Documento de Arquitectura de Software

Grupo [15] - 2023

Taller de Sistemas Empresariales
Taller de Sistemas de Información Java EE

Pablo Cristiani

Pedro Aldama

Facundo Aparicio

Federico Acosta

Marzo 2023

Tabla de contenidos

1. Introducción

Este documento es parte del desarrollo de la propuesta planteada por la Facultad de Ingeniería en el Taller de JavaEE/Sistemas Empresariales. “carga.uy” es una aplicación web y móvil, de gestión de empresas de transporte de cargas en Uruguay, la cual será diseñada, implementada y testeada por el equipo de trabajo compuesto por Adriana Pisano, Facundo Aparicio, Federico Acosta, Pablo Cristiani y Pedro Aldama, estudiantes de la Universidad del Trabajo del Uruguay (UTU).

Esta plataforma apunta a brindar soporte a la fiscalización y gestión del transporte de carga en Uruguay.

1.1. Objetivo del Documento

El objetivo del presente documento es ampliar el conocimiento de la estructura del sistema informático que se llevará a cabo por el equipo de producción. Es de suma importancia poder detallar cada etapa, metodología y tecnología con la que la aplicación va contar. La completa documentación logra satisfacer los requerimientos funcionales y no funcionales establecidos por los interesados.

1.2. Representación de la Arquitectura

La arquitectura de la plataforma ‹ carga.uy › está representada por diferentes vistas que permiten visualizar, entender y razonar sobre los elementos significativos de la arquitectura e identificar áreas de riesgo que requieran mayor detalle de elaboración. En particular, las vistas utilizadas para representar la arquitectura de la plataforma ‹ carga.uy › son:

1. **Vista de Casos de Uso:** Representa las funcionalidades más importantes del sistema a modelar desde el punto de vista de los distintos usuarios del mismo. Esta vista describe los comportamientos esperados del sistema.

2. **Vista de Restricciones:** Describe las diversas restricciones (e.g. tecnológicas, regulatorias, estándares) que deben ser respetadas por el sistema.

3. **Vista de Atributos de Calidad:** La vista de atributos de calidad lista las propiedades relacionadas a requerimientos no funcionales (e.g. escalabilidad, seguridad, mantenibilidad, modularidad) que deben ser cumplidas por el sistema.

4. **Vista Lógica:** Presenta las estructuras conceptuales (componentes/paquetes/ módulos) que conforman el sistema y sus relaciones.

5. **Vista de Despliegue (Distribución/Deployment):** Presenta cómo va a funcionar el sistema en el entorno productivo. Muestra cómo están conectados los diferentes componentes de infraestructura del sistema (e.g. firewall, balanceadores de carga, servidores, base de datos). La vista de despliegue representa la disposición física del sistema.

6. **Vista de Implementación:** Representa las opciones arquitectónicas y tecnológicas. Aquí se incluyen las particularidades del estilo arquitectónico seleccionado para implementar la solución (e.g. las capas de una aplicación, basada en eventos) junto con el razonamiento que la sustenta. También se detallan las opciones tecnológicas, junto con los pros y los contras de su uso. En la vista de implementación se deberían justificar los recursos y las habilidades necesarias para ejecutar el proyecto.

7. **Vista de Decisiones de Arquitectura:** La vista de decisiones de arquitectura presenta y describe las principales decisiones de arquitectura tomadas sopesando las diferentes alternativas que se presentaban al momento de la decisión.

1.3. Partes Interesadas (i.e. *stakeholders*)

Las partes interesadas en la plataforma carga.uy son:

1. **Estado/MTOP:** objetivos de recaudación, fiscalización de la actividad y planificación de infraestructura para el futuro.
2. **Administrativos:** gestión de usuarios al sistema. Monitoreo de empresas y vehículos.
3. **Ciudadanos:** consulta e información general.
4. **Empresas transportistas:** control y gestión. Información precisa sobre su negocio.
5. **Empresas tercerizadas:** validación de permisos.
6. **Choferes:** registro de su trabajo, garantía de exigir cumplimientos de sus derechos.
7. **Equipo de desarrollo:** conocimiento, experiencia.

La Tabla 1 indica a qué parte interesada está orientada cada una de las vistas de la arquitectura.

Tabla 1: Partes Interesadas y Vistas

	Vista de Casos de Uso	Vista de Restricciones	Vista de Atributos de Calidad	Vista Lógica	Vista de Procesos	Vista de Datos	Vista de Decisiones de Arquitectura
Estado/MTOP	✓	✓	✓		✓		✓
Administrativos	✓	✓	✓		✓		✓
Ciudadanos		✓					
Emp. Transportistas		✓					✓
Emp. Tercerizadas		✓					✓
Choferes		✓					✓
Eq. Desarrollo	✓	✓	✓	✓	✓	✓	✓

1.4. Organización del Documento

El resto del documento se organiza en ocho secciones, cada una de las cuales describe una de las vistas que representan la arquitectura:

- Sección 2: Vista Conceptual
- Sección 3: Vista de Casos de Uso
- Sección 4: Vista de Restricciones
- Sección 5: Vista de Atributos de Calidad
- Sección 6: Vista lógica
- Sección 7: Vista de Distribución
- Sección 8: Vista de Implementación
- Sección 9: Vista de Decisiones de Arquitectura

2. Vista Conceptual

La Vista Conceptual brinda una descripción general de la plataforma y presenta los principales conceptos asociados a la misma.

2.1. Descripción General de la Plataforma

La plataforma surge debido a que el transporte es un área fundamental, donde múltiples personas y entidades desean poder llevar un control, gestión y uso de una plataforma. Para ello se necesitaba un lugar donde poder realizar grandes operaciones, gestiones automáticas, controles tanto de las empresas como del estado para la correcta fiscalización; Todo ello de yendo de la mano con el avance del gobierno tecnológico. Pudiendo así mantener el flujo constante de transporte de diferentes mercaderías, lograr generar reportes estadísticos útiles para el negocio y adicionalmente todo regulado por el estado. Siendo posible a futuro la expansión de la misma y llegar a un punto donde el programa sea un uso estándar para todas las empresas y que genere fácil gestión de todo para todos los involucrados.

El sistema se compone de un componente central, ejecutado sobre un servidor de aplicaciones Wildfly dentro de la plataforma Elastic Cloud. Este nodo central será el encargado de la mayor parte de la lógica de la plataforma, presentando una capa denominada *FrontOffice*, donde los usuarios podrán ver de forma rápida ciertas funciones para poder realizar su trabajo. Adicionalmente este *FrontOffice* cuenta con una aplicación móvil para poder facilitar ciertas acciones de parte de los usuarios. Por otro lado para el resto de usuarios que ingresen por el componente central será mediante un *BackOffice*, el cual mediante un login ingresarán al programa y podrán usar funcionalidades especiales acordes a su rango. Todo esto contará con una diferenciación de tipos de usuario para poder mantener la seguridad y los accesos a quienes lo debían tener.

Dentro de este componente central el ciudadano puede ingresar y gestionar la información sobre los vehículos de carga de su empresa y de los viajes. Por otro lado, la aplicación móvil permite a los conductores de los vehículos indicar que comienzan un viaje de carga, que lo terminan, etc.

Los funcionarios pueden gestionar las empresas de transporte de carga. Esto es, darlas de alta, asociarlos usuarios, consultar la información de fiscalización, etc.

Las autoridades podrán acceder a través del *BackOffice*, mediante el cual las autoridades pueden consultar la información de las empresas de transporte y tener acceso a reportes de distinto tipo.

Los administradores están disponibles a través de un *BackOffice*, mediante el cual los administradores pueden gestionar usuarios, roles, nodos periféricos y tipos de datos.

Apartado del componente central se cuenta con diferentes nodos periféricos o plataformas a las cuales estará enlazado para poder generar una mayor interoperabilidad y permitirán al componente central nutrirse de información relevante para su funcionamiento y operatividad. Una de las que usaremos será la plataforma de interoperabilidad que provee el estado.

