

PROYECTO DE PORTAFOLIO

COMUNIDAD VROL

Autor:
Cristian Carrillo

Introducción

Como proyecto de portafolio aborde parte de las necesidades de la Comunidad VRol de la cual participo, así que hagamos una leve introducción para repasar los puntos que dan razón a este proyecto:

¿Qué es la Comunidad VRol?

La comunidad de VRol, es como indica una Comunidad de distintas personas, siendo la mayoría de la Quinta Region (ende la V para indicar Quinta) y de otros lugares de Chile, siendo su nombre oficial Quinta Rol. Esta parte como la iniciativa de ser un punto de encuentro para los amantes de los [Juegos de Rol](#). Esto ya que en la Quinta región no se encontraban tan organizados como los que juegan en la región Metropolitana, y que sobre todo en tiempos de COVID donde las juntas presenciales no se pudiesen realizar con normalidad (o realizarse siquiera) hiciera que varios quedaran aislados. El grupo en si parte en Facebook por miembros integrantes de un Grupo mas grande en Chile que es Roleros.cl, también organizando un grupo de chat de WhatsApp para tratar de reunir a todos los roleros de la región para así poder organizar partidas online, y su ocasional presencial, y es aquí en este punto donde comienza a surgir esta necesidad.

Textos de chat perdidos

Una de las primeras cosas que se empezaron a notar, es que pasaba a menudo que la información pertinente a la organización de las partidas se perdía en el gran flujo de mensajes del chat (estamos hablando que hay mínimo un flujo de 200 mensajes al día) por lo cual siempre se tenia que estar preguntando cada tanto a los administradores por esta información pertinente. Esto creo la primera herramienta de control que fue crear una Calendario Google que tiene una información básica de las partidas que se están llevando a cabo y parte de la planificación de eventos de la comunidad.

Información relevante poco accesible.

Depender del calendario ayudo, pero no eliminaba todo, sin mencionar que el calendario era visible solo para administradores, quienes procedían a sacar una leve captura cada vez que se preguntaba (hasta el día de hoy aun se hace), lo que llevo a que se usara el medio usual para realizar las partidas on-line para también dejar una agendacion de los eventos, Discord. Pero esta plataforma aun funciona como chat, por lo cual si bien las secciones dejadas para el posteo de eventos estaban bloqueadas para posteo, esto aun no deja una claridad inmediata sobre algunos de los datos de la partida, aunque ha sido una mejora sin duda.

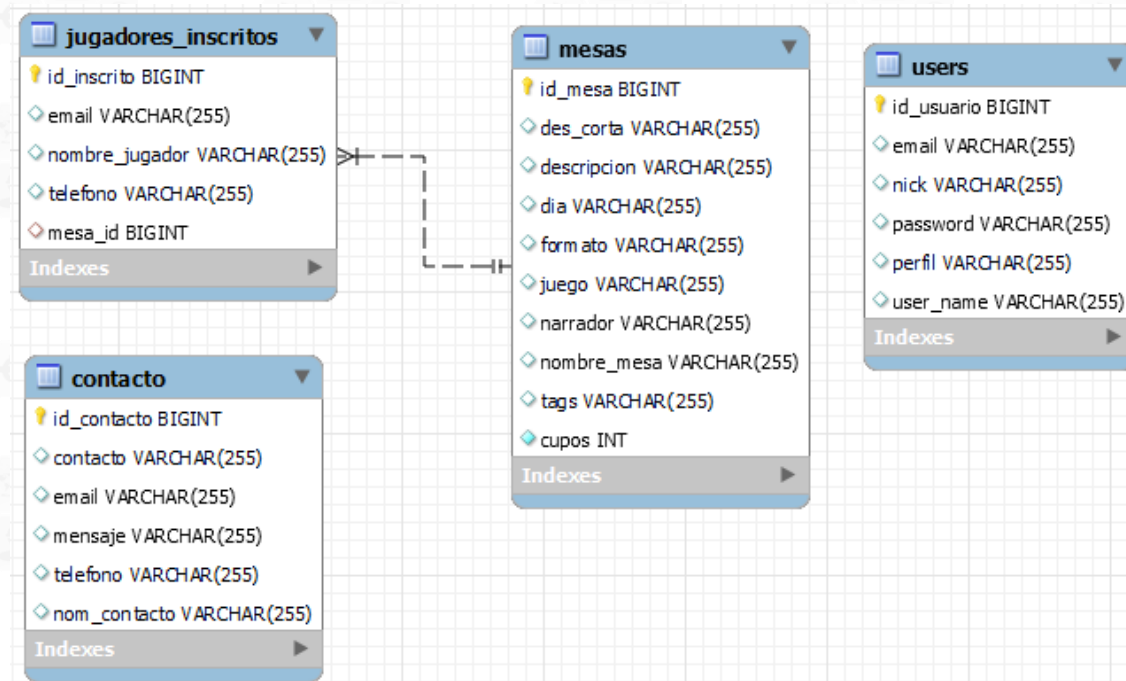
Entonces, como se puede apreciar, en favor de crear algo que sirva para unificar las fuentes de información de las mesas y eventos es que decidí tomar esto como caso de desarrollo para mi Portafolio.

DESARROLLO

Vamos entonces a abordar el desafío de este proyecto, el cual ha sufrido de bastantes transformaciones desde su primer génesis como simplemente un portal web el cual aun es visible a través del sitio web que sigue publicado [Vrol](#) a través de [Netlify](#) (también pueden ver una que realice en base a modelo pero en Angular [VrolSite](#)).

Análisis del modelo

Después de decidir utilizar un modelo en base a [JPA](#), lo primero fue derivar una estructura básica de lo que seria el modelo de la base de datos la cual para las necesidades mas inmediatas cuenta solo con las siguientes 4 Tablas: la tabla de Mesas, Jugadores Inscritos, Contacto y Usuarios. Por lo tanto el primer paso fue crear estas Clases, y colocar las anotaciones necesarias para definirlas como Entidades, dejo una imagen del modelo final (a la fecha).



Al ocupar JPA, nuestro principal objetivo es simplemente crear el Schema y dejarlo definido en el archivo de las propiedades de la base de datos para poder así indicar al sistema que estamos trabajando con él y de ahí empezar a trabajar en las clases.

Dependencias

Un punto importante antes de siquiera hablar de crear Entidades, vamos a por las dependencias usadas para este proyecto Springboot, de las cuales aquí dejo marcada la primera más importante:

```
<dependencies>
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-data-jpa</artifactId>
  </dependency>
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-thymeleaf</artifactId>
  </dependency>
  <dependency>
    <groupId>nz.net.ultraq.thymeleaf</groupId>
    <artifactId>thymeleaf-layout-dialect</artifactId>
    <version>3.2.1</version>
  </dependency>

  <dependency>
    <groupId>org.hibernate.validator</groupId>
    <artifactId>hibernate-validator</artifactId>
    <version>7.0.1.Final</version>
  </dependency>

  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web</artifactId>
  </dependency>
</dependencies>
```

```
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-devtools</artifactId>
  <scope>runtime</scope>
  <optional>true</optional>
</dependency>
<dependency>
  <groupId>mysql</groupId>
  <artifactId>mysql-connector-java</artifactId>
  <version>8.0.32</version>
</dependency>

<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-test</artifactId>
  <scope>test</scope>
</dependency>

<dependency>
  <groupId>org.projectlombok</groupId>
  <artifactId>lombok</artifactId>
  <version>1.18.26</version>
  <scope>provided</scope>
</dependency>
```

```
<!-- JUNIT -->
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-test</artifactId>
  <scope>test</scope>
</dependency>

<dependency>
  <groupId>org.junit.jupiter</groupId>
  <artifactId>junit-jupiter-api</artifactId>
  <version>5.7.2</version>
  <scope>test</scope>
</dependency>

<dependency>
  <groupId>org.junit.jupiter</groupId>
  <artifactId>junit-jupiter-engine</artifactId>
  <version>5.7.2</version>
  <scope>test</scope>
</dependency>
```

