Técnicas Avanzadas de la Programación

"Spring Nro. 4"

Alumno: Leandro Pasteur

Profesora: Antonieta Kuz

Fecha: 09/05/2025

Índice:

I. Diagrama de clases UML "Jardín Maternal"

El propósito de este diagrama es modelar las entidades clave del sistema web del jardín maternal "Luz de luna", representando las clases, atributos, métodos y relaciones entre los actores del sistema.

El diagrama de Clases UML fue realizado con la aplicación Miro, a continuación, dejamos el link de acceso:

https://miro.com/app/board/uXjVI8EG0ak=/

El diagrama fue organizado en una disposición horizontal) para mejorar la legibilidad y evitar el cruce de flechas. Se definieron los siguientes grupos principales:

1) Clase Usuario:

Clase para autenticar y administrar usuarios del sistema.

Atributos	Métodos
-ID:int -nombreUsuario: string -contraseña:string -fechaRegistro:date -estadoCuenta:string	+iniciarSesion() +cerrarSesion() +recuperarContraseña() +actualizarContraseña()

Relaciones:

- Herencia: Es la clase base generalizada de Tutor, Maestra y Administrador, quienes heredan sus atributos y métodos básicos de autenticación.

2) Clase Tutor: (Padre/Madre) – Tipo: Entidad.

Atributos	Métodos
-ID:int -nombre:string -apellido:string -DNI:int -domicilio:string -telefono:int -mail:string	+consultarPagos() +consultarMensajes() +consultarRutina() +consultarAsistenciaYSalud() +descargarReporteSemanal() +pagarCuota() +editarPerfil()

Relaciones:

- **Herencia:** Hereda de Usuario.

- **Composición:** Compone a Niño (1..1) → Cada niño debe estar vinculado a un único tutor.

- **Agregación:** Agrega a Pago (0..*) → Un tutor puede tener múltiples pagos, que pueden subsistir sin él.

- Asociación: Se asocia con Mensaje (0..*) → Puede enviar y recibir múltiples mensajes.

3) Clase Maestra: (Encargada de registrar actividades) – Tipo: Entidad.

Atributos	Métodos
-ID:int -nombre:string -apellido:string -DNI:int -domicilio:string -telefono:int -mail:string	+registrarAsistencia() +cargarRutinasDiarias() +registrarSalud() +verMensajes() +editarPerfil()

Relaciones:

-Herencia: Hereda de Usuario.

-Dependencia:

Depende de Inscripcion (0..1) → Puede estar involucrado en el proceso de inscripción.

Depende de RutinaDiaria y AsistenciaYSalud (1..1) → Es quien registra rutinas y controles diarios.

-Asociación: Se asocia con Mensaje (0..*).

4) Clase Administrador: Tipo: Entidad / Posible controlador.

Atributos	Métodos
-ID:int -nombre:string -mail:string -contraseña:string	+gestionarUsuarios() +administrarPagos() +administrarMensajes() +generarReportes()

Relaciones:

-Herencia: Hereda de Usuario.

-Asociación: Se asocia con Mensaje (0..*) → Participa en la comunicación institucional.

-Dependencia: Depende de Administrador (0..1) → Un administrador puede gestionar y supervisar los pagos realizados, sin ser parte directa del objeto

5) Clase Niño: Tipo: Entidad.

Atributos	Métodos
-ID:int -nombre:string -apellido:string -fechaNacimiento:date -grupoSala:string -tutorID:int	+obtenerHistorialSalud() +obtenerReporteDiario() +verHistorialAsistencias() +verPagosAsociados()

Relaciones:

-Composición:

Compuesto por Tutor (1..1) → Su existencia depende de un adulto responsable.

Compone a Inscripcion, RutinaDiaria, AsistenciaYSalud (1..1 cada una) → Elementos del historial diario del niño.

6) Clase Inscripcion: Tipo: Entidad.

Atributos	Métodos
-IDInscripción:int -niñoID:int -fechaInscripción:int -estado:string -observaciones:string	+crearInscripción() +modificarInscripción() +eliminarInscripción()

Relaciones:

-Composición: Compone a Niño (1..1) → Cada inscripción está ligada a un niño específico.