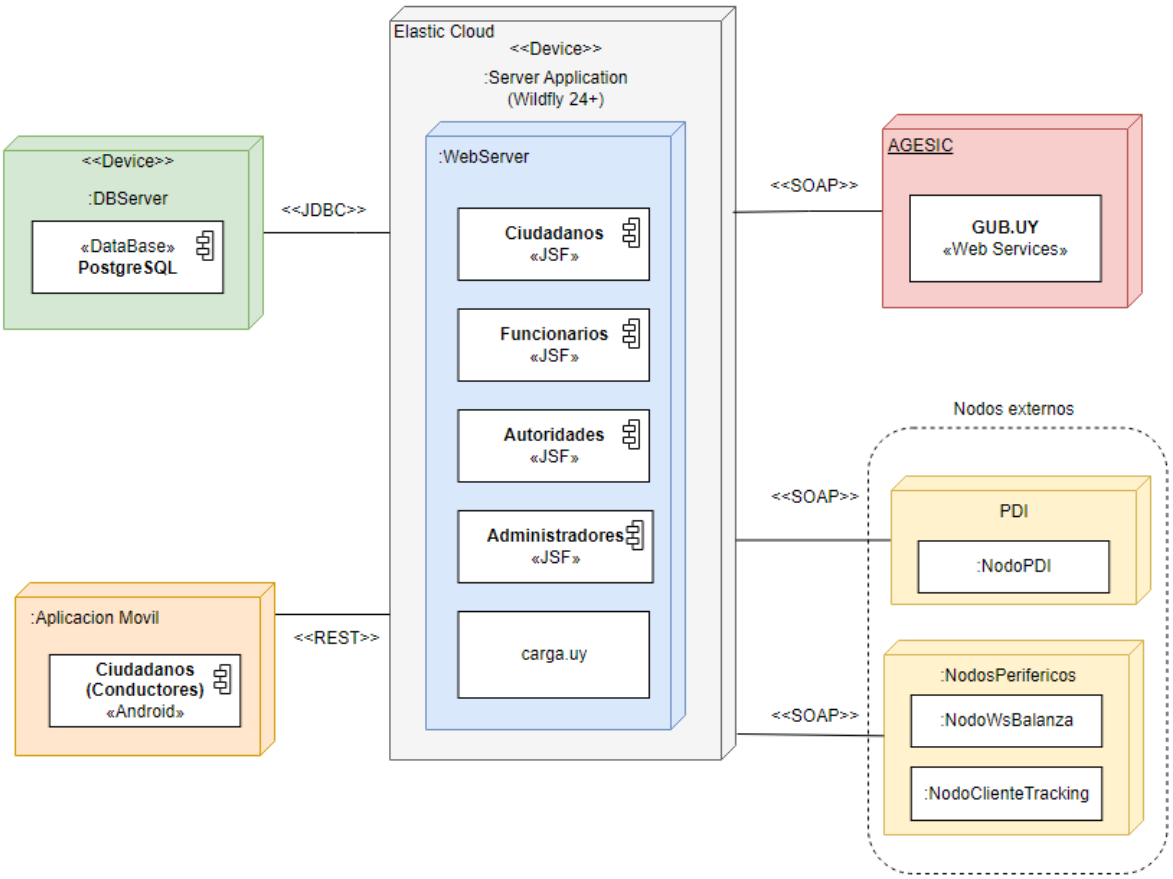


Figura 1: Diagrama de implementación en la aplicación

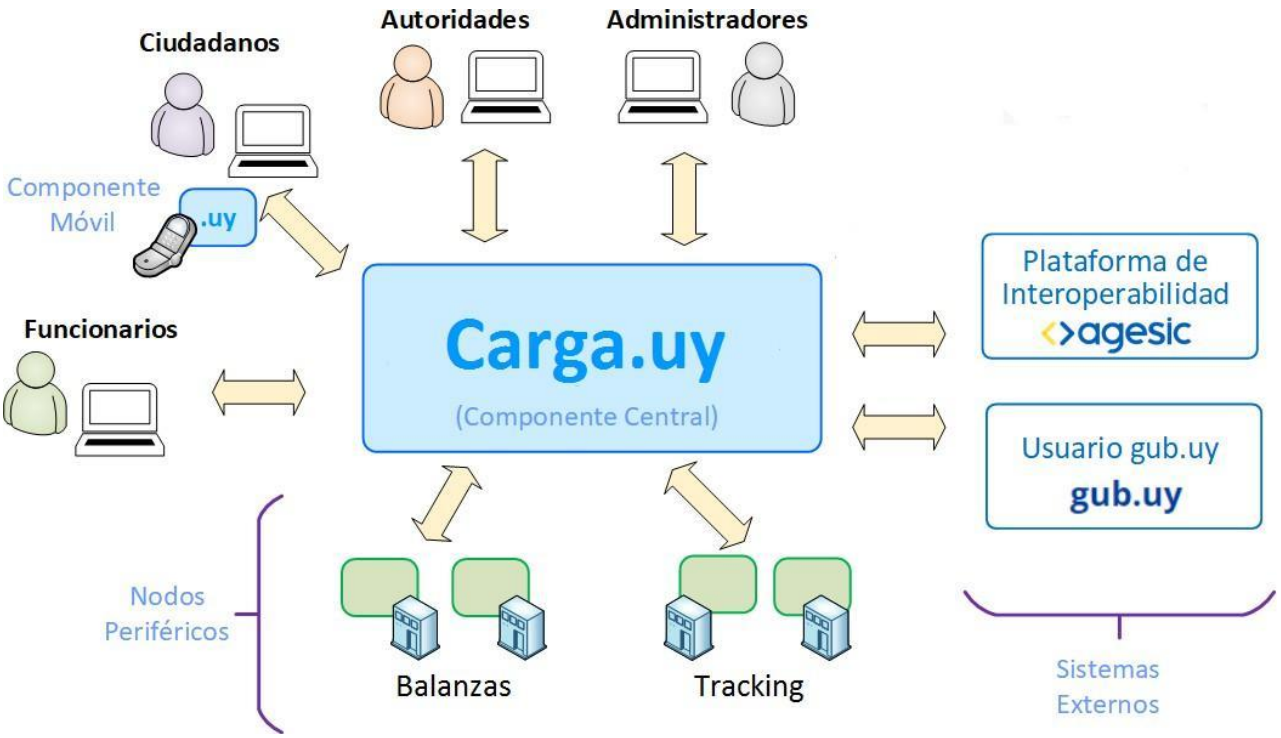
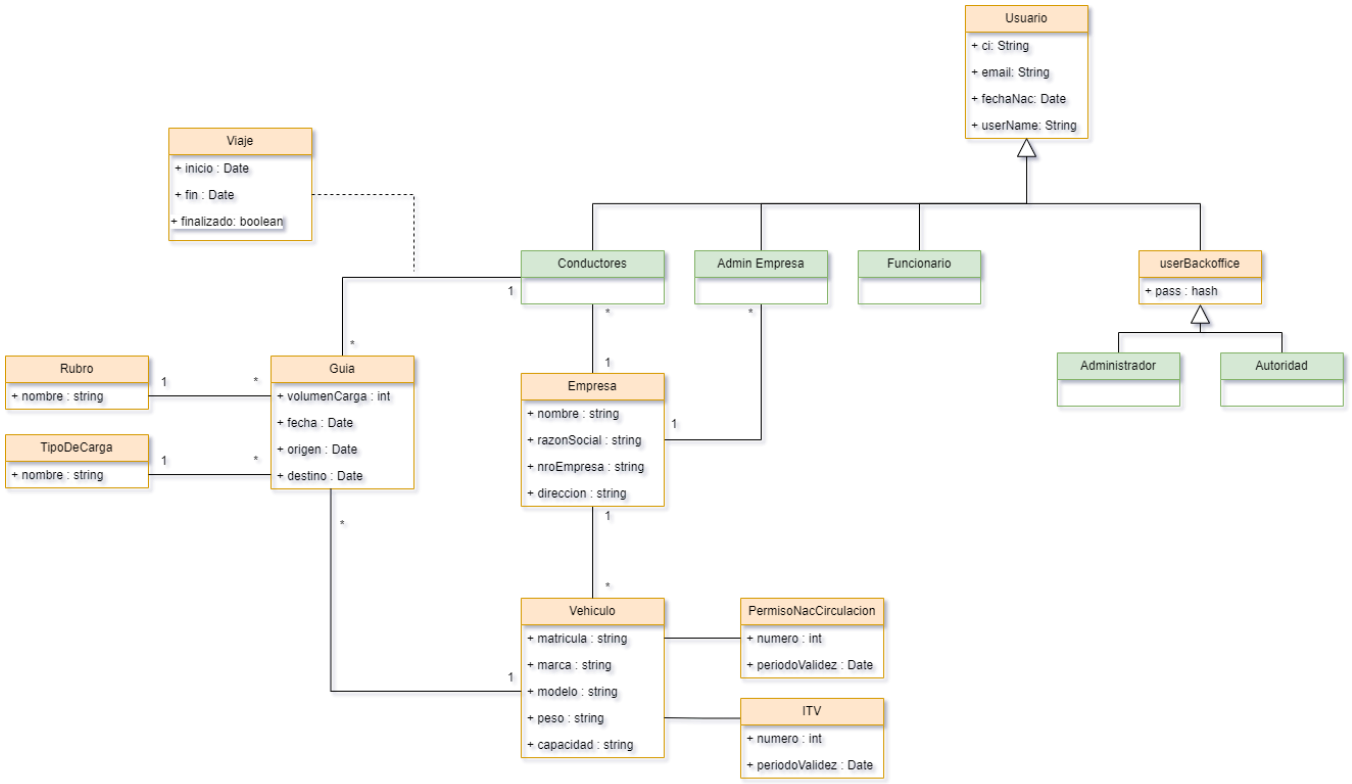