Por supuesto la Dependencia de JPA, también voy a destacar otras que son de relevancia en post de este proyecto, que son las Dependencias de [Thymeleaf](#) pues marcan una diferencia marcada con respecto a la de mis compañeros de curso: Thymeleaf no soporta JSP, solo trabaja con HTML y otros, ¿ahora porque Thymeleaf? La verdad lo vi en mas de un tutorial y me pareció interesante y que trabajaba de una forma más interesante, aunque similar al uso de las JSTL (tiene comandos que incluso operan en integración con Spring Security) por lo que decidí darle una prueba y así permitirme aprender a ocupar algo nuevo. De ahí las otras dependencias son algo mas clásicas, como las necesarias de Spring para trabajar, el Conector J para trabajar con MySQLL, Junit y otras.

Entidades

El concepto de Entidad es clave para su interacción con JPA a diferencia del modelo por conexión JDBC, ya que en este se necesita insertar la anotación exclusiva que indica que esta Clase se indica como entidad y de ahí poder crear su espejo en la base de datos.

```
1 package cl.vrol.models.entity;
2
3 import java.io.Serializable;
4
5
6 @Entity
7 @Table(name = "jugadores_inscritos")
8 public class Jugador implements Serializable {
9
10    /**
11     *
12     */
13    private static final long serialVersionUID = 1L;
14
15    @Id
16    @GeneratedValue(strategy=GenerationType.IDENTITY)
17    private Long idInscrito;
18    @NotEmpty
19    private String nombreJugador;
20    @NotEmpty
21    @Email
22    private String email;
23    @NotEmpty
24    @Pattern(regexp="[0-9]{9}")
25    private String telefono;
26
27    @ManyToOne
28    @JoinColumn(name="mesa_id")
29    private Mesa mesa;
30}
```

@Entity: Anotación Principal para indicar que esta es una Clase tipo Entidad.

@Table: Anotación que Indica que esta clase es una Tabla, y que nombre llevara.

@Id: Anotación que indica que este atributo es del tipo Identidad.

@GeneratedValue: generalmente usada junto con Id, para indicar que se autogenera.

@NotEmpty: esto indica que este valor no puede estar nulo.

@Email: anotación que da un parámetro general para indicar que el valor indicado aquí debe de llevar mínimo el @.

@Pattern: sirve para determinar que el valor ingresado debe cumplir con un patrón, en este caso, que solo serán números y son 9 dígitos solamente.

@ManyToOne y @JoinColumn: estas anotaciones son las que hacen posible determinar que esta tabla esta enlazada a otra (en este caso Mesa), y que su relación es de una a muchas.

Aquí dejo un ejemplo con el caso de la Clase Jugador, la cual crea entonces su Tabla jugadores_inscritos, e inserto los atributos que tendrá esta clase los cuales son idInscrito, nombreJugador, email, teléfono y Mesa. Esta clase tiene como objetivo ser el medio para que cualquier persona que desea se pueda inscribir a una Mesa de Juego, por ende, su relación de Uno es a Muchos, pues en este caso 1 Mesa puede tener X jugadores.

Sobre los Usuarios

Como clarificación, en este sistema, hay tres tipos de usuarios: usuario general o visita, Narrador, y Administrador. El primero es el usuario genérico que visite la página, no necesita permiso especial para visitar casi todos los menús y vistas. El segundo, Narrador, puede ver los jugadores inscritos en las mesas y crear mesas. Mientras que solo el Administrador puede crear otro usuario Administrador o Narrador, crear mesas, listar usuarios y en general total acceso a la aplicación.

Análisis de la Rubrica

A continuación, me basare en la rubrica para cubrir aspectos del proyecto y demostrar su aplicación a lo largo del proyecto:

Consulta a Base de Datos

1.- Selecciona las columnas requeridas para presentar la información solicitada

Como mencionado anteriormente, al estar usando JPA, el uso y selección de muchas de las posibles "queries" esta ya hecha en si, JPA a través de JpaRepository, ya nos entrega las Queries mas básicas que componen el Select que son las funciones que parten de un find, Insert Into y Update, que se absorben en el save, y Delete por delete.

```
List<Jugador> listaInscritos = playerService.listarTodos();|
model.addAttribute("titulo", "Lista de Inscritos");
model.addAttribute("inscritos", listaInscritos);
return "/views/jugadores/listarinscritos";
```

Aquí por ejemplo se está llamando a una función que pide llamar a todos los jugadores inscritos, y pasarlos como objeto al modelo de vista para obtener lo siguiente

Lista de Inscritos									
Default	Ord Asc	Ord Des	By Mesa 1	By Mesa 2	Group 1	Group 2			
Id Jugador:	Jugador:	E-mail:	Telefono:	En la Mesa:	Juego:	Narrador:	Editar:	Eliminar:	
1	Alexis	test@test.cl	1234124	De quien es el Dragon?	Changeling	Naitsirc	Editar	Eliminar	
2	Savandija	test@test.cl	12341234	De quien es el Dragon?	Changeling	Naitsirc	Editar	Eliminar	
3	Milka	test@test.cl	12341234	De quien es el Dragon?	Changeling	Naitsirc	Editar	Eliminar	
4	Tomate	tomate@tes.cl	12341234	El Mercader Perdido	Mousguard	Alexis	Editar	Eliminar	

2.- Utiliza JOIN para relacionar la información de distintas tablas

Como se indico al mostrar la clase Jugador, la Clase Jugador, esta estaba vinculada a la Clase Mesa, lo que nos indica están enlazadas en una relación de 1 es a muchas. Aquí dejo señalado de forma explicita cuales datos son de la tabla Jugador y cual de Mesa

Lista de Inscritos								
Default				Jugador		Mesa		
Ord Asc				By Mesa 1		Group 1		
Ord Des				By Mesa 2		Group 2		
By Mesa 1				Group 1		Group 2		
By Mesa 2				Group 1		Group 2		
Group 1				Group 1		Group 2		
Group 2				Group 1		Group 2		
Id Jugador:				Jugador:		Mesa		
E-mail:				Telefono:		Mesa		
1				Alexis		test@test.cl		
2				Savandija		test@test.cl		
3				Milka		test@test.cl		
4				Tomate		tomate@tes.cl		
En la Mesa:				Juego:		Narrador:		
De quien es el Dragon?				Changeling		Naitsirc		
De quien es el Dragon?				Changeling		Naitsirc		
De quien es el Dragon?				Changeling		Naitsirc		
El Mercader Perdido				Mousguard		Alexis		
Editar:				Eliminar:				
Editar				Eliminar				
Editar				Eliminar				
Editar				Eliminar				
Editar				Eliminar				

