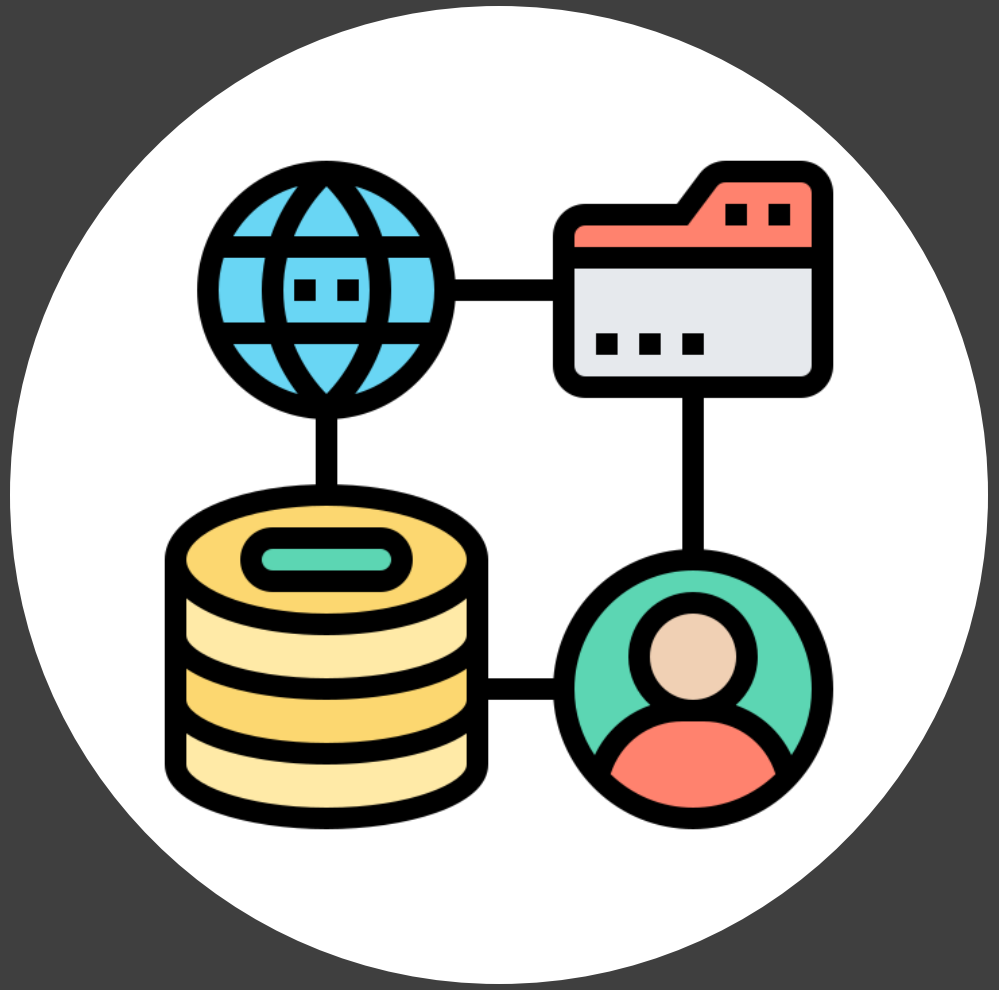


Arquitectos del Software

Mar Alonso García

Sebastián Alfredo Castro Rampersad

Adrián Castro Vilar

Adolfo Fanjul Sánchez

David Gayoso Salvado

Carla Rodríguez Estévez

Índice

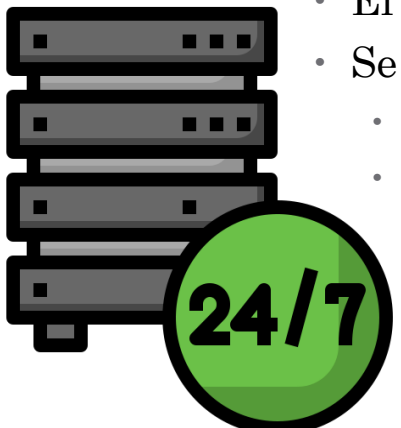
DESCRIPCIÓN DE PROYECTO

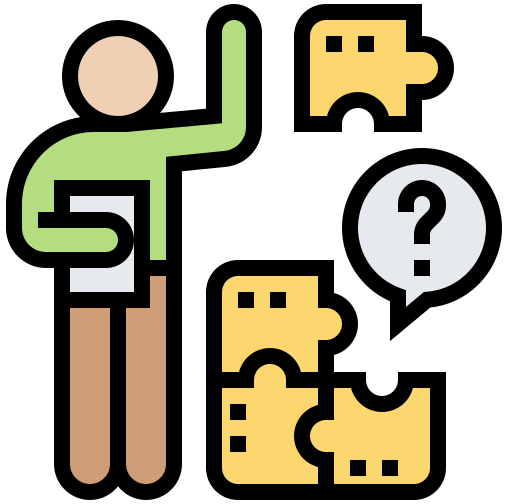
Descripción del proyecto I

- El proyecto consiste en un servicio de mensajería distribuido.
- Los usuarios podrán registrarse, enviarse mensajes entre ellos y recibir los mensajes que le han enviado.
- Un usuario lo podrá usar a través de una interfaz de línea de comandos.
- Los requisitos funcionales son:
  - Registrarse con un nombre de usuario y una contraseña, nombre que ha de ser único.
  - Loguearse.
  - Enviar un mensaje a otro usuario que esté registrado.
  - Revisar los mensajes no leídos
  - Revisar el histórico de mensajes.
  - Borrar mensajes leídos.
  - Listar los usuarios registrados en el sistema.

Descripción del proyecto II

- Entre los requisitos no funcionales que se han considerado para el proyecto se debatió entre seguridad y disponibilidad.
- Se eligió centrarse en disponibilidad
  - Cada intento del usuario de usar la aplicación será exitoso
  - Para la experiencia de usuario la disponibilidad es fundamental ya que evita la experiencia de entrar a un servicio que no funciona o no responde.
  - Tiene un funcionamiento más semejante a servicios de mensajería reales, donde no es usual verlo no disponible.
- Además el sistema se ha implementado de forma que es escalable
  - El sistema tiene capacidad para adaptarse manualmente a la demanda existente
  - Se puede aumentar/disminuir la escalabilidad del sistema de dos formas:
    - Añadiendo/eliminando balanceadores de carga.
    - Añadiendo/eliminando servicios a los balanceadores.