Figura 2: Descripción General de la Plataforma

2.2. Modelo Conceptual



RESTRICCIÓN NO ESTRUCTURAL

- En la guía, tanto el vehículo como el conductor deben pertenecer a la misma empresa.

Figura 3: Modelo conceptual

3. Vista de Casos de Uso

La Vista de Casos de Uso se centra en los aspectos funcionales de la plataforma. En esta vista se presentan los actores así como los casos de uso de la plataforma, y se detallan los casos de uso que se consideran críticos para la arquitectura.

3.1. Actores

En esta sección se presentan los actores que interactúan con la plataforma carga.uy.

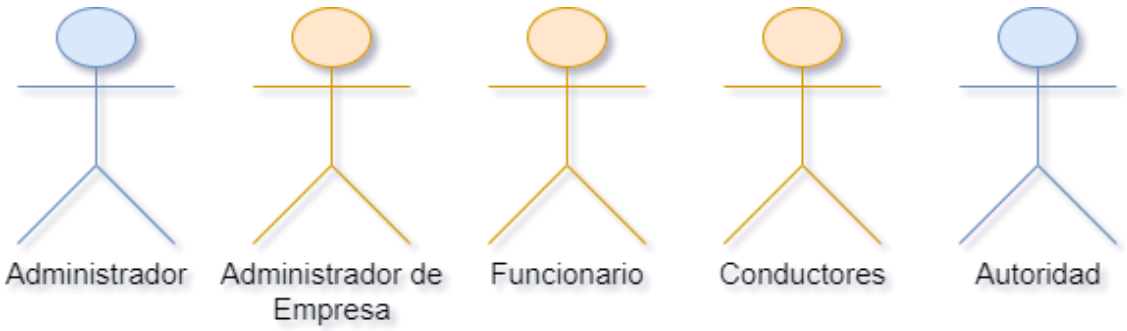


Figura 4: Vista Autores

3.2. Diagrama de Casos de Uso

En esta sección se utilizan Diagramas de Casos de Uso UML para presentar los casos de uso de la plataforma, indicando cuáles se consideran críticos para la arquitectura.