-Dependencia: Depende de Maestra (0..1) → Puede ser iniciada por una maestra.

7) Clase Rutina Diaria: Tipo: Entidad.

Atributos	Métodos
-ID:int -fecha:date -niñoID:int -alimentación:string -sueño:string -higiene:string -actividades:string	+registrarRutina() +consultarRutina() +generarReporteSemanal()

Relaciones:

-Composición: Compone a Niño (1..1) → Cada rutina está registrada para un niño.

-Dependencia: Depende de Maestra (1..1) → Solo una maestra puede registrar la rutina.

8) Clase Asistencia y Salud: Tipo: Entidad.

Atributos	Métodos
-ID:int -niñoID:int -fecha:date -presente:string	+registrarAsistencia() +registrarSalud() +generarFichaSalud()

Relaciones:

-Composición: Compone a Niño (1..1) → La información se registra por niño.

-Dependencia: Depende de Maestra (1..1) → Es registrada por maestras a cargo.

9) Clase Pago: Tipo: Entidad.

Atributos	Métodos
-ID:int -tutorID:int -monto:double -fechaPago:date -fechaVencimiento:date	+generarPago() +consultarPago() +emitirComprobante() +pagar() +descargarRecibo()

Relaciones:

- Agregación: Agregado por Tutor (0..*) → Un tutor puede realizar múltiples pagos relacionados.

-Dependencia: Depende de Administrador (0..1) → Un administrador puede gestionar y supervisar los pagos realizados, sin ser parte directa del objeto

10) Clase Mensaje: Tipo: Entidad.

Atributos	Métodos
-ID:int -emisor:string -receptor:string -fecha:date -contenido:string	+enviarMensaje() +recibirMensaje() +marcarComoLeido()

Relaciones:

-Asociación: Se asocia con Tutor, Maestra, Administrador (0..*) → Todos los perfiles pueden enviarse mensajes entre sí.

II. El diagrama de Arquitectura fue realizado con la aplicacion Figma, a continuación, dejamos el link de acceso:

https://www.figma.com/design/cpMl0WHWOi7MSfejJpVhMR/Proyecto-Jardin-Maternal?node-id=0-1&p=f&t=b0dSUUfI9tzPL2IQ-0

II. Descripción del Diagrama de Arquitectura en capas “Luz de Luna”

La estructura sigue un patrón de capas apiladas, donde cada capa ofrece servicios a la capa superior y consume servicios de la capa inferior, promoviendo una clara separación de preocupaciones y facilitando la modularidad y el mantenimiento.

El diagrama se estructura verticalmente, con las capas superiores representando niveles de abstracción más cercanos al usuario final y las capas inferiores gestionando aspectos de bajo nivel, como la persistencia y el acceso a los datos fundamentales del sistema.

Nivel de Interacción del Usuario:

En la parte superior del diagrama, se identifican los Usuarios o Actores del sistema. Estos corresponden a los distintos roles que interactuarán activamente con la aplicación para realizar tareas específicas dentro del jardín maternal: maestras, padres, directora y el personal administrativo.

Estos actores inician las interacciones con el sistema a través de un Navegador Web. El Navegador actúa como el cliente de la aplicación, ejecutando el código de la interfaz de usuario y sirviendo como el medio a través del cual el usuario accede y manipula el sistema. La comunicación entre el Usuario y la Capa de Presentación se realiza mediante el protocolo HTTP/S. Este protocolo, garantiza que las solicitudes y respuestas se transmitan de forma cifrada, protegiendo la confidencialidad e integridad de los datos intercambiados a través de la red.

Primera Capa: Capa de Presentación (Frontend)

Es la responsable primaria de la interfaz de usuario y de la gestión de la interacción directa con los actores. Su objetivo fundamental es capturar las entradas del usuario (acciones, datos) y enviar estas interacciones a la capa subsiguiente para su procesamiento.

Implementada utilizando React, una biblioteca de JavaScript para construir interfaces de usuario. React permite desarrollar componentes reutilizables, lo que acelera el desarrollo y mejora la consistencia de la UI.

Contiene las Interfaces de Usuario, que son las representaciones visuales de las distintas funcionalidades y flujos de trabajo del sistema. Esto incluye pantallas dedicadas para el Login y Registro de usuarios, el módulo de Inicio con información relevante según el rol, secciones específicas para la gestión de Asistencia y Salud de los niños, Rutinas Diarias, Administración y Pagos, el módulo de Mensajes (Chat) para la comunicación, y la gestión del Perfil de Usuario.

Comprende los Elementos de Interfaz, que son los controles interactivos básicos y componentes visuales. Esto abarca desde botones (para iniciar acciones), campos de texto (para la entrada de datos alfanuméricos), tablas (para la visualización estructurada de listados de datos como asistencias o pagos), iconos (para acciones rápidas o indicativos), barras de navegación y menús (para la orientación dentro del sistema), y otros componentes interactivos.

Incorpora el Diseño Adaptable. Este aspecto asegura que la interfaz se adapte dinámicamente a las características del dispositivo de visualización (tamaño de pantalla, orientación), en dispositivos (desktops, laptops, celulares, tablets)

Define el Estilo Visual, aplicando la paleta de colores, la guía tipográfica y otros elementos gráficos definidos en el proyecto.

La Capa de Presentación se comunica exclusivamente con la Capa de Aplicación. Esta comunicación se realiza mediante el envío de Solicitudes API (Application Programming Interface), estructuradas típicamente en formato JSON (JavaScript Object Notation). Estas solicitudes encapsulan las acciones del usuario (ej. registrar asistencia) o las peticiones de información (ej. obtener registros diarios de un niño). A continuación, recibe Respuestas API (también en JSON) que contienen los resultados de las operaciones ejecutadas en el backend o los datos requeridos para actualizar dinámicamente la interfaz de usuario.

Segunda Capa: Capa de Aplicación (Backend)

Situada directamente debajo de la Capa de Presentación, la Capa de Aplicación (Backend o Capa de Lógica de Negocio). Su función principal es procesar las solicitudes recibidas del Frontend, ejecutar la lógica de negocio, aplicar las reglas definidas por el jardín maternal y coordinar las interacciones con la capa de datos y otros componentes del sistema.

Compuesta por componentes internos que colaboran para procesar las solicitudes:

Controladores API: Reciben los datos enviados por el cliente, realizan validaciones preliminares (ej. formato de datos) y delegan la ejecución de la lógica de negocio compleja a los Servicios correspondientes.

Servicios (Lógica): Cada Servicio se especializa en una funcionalidad o área del negocio (Ej: Servicio de Asistencia gestiona todo lo relacionado con el registro de entradas y salidas; Servicio de Pagos maneja la generación, procesamiento y seguimiento de cuotas). Interactúan con los Repositorios para acceder a los datos y pueden coordinar acciones con otros Servicios o componentes transversales (como el envío de notificaciones).

Entidades (Modelos): Clases que representan los conceptos fundamentales del dominio del negocio del sistema (ej. Niño, Usuario, Pago). Actúan como la estructura de datos principal, encapsulando los atributos relevantes de cada concepto. Sirven como los objetos con los que interactúa la lógica de negocio (Servicios) y que son gestionados por los Repositorios para su persistencia en la Capa de Datos.

Repositorios (Spring Data JPA): Constituyen una capa de abstracción para el acceso a la Base de Datos. Su función es proporcionar una interfaz clara para realizar operaciones de persistencia (CRUD) sobre las Entidades. Utilizando Spring Data JPA, implementan la lógica necesaria para interactuar con la Capa de Datos (MySQL) mediante JPA/JDBC, desacoplando los Servicios de los detalles técnicos del acceso a datos de bajo nivel.

Manejo de Errores: Conjunto de mecanismos implementados para interceptar, procesar y responder de manera controlada a las excepciones y errores que pueden ocurrir durante la ejecución de la lógica en los Controladores y Servicios. Su función es proporcionar respuestas de error informativas y estandarizadas al Frontend, y registrar los incidentes para su posterior análisis y depuración.

La Capa de Aplicación recibe solicitudes JSON de la Capa de Presentación y, tras procesarlas, envía respuestas JSON de vuelta. Su interacción principal con la Capa de Datos se realiza a través de los Repositorios.

Tercera Capa: Capa de Datos

Responsable de la persistencia de toda la información del sistema. Su función principal es almacenar, organizar, gestionar y proporcionar acceso seguro y eficiente a los datos de manera duradera.