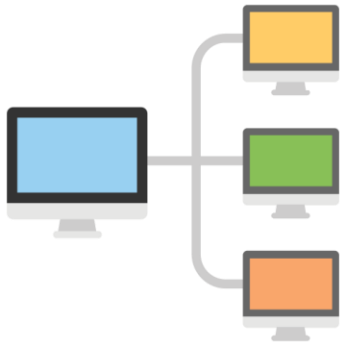


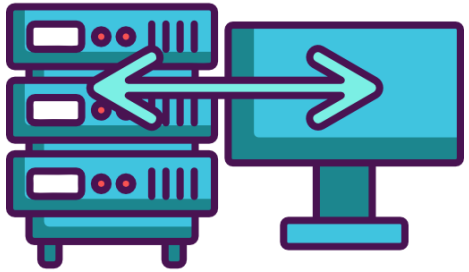


SOLUCIÓN ARQUITECTÓNICA

Solución arquitectónica I

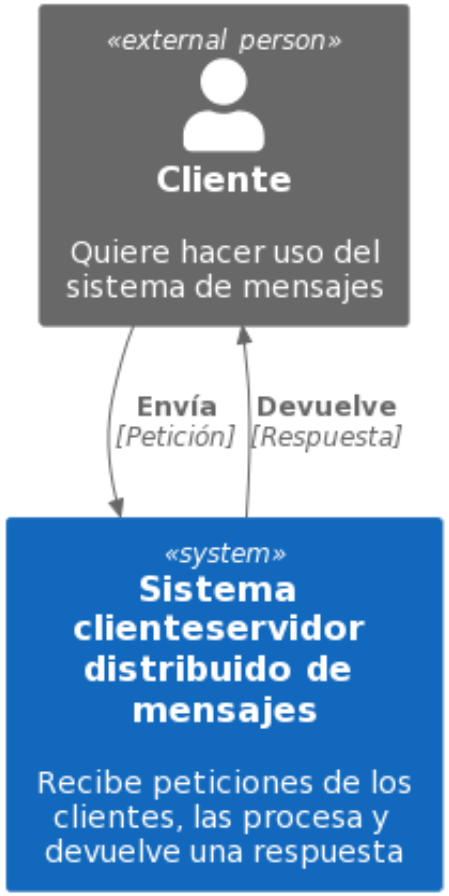
- La arquitectura seleccionada es cliente-servidor distribuido.
- El cliente se encuentra en un nodo distinto al servidor.
- En el caso particular de este proyecto a parte del cliente y el directorio, estarían distribuidos:
  - Los servicios de mensajes y usuarios.
  - Las bases de datos de cada uno de los servicios.
  - Los balanceadores de carga.
- Entre estos elementos se envían mensajes para poder solucionar las consultas relacionadas con uno u otro servicio.





