729
2022-09-28 12:25:46.281 INFO 17840 --- [ restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 8001 (http) with context path
"/"
2022-09-28 12:25:46.295 INFO 17840 --- [ restartedMain] com.example.demo.DemoApplication : Started DemoApplication in 3.859 seconds (JVM running for
4.341s)
  
```

3. Informe de retrospectiva.

En esta tercera etapa del desarrollo del proyecto (**SPRINT 3**) se realizaron avances de acuerdo con la respectiva elaboración del front y del back del proyecto como tal, el cual estuvo sujeto a aprobación y debidas modificaciones por parte del equipo de trabajo y contando con la respectiva supervisión de los docentes encargados. A continuación, se encuentran estipulados una serie de puntos, los cuales serán tratados en el desarrollo del SPRINT 4 para cumplir a cabalidad con la realización del proyecto en curso.

- Realizar las debidas pruebas para el correcto desarrollo del proyecto.
- Actualizaciones periódicas de los repositorios (GitHub - Trello).
- Revisiones continuas de los correspondientes repositorios del proyecto (GitHub - Trello).
- Revisiones de funcionalidades de todo el proyecto para aprobar su viabilidad.

En cuanto a best practice desarrollaremos una serie de puntos a tratar, los cuales son los siguientes:

- Implementación de nuevas prácticas para el correcto desarrollo del proyecto.
- Seguir con el aprendizaje continuo por parte de cada uno de los integrantes del equipo de trabajo.
- Continuar con el debido cronograma de actividades a realizar.
- Continuar con el debido control y actualización de los repositorios del proyecto.

INTEGRANTES DEL EQUIPO DE TRABAJO

- **Leiner Bracho Ortega:** Gestor Bases de Datos.
- **Miguel Ángel Duncan:** Desarrollador Backend.
- **Diego Armando Moreno:** Tester.
- **Eduar Yofree Muñoz:** Gestor del Proyecto.
- **Juan Camilo Salcedo Rodríguez:** Desarrollador Frontend.