

UNIVERSIDAD TECNOLÓGICA NACIONAL FACULTAD REGIONAL MAR DEL PLATA

PROGRAMACION III – COMISION 6 Y 7 DOCENTE: DANIEL DIAZ

Trabajo Práctico Final – Programación III

Consigna General

El presente trabajo práctico tiene como objetivo principal el desarrollo de una **API RESTful** utilizando **Spring Boot** y **Java**. La temática del sistema es **libre** (puede ser un sistema de gestión, inventario, reservas, etc.), pero debe cumplir con los requisitos funcionales, no funcionales y técnicos que se detallan a continuación.

Requisitos

1. Funcionales

- Se debe implementar un CRUD básico (Create, Read, Update, Delete) sobre al menos una entidad principal del sistema.
- El sistema deberá contemplar entre **5 y 10 requisitos funcionales** que describan las operaciones esperadas, tales como:
 - Alta, baja y modificación de registros.
 - Listados filtrados o búsquedas.
 - Asociación entre entidades (por ejemplo, relaciones uno a muchos o muchos a muchos).
 - Acciones específicas según el tipo de usuario o rol, si aplica.

Los requisitos funcionales deben estar claramente documentados y ser aprobados antes de comenzar el desarrollo.

2. No Funcionales

- El sistema debe contemplar 5 requisitos no funcionales, como:
 - Asegurar que los datos sensibles (como contraseñas) estén encriptados en la base de datos.
 - Uso de arquitectura en capas (Controller, Service, Repository).
 - Implementación de estándares REST (nombres de endpoints, métodos HTTP adecuados, etc.).
 - Autenticación y autorización segura mediante JWT.
 - La API debe estar documentada de forma clara mediante OpenAPI/Swagger.



UNIVERSIDAD TECNOLÓGICA NACIONAL FACULTAD REGIONAL MAR DEL PLATA

PROGRAMACION III – COMISION 6 Y 7 DOCENTE: DANIEL DIAZ

3. Tecnologías y Herramientas Obligatorias

Se debe utilizar **obligatoriamente**:

- Spring Boot Web: para la creación de los controladores y endpoints REST.
- Spring Data JPA: para la gestión de la persistencia de datos con MySQL como motor de base de datos.
- Spring Security con JWT: para la implementación de autenticación y autorización segura.
- Manejo de excepciones de manera centralizada utilizando controladores de excepciones (@ControllerAdvice).
- Validaciones de entrada de datos utilizando anotaciones (@Valid, @NotNull, @Size, etc.).
- Sistema de control de versiones Git y repositorio en GitHub.

4. Modalidad de Trabajo

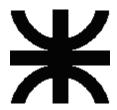
• El proyecto debe ser realizado de manera **colaborativa** en grupos de **hasta 5 integrantes** como máximo.

Se recomienda que los equipos adopten alguna **metodología ágil** de trabajo, como por ejemplo **Scrum** o **Kanban**, para facilitar la organización de tareas, mejorar la comunicación y fomentar la colaboración.

Algunas prácticas sugeridas son:.

- Dividir el trabajo en tareas pequeñas (issues) y asignarlas claramente a los integrantes.
- Planificar reuniones breves (dailys o encuentros periódicos) para revisar avances, obstáculos y próximas actividades.
- **Utilizar tableros Kanban** (por ejemplo, en GitHub Projects, Trello o Jira) para visualizar el estado de las tareas y facilitar el seguimiento del proyecto.
- Registrar las decisiones importantes y documentar los cambios relevantes durante el desarrollo.

El objetivo de aplicar estas prácticas es favorecer un desarrollo ordenado, responsable y eficiente, simulando dinámicas de trabajo reales en proyectos de software profesionales.



UNIVERSIDAD TECNOLÓGICA NACIONAL FACULTAD REGIONAL MAR DEL PLATA

PROGRAMACION III – COMISION 6 Y 7 DOCENTE: DANIEL DIAZ

5. Entregables

- Repositorio público o privado en GitHub con:
 - Código fuente completo y organizado.
 - o Archivo de documentación de OpenAPI (o Swagger UI configurado en el proyecto).
- Video demostrativo (opcional pero recomendado) presentando el funcionamiento general de la API.

6. Entregas

• Primera entrega: 13/06

• Segunda entrega (Recuperatorio): 26/06