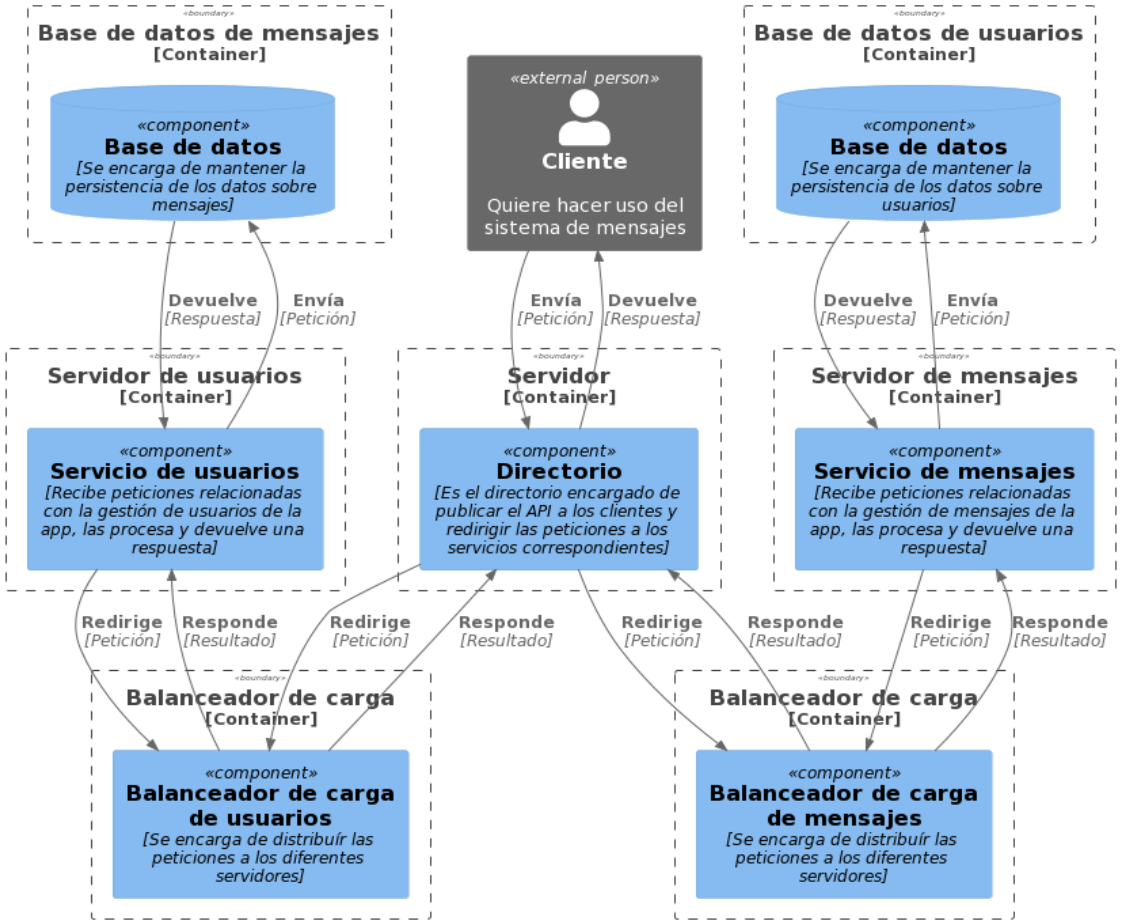
Solución arquitectónica II

- Diagrama de contexto



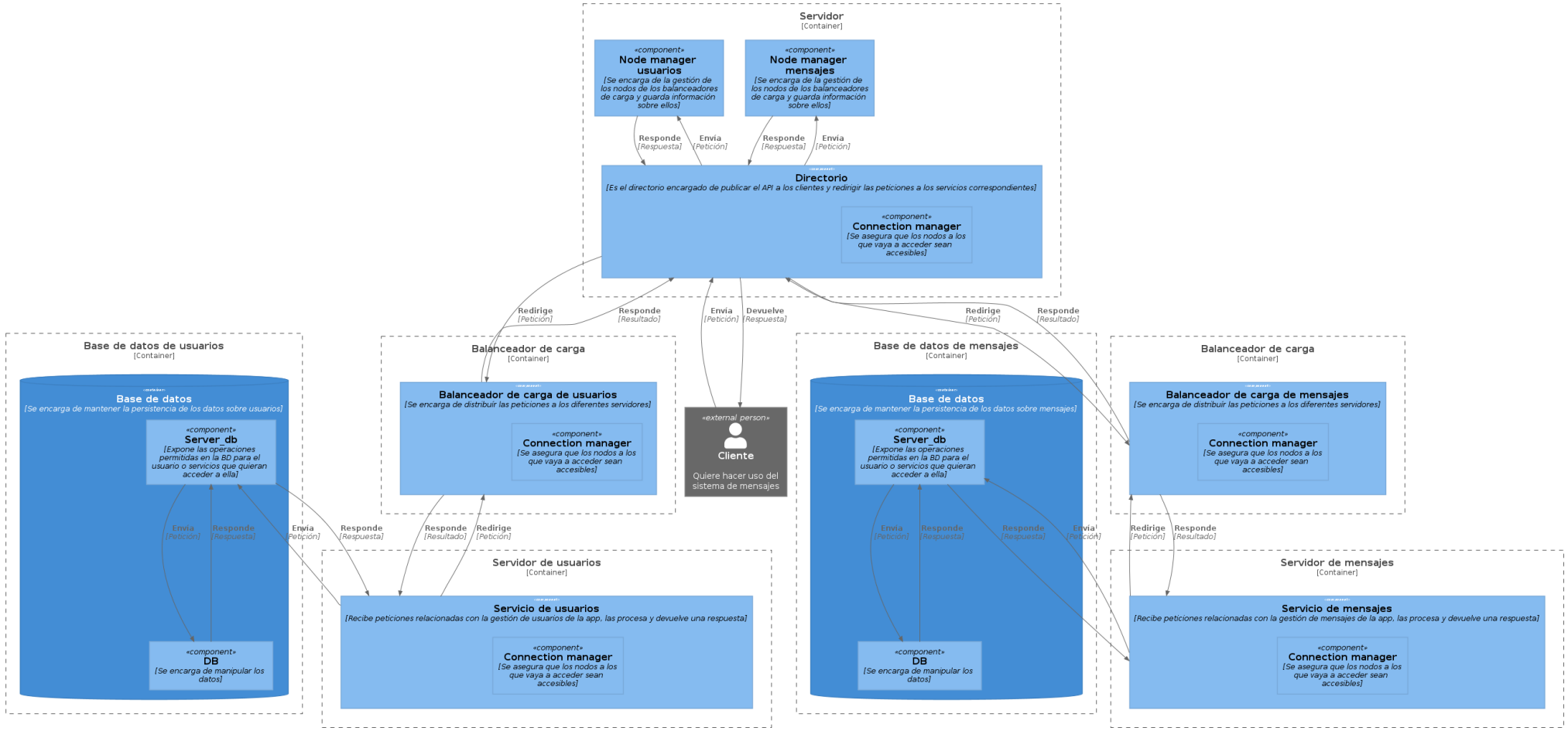
Solución arquitectónica III

- Diagrama de contenedor



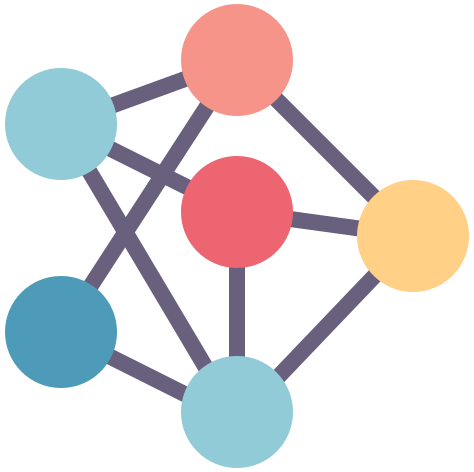
Solución arquitectónica IV

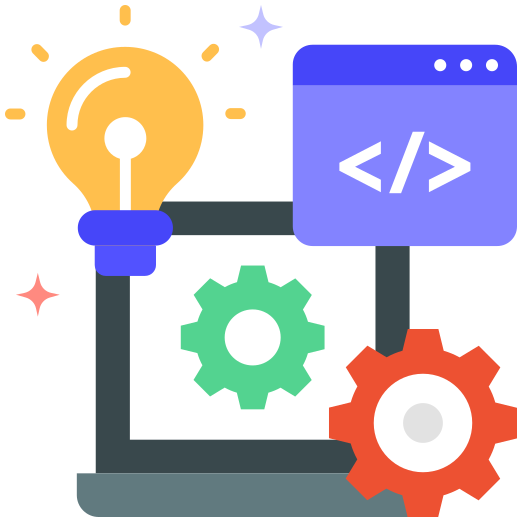
- Diagrama de componentes



Tácticas aplicadas: Ping/echo & Excepciones

- El módulo **ConnectionManager** implementa la táctica de Ping y la táctica de excepciones.
- La táctica de ping la emplea realizando un **Node.connect** al nodo al que se pretende hacer una petición.
- En caso de conseguir conexión (true), se realiza la petición.
- En caso contrario (false) se lanza una excepción que es controlada en el mismo componente y que se envía en última instancia al cliente.

