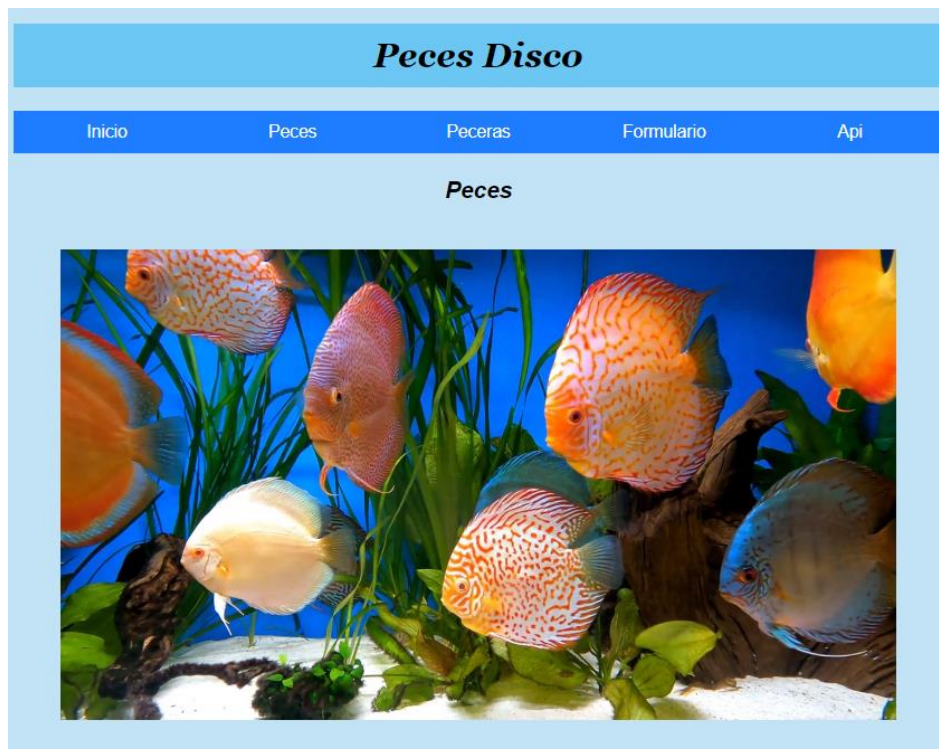


Acuario



Sprint 1

Carlos Marín, Santiago Sicilia, Gonzalo de Alfonso y Jaime Landa

abril del 2024

Contents

SPRINT BACKLOG	3
DIAGRAMA DE CASOS DE USO	3
API REST.....	4
BASES DE DATOS.....	5
DEPENDENCIAS	6
WORKFLOW CI/CD	¡Error! Marcador no definido.

SPRINT BACKLOG

Tras algunas reuniones de equipo, se decidió que, para añadir los últimos conceptos vistos en clase, estos se realizarían sobre la web de acuario. En esta web el usuario tiene acceso a ver los diferentes peces y tamaños de peceras y se han querido añadir una serie de funcionalidades extra, como funcionalidades de login o registro y la funcionalidad de añadir peces a una pecera. La funcionalidad que se va a llevar a cabo en este sprint es la siguiente.

- Usuario y token.

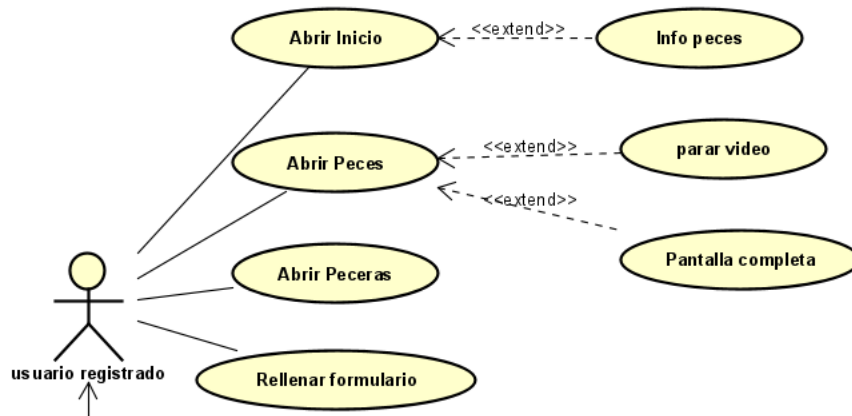
El reparto de las tareas es el siguiente:

- Backend:
 - Jaime y Gonzalo.
 - Controlador.
 - Entidades.
 - Modelo.
 - Repositorios.
 - Servicios.
 - Útil.
- Frontend:
 - Carlos y Yago.
 - Html.
 - Js.
 - Test.
 - Css.

DIAGRAMA DE CASOS DE USO

El diagrama de casos de uso es una forma de diagrama de comportamiento en lenguaje de modelado unificado (UML), con la que se representan los casos de uso, actores y sus relaciones. En primer lugar, para acceder a la página web no será necesario registrarse, esto sirve como “gancho” para atraer a nuevos usuarios. Sin registrarse el usuario no podrá acceder a todas las funcionalidades de la página. El usuario no registrado podrá visualizar el índice, los peces, las peceras y rellenar el formulario. Este sprint se ha centralizado en que las funciones de login y registro funcionen, sin embargo, los usuarios registrados en este sprint no tendrán ninguna funcionalidad adicional, esta se implementará en el siguiente sprint. El usuario se podrá registrar como usuario o administrador.

A continuación, se muestra un diagrama detallado sobre los casos de uso:

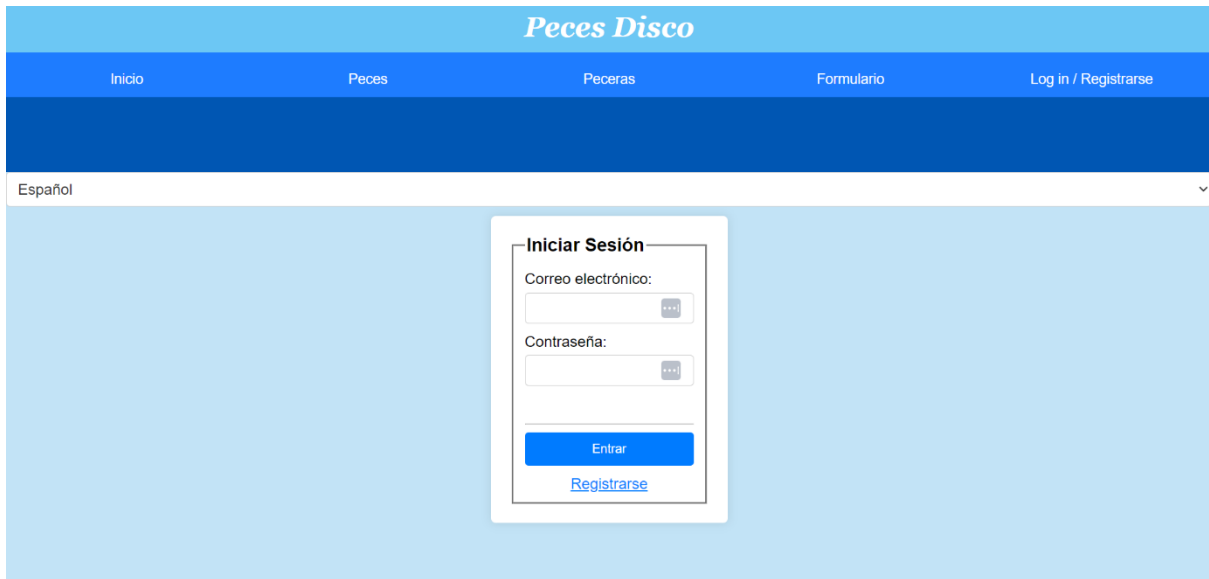


API REST

El Api rest del primer sprint:

@PostMapping("/api/users"): permite a los usuarios registrarse en la web. Este campo requiere nombre, rol, email, contraseña y repetir contraseña.

@PostMapping("/api/users/me/session") : permite a los usuarios registrarse en la web. Este campo requiere nombre, rol, email, contraseña y repetir contraseña.

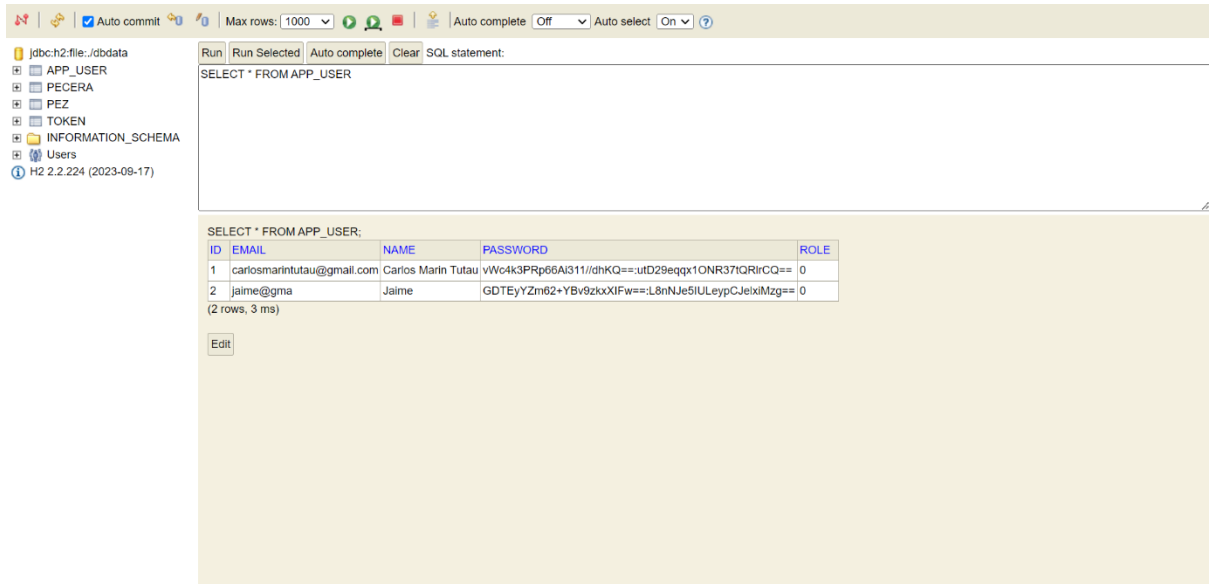


BASES DE DATOS

En este caso se ha realizado una base de datos relacional, ya que es la que mejor se adapta a nuestro proyecto, nuestra base de datos tendrá las siguientes tablas:

- Usuario.
- Token.

La relación entre usuario y token es OneToOne.



ID	EMAIL	NAME	PASSWORD	ROLE
1	carlosmarintutau@gmail.com	Carlos Marín Tutau	vWc4k3PRp66AI311//dhKQ==:utD29eqqx1ONR37iQRirCQ==	0
2	jjaime@gma	Jaime	GDTEyYZm62+YBv9zxxIFw==:L8nNJe5IULeypCJelxiMzg==	0

(2 rows, 3 ms)

Como se puede comprobar los usuarios quedan registrados de manera correcta.

DEPENDENCIAS

Las dependencias más relevantes utilizadas en el proyecto son las siguientes:

```
<dependency>
  <groupId>com.h2database</groupId>
  <artifactId>h2</artifactId>
  <scope>runtime</scope>
</dependency>
```

Esta primera dependencia nos permite trabajar con bases de datos y poder almacenar la información de los usuarios

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

Esta segunda dependencia permite realizar diferentes pruebas para verificar que no ocurre ningún error.