CU001 - Autenticación con Gub.uy

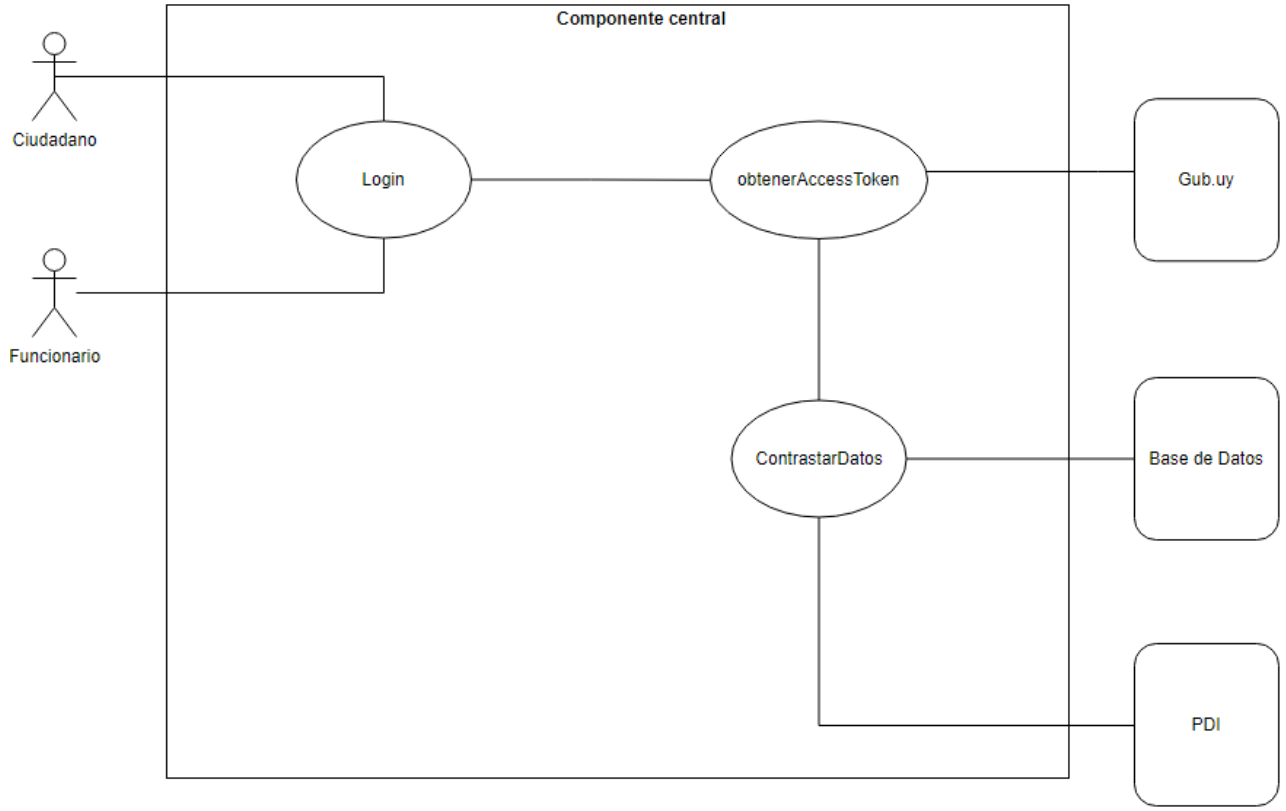


Figura 5: CU autenticación con Gub.uy

CU002 - Inicio y fin de viaje

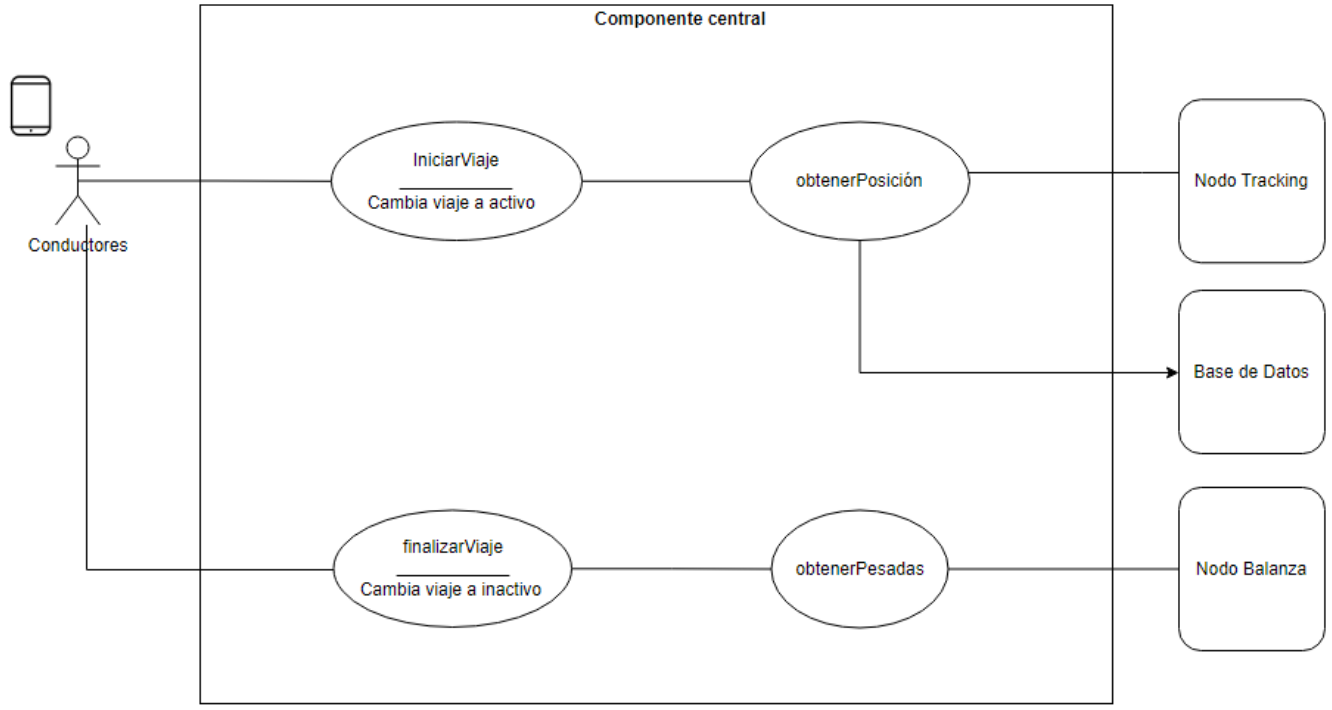


Figura 6: CU inicio/fin del viaje

3.3. Caso de Uso Crítico 1

Identificador	CU001
Nombre	Autenticación gub.uy
Descripción	Permite ingresar al sitio web desde varios dispositivos (web o móvil) al frontoffice para los usuarios Ciudadanos y Funcionarios.
Pre-Condiciones	● El usuario está dado de alta en gub.uy. ● El usuario ingresa desde *FrontOffice* web o un dispositivo móvil.
Flujo normal de eventos	1. Se ingresa a la web. 2. El usuario ingresa: usuario y contraseña (credenciales válidas). 3. Se retorna al sistema con un código y estado 4. Se usa el código para obtener un access token con la API de gub.uy 5. Se usa el access token para obtener la información del usuario con la API de gub.uy 6. Se usa la información Obtenida para verificar el usuario con la BD y la PDI 7. El usuario queda logueado en el sistema (Si no está en el sistema que lo manda al inicio para loguearse y tira un error)
Flujos alternativos	2.1. El usuario ingresa: usuario y contraseña (credenciales inválidas). 3.1. El sistema valida las credenciales en gub.uy y notifica que no son correctas las credenciales. 3.2. Se detecta que es el primer ingreso y se le solicita correo cédula, luego se lo redirige al menú principal.
Post-condiciones	Obtenemos la información del Ciudadano en el componente central para utilizar posteriormente

3.4. Caso de Uso Crítico 2

Identificador	CU002
Nombre	Inicio/fin de viaje
Descripción	Permite al conductor indicar el inicio y fin de un viaje.
Pre-condiciones	● Se encuentra un usuario (Conductor) logueado en el sistema. ● El usuario cuenta con los permisos necesarios. ● Existe el viaje en el sistema con el conductor asociado.
Flujo normal de eventos	1. Se selecciona “Iniciar el viaje”. 2. Durante el viaje el componente Nodo Tracking envía la información de la posición del conductor al componente central. 3. Se realiza el viaje. 4. Se selecciona “Finalizar viaje”. 5. Al finalizar el viaje se le solicita al Nodo Balanza las pesadas realizadas en este viaje y se guardan en la base de datos
Post-condiciones	Quedan las pesadas y tracking del viaje. El estado del viaje queda en “Finalizado”

4. Vista de Restricciones

En esta vista se muestran las restricciones que hay tanto en el desarrollo como en el producto terminado de este software las cuales pueden ser normativas, de estándares y tecnologías.

4.1. Normativas

Cumplir con las regulaciones del transporte de carga en Uruguay y las normativas del MTOP.

4.2. Nodos Periféricos

Utilizar los servicios de la Plataforma de Interoperabilidad (PDI) y proveedores de identidad (Usuario Gub.uy) para garantizar la interoperabilidad y la autenticación.

4.3. Seguridad

Garantizar la privacidad y seguridad de la información almacenada y transmitida.

4.4. Escalabilidad

Asegurar la escalabilidad de la plataforma para manejar grandes cantidades de datos y usuarios.

4.5. Compatibilidad

Interfaz Web

El usuario debe al menos contar con las siguientes versiones de los navegadores posibles para acceder a la web: Google Chrome (Versión 84.0+), Firefox (Versión 78.10+), Edge (Version 90.0+) y Opera (76.0.4+).

Interfaz Móvil

El usuario debe al menos contar con las siguientes versiones de Android para que pueda funcionar correctamente el componente móvil: Android 6.0 en adelante.

4.6. Tecnologías

El desarrollo del sistema debe estar en:

Backend

- Componente Central
  - Java EE.
  - WildFly.
  - Elastic Cloud (Antel).
- Servicios Externos
  - AGESIC
  - gub.uy

Frontend

- Web
  - JSF.
  - JSP.
  - JavaScript.
- Móvil
  - Android

5. Vista de Atributos de Calidad

Describe los requerimientos no funcionales del sistema.

5.1. Fiabilidad

La plataforma debe ser capaz de manejar grandes cantidades de datos de manera confiable y precisa. Además, debe garantizar la seguridad y privacidad de la información almacenada y transmitida.

5.2. Eficiencia

La plataforma debe ser capaz de procesar grandes cantidades de datos y proporcionar resultados en tiempo real.

5.3. Usabilidad

La plataforma debe ser fácil de usar y navegar tanto para los funcionarios del MTOP como para las empresas transportistas y los clientes.

5.4. Mantenibilidad

La plataforma debe ser fácil de mantener y actualizar, ya que se espera que se realicen cambios en las normativas y regulaciones del transporte de carga.

6. Vista Lógica

La Vista Lógica describe la arquitectura lógica de la plataforma, utilizando varios niveles de refinamiento. En particular, se presentan y describen los principales componentes lógicos de la plataforma así como sus responsabilidades, dependencias e interacciones.

6.1. Arquitectura General del Sistema

Vista de alto nivel de la arquitectura lógica del sistema.