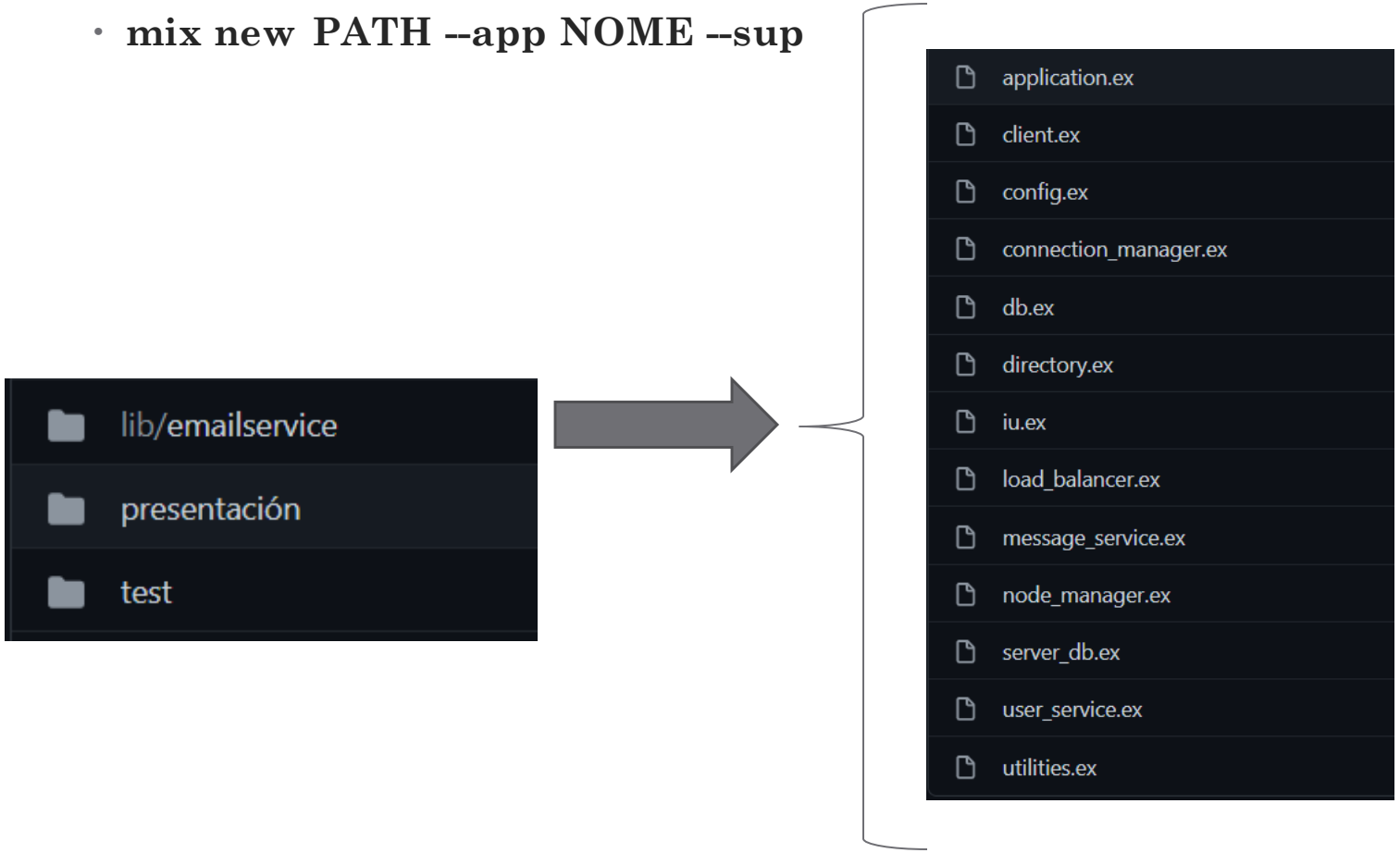




IMPLEMENTACIÓN