3.- Utiliza WHERE para filtrar la información requerida,

4.- Utiliza cláusulas de ordenamiento para presentar la información, Y

5.- Utiliza cláusulas de agrupación de información para obtener datos agregados

```
public interface JugadorRepository extends JpaRepository<Jugador, Long> {  
  
    public List<Jugador> findAllByOrderByByIdInscritoAsc();  
    public List<Jugador> findAllByOrderByByIdInscritoDesc();  
    public List<Jugador> findAllByMesa(Mesa mesa);  
    @Query("SELECT COUNT(j) FROM Jugador j JOIN j.mesa m WHERE m.idMesa =:idMesa")  
    public Long findByMesa(@Param("idMesa") Long mesa);  
}
```

Aquí, en la clase repositorio podemos ver estos métodos que salvo el que lleva la anotación @Query, las otras solo por "escribirse" tal cual conllevan a decir que son queries en si lo que se llama "queries nativas" donde simplemente por decir "find" es sinónimo de decir Select, findAll es "Select *", By viene siendo el "Where", OrderBy es evidente, y acompañando al OrderBy es el campo por el cual queremos que ordene, si escribo Asc ordenara por ascendente, y si acompañó con Desc e forma descendente. Por ultimo tenemos el caso del método findByMesa en donde tenemos la anotación @Query e indicamos la Query para expresar lo siguiente "cuenta y agrupa todos los jugadores donde la mesa sea X. Dejare las vistas de los casos 3, 4 y 5

Utiliza *WHERE* para filtrar La información requerida

Aquí es "Buscame a todos los jugadores que están en la mesa 1".

O "SELECT * FROM jugadores_inscritos WHERE idMesa = 1."

Lista de Inscritos - Por Mesa								
Default	Ord Asc	Ord Des	By Mesa 1	By Mesa 2	Group 1	Group 2		
Id Jugador:	Jugador:	E-mail:	Telefono:	En la Mesa:	Juego:	Narrador:	Editar:	Eliminar:
1	Alexis	test@test.cl	1234124	De quien es el Dragon?	Changeling	Naitsirc	Editar	Eliminar
2	Savandija	test@test.cl	12341234	De quien es el Dragon?	Changeling	Naitsirc	Editar	Eliminar
3	Milka	test@test.cl	12341234	De quien es el Dragon?	Changeling	Naitsirc	Editar	Eliminar

Utiliza *cláusulas de ordenamiento* para presentar La información

"Buscame a todos los jugadores y ordenalos de forma descendiente según su id"

"SELECT * FROM jugadores_inscritos ORDER BY idInscrito Desc"

Lista de Inscritos - Descendente								
Default	Ord Asc	Ord Des	By Mesa 1	By Mesa 2	Group 1	Group 2		
Id Jugador:	Jugador:	E-mail:	Telefono:	En la Mesa:	Juego:	Narrador:	Editar:	Eliminar:
4	Tomate	tomate@tes.cl	12341234	El Mercader Perdido	Mousguard	Alexis	Editar	Eliminar
3	Milka	test@test.cl	12341234	De quien es el Dragon?	Changeling	Naitsirc	Editar	Eliminar
2	Savandija	test@test.cl	12341234	De quien es el Dragon?	Changeling	Naitsirc	Editar	Eliminar
1	Alexis	test@test.cl	1234124	De quien es el Dragon?	Changeling	Naitsirc	Editar	Eliminar

Utiliza *cláusulas de agrupación de información* para obtener datos agregados

"Cuéntame todos los jugadores que están en la mesa 1"

"SELECT nombre_mesa, count(id_inscrito) from jugadores_inscritos j JOIN mesas m WHERE m.id_mesa = 1 group by id_mesa"

Cantidad de Jugadores Inscritos por Mesa

Default

Ord Asc

Ord Des

By Mesa 1

By Mesa 2

Group 1

Group 2

Mesa:

Cant. Jugadores:

De quien es el Dragon?

3

Algoritmo de cálculo y unidades de prueba

6.- Utilización general del lenguaje, sintaxis, selección de tipos de datos, sentencias lógicas, expresiones, operaciones, comparaciones

Ejemplo de selección de datos

```
@Id
@GeneratedValue(strategy=GenerationType.IDENTITY)
private Long idInscrito;
```

Ejemplo de comparaciones, sentencias expresiones

```
21 @Controller
22 @RequestMapping("/views/usuarios")
23 public class UsuarioController {
24
25     @Autowired
26     private IUsuarioService userService;
27
28     @GetMapping("/")
29     public String listarUsuarios(Model model, HttpSession session, RedirectAttributes attribute) {
30
31         if (session.getAttribute("perfil") == null || "Narrador".equals(session.getAttribute("perfil"))) {
32             System.out.println("entro");
33             // anadir error x login no autorizado
34             attribute.addFlashAttribute("error", "No Tienes Permiso para ver esta página");
35             // Redirigir a home
36             return "redirect:/";
37         }
38
39         List<Usuario> listaUsuarios = userService.listarTodos();
40         model.addAttribute("titulo", "Lista de Usuarios");
41         model.addAttribute("usuarios", listaUsuarios);
42         return "/views/usuarios/listarusuarios";
43     }
44 }
```

7.- Utilización de sentencias repetitivas

Ejemplo

```
<tbody>
<tr th:each="pj:${inscritos}" class="table-dark">
    <th scope="row" th:text="${pj.idInscrito}"></th>
    <td th:text="${pj.nombreJugador}"></td>
    <td th:text="${pj.email}"></td>
    <td th:text="${pj.telefono}"></td>
    <td th:text="${pj.mesa.nombreMesa}"></td>
    <td th:text="${pj.mesa.juego}"></td>
    <td th:text="${pj.mesa.narrador}"></td>
    <td>
        <a class="btn btn-outline-warning btn-sm" th:href="@{/views/jugadores/edit/}+${pj.idInscrito}"
        th:text="Editar" title="Editar Registro"></a>
    </td>
    <td>
        <a class="btn btn-outline-danger btn-sm" th:href="@{/views/jugadores/delete/}+${pj.idInscrito}"
        th:text="Eliminar" title="Eliminar Registro"
        onclick="return confirm('Esta Seguro de querer Eliminar?');"></a>
    </td>
</tr>
</tbody>
```

8.- Utilización de clases, encapsulamiento y responsabilidad única

Ejemplo

```
1 package cl.vrol.controller;
2
3 import org.springframework.beans.factory.annotation.Autowired;
4
14 @Controller
15 @RequestMapping("/views/contacto")
16 public class ContactoController {
17
18     @Autowired
19     private IContactoService contactoService;
20
21
22     @GetMapping("/")
23     public String contacto(Model model) {
24         Contacto contacto = new Contacto();
25         model.addAttribute("titulo", "Nuevo Mensaje");
26         model.addAttribute("contacto", contacto);
27         return "/views/contacto/contacto";
28     }
29
30     @PostMapping("/enviar")
31     public String enviarMsj(@ModelAttribute Contacto contacto, RedirectAttributes attribute) {
32         contactoService.guardar(contacto);
33         System.out.println("Contacto Enviado: "+contacto);
34         attribute.addFlashAttribute("success", "Mensaje enviado con Exito!");
35         return "redirect:/views/contacto/";
36     }
37
38 }
39
```

9.- Se utilizan correctamente interfaces o relaciones de herencia para hacer polimorfismo donde fuese necesario

En el caso de este proyecto, el uso de interfaces, herencia es marcado sobretodo entre las clases de repositorio y servicio

Ejemplo Clase Jugador Repository la cual extiende (hereda) de JpaRepository

```
1 package cl.vrol.models.repository;
2
3 import java.util.List;
4
11
12
13 public interface JugadorRepository extends JpaRepository<Jugador, Long> {
14
15     public List<Jugador> findAllByOrderByIdInscritoAsc();
16     public List<Jugador> findAllByOrderByIdInscritoDesc();
17     public List<Jugador> findAllByMesa(Mesa mesa);
18     @Query("SELECT COUNT(j) FROM Jugador j JOIN j.mesa m WHERE m.idMesa =:idMesa")
19     public Long findByMesa(@Param("idMesa") Long mesa);
20
21 }
22
```