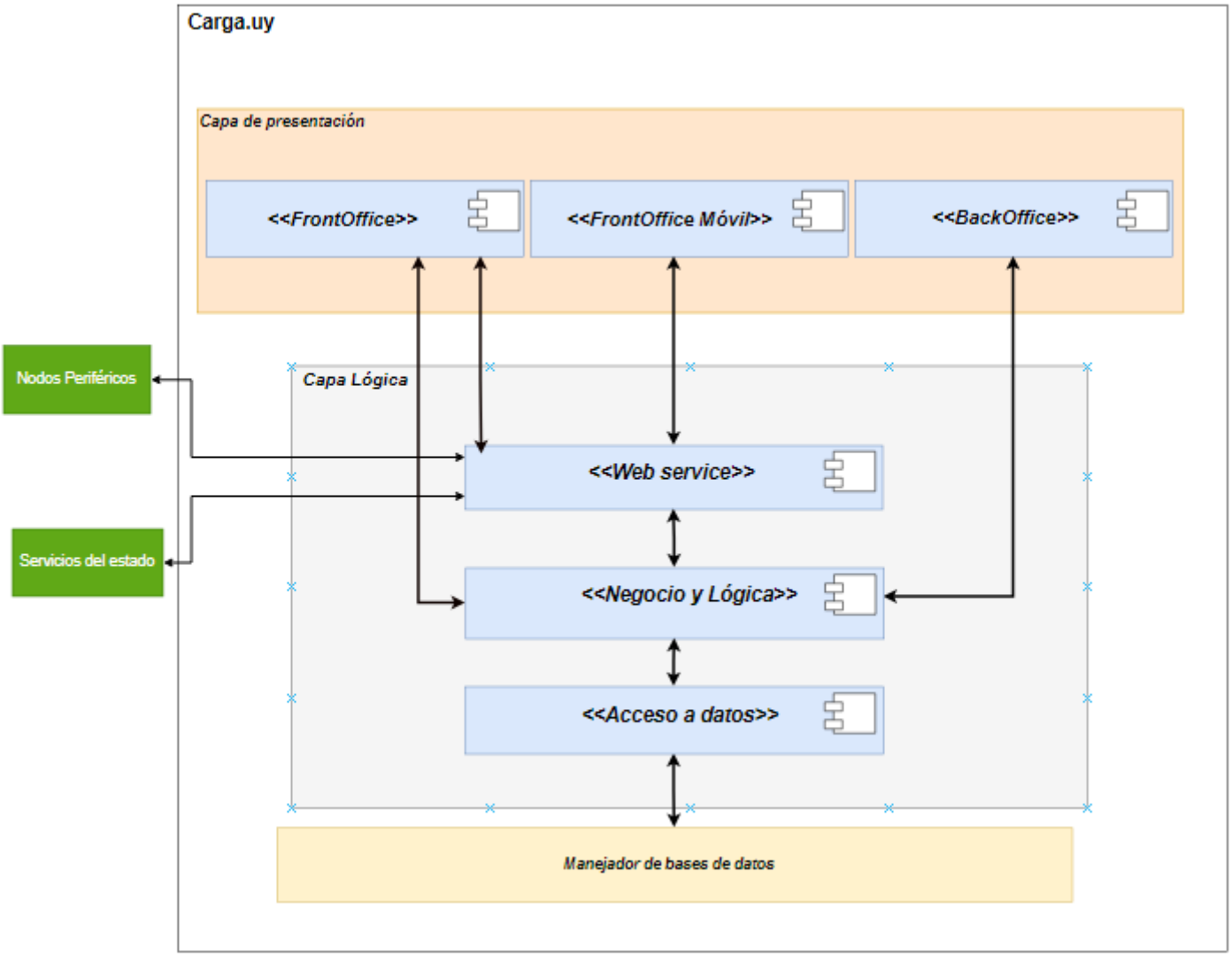


Figura 7: Vista Lógica

A grandes rasgos, el sistema se divide en tres capas principales:

La capa Lógica, define y ejecuta toda la lógica de negocio, brinda los servicios necesarios para interactuar con ella, así como también gestionar el acceso a la persistencia de los datos.

La capa de Presentación, contiene a los componentes que realizan la interacción con los usuarios finales para que estos puedan interactuar de manera sencilla y eficaz.

La capa de persistencia, es la encargada de almacenar los datos de la aplicación para su persistencia en el tiempo, uso y procesado según sea necesario.

6.2. Componentes del Sistema

Se presentan los distintos componentes que integran las capas de la arquitectura general, y el flujo de interacción entre los mismos.

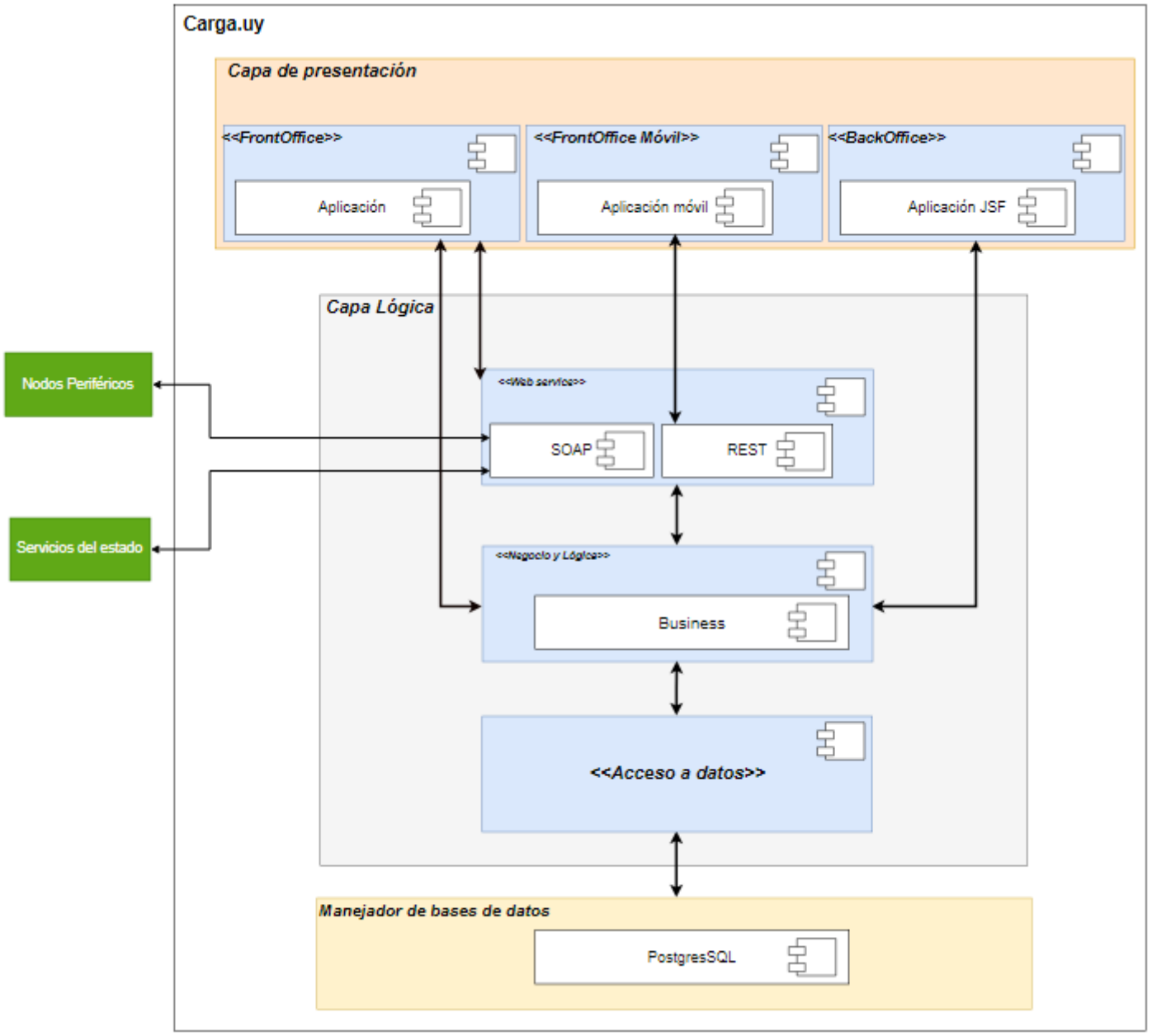


Figura 8: Vista Lógica, Componentes

Aplicación Android

Componente que sirve como interfaz para dispositivos móviles con sistema operativo Android para el uso de los ciudadanos. Consume servicios prestados por el componente REST.

Aplicación JSF

Este componente servirá como interfaz web para la interacción con las autoridades y administradores del sistema. Consume servicios prestados por el componente Servicios de Negocio.

REST

Ofrece servicios en formato de Web Services REST para que puedan ser consumidos por los componentes de Frontoffice tanto web como móvil.

Business

Implementa la lógica de todas las operaciones ofrecidas por esta capa. Se comunica directamente con el componente de acceso a datos para persistir la información.