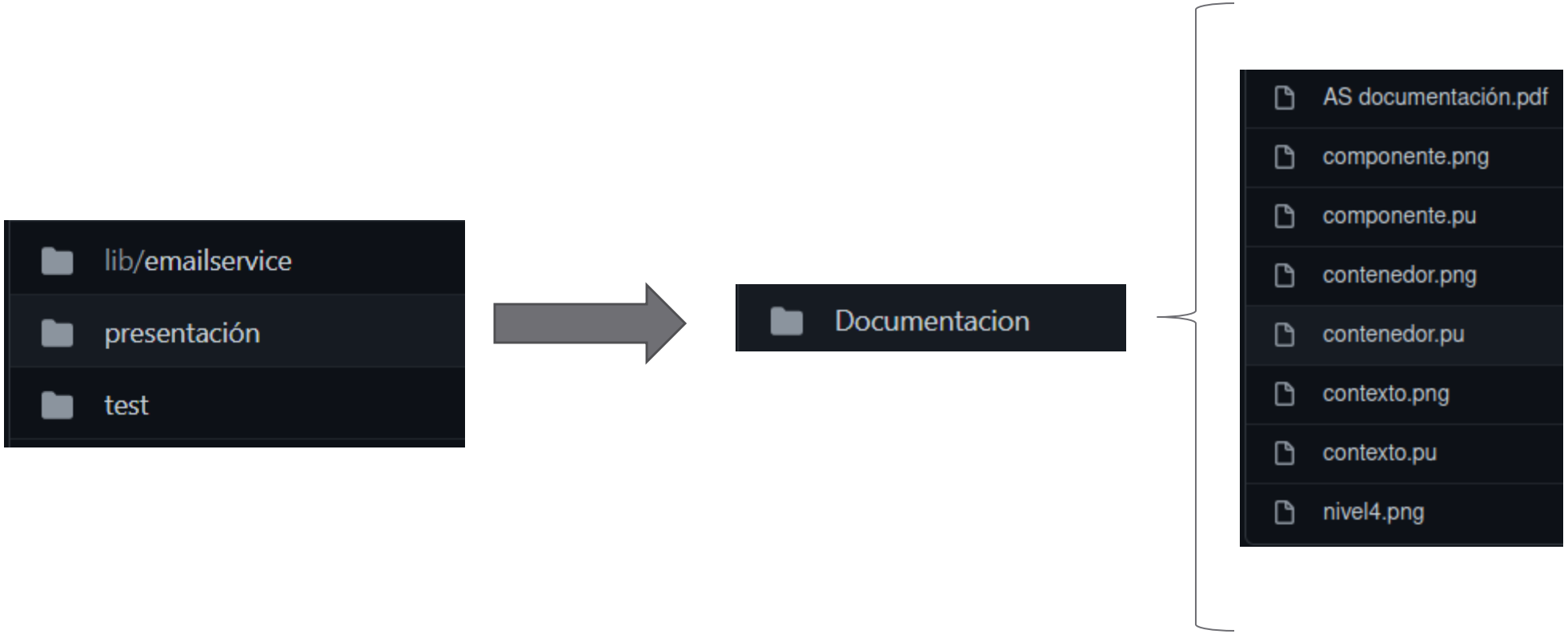
Estructura del proyecto en el repositorio I

