

Documentación Scrum

Práctica Final de Curso: Desarrollo de una aplicación Java con MySQL y Scrum





27 de junio de 2025

INDRA Minsait

**PLANIFICACIÓ EN 4 DIES (28 hores totals)**

**✅ DIA 1 – Estructura inicial, model ER, taules i CRUD bàsic (7h)**

**Objectiu general**: Posar les bases del projecte (estructura, base de dades, models i CRUD bàsic).

**Tareas:**

| **Temps** | **Tasca** |
| --- | --- |
| 1h | Crear projecte Maven amb estructura (pom.xml) |
| 1h | Dissenyar el model ER amb [dbdiagram.io](https://dbdiagram.io/) |
| 1h | Crear database.sql amb taules (Sala, Empleado, Reserva) i dades de prova |
| 1h | Implementar ConexionMySQL.java amb JDBC |
| 1.5h | Crear classes Sala, Empleado, Reserva |
| 1.5h | Crear DAO per a Sala i Empleado (CRUD complet) |

**Commits recomanats:**

* feat: estructura inicial del proyecto con Maven
* feat: script SQL con tablas y datos de prueba
* feat: modelo y DAO para salas y empleados

**✅ DIA 2 – CRUD Reserva + Control de solapaments (7h)**

**Objectiu general**: Implementar les funcionalitats de reserva i el control de conflictes horaris.

**Tareas:**

| **Temps** | **Tasca** |
| --- | --- |
| 2h | Crear DAO ReservaDAO amb CRUD complet |
| 2h | Implementar control de solapaments (verificar que no hi ha conflictes de data i hora) |
| 1.5h | Millorar els DAOs i afegir validacions |
| 1.5h | Provar manualment amb classe Main.java (menú simple per CRUDs) |

**Commits recomanats:**

* feat: CRUD de reservas implementado
* feat: verificación de solapamiento de reservas
* fix: validaciones y pruebas básicas con menú

**✅ DIA 3 – Testing i documentació Scrum (7h)**

**Objectiu general**: Afegir proves unitàries i preparar la documentació Scrum.

**Tareas:**

| **Temps** | **Tasca** |
| --- | --- |
| 2h | Escriure 5 proves unitàries amb JUnit (DAO i solapaments) |
| 2h | Crear README.md amb instruccions d’execució i DB |
| 2h | Escriure el document Documentacion\_Scrum\_NomCognoms.docx amb 2 sprints, històries, reflexions |
| 1h | Organitzar carpeta /documentacion, revisar projectes i pujar a GitHub |

**💬 Commits recomanats:**

* test: pruebas unitarias con JUnit
* docs: añadido README y documentación Scrum Sprint 1 y 2

**✅ DIA 4 – Revisió final, millores i neteja (7h)**

**Objectiu general**: Millorar, revisar i finalitzar el projecte per entregar.

**Tareas:**

| **Temps** | **Tasca** |
| --- | --- |
| 2h | Revisió completa del codi, neteja i refactorització |
| 1h | Millorar format del README i documentació |
| 1.5h | Afegir comentaris al codi i revisar bones pràctiques |
| 1h | Crear .gitignore, netejar arxius innecessaris |
| 1.5h | Commits finals ben comentats, comprovació execució final i entrega GitHub |

**Commits recomanats:**

* refactor: limpieza de código y mejoras finales
* chore: preparado el proyecto para entrega final

**Recomanacions extra:**

* **Trello o Notion**: crea un petit tauler Kanban (To Do, In Progress, Done) per fer la part Scrum visual.
* **Commits significatius**: fes 2-3 per dia, ben separats per funcionalitat o bloc de treball.
* **README atractiu**: que expliqui què fa el projecte i com executar-lo pas a pas.
* **No esperis a l’últim dia per pujar tot a GitHub**: puja canvis cada dia!

**Sprint 1: Planificación y Configuración del Proyecto**

**Objetivos del Sprint 1**

* Crear la estructura inicial del proyecto con Maven.
* Diseñar el modelo entidad-relación.
* Crear el script SQL con las tablas y datos de prueba.
* Implementar las operaciones CRUD para Salas y Empleados.
* Configurar Git y GitHub correctamente.

### 📖 Historias de Usuario y Tareas Técnicas

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | **Tareas Técnicas** | **Estimación** |
| **HU01 - Alta de sala:** Como administrador quiero dar de alta nuevas salas para que se puedan reservar | - Crear clase Sala- Crear tabla sala en SQL- Implementar método crearSala() en DAO- Prueba de inserción | 5 pts |
| **HU02 - Listado de salas:** Como empleado quiero ver un listado de salas para elegir cuál reservar | - Implementar método getAllSalas() en DAO- Crear método para mostrar lista por consola o interfaz | 3 pts |
| **HU03 - Modificar sala:** Como administrador quiero modificar los datos de una sala para mantener la información actualizada | - Implementar método updateSala() en DAO- Validaciones básicas | 2 pts |
| **HU04 - Eliminar sala:** Como administrador quiero eliminar una sala para que no pueda reservarse | - Implementar método deleteSala() en DAO | 1 pt |
| **HU05 - Alta de empleado:** Como administrador quiero registrar empleados para que puedan reservar salas | - Crear clase Empleado- Crear tabla empleado en SQL- Método crearEmpleado() | 4 pts |
| **HU06 - Listado de empleados:** Como administrador quiero ver empleados registrados para gestionarlos | - Método getAllEmpleados()- Visualización | 2 pts |
| **HU07 - Modificar empleado:** Como administrador quiero modificar los datos de una sala para mantener la información actualizada | - Implementar método updateEmpleado() en DAO- Validaciones básicas | 2 pts |
| **HU08 - Eliminar empleado:** Como administrador quiero eliminar una sala para que no pueda reservarse | - Implementar método deleteEmpleado() en DAO | 1 pt |
| **Setup técnico inicial** | - Inicializar proyecto Maven (pom.xml)- Crear clase de conexión JDBC- Diseñar diagrama ER en dbdiagram.io Crear script database.sql- Crear repositorio GitHub- Hacer commits frecuentes | 6 pts |

**Reflexión Sprint 1**

Se completaron todas las tareas planificadas. La configuración del proyecto y la base de datos fueron exitosas. Se detectaron pequeños errores en validaciones que se corrigieron al final del día.

**Sprint 2: Funcionalidad de Reservas y Testing**

**Objetivos del Sprint 2**

* Implementar la funcionalidad de reservas con control de solapamientos.
* Realizar pruebas unitarias con JUnit.
* Documentar los sprints con enfoque Scrum.
* Finalizar el README.md y subir el proyecto completo a GitHub.

### **📖 Historias de Usuario y Tareas Técnicas**

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | **Tareas Técnicas** | **Estimación** |
| **HU07 - Crear reserva:** Como empleado quiero reservar una sala indicando fecha y hora para organizar una reunión | - Crear clase Reserva- Crear tabla reserva en SQL- Implementar método crearReserva()- Interfaz o consola para introducir datos | 5 pts |
| **HU08 - Evitar solapamientos:** Como empleado quiero que el sistema impida reservar una sala ocupada para evitar conflictos | - Añadir lógica en DAO para validar disponibilidad- Consulta SQL con comprobación de horario- Lanzar error o mensaje informativo | 4 pts |
| **HU09 - Ver mis reservas:** Como empleado quiero consultar mis reservas para saber qué tengo programado | - Método getReservasPorEmpleado()- Mostrar resultados en consola/interfaz | 2 pts |
| **HU10 - Cancelar reserva:** Como empleado quiero cancelar una reserva para liberar la sala | - Método deleteReserva() en DAO- Confirmación de acción | 2 pts |
| **Testing** | - Crear mínimo 5 pruebas unitarias con JUnit:• DAO de salas• DAO de empleados• DAO de reservas• Comprobación de solapamiento• Inserción correcta | 5 pts |
| **Documentación y entrega final** | - Redactar Documentacion\_Scrum\_NombreApellidos.docx- Crear README.md claro y completo- Subir proyecto completo a GitHub- Asegurar historial limpio de commits | 5 pts |

**Reflexión Sprint 2**

Se implementaron correctamente las reservas y las pruebas unitarias. El control de solapamientos presentó un reto interesante resuelto con consultas SQL. La documentación fue completada según las indicaciones.