Acceso a Datos

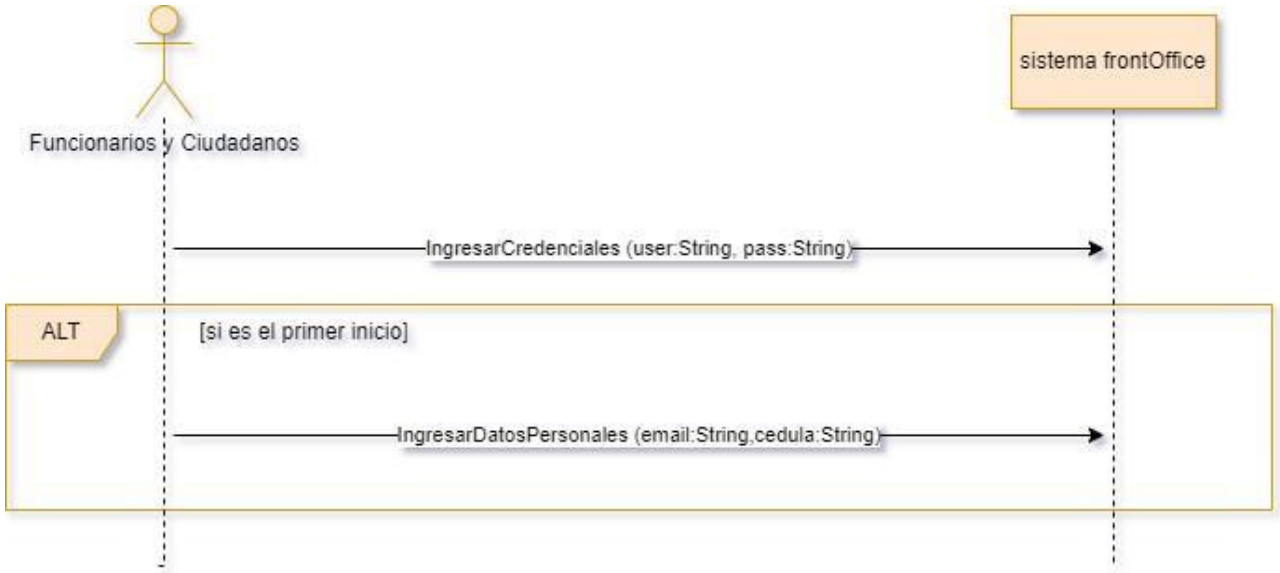
Encargado de gestionar el acceso a los datos con la capa de persistencia, ofreciendo operaciones para facilitar la interacción.

Base de datos PostgreSQL

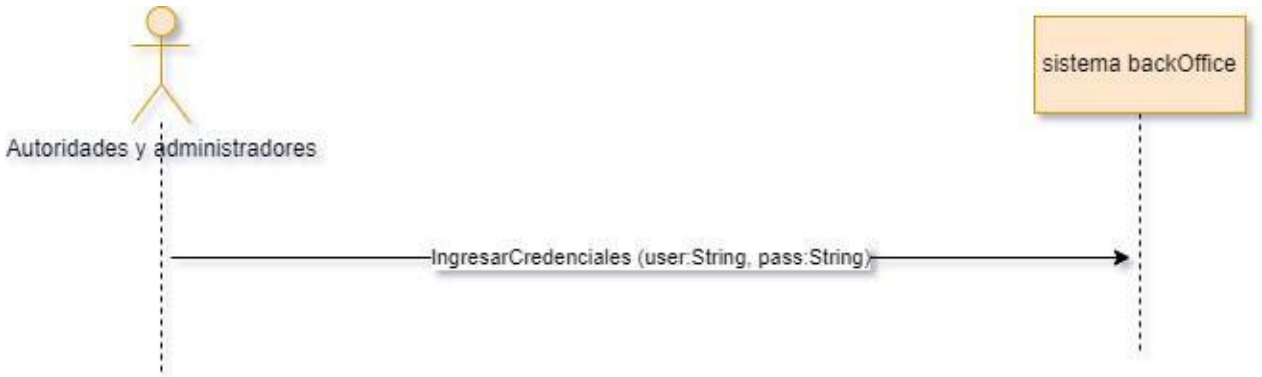
Base de datos en la cual se guarda toda la información que gestiona el sistema.