- Estructura del proyecto creada utilizando mix, concretamente utilizando el siguiente comando:
  - **mix new PATH --app NOME --sup**

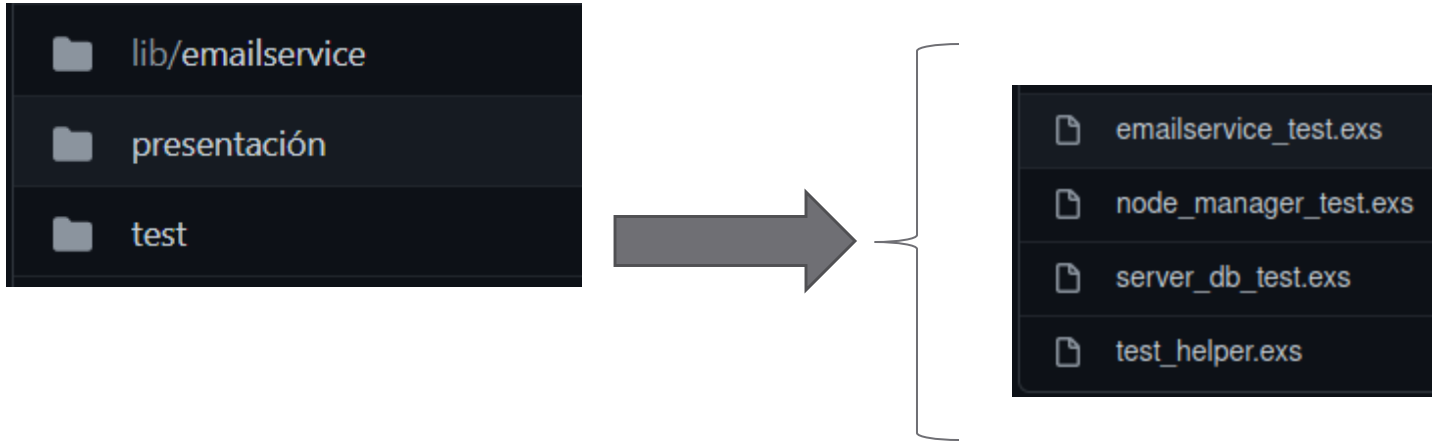


Elementos que componen la arquitectura desarrollada y herramientas de configuración

Estructura del proyecto en el repositorio II



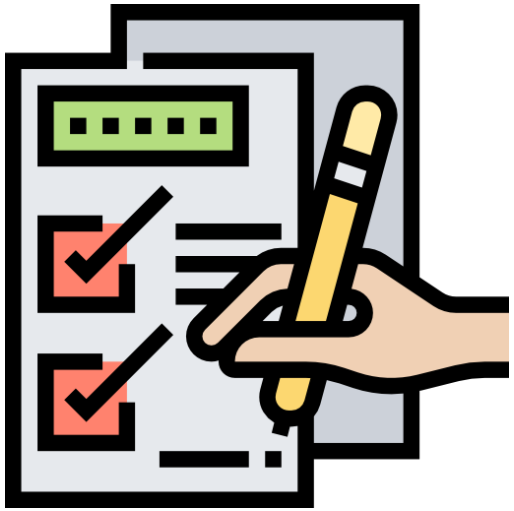
Estructura del proyecto en el repositorio III



Pruebas realizadas I

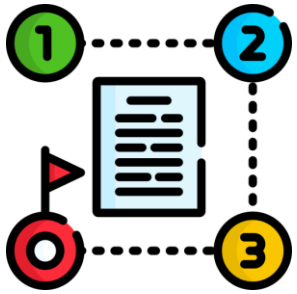
- Se realizaron pruebas unitarias en **NodeManager** y en **ServerDb**.
- En las pruebas de integración se ha probado de forma exhaustiva el sistema, destacando los siguientes escenarios:

1. Registrarse de forma satisfactoria en el sistema, indicando un nombre de usuario y contraseña.
2. Registrarse con un usuario ya existente. (**ERROR**).
3. Loguearse correctamente con un usuario ya existente.
4. Loguearse como un usuario no registrado. (**ERROR**).
5. Loguearse con un usuario registrado y una contraseña incorrecta. (**ERROR**).
6. Enviar un mensaje a un usuario registrado.
7. Enviar un mensaje a un usuario NO registrado. (**ERROR**).



Pruebas realizadas II

1. Revisar los mensajes no leídos del buzón de correo como usuario registrado.
2. Revisar los mensajes no leídos del buzón de correo sin estar registrado. (**ERROR**).
3. Revisar todos los mensajes del buzón.
4. Revisar todos los mensajes del buzón sin estar registrado. (**ERROR**).
5. Borrar mensajes leídos.
6. Borrar mensajes leídos sin estar registrado. (**ERROR**).
7. Listar los usuarios registrados a los que puede enviar un mensaje.



Elementos destacados

- Destacamos el uso de **GenServer** en los siguientes componentes:

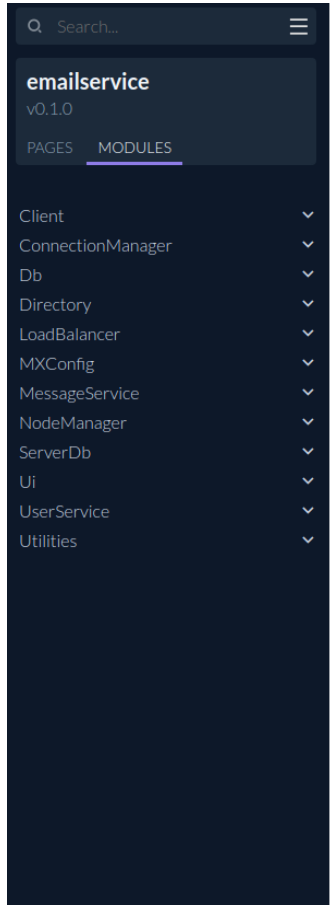
- **Servidor de la base de datos de usuario**
- **Servidor de la base de datos de mensajes**

- Destacamos el uso de **Agents** en los siguientes componentes:
- **Directorio** (para guardar la lista de balanceadores de carga)
- **Balanceadores de carga** (para guardar la lista de servicios)
- **Servicios** (para guardar la referencia a la base de datos)

- Todos los componentes del sistema son **Nodos**

Documentación

- Toda la documentación del código ha sido implementada usando **ExDoc**



API Reference

Modules

Client

Módulo que implementa el cliente.

ConnectionManager

Db

Módulo que se encarga de implementar las funcionalidades de las BDs.

Directory

Módulo que implementa el directorio.

LoadBalancer

Módulo que implementa un balanceador de carga.

MXConfig

Este módulo se encarga de inicializar una configuración particular del sistema implementado.

MessageService

Módulo que implementa el servicio de mensajes.

NodeManager

Módulo que implementa un gestor de nodos.

ServerDb

Módulo que implementa un servidor de una base de datos.

Ui

UserService

Módulo que implementa el servicio de usuario.

Utilities

Demostración...