Ejemplo Interface (lo cual antes seria un DAO) IJugadorService

```
1 package cl.vrol.models.service;
2
3 import java.util.List;
4
10
11 public interface IJugadorService {
12
13     public List<Jugador> listarTodos();
14     public void guardar(Jugador jugador);
15     public Jugador buscarPorId(Long idInscrito);
16     public void eliminar(Long idInscrito);
17     public List<Jugador> findAllbyOrderIdAsc();
18     public List<Jugador> findAllbyOrderIdDes();
19     public List<Jugador> findAllByMesa(Mesa mesa);
20     public Long findByMesa(Long idMesa);
21
22
23 }
24
```

Ejemplo de Clase JugadorServiceImpl la cual implemente Interface IJugadorService

```
1 package cl.vrol.models.service;
2
3 import java.util.List;
4
11
12 @Service
13 public class JugadorServiceImpl implements IJugadorService {
14
15     @Autowired
16     private JugadorRepository playerRepository;
17
18     @Override
19     public List<Jugador> listarTodos() {
20         return (List<Jugador>) playerRepository.findAll();
21     }
22
23     @Override
24     public void guardar(Jugador jugador) {
25         playerRepository.save(jugador);
26     }
27
28     @Override
29     public Jugador buscarPorId(Long idInscrito) {
30         return playerRepository.findById(idInscrito).orElse(null);
31     }
32
33     @Override
34     public void eliminar(Long idInscrito) {
35         playerRepository.deleteById(idInscrito);
36     }
37
38     @Override
39     public List<Jugador> findAllbyOrderIdAsc() {
40         return playerRepository.findAllByOrderByInscritoAsc();
41     }
42
43 }
```

```
43 @Override
44 public List<Jugador> findAllByIdDes() {
45     return playerRepository.findAllByIdInscritoDesc();
46 }
47
48 @Override
49 public List<Jugador> findAllByMesa(Mesa mesa) {
50     return playerRepository.findAllByMesa(mesa);
51 }
52
53 @Override
54 public Long findByMesa(Long mesa) {
55     return playerRepository.findByMesa(mesa);
56 }
57
58
59
60 }
61
```

10.- Convenciones y estilos de programación

Varias desde el Uso de camelCase y otras como uso de getters, Setters, etc.

Como ejemplo en la siguiente clase podemos apreciar las siguientes:

- El nombre de la clase "Jugador" usa una convención de nomenclatura en CamelCase, donde la primera letra de cada palabra se escribe en mayúscula, excepto la primera letra de la primera palabra.
- Las anotaciones utilizadas, como @Entity, @Table, @Id, @GeneratedValue, @ManyToOne, @JoinColumn, son convenciones que pertenecen a la especificación JPA (Java Persistence API), y se utilizan para mapear la clase a una tabla de base de datos.
- Los nombres de los atributos y métodos siguen la convención de nomenclatura en camelCase, donde la primera letra de la primera palabra se escribe en minúscula, y la primera letra de cada palabra siguiente se escribe en mayúscula.
- Las constantes como "serialVersionUID" se nombran utilizando una convención de nomenclatura en mayúsculas y minúsculas (camelCase) y en mayúsculas, respectivamente.
- Los nombres de los métodos, como "getIdInscrito", "setNombreJugador", "getEmail", etc., siguen la convención de nomenclatura de los métodos getter y setter.
- La clase tiene un método toString() que sigue la convención de nomenclatura de este método y escribe la representación en cadena del objeto.
- En general, la clase sigue el estilo de codificación en Java, que incluye indentación, uso de llaves, comentarios, etc.

```
1 package cl.vrol.models.entity;
2
3 import java.io.Serializable;
15
16 @Entity
17 @Table(name = "jugadores_inscritos")
18 public class Jugador implements Serializable {
19
20     /**
21      *
22      */
23     private static final long serialVersionUID = 1L;
24
25     @Id
26     @GeneratedValue(strategy=GenerationType.IDENTITY)
27     private Long idInscrito;
28
29     @NotEmpty
30     private String nombreJugador;
31
32     @Email
33     private String email;
34
35     @Pattern(regexp="[0-9]{9}")
36     private String telefono;
37
38     @ManyToOne
39     @JoinColumn(name="mesa_id")
40     private Mesa mesa;
41
42     public Long getIdInscrito() {
43         return idInscrito;
44     }
45
46     public void setIdInscrito(Long idInscrito) {
47         this.idInscrito = idInscrito;
48     }
49
50     public String getNombreJugador() {
51         return nombreJugador;
52     }
53
54     public void setNombreJugador(String nombreJugador) {
55         this.nombreJugador = nombreJugador;
56     }
57
58     public String getEmail() {
59         return email;
60     }
61
62     public void setEmail(String email) {
63         this.email = email;
64     }
65
66     public String getTelefono() {
67         return telefono;
68     }
69
70     public void setTelefono(String telefono) {
71         this.telefono = telefono;
72     }
73
74     public Mesa getMesa() {
75         return mesa;
76     }
```



```

77 public void setMesa(Mesa mesa) {
78     this.mesa = mesa;
79 }
80
81 @Override
82 public String toString() {
83     StringBuilder builder = new StringBuilder();
84     builder.append("Jugador [idInscrito=");
85     builder.append(idInscrito);
86     builder.append(", nombreJugador=");
87     builder.append(nombreJugador);
88     builder.append(", email=");
89     builder.append(email);
90     builder.append(", telefono=");
91     builder.append(telefono);
92     builder.append(", mesa=");
93     builder.append(mesa);
94     builder.append("]");
95     return builder.toString();
96 }
97
98 }

```

11.- Utilización de unidades de prueba

Dejo muestra de Caso de Prueba de Clase ContactoServiceImpl

```

1 package cl.vrol.tests;
2
3 import static org.junit.jupiter.api.Assertions.assertEquals;
4
5 public class ContactoServiceImplTest {
6
7     @Mock
8     private ContactoRepository contactoRepository;
9
10    @InjectMocks
11    private ContactoServiceImpl contactoService;
12
13    @Captor
14    private ArgumentCaptor<Contacto> contactoCaptor;
15
16    @BeforeEach
17    public void setUp() {
18        MockitoAnnotations.openMocks(this);
19    }
20
21    @Test
22    public void guardarTest() {
23        Contacto contacto = new Contacto();
24        contacto.setNomContacto("John");
25        contacto.setEmail("johndoe@example.com");
26        contactoService.guardar(contacto);
27
28        verify(contactoRepository).save(contactoCaptor.capture());
29        assertEquals(contacto, contactoCaptor.getValue());
30    }
31 }