6.3. Diagramas de Secuencia del sistema

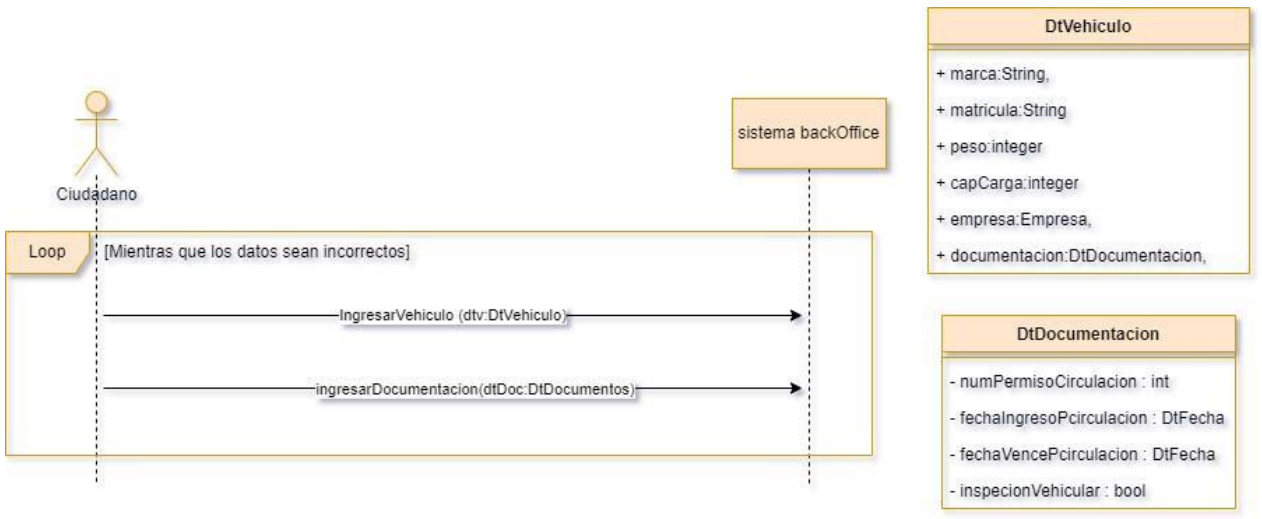
Autenticacion gub.uy



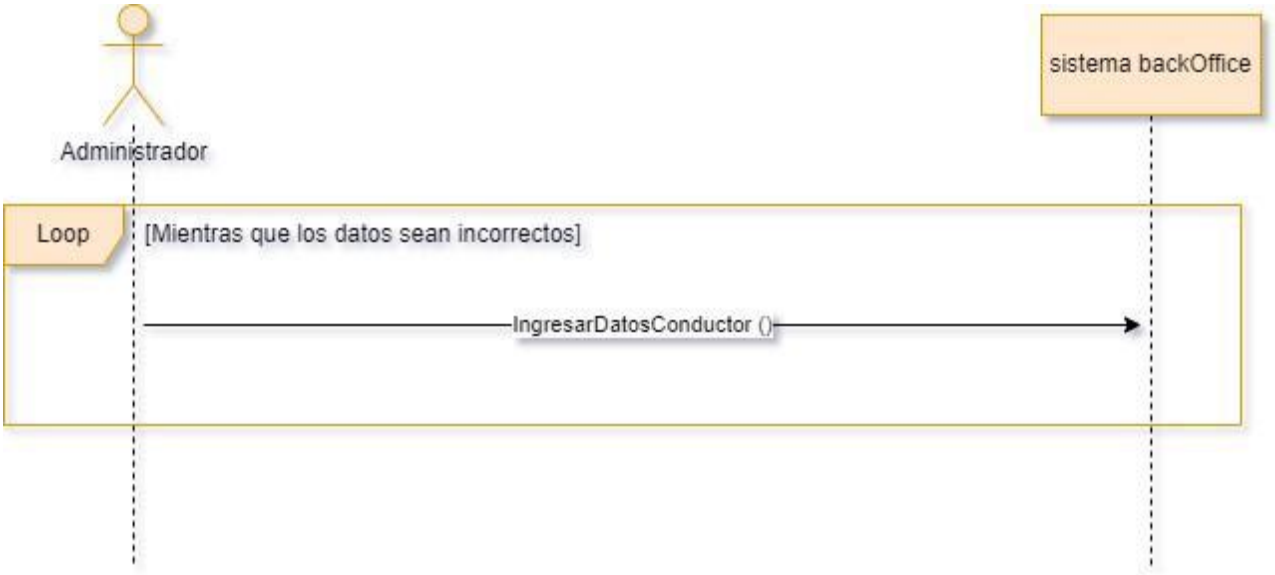
Autenticación interna



Alta vehiculo

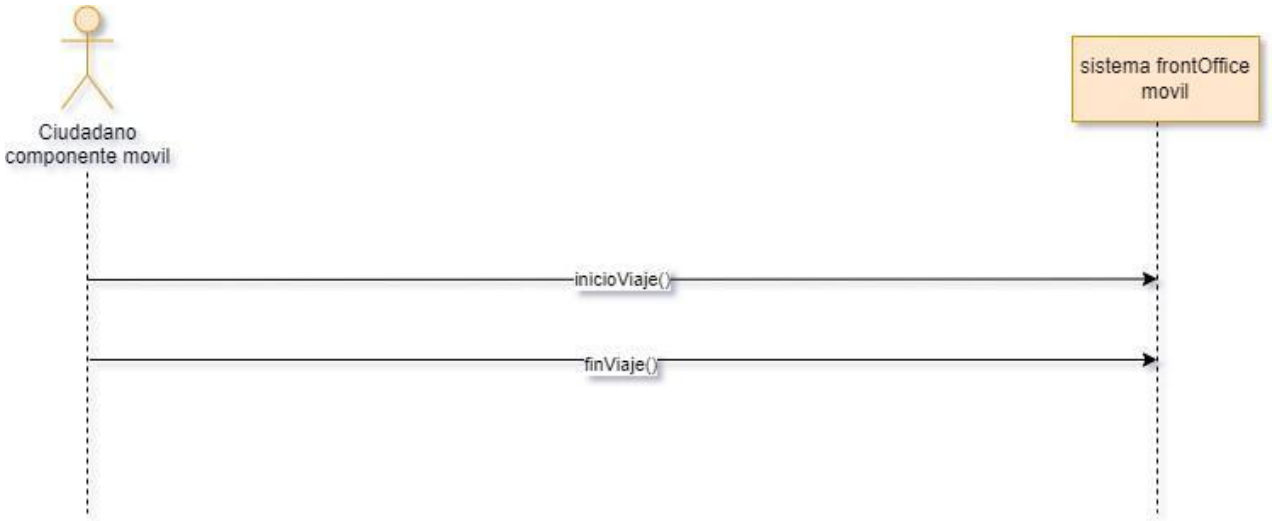


Alta conductor

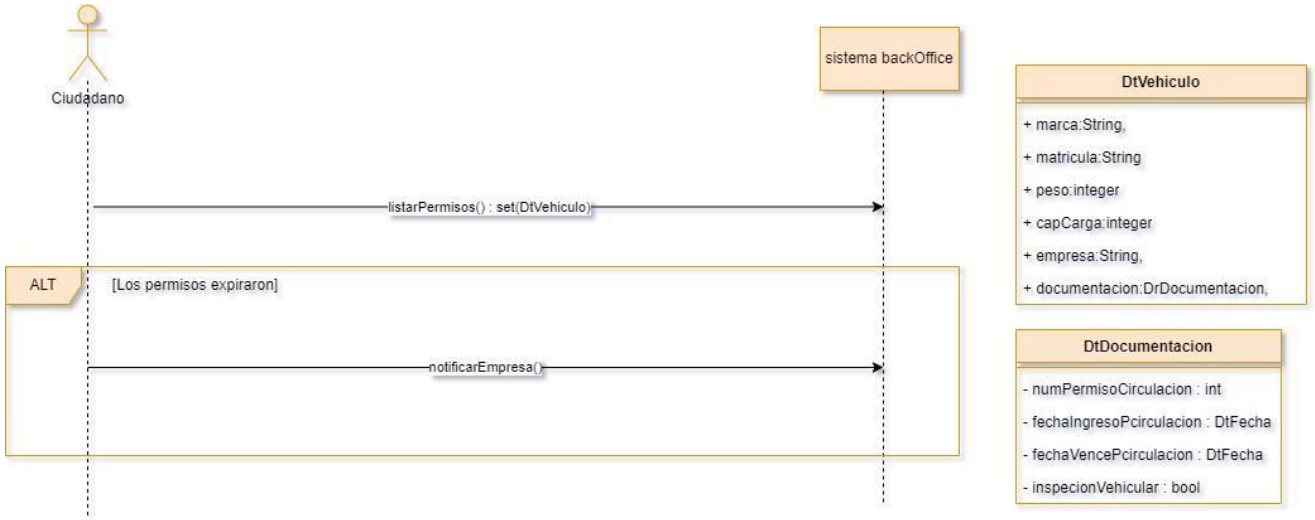


Gestión de viajes

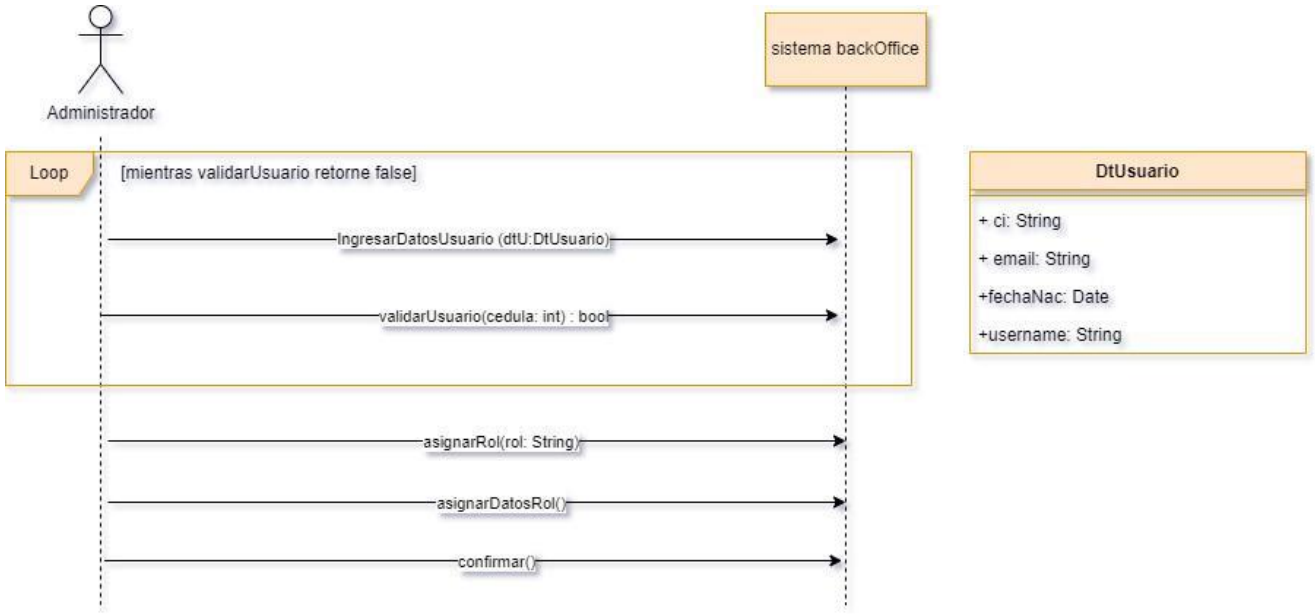
Inicio/fin de viaje



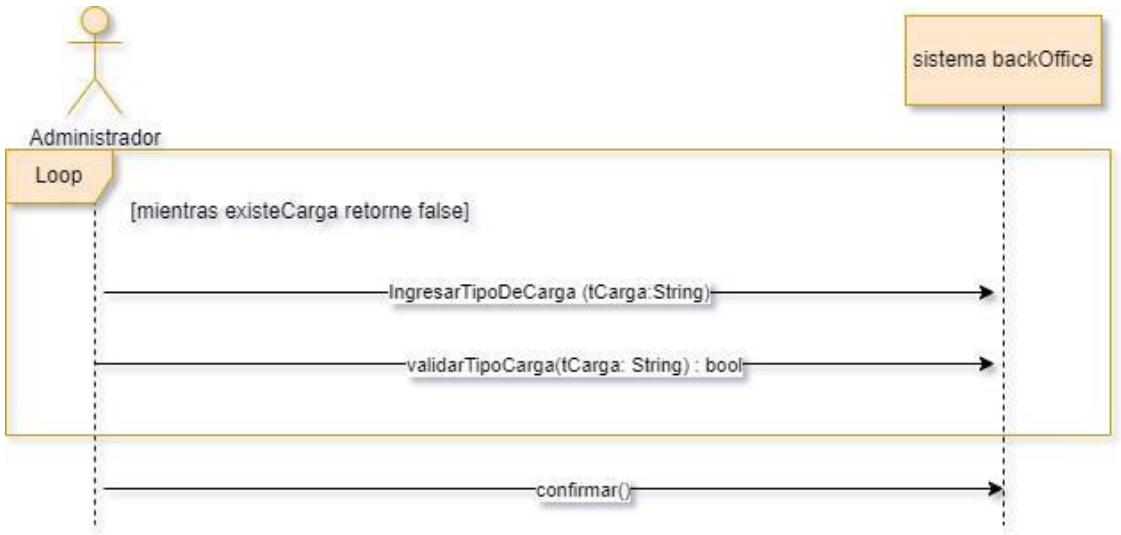
Fiscalización de permisos



Alta usuario



Alta tipo de carga



7. Vista de Distribución

En este apartado se plantea la estructura de deploy presente en la figura a continuación. La misma plantea un deploy usando la plataforma de Elastic Cloud, la cual por motivos de configuraciones y errores en la licencia, se desestimó al final para terminar usando un deploy en un servidor local. Se presenta por dentro todos los servicios expuestos en servidores Wildfly 24+ y a su vez el uso de una base de datos interna para almacenamiento de datos, todo esto con los tipos de comunicaciones y protocolos especificados. Adicionalmente dentro de cada servidor se detalla qué tipo de aplicación se encuentra y cómo será su comunicación con el resto de los demás módulos. En esta figura se muestra como nuestro backend es el núcleo de la aplicación, quien recibiendo y mandando datos genera el dinamismo y lógica necesaria para el despliegue completo, siendo estos componentes conectados su capa visual y sistemas externos para recolectar datos.