Utiliza MySQL como sistema de gestión de base de datos relacional que gestiona el almacenamiento y la recuperación de los datos.

Define el Esquema DB y las Tablas. El esquema de base de datos es la estructura lógica que organiza los datos. Las Tablas son las unidades fundamentales de almacenamiento en una base de datos relacional, donde se guardan los datos en filas y columnas. Cada tabla corresponde generalmente a una Entidad definida en la Capa de Aplicación (Ej: la tabla usuarios almacena los datos de los usuarios, la tabla registros_asistencia guarda los registros de asistencia). Las relaciones entre las tablas (mediante claves primarias y foráneas) aseguran la integridad y consistencia de los datos.

Asegura que los datos guardados en la base de datos se mantengan disponibles de forma duradera a lo largo del tiempo, independientemente del estado de ejecución de la aplicación o de posibles reinicios del servidor.

La Capa de Datos es accedida exclusivamente por la Capa de Aplicación, específicamente por los Repositorios, para realizar operaciones de lectura y escritura de datos. Esta interacción se lleva a cabo utilizando protocolos y tecnologías de acceso a bases de datos como JPA / JDBC.

Componente Transversal: Seguridad

Representado lateralmente a la Capa de Aplicación, el componente de Seguridad, implementado típicamente con Spring Security en un entorno Spring Boot. Su rol es proteger el sistema y sus recursos de accesos no autorizados y garantizar que los usuarios solo puedan realizar las acciones para las que tienen permiso.

Se encarga de la Autenticación, el proceso de verificar la identidad de un usuario que intenta acceder al sistema (ej. validando credenciales como usuario y contraseña).

Contribuye activamente a la Protección de Datos Sensibles, asegurando que la información confidencial, especialmente la relacionada con los niños y sus registros, solo sea accesible por usuarios debidamente autorizados.

Interactúa con los Repositorios (en la Capa de Aplicación) para obtener la información de usuario, roles y permisos necesaria para realizar sus procesos de autenticación y autorización.

III. Flujo de datos

El flujo de datos fue realizado con la aplicación Miro. A continuación, dejamos el link de acceso:

https://miro.com/app/board/uXjVI3qWCa8=/

Componentes del Diagrama

Usuario: Representa a cualquier persona que interactúa con el sistema (maestra, padre, directora, administrativo).

Se conecta al navegador para iniciar la interacción.

Navegador Web: Medio por el cual el usuario accede a la aplicación web.

Envía solicitudes HTTP/S al Frontend.

Frontend (Capa de Presentación): Interfaz hecha en React que captura la acción del usuario.

Genera una solicitud API (en JSON) hacia los controladores del backend.

Controladores API: Reciben solicitudes y las redirigen.

Seguridad (Spring Security): Intercepta las solicitudes y verifica si el usuario está autenticado y tiene permiso.

Servicios: Contienen la lógica de negocio (por ejemplo, registrar asistencia, enviar mensaje, etc.).

Repositorios: Se encargan de acceder a la base de datos mediante JPA o JDBC.

Base de Datos MySQL (Capa de Datos): Almacena la información persistente.

Recibe consultas y operaciones de guardado desde los repositorios.

Responde con datos (entidades) que se devuelven hacia atrás por el mismo flujo.

Uniones / Flujo de Datos

Estas son las conexiones clave:

Usuario → Navegador: El usuario accede al sistema.

Navegador → Frontend: El navegador envía una solicitud al frontend.

Frontend → Controladores API: Envía la solicitud de acción en formato JSON.

Controladores API → Seguridad: Verifica si el usuario puede realizar la acción.

Seguridad → Controladores API: Devuelve si está autorizado.

Controladores API → Servicios: Si todo está bien, se llama a la lógica de negocio.

Servicios → Repositorios: Si se necesita guardar o leer datos.

Repositorios → BD: Se ejecutan consultas o escrituras en la base de datos.

BD → Repositorios → Servicios: Los datos fluyen de regreso.

Servicios → Controladores API: Se genera una respuesta.

Controladores API → Frontend: Se envía respuesta JSON.

Frontend → Navegador: Se actualiza la interfaz del usuario.