```

Página web html y css

12.- Utilización de tags html, estilos y responsividad

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head th:fragments="head">
<meta charset="UTF8">
<title>Comunidad VROL</title>
<link rel="stylesheet" type="text/css" th:href="@{/css/bootstrap.min.css}" />
<link rel="stylesheet" type="text/css" th:href="@{/css/style.css}" />
<link rel="preconnect" href="https://fonts.googleapis.com"><link rel="preconnect" href="https://fonts.gstatic.com" crossorigin><link
<link rel="shortcut icon" th:href="@{/imgs/LogoVrol.png}" type="image/x-icon">
</head>
<!-- NAVBAR -->
<header th:fragments="header">
<nav class="navbar navbar-expand-lg bg-body-tertiary bg-dark" data-bs-theme="dark">
<div id="navs" class="container-fluid">
 <a
class="navbar-brand" th:href="@{/}">&nbsp;<strong>Comunidad VROL</strong></a>
<button class="navbar-toggler" type="button" data-bs-toggle="collapse"
data-bs-target="#navbarNavDropdown" aria-controls="navbarNavDropdown"
aria-expanded="false" aria-label="Toggle navigation">
<span class="navbar-toggler-icon"></span>
</button>
<div class="collapse navbar-collapse" id="navbarNavDropdown">
<ul class="navbar-nav">
<li class="nav-item"><a class="nav-link" th:href="@{/juegos/}">Juegos</a>
</li>
<li class="nav-item"><a class="nav-link" th:href="@{/Calendario/}">Calendario</a>
</li>
<li class="nav-item"><a class="nav-link"
th:href="@{/views/contacto/}">Contacto</a></li>
<li class="nav-item"><a class="nav-link" th:href="@{/mesas/}">Mesas</a>
</li>
```

```

class="nav-link dropdown-toggle" href="#" role="button"
data-bs-toggle="dropdown" aria-expanded="false"> Links Útiles </a>
<ul class="dropdown-menu">
<li><a class="dropdown-item"
href="https://chat.whatsapp.com/C6RxtSo2z5b44229GhpIhH">Chat
whatsapp</a></li>
<li><a class="dropdown-item"
href="https://web.facebook.com/groups/1168027804123411">Facebook</a></li>
<li><a class="dropdown-item"
href="https://discord.gg/vfs4aVQfHp">Discord</a></li>
<li><a class="dropdown-item"
href="https://www.instagram.com/v_r0lc1ub/">Instagram</a></li>
<li><hr class="dropdown-divider"></li>
<li><a class="dropdown-item disabled"
href="https://forms.gle/xUiy9rVNs82mVQ228">Inscripcion Junta</a></li>
</ul></li>
<li class="nav-item dropdown" th:if="${session.perfil == 'Administrador' || session.perfil == 'Narrador'}">
<a class="nav-link dropdown-toggle" href="#" role="button"
data-bs-toggle="dropdown" aria-expanded="false"> Narrador</a>
<ul class="dropdown-menu">
<li class="nav-item">
<a class="nav-link text-muted" th:href="@{/}">Crear Mesa</a>
</li>
<li class="nav-item">
<a class="nav-link" th:href="@{/views/jugadores/}">Listar Inscripciones</a>
</li>
</ul>
</li>
</div>
<li class="nav-item dropdown" th:if="${session.perfil == 'Administrador'}">
<a class="nav-link dropdown-toggle" href="#" role="button" data-bs-toggle="dropdown" aria-expanded="f
<ul class="dropdown-menu">
<div th:if="${session.perfil == 'Administrador'}">
<li class="nav-item">
```

```

65<div>
66<li class="nav-item dropdown" th:if="${session.perfil == 'Administrador'}">
67<a class="nav-link dropdown-toggle" href="#" role="button" data-bs-toggle="dropdown" aria-expanded="f
68<ul class="dropdown-menu">
69<div th:if="${session.perfil == 'Administrador'}">
70<li class="nav-item">
71<a class="nav-link" th:href="@{/views/usuarios/}">Listar Usuarios</a>
72</li>
73<li class="nav-item">
74<a class="nav-link" th:href="@{/views/usuarios/nuevousuario}">Crear Usuario</a>
75</li>
76</div>
77</ul>
78</li>
79</div>
80</ul>
81<div th:if="${session.perfil == null}" id="loginSector">
82<form class="d-flex" th:action="@{/views/Login/}">
83<button class="btn btn-outline-info btn-sm" type="submit">Login</button>
84</form>
85</div>
86<div id="loginSector" th:if="${session.perfil == 'Narrador' || session.perfil == 'Administrador'}">
87<form class="d-flex align-items-center ml-auto">
88<p class="mb-0 mr-4 text-info" th:text="'Welcome: ' + ${session.nick}">&nbsp;&nbsp;&nbsp;</p>&nbsp;&nbsp;&nbsp;</p>
89<a class="btn btn-outline-primary dropdown-toggle ml-3" href="#" role="button" id="dropdownMenuLink"
90data-bs-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
91<span class="d-none d-md-inline-block">Menu</span>
92</a>
93<div class="dropdown-menu dropdown-menu-end ml-auto" aria-labelledby="dropdownMenuLink"
94style="flex: 1; justify-content: flex-end;">
95<a class="dropdown-item" href="#">Editar Perfil</a>
96<a class="dropdown-item text-danger ml-auto" th:href="@{/views/Login/Logout}" type="button">Logout</a>
97</div>
98</form>
99</div>

```

```

99</div>
100</div>
101</div>
102</nav>
103<!-- MENSAJES -->
104<div class="alert alert-success alert-dismissible position-absolute" th:if="${success != null}">
105<label th:text="${success}"></label>
106<button type="button" class="close" data-dismiss="alert" aria-label="Close">&times;</button>
107</div>
108<div class="alert alert-danger alert-dismissible position-absolute" th:if="${error != null}">
109<label class="textvis" th:text="${error}"></label>
110<button type="button" class="close float-right" data-dismiss="alert">&times;</button>
111</div>
112<div class="alert alert-warning alert-dismissible" th:if="${warning != null}">
113<label th:text="${warning}"></label>
114<button type="button" class="close" data-dismiss="alert" aria-label="Close">&times;</button>
115</div>
116<div class="alert alert-info alert-dismissible position-absolute" th:if="${info != null}">
117<label th:text="${info}"></label>
118<button type="button" class="close" data-dismiss="alert" aria-label="Close">&times;</button>
119</div>
120</header>
121<!-- CONTENT -->
122<div class="container-fluid">
123
124
125</div>
126<!-- Footer -->

```

```

125 </div>
126 <!-- Footer -->
127 <footer th:fragments="footer" id="footer">
128 <div class="d-flex flex-wrap justify-content-between align-items-center py-3 my-4 border-top fixed-bottom" id="footersp">
129 <p class="col-md-4 mb-0" id="footertext">© Naitsirc, Inc 2023. </p>
130 <ul class="nav col-sm-5 col-md-5 justify-content-end">
131 <li class="nav-item footlink"><a href="/" class="nav-link px-2 footlink" id="home">Home</a></li>
132 <li class="nav-item footlink"><p class="nav-link px-2 footlink" id="min">Siguenos en nuestras RR.SS.</p></li>
133 <li class="nav-item footlink"><a href="https://web.facebook.com/groups/1168027804123411" class="nav-link px-2 footlink te
134 <li class="nav-item footlink"><a href="https://chat.whatsapp.com/C6RxtSo2zSb44229GhpIhH" class="nav-link px-2 footlink te
135 <li class="nav-item footlink"><a href="https://discord.gg/vfs4aVQfHp" class="nav-link px-2 text-muted footlink"><img th:src
136 <li class="nav-item footlink"><a href="https://www.instagram.com/v_roclub/" class="nav-link px-2 text-muted footlink"><im
137 </ul>
138 </div>
139 <!-- JS -->
140 <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
141 <script th:src="@{/js/popper.min.js}"></script>
142 <script th:src="@{/js/bootstrap.min.js}"></script>
143 <script th:src="@{/js/main.js}"></script>
144 <script src="https://cdn.jsdelivr.net/npm/sweetalert2@11"></script>
145 </footer>
146 </body>
147 </html>

```

13.- Utilización de Bootstrap

Tal cual como demostrado en [punto 12](#) en el cual se muestra la plantilla de HTML, se usa archivos de estilos de Bootstrap, además de archivos de css propio, mismo caso ocurre con el archivo de JS de Bootstrap

Spring MVC

14.- Utilización de controladores

Este proyecto contempla la creación (al momento) de 10 Controladores activos los cuales son: CalendarController (controla la vista de la sección de Calendario en la pagina), ContactoController (controla la vista de formulario y el envío de mensaje), HomeController (controla el despliegue de home o vista principal), JuegosController (controla el despliegue de la vista de juegos y a futuro también controlara los juegos por individual), JugadorController (controla la inscripción de un nuevo jugador, listarlos, editarlos y borrarlos), JuntasController (controla el despliegue de la vista de álbum con fotos de las juntas), ListarJController (controlador REST, despliega información de los jugadores que se han inscrito en mesas, se espera que también la información sea descargada como un pdf próximamente), LoginController (controla el login, desde vista, proceso y logout), MesasController (controla la vista de las mesas, y controlara la creación, edición y borrado de mesas), y UsuarioController (controla la creación, listado, edición y eliminación de usuarios). En el [punto 8](#) deje un print de la clase controladora ContactoController.

15.- Utilización de vistas JSP y Taglib

Tal y como señalo en el ítem de [Dependencias](#) casi al final, decidí utilizar Thymeleaf, la cual no soporta el uso de JSP. Las TagLib de JSP son etiquetas personalizadas que se pueden utilizar para simplificar y mejorar la legibilidad de los archivos JSP. Thymeleaf, como motor de plantillas, no utiliza taglib en el sentido tradicional de JSP, pero proporciona una serie de atributos y directivas personalizadas para simplificar la escritura de plantillas HTML.

Aquí hay algunos ejemplos de etiquetas Thymeleaf que imitan el uso de TagLib en JSP:

th:include - Utilizado para incluir fragmentos de plantilla en otra plantilla.

```
<th:block th:include="fragments/footer :: footer"></th:block>
```

th:replace - Utilizado para reemplazar un elemento HTML con otro elemento o fragmento de plantilla.

```
<th:block th:replace="fragments/header :: header"></th:block>
```

th:with - Utilizado para definir una variable local en una plantilla.

```
<th:block th:with="titulo='Bienvenido'">
```

```
  <h2 th:text="${titulo}"></h2>
```

```
</th:block>
```

th:attr - Utilizado para establecer atributos HTML en un elemento con el valor de una variable o expresión.

```
<a th:attr="href=@{/producto/{id}(id=${producto.id})}, title=${producto.nombre}">
```

```
  <h2 th:text="${producto.nombre}"></h2>
```

```
</a>
```

th:text - Utilizado para establecer el contenido de un elemento HTML con el valor de una variable o expresión.

th:if y th:unless - Utilizados para establecer estructuras condicionales en una plantilla.

```
<th:block th:if="${usuario != null}">
```

```
  <h2>Bienvenido, <span th:text="${usuario.nombre}"></span>!</h2>
```

```
</th:block>
```

th:each - Utilizado para iterar sobre una lista o colección de elementos.

```
<th:block th:each="producto : ${productos}">
```

```
  <h2 th:text="${producto.nombre}"></h2>
```

```
  <p th:text="${producto.descripcion}"></p>
```

```
  <p th:text="${producto.precio}"></p>
```

```
</th:block>
```


16.- Creación de Servicio Spring y

17.- Creación de DAO acceso a Datos

Este al ser un proyecto SpringBoot con JPA no tiene la presencia de las clases DAO tan explicitas como lo es en un proyecto Spring MVC normal, en ves de los DAO tenemos las clases Repositorio y los Service que son Interfaces (estos últimos los cuales se implementan en una clase que las Implementa). En el [punto 9](#) deje prints de JugadorRepository, IJugadorService y JugadorServiceImpl

18.- Creación del Proyecto y Configuración

```
https://maven.apache.org/xsd/maven-4.0.0.xsd (xsi:schemaLocation)
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">
4   <modelVersion>4.0.0</modelVersion>
5   <parent>
6     <groupId>org.springframework.boot</groupId>
7     <artifactId>spring-boot-starter-parent</artifactId>
8     <version>3.0.6</version>
9     <relativePath/> <!-- lookup parent from repository -->
10  </parent>
11  <groupId>cl.vrol</groupId>
12  <artifactId>VrolSite</artifactId>
13  <version>0.0.1-SNAPSHOT</version>
14  <name>VrolSite</name>
15  <description>VrolSite</description>
16  <properties>
17    <java.version>17</java.version>
18  </properties>
19  <dependencies>
20    <dependency>
21      <groupId>org.springframework.boot</groupId>
22      <artifactId>spring-boot-starter-data-jpa</artifactId>
23    </dependency>
24    <dependency>
25      <groupId>org.springframework.boot</groupId>
26      <artifactId>spring-boot-starter-thymeleaf</artifactId>
27    </dependency>
28    <dependency>
29      <groupId>nz.net.ultraq.thymeleaf</groupId>
30      <artifactId>thymeleaf-layout-dialect</artifactId>
31      <version>3.2.1</version>
32    </dependency>
```

```

34<  <dependency>
35    <groupId>org.hibernate.validator</groupId>
36    <artifactId>hibernate-validator</artifactId>
37    <version>7.0.1.Final</version>
38  </dependency>
39
40<  <dependency>
41    <groupId>org.springframework.boot</groupId>
42    <artifactId>spring-boot-starter-web</artifactId>
43  </dependency>
44
45<  <dependency>
46    <groupId>org.springframework.boot</groupId>
47    <artifactId>spring-boot-devtools</artifactId>
48    <scope>runtime</scope>
49    <optional>true</optional>
50  </dependency>
51<  <dependency>
52    <groupId>mysql</groupId>
53    <artifactId>mysql-connector-java</artifactId>
54    <version>8.0.32</version>
55  </dependency>
56
57<  <dependency>
58    <groupId>org.springframework.boot</groupId>
59    <artifactId>spring-boot-starter-test</artifactId>
60    <scope>test</scope>
61  </dependency>

```

```

63<  <dependency>
64    <groupId>org.projectlombok</groupId>
65    <artifactId>lombok</artifactId>
66    <version>1.18.26</version>
67    <scope>provided</scope>
68  </dependency>
69
70<!-- JUNIT -->
71<  <dependency>
72    <groupId>org.springframework.boot</groupId>
73    <artifactId>spring-boot-starter-test</artifactId>
74    <scope>test</scope>
75  </dependency>
76
77<  <dependency>
78    <groupId>org.junit.jupiter</groupId>
79    <artifactId>junit-jupiter-api</artifactId>
80    <version>5.7.2</version>
81    <scope>test</scope>
82  </dependency>
83
84<  <dependency>
85    <groupId>org.junit.jupiter</groupId>
86    <artifactId>junit-jupiter-engine</artifactId>
87    <version>5.7.2</version>
88    <scope>test</scope>
89  </dependency>


```

```

105     </dependencies>
106
107     <build>
108         <plugins>
109             <plugin>
110                 <groupId>org.springframework.boot</groupId>
111                 <artifactId>spring-boot-maven-plugin</artifactId>
112             </plugin>
113         </plugins>
114     </build>
115
116 </project>
117

```

19.- Funcionamiento General del Aplicativo


Comunidad VROL
[Juegos](#)
[Calendario](#)
[Contacto](#)
[Mesas](#)
[Juntas](#)
[Links Utiles](#)
[Login](#)

Quienes Somos

Somos una Comunidad que nacio de la Quinta Región y alrededores, quienes tenemos un gran amor a los Juegos de Rol.

Nuestro objetivo es jugar, promover, difundir, gestionar espacios (virtuales y presenciales), generar comunidad, y fomentar la cultura de los juegos de rol, para que estos sean mas conocidos y así poder lograr que mas personas se unan a esta gran comunidad.

Queremos brindar un espacio seguro para que todos puedan aportar y sentirse parte de la comunidad, por eso tenemos un conjunto de normas de convivencia, para así velar para que todos y todas se sientan a gusto. Si no tienes experiencia en los Juegos de Rol, no te preocupes, todos contamos con la vocacion de enseñar y nos encanta tener mas personas que quieran aprender.


UNETE


Normas

1. Como eje principal de nuestra convivencia, debemos fomentar el **RESPECTO** entre todos los participantes de VROL, tanto en actividades presenciales como en sus plataformas, para compartir en un espacio seguro y agradable.
2. NO se tolerará el bullying en cualquiera de sus formas, y en ninguna de las plataformas que conforman VROL.
3. NO aceptaremos ningún tipo de discriminación, ya sea por género, religión, nacionalidad, creencias, gustos, etc.
4. Debemos respetar la privacidad de todos los integrantes de la comunidad.
5. Es necesario que informen a los administradores si sufren o son testigos de alguna infracción a estas normas, para poder seguir el protocolo correspondiente

© Naitsirc, Inc 2023.

Home Siguenos en nuestras RR.SS. [f](#) [w](#) [m](#) [i](#)



Comunidad VROL
[Juegos](#)
[Calendario](#)
[Contacto](#)
[Mesas](#)
[Juntas](#)
[Links Utiles](#)
[Login](#)



Changeling: The Dreaming v2.0

S: Sistema Narrativo(d10)


Conocer mas



Dungeons and Dragons

S: d20


Conocer mas



Pathfinder

S: d20


Conocer mas



Vampiro: La Mascarada

S: Sistema Narrativo(d10)

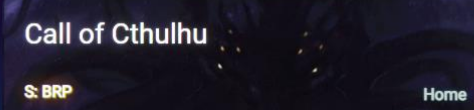
Conocer mas



Mage: The Ascension

S: Sistema Narrativo(d10)

Conocer mas



Call of Cthulhu

S: BRP

Conocer mas

Mesas On-Line y Presenciales

La Mina Perdida de Phandelver

The Screaming Nights

Aventura Antartica

Mahou Shoujo: Miraculous Adv

PokeRole

Mouse Guard: Aventuras ratoniles

Conoce mas

Home Siguenos en nuestras RR.SS. [f](#) [w](#) [m](#) [i](#)

Comunidad VROL
Juegos
Calendario
Contacto
Mesas
Juntas
Links Útiles
Login

Calendario de Eventos Actuales

Hoy
mayo 2023
Imprimir

dom	lun	mar	mié	jue	vie	sáb
30 Starfinder: proyecto TITAN	1 may	2	3	4	5 Dnd - Campaña por la suc Dnd - Mundos de Fantasía	6 [7mo Mar] Oro y Sombras Scion 2e - Las Semillas de
7 Starfinder: proyecto TITAN	8 F.I.S.T. - Quiero matarla -	9	10	11	12 Dnd - Campaña por la suc	13 [7mo Mar] Oro y Sombras Scion 2e - Las Semillas de
14 Starfinder: proyecto TITAN	15 F.I.S.T. - Quiero matarla - Intro D&D (Online DS)	16 Semana Temática - 7mo M	17 No te duermas - Estación I Semana Temática - F.I.S.T.	18 Semana Temática - Himbo	19 Dnd - Campaña por la suc Semana Temática - Monst	20 [7mo Mar] Oro y Sombras Scion 2e - Las Semillas de Semana Temática - Dnd -
21 Starfinder: proyecto TITAN	22 F.I.S.T. - Quiero matarla -	23	24 No te duermas - Estación I	25	26 Dnd - Campaña por la suc	27
28 Starfinder: proyecto TITAN	29	30	31 No te duermas - Estación I	1 jun Dune - aGbo!	2	3

Eventos que se muestran en la zona horaria: Hora de Chile
Google Calendar

© Naitsirc, Inc 2023.
Home
Siguenos en nuestras RR.SS.

Comunidad VROL
Juegos
Calendario
Contacto
Mesas
Juntas
Links Útiles
Login

Nuevo Mensaje

Nombre de Remitente

No puede estar vacío

Email

No puede estar vacío

Numero Telefonico

No puede estar vacío

Ingresa tu mensaje. Te responderemos a la brevedad.

No puede estar vacío

Enviar Mensaje

© Naitsirc, Inc 2023.
Home
Siguenos en nuestras RR.SS.

Comunidad VROL
Juegos
Calendario
Contacto
Mesas
Juntas
Links Utiles
Login

Changelling

De quien es el Dragon?

Narrador: Naitsirc

Cupos: 4

Formato: presencial

Fecha: 2023/04/17

Hay un Dragon Quimerico suelto en la Ciudad

TAGS: traumas, abuso

Inscribirse

Mousguard

El Mercader Perdido

Narrador: Alexis

Cupos: 4

Formato: presencial

Fecha: 2023/04/17

Se ha perdido un Cargamento de Arroz, encuéntralo

TAGS: aventuras ratoniles

Inscribirse

© Naitsirc, Inc 2023.
Home
Siguenos en nuestras RR.SS.

Comunidad VROL
Juegos
Calendario
Contacto
Mesas
Juntas
Links Utiles
Login

Sector Social - Las Juntas Mensuales

Pequena Galeria de imagenes para dejar un registro breve de las Juntas Mensuales del grupo. Para ver todas las fotos que se toman visiten el

Instagram

Thumbnail

This is a wider card with supporting text below as a natural lead-in to additional content. This content is a little bit longer.

View Edit

9 mins

Thumbnail

This is a wider card with supporting text below as a natural lead-in to additional content. This content is a little bit longer.

View Edit

9 mins

Thumbnail

This is a wider card with supporting text below as a natural lead-in to additional content. This content is a little bit longer.

View Edit

9 mins


Comunidad VROL

[Juegos](#)
[Calendario](#)
[Contacto](#)
[Mesas](#)
[Juntas](#)
[Links Útiles](#)

Quienes Somos

Somos una Comunidad que nacio de la Quinta Región y alrededores, quienes tenemos un gran amor a los Juegos de Rol.

Nuestro objetivo es jugar, promover, difundir, gestionar espacios (virtuales y presenciales), generar comunidad, y fomentar la cultura de los juegos de rol, para que estos sean mas conocidos y así poder lograr que mas personas se unan a esta gran comunidad.


[Chat whatsapp](#)
[Facebook](#)
[Discord](#)
[Instagram](#)

[Inscripcion Junta](#)


Comunidad VROL

[Juegos](#)
[Calendario](#)
[Contacto](#)
[Mesas](#)
[Juntas](#)
[Links Útiles](#)

Login



Ingreso al Sistema


Nombre de Usuario:

Password:

Ingresar

© PDLY 2023

© Naitsirc, Inc 2023.
 [Home](#)
 Siguenos en nuestras RR.SS.
 [f](#)
[w](#)
[d](#)
[i](#)


Comunidad VROL

[Juegos](#)
[Calendario](#)
[Contacto](#)
[Mesas](#)
[Juntas](#)
[Links Útiles](#)

[Narrador](#)
[Admin M.](#)

Welcome: Admin

Menu

Se ha logueado: Admin

Quienes Somos

Somos una Comunidad que nacio de la Quinta Región y alrededores, quienes tenemos un gran amor a los Juegos de Rol.

Nuestro objetivo es jugar, promover, difundir, gestionar espacios (virtuales y presenciales), generar comunidad, y fomentar la cultura de los juegos de rol, para que estos sean mas conocidos y así poder lograr que mas personas se unan a esta gran comunidad.

Queremos brindar un espacio seguro para que todos puedan aportar y sentirse parte de la comunidad, por eso tenemos un conjunto de normas de convivencia, para así velar para que todos y todas se sientan a gusto. Si no tienes experiencia en los Juegos de Rol, no te preocupes, todos contamos con la vocacion de enseñar y nos encanta tener mas personas que quieran aprender.

UNETE

Normas

1. Como eje principal de nuestra convivencia, debemos fomentar el **RESPECTO** entre todos los participantes de VROL, tanto en actividades presenciales como en sus plataformas, para compartir en un espacio seguro y agradable.
2. NO se tolerará el bullying en cualquiera de sus formas, y en ninguna de las plataformas que conforman VROL.
3. NO aceptaremos ningún tipo de discriminación, ya sea por género, religión, nacionalidad, creencias, gustos, etc.
4. Debemos respetar la privacidad de todos los integrantes de la comunidad.
5. Es necesario que informen a los administradores si sufren o son testigos de alguna infracción a estas normas, para poder seguir el protocolo correspondiente

Comunidad VROL

JuegosCalendarioContactoMesasJuntasLinks Utiles

Login

No Tienes Permiso para ver esta página

Quienes Somos

Somos una Comunidad que nacio de la Quinta Región y alrededores, quienes tenemos un gran amor a los Juegos de Rol.

Nuestro objetivo es jugar, promover, difundir, gestionar espacios (virtuales y presenciales), generar comunidad, y fomentar la cultura de los juegos de rol, para que estos sean mas conocidos y así poder lograr que mas personas se unan a esta gran comunidad.

Queremos brindar un espacio seguro para que todos puedan aportar y sentirse parte de la comunidad, por eso tenemos un conjunto de normas de convivencia, para así velar para que todos y todas se sientan a gusto. Si no tienes experiencia en los Juegos de Rol, no te preocupes, todos contamos con la vocacion de enseñar y nos encanta tener mas personas que quieran aprender.

UNETE

Naitsirc, Inc 2023.

HomeSíguenos en nuestras RR.SS.

Normas

1. Como eje principal de nuestra convivencia, debemos fomentar el **RESPECTO** entre todos los participantes de VROL, tanto en actividades presenciales como en sus plataformas, para compartir en un espacio seguro y agradable.

2. NO se tolerará el bullying en cualquiera de sus formas, y en ninguna de las plataformas que conforman VROL.

3. NO aceptaremos ningún tipo de discriminación, ya sea por género, religión, nacionalidad, creencias, gustos, etc.

4. Debemos respetar la privacidad de todos los integrantes de la comunidad.

5. Es necesario que informen a los administradores si sufren o son testigos de alguna infracción a estas normas, para poder seguir el protocolo correspondiente

Comunidad VROL

JuegosCalendarioContactoMesasJuntasLinks Utiles

Login

Inscribir - Nuvo Jugador

Nombre del Jugador:

Tal Ivan

E-mail:

test@test.cl

Telefono:

912341234

Mesa:

El Mercader Perdido

Inscribirse

Comunidad VROL

JuegosCalendarioContactoMesasJuntasLinks Útiles

Login

Jugador Inscrito!

Quienes Somos

Somos una Comunidad que nació de la Quinta Región y alrededores, quienes tenemos un gran amor a los Juegos de Rol.

Nuestro objetivo es jugar, promover, difundir, gestionar espacios (virtuales y presenciales), generar comunidad, y fomentar la cultura de los juegos de rol, para que estos sean mas conocidos y así poder lograr que mas personas se unan a esta gran comunidad.

Queremos brindar un espacio seguro para que todos puedan aportar y sentirse parte de la comunidad, por eso tenemos un conjunto de normas de convivencia, para así velar para que todos y todas se sientan a gusto. Si no tienes experiencia en los Juegos de Rol, no te preocupes, todos contamos con la vocación de enseñar y nos encanta tener mas personas que quieran aprender.

UNETE

Normas

1. Como eje principal de nuestra convivencia, debemos fomentar el **RESPECTO** entre todos los participantes de VROL, tanto en actividades presenciales como en sus plataformas, para compartir en un espacio seguro y agradable.

2. NO se tolerará el bullying en cualquiera de sus formas, y en ninguna de las plataformas que conforman VROL.

3. NO aceptaremos ningún tipo de discriminación, ya sea por género, religión, nacionalidad, creencias, gustos, etc.

4. Debemos respetar la privacidad de todos los integrantes de la comunidad.

5. Es necesario que informen a los administradores si sufren o son testigos de alguna infracción a estas normas, para poder seguir el protocolo correspondiente

Naitsirc, Inc 2023.

Home

Síguenos en nuestras RR.SS.

Comunidad VROL

JuegosCalendarioContactoMesasJuntasLinks Útiles

NarradorAdmin M.

Welcome: Admin

Menu

Lista de Usuarios

Id Usuario:	Usuario:	Nick:	e-mail:	Perfil:	Editar:	Eliminar:
1	admin	Admin	admin@test.cl	Administrador	Editar	Eliminar
2	larcon	Naitsirc	naitsirc@gmail.com	Narrador	Editar	Eliminar

Naitsirc, Inc 2023.

Home

Síguenos en nuestras RR.SS.

API REST

20.- Creación servicio Rest

De momento tengo solo la Clase Controladora ListarJController, la cual permite a alguien acceder a el listado de los jugadores inscritos a mesas en formato JSON al acceder a través de la dirección /ListarJ y ver la información de los jugadores en forma individual a través de cada numeral (ej: /ListarJ/1)

```
1 package cl.vrol.controller;
2
3 import java.util.List;
4
5 @RestController
6 @RequestMapping("/listarJ")
7 public class ListarJController {
8
9     @Autowired
10     private IJugadorService playerService;
11
12     @GetMapping("/")
13     public List<Jugador> listarJugadores(Model model) {
14         List<Jugador> listaInscritos = playerService.listarTodos();
15         return listaInscritos;
16     }
17
18     @GetMapping("/{id}")
19     public Jugador jugadaorXId(@PathVariable("id") Long idInscrito){
20         Jugador player = playerService.buscarPorId(idInscrito);
21         return player;
22     }
23 }
```

Conclusión

Puedo decir que ha sido una Aventura, sin más, entre buscar información, documentarse, ir, intentar, probar, seguir probando, ver las innumerables veces que el código me ha estallado. Pero el resultado final me deja satisfecho. Hay mucho mas por hacer para seguir perfeccionando este proyecto muchas mas funcionalidades que le quiero agregar y mucho más. ¿Y quién sabe?, aprendiendo nuevas tecnologías, permitirme darle una vuelta mas tarde y convertirla en una app para celular.

A destacar debo dejar que hacer la lógica de la construcción de la inscripción de Jugadores y la de mesas es una que me hizo pasar bastante tiempo.