Trabajo Práctico Nº 3

Despliegue y Clases

1. Objetivo

- Establecer el puente entre los modelos de arquitectura lógica (componentes y clases) y los modelos de infraestructura física (despliegue).

2. Fechas de trabajo

Semana del 06/10/2025 al 13/10/2025, inclusive.

3. Contextualización y estado de situación

A esta altura, los grupos ya:

- Identificaron requerimientos funcionales y no funcionales.
- Definieron una arquitectura preliminar de alto nivel con el modelo C4 (al menos hasta
  la vista de componentes).
- Modelaron comportamiento con diagramas de secuencia.

El siguiente paso es profundizar en el diseño detallado de su solución, vinculando:

- UML de clases (estructura interna de los componentes),
- UML de despliegue (infraestructura y asignación de software a hardware).

4. Directivas

1. Revisión teórica. Repasar los conceptos de:
   - ○ Diagrama de clases UML.
   - ○ Diagrama de despliegue UML.

2. Ejercicio de análisis
   - ○ Explicar qué comunica cada uno de estos diagramas y qué diferencia hay entre ellos.

Criterio	Diagrama de Clases UML	Diagrama de Despliegue UML
Tipo de diagrama	Estructural (lógico)	Estructural (físico)
Propósito principal	Representar la estructura interna del sistema	Representar la arquitectura física del sistema
Representa	clases, atributos, métodos y relacione	hardware, nodos y componentes desplegados
Explica	Cómo se organiza el código y cómo se relacionan las clases	Dónde se ejecutan los componentes del software y cómo se conectan físicamente
Nivel de abstracción	Lógico / de diseño de software	Físico / de implementación e infraestructura
Momento de uso	Durante el análisis y diseño del sistema	Durante la implementación, pruebas o despliegue del sistema
Ventajas	Permite entender la estructura y responsabilidades del código	Permite planificar la instalación, comunicación y operación del sistema.

   - ○ Justificar por qué es importante mantener la trazabilidad entre C4 → Clases
   → Despliegue.

Mantener la trazabilidad en el modelo C4 es importante porque permite conectar el diseño con la implementación, asegurando que lo que se planifica se cumpla en el código y en el despliegue.
Además, facilita la comunicación entre los equipos, ayuda a detectar impactos de cambios y mantiene la coherencia entre todas las partes del sistema.

3. Aplicación a su proyecto
   A partir de la solución que se viene diseñando:
   - ○ C4 (revisión): ajustar o completar su vista de componentes.
   - ○ Clases: para al menos tres componentes críticos (ej. autenticación, gestión de datos, procesamiento principal), desarrollar el diagrama de clases detallado. → tres componentes críticos
   - ○ Despliegue: representar cómo los contenedores identificados en C4 se distribuyen en nodos físicos/lógicos de ejecución (servidores, bases de datos, clientes, etc.).

4. Relación entre modelos
   Incluir una breve explicación donde muestren explícitamente:
   - ○ Cómo los componentes de C4 se traducen en clases UML.
   - ○ Cómo esos mismos componentes/clases se asignan en el diagrama de despliegue.

En el modelo C4, los componentes de alto nivel se traducen directamente en clases UML que reflejan sus responsabilidades. Los controladores del diagrama C4 (como Authentication Controller o DICOM File Controller) se convierten en clases con estereotipo Controller —por ejemplo, LoginController y MainViewController—, que gestionan la interacción con el usuario y delegan tareas a servicios. Por su parte, los servicios del modelo C4 (como Authentication Service, MCS Calculation Service o Report Service) se implementan como clases Service en UML, como AuthenticationService, McsCalculatorService y ResultHistoryService, encapsulando la lógica de negocio específica de cada dominio.

Las entidades del dominio, aunque no siempre visibles como componentes en C4, aparecen claramente en UML como clases con estereotipo Entity o Enumeration. Ejemplos son User, Role, DicomFile y ProcessingResult, que representan los datos fundamentales que fluyen entre los servicios y persisten en el sistema. Estas entidades son manipuladas por los servicios y referenciadas por los controladores, manteniendo una clara separación de responsabilidades acorde con los principios del diseño orientado a objetos.

Finalmente, aspectos transversales como el registro de eventos o la gestión de errores, representados en C4 como Error Logging Service, se materializan en UML mediante clases como LoggerService, utilizadas por múltiples componentes. Aunque los repositorios o integraciones externas (como Hospital API Client) no aparecen como clases independientes, su funcionalidad está implícita en los servicios —por ejemplo,

AuthenticationService gestiona usuarios en memoria, simulando el acceso a un repositorio—, lo que permite una transición coherente desde la arquitectura conceptual (C4) hasta el diseño detallado (UML).
En el diagrama de despliegue, se asignó cada componente y clase del modelo UML a su correspondiente infraestructura real: toda la lógica de la aplicación —incluyendo los controladores como LoginController y MainViewController, los servicios como AuthenticationService, ProcessingOrchestrator y LoggerService, y las entidades como User y ProcessingResult— se ejecuta dentro de la aplicación backend (implementada en Java Spring Boot). Esta aplicación es accesible para los usuarios finales mediante un navegador web, cuyas peticiones HTTPS son gestionadas por Nginx, que actúa como proxy inverso. Para la persistencia, el sistema se conecta a dos recursos externos: una base de datos PostgreSQL, donde se almacenan datos estructurados como usuarios, roles y resultados de procesamiento, y un almacenamiento NFS local, que guarda los archivos DICOM cargados y los logs generados. Esta distribución refleja una arquitectura limpia y escalable, en la que la aplicación es stateless y delega el almacenamiento a sistemas especializados, manteniendo coherencia entre el diseño lógico (C4/UML) y su implementación física.

5. Entregables

- Informe en PDF con:
    1. Explicación breve de cada tipo de diagrama.
    2. Diagramas elaborados (C4 – componentes, UML clases, UML despliegue).
    3. Relación y justificación entre los tres niveles.