**Servidor Java**

**Tecnologías**

Dado que tu proyecto original estaba en **Java**, podemos usar un stack moderno y sencillo:

* **Lenguaje:** Java
* **Framework:** Spring Boot (muy bueno para REST y fácil de escalar)
* **Base de datos:** MySQL o PostgreSQL (según prefieras)
* **ORM (Object Relational Mapping):** Hibernate (ya lo conoces)
* **Formato de datos:** JSON (ideal para comunicar con React)
* **Seguridad:** Spring Security (para autenticación y roles)

**2️⃣ Arquitectura básica**

**Front ↔ Controlador ↔Servicio ↔ Repositorio ↔ Modelo**

**Modelo → DTO → Servicio → Repositorio → Controlador**

* **Modelo:** Clases Java que representan las **tablas de la base de datos**. Cada atributo de la clase corresponde a una columna.Hibernate (o JPA) convierte estas clases en registros de la base de datos y maneja la persistencia automáticamente.
* **DTO:** objetos que intercambias con el front. Evita exponer directamente las entidades del modelo y permite: Validaciones, transformaciones y evita que el front vea campos sensibles.
* **Servicio:** lógica de negocio. Sirve para mantener el control sobre operaciones complejas, reutilizar el código y separar la lógica del controlador, que solo recibe y devuelve datos.
* **Repositorio:** Interfaz que se comunica directamente conla base de datos usando Hibernate/JPA. Encapsula operaciones CRUD y evita escribir SQL a mano. Spring Boot ya implementa estos métodos CRUD.
* **Controlador:** Exposición de los endpoints REST que el front va a consumir. Maneja las utilidades HTTP (Get, Post, Put, Del), llama al servicio para ejecutar la lógica y devuelve DTOs al front.

**3️⃣ Entidades principales**

Para Work O’Clock, mínimo necesitas:

* **Empleados**
  + id, nombre, apelidos, email, contraseña(hasheada), fecha alta, activo
* **Roles**
  + Id, nombre, descripción
* **Empleados\_Roles**
  + Empleado\_id, rol\_id
* **Jornada**
  + id, empleado\_id, hora\_inicio, hora\_fin, días\_semana
* **Fichajes**
  + id, empleado\_id, tipo, fichaje, latitud, longitud
* **Auditoría**
  + id, empleado\_id, accion, descripcion, fecha, ip

**4️⃣ Endpoints básicos**

1. **Empleados**
   * **GET: /empleados: Listar todos los empleados**
   * **GET: /empleados/{id}: Obtener un empleado por ID**
   * **POST: /empleados: Crear un nuevo empleado**
   * **PUT: /empleados/{id}: Actualizar datos de un empleado**
   * **DELETE: /empleados/{id}: Desactivar o eliminar un empleado**
2. **Empleados Roles**
   * **GET: /empleados/{id}/roles: Listar roles de un empleado**
   * **POST: /empleados/{id}/roles: Asignar uno o varios roles a un empleado**
   * **DELETE: /empleados/{id}/roles/{rol\_id}: Quitar un rol a un empleado**
3. **Roles**
   * **GET: /roles: Listar todos los roles**
   * **GET: /roles/{id}: Obtener un rol por ID**
   * **POST: /roles: Crear un nuevo rol**
   * **PUT: /roles/{id}: Actualizar un rol**
   * **DELETE: /roles/{id}: Eliminar un rol**
4. **Jornadas**
   * **GET: /jornadas: Listar todas las jornadas**
   * **GET: /jornadas/{id}: Obtener jornada por ID**
   * **GET: /empleados/{id}/jornadas: Obtener jornadas de un empleado**
   * **POST: /jornadas: Crear una nueva jornada**
   * **PUT: /jornadas/{id}: Actualizar jornada**
   * **DELETE: /jornadas/{id}: Eliminar jornada**
5. **Fichajes**
   * **GET: /fichajes: Listar todos los fichajes**
   * **GET: /fichajes/{id}: Obtener un fichaje por ID**
   * **GET: /empleados/{id}/fichajes: Listar fichajes de un empleado**
   * **GET: /empleados/{id}/fichajes/ultimo: Obtener último fichaje de un empleado**
   * **POST: /fichajes: Registrar un fichaje (entrada/salida)**
   * **PUT: /fichajes/{id}: Modificar un fichaje**
   * **DELETE: /fichajes/{id}: Eliminar un fichaje**
6. **Auditoría**
   * **GET: /auditoria: Listar todas las acciones registradas**
   * **GET: /empleados/{id}/auditoria: Listar auditoría de un empleado**
   * **POST: /auditoria: Registrar una acción manualmente**
7. **Autenticación**
   * **POST: /auth/login: Autenticación por email/contraseña**
   * **POST: /auth/logout: Cerrar sesión**
   * **POST: /auth/refresh: Renovar token de sesión**
8. **Reportes / Extras**
   * **GET: /empleados/{id}/estado: Estado actual (dentro/fuera según último fichaje)**
   * **GET: /reportes/fichajes: Generar informes de fichajes por empleado o fechas**
   * **GET: /reportes/jornadas: Comparar horarios planificados vs fichajes reales**