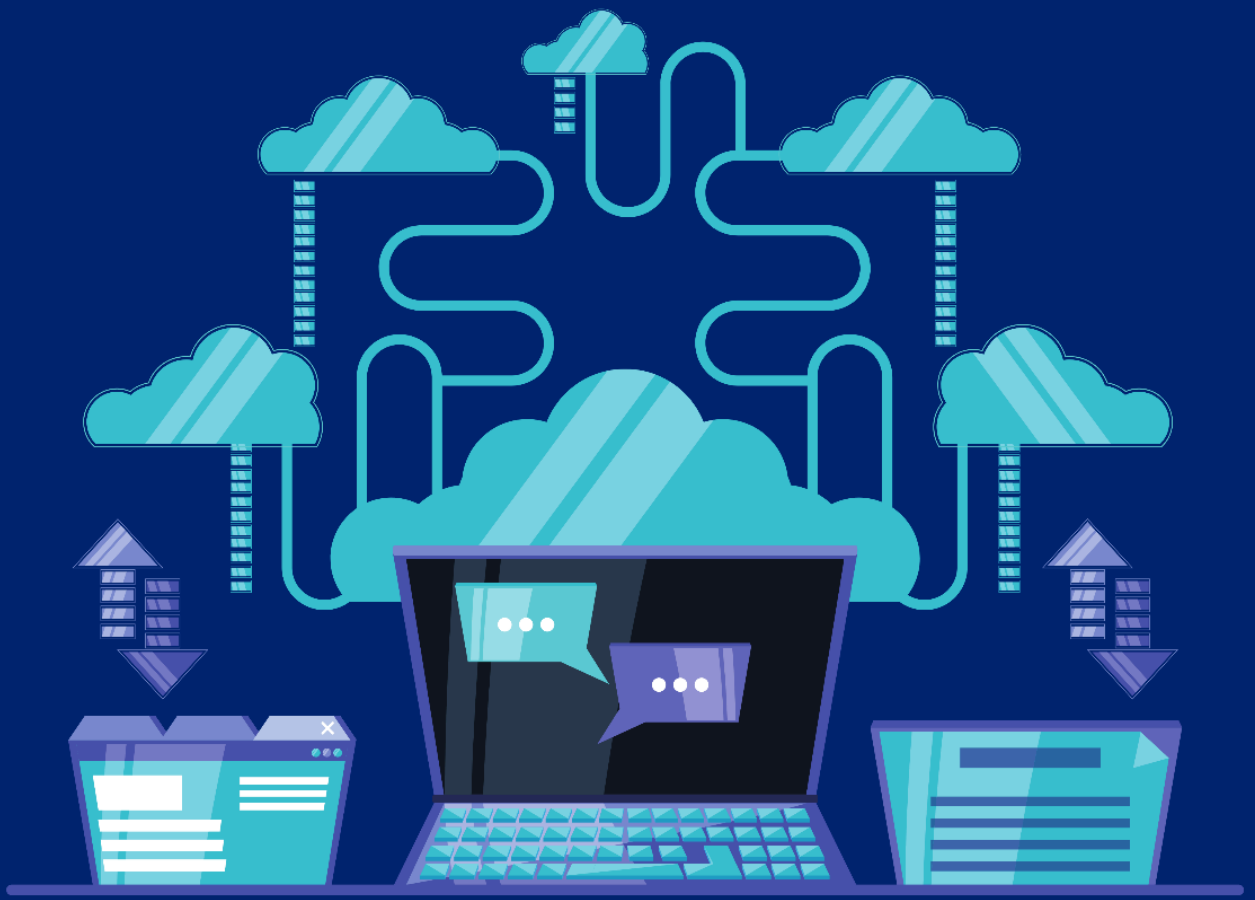
PROGRMACIÓN WEB

»»» INFORME «««

PRACTICA 2



Mohssin Bassat Sidki, Javier Castilla Arroyo,
María José García Aragón y Alejandro Gómez Amaro

GRUPO 1 GM 3

Índice

Repositorio en GitHub → https://github.com/alegomezamaro/pw2526-gm3-1

1. Introducción

El objetivo de esta segunda práctica ha sido ampliar la funcionalidad del sistema web del Club Náutico desarrollado en la Práctica 1, implementando una **API REST**. Esta interfaz permite el acceso programático a los recursos del sistema (Socios, Embarcaciones, Alquileres, Reservas) mediante los verbos HTTP estándar (GET, POST, PUT, PATCH, DELETE), facilitando la integración con sistemas externos o clientes automatizados.

Además, se ha aprovechado esta iteración para refactorizar la base de datos y el código base, corrigiendo las deficiencias de diseño y consistencia detectadas en la evaluación de la entrega anterior.

2. Diseño y estructura del proyecto

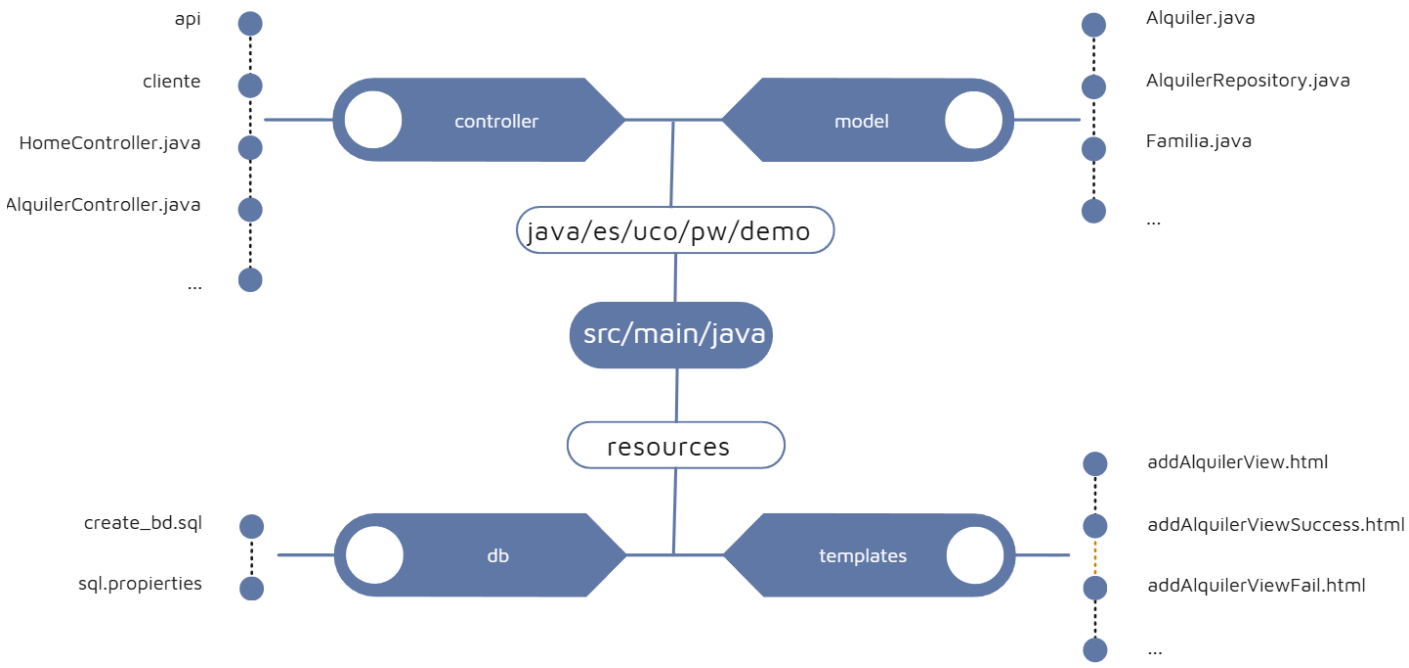
La estructura del proyecto se basó en tres paquetes principales:

1. **controller** → Está dividido en 3. Tipos:
    1. Los controladores MVC con los datos para las vistas de la práctica 1.
    2. Los controladores API creados para la práctica 2.
    3. Controladores cliente de ejemplo para API.

2. **model** → Contiene las clases de dominio (Socio, Embarcación, Inscripción, etc.) y sus respectivos repositorios.

3. **db** → Contiene la base de datos SQL
4. **templates** → Contiene todas las vistas Thymeleaf utilizadas por el usuario final.

Podemos verlo en el mapa de navegación:



2.1. Modelo de dominio

Cada entidad del proyecto representa un elemento gestionado por el club náutico:

- **Alquiler** → Registra la matrícula de la embarcación, el dni del titular, la fecha de inicio y fin del alquiler, las plazas solicitadas y el precio total.
- **Embarcación** → Recoge la matrícula, nombre, tipo de embarcación, plazas, dimensiones y patrón asignado (Si lo hay) de la embarcación.
- **Familia** → Agrupa el identificador de la familia, el dni del titular, el número de adultos y el de niños.
- **Inscripción** → Mantiene información sobre la cuota (individual o familiar), fecha de inscripción, dni del titular y el identificador de la familia asociada.
- **Patrón** → Almacena el DNI, nombre y apellidos, fecha de nacimiento y fecha en la que obtuvo el título de patrón.

- **Reserva** → Vincula al socio solicitante con una embarcación que tenga patrón asignado, detallando la fecha y el motivo del evento
- **Socio** → Incluye atributos como DNI, nombre, apellidos, dirección, fecha de nacimiento, fecha de alta y un booleano que indica si es patrón.

3. Dificultades encontradas

1. Inconsistencias entre modelo Java, BD y sql.properties.

Había consultas/updates que referían columnas que no existían en el modelo ('socioSolicitante', 'matricula') o nombres distintos ('plazasreserva' vs 'plazasReserva', 'fechareserva' vs 'fechaReserva'). Eso rompía endpoints que antes funcionaban o hacía que métodos devolvieran resultados inesperados.

2. Diferencia entre "la sentencia se ejecuta" y "filas afectadas".

En algunos UPDATE el rows > 0 podía dar false aunque no hubiese error real (o aunque el cambio fuera "igual"). Eso generaba respuestas 500 engañosas; lo correcto fue basarse en excepción/no excepción o controlar mejor los casos.

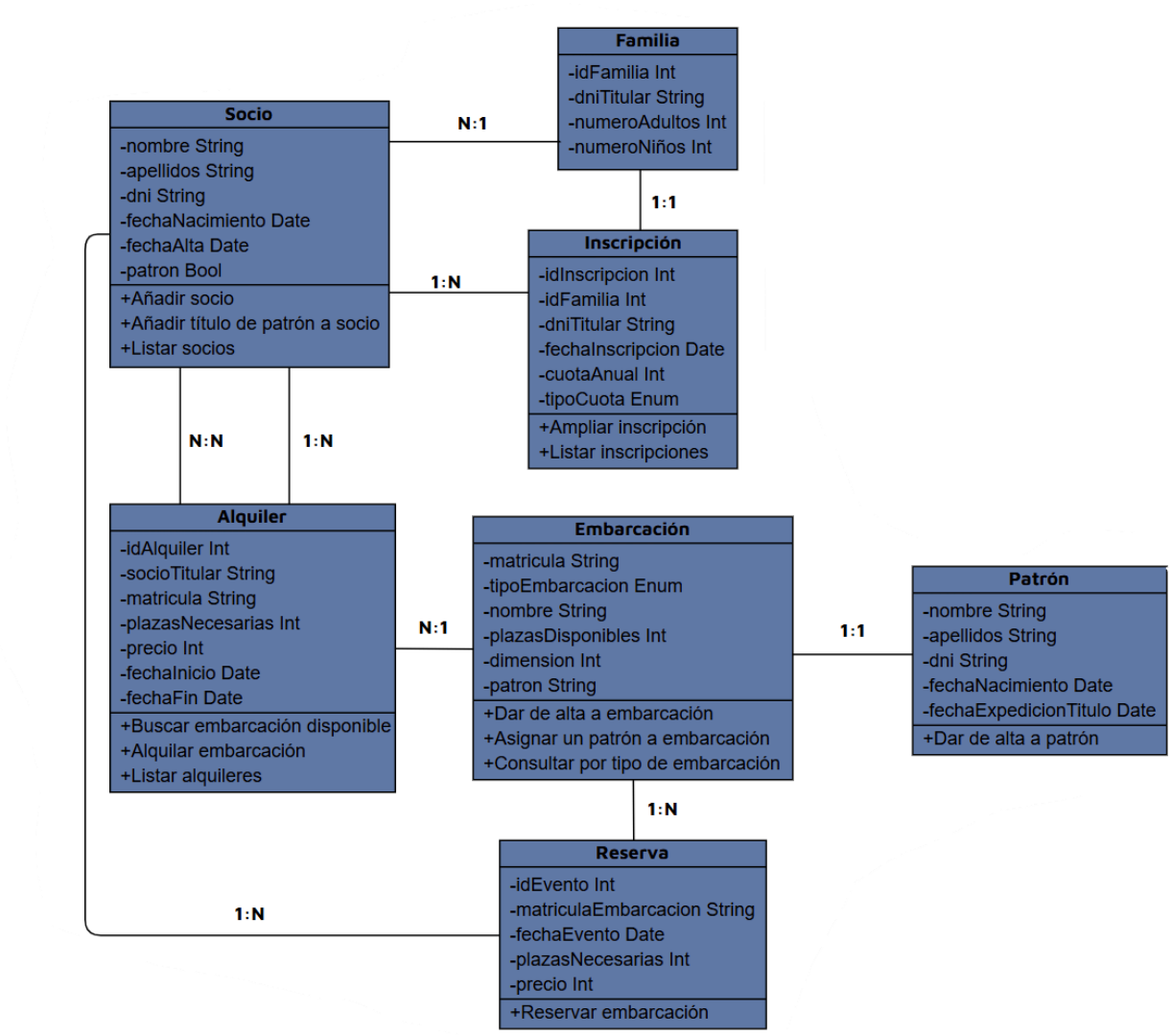
3. Funcionalidades del enunciado no reflejadas en el dominio actual.

El enunciado pedía "descripción" en reservas, pero el modelo/tabla Reserva no lo tenía. Hubo que adaptar el requisito a lo implementable (plazas + precio) o decidir ampliación del modelo si se quisiera cubrirlo literal.

4. Definición de clases

El diagrama representa el modelo de clases de un sistema para la gestión integral de un **Club NautiUCO**.

La clase **Socio** es la entidad central, almacenando datos personales y permitiendo operaciones como añadir socios o asignarles el título de patrón. La clase **Familia** vincula a un socio titular con otros miembros mediante el número de adultos y niños. La clase **Inscripción** registra las cuotas anuales, su tipo y la relación con socios o familias. El módulo de navegación se estructura en las clases **Patrón** y **Embarcación**, que permiten gestionar los trabajadores y asignarlos a embarcaciones. El sistema también incorpora las clases **Alquiler** y **Reserva**, que gestionan la disponibilidad de embarcaciones, las fechas de uso y el coste asociado.



En conjunto, el diagrama refleja un sistema modular y bien organizado, donde cada clase cumple una función específica dentro de la operativa del club.