Por último y apartado de nuestro componente central, se encuentra el componente móvil, quien de forma independiente genera un stack completo para la ejecución, todo esto siendo un enlace con el componente central quien será el que guarde la preciada lógica del negocio.

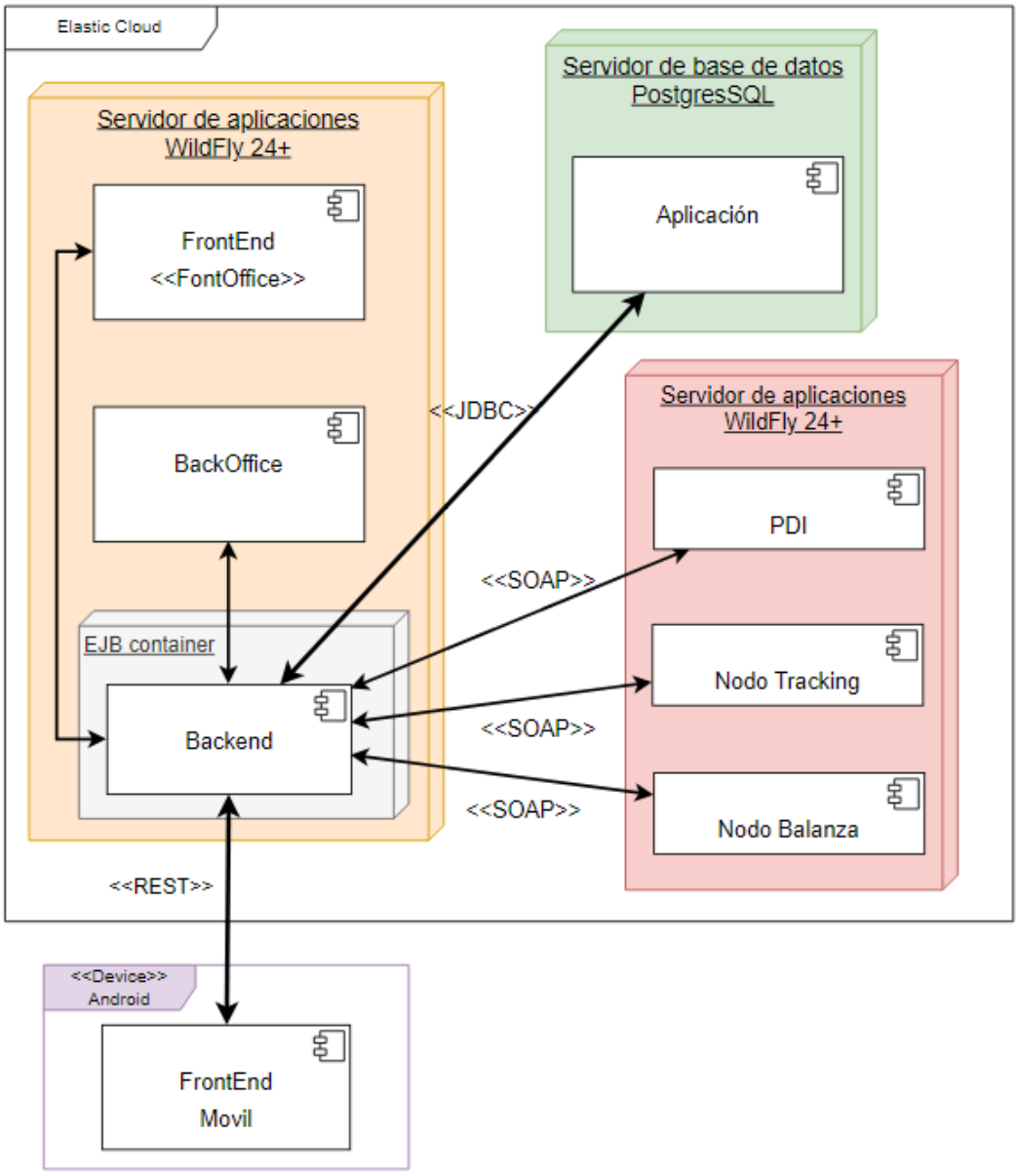


Figura 9: Vista Distribucion

7.1. Arquitectura de despliegue

Para el despliegue de la aplicación, se pensó originalmente realizar lo en Elastic cloud como fue solicitado. Debido a inconvenientes en el desarrollo por temas de licencia y configuraciones, se tuvo que migrar al uso de un servidor local. Echa la aclaración, se pasa a explicar el despliegue del mismo en local.

Primero se desplegó el componente central junto a los nodos en un mismo Wildfliy 24+ (separado en el diagrama debido a que originalmente se pensó en desplegarlo en Elastic cloud en diferentes wildFly 24+), con esto se generó el despliegue de del componente central junto a los 2 nodos y la PDI. Estos dos últimos se comunican gracias a los WS SOAP ofrecidos por el componente central.

Luego se realizó el deploy del nodo servidor de base de datos, donde usando el manejador de bases de datos Postgres SQL, se provee el almacenamiento de los datos, manteniendo la integridad gracias al diseño relacional con el que cuenta.

Por último el módulo móvil se levanta en otro componente aparte para poder generar un apartado separado, siendo consumidor de los servicios del componente central y su backend a través del uso de REST para consumir operaciones desde un dispositivo portable emulado.

De contar con la posibilidad se hubiese realizado un despliegue en Elastic cloud con diferentes nodos, uno para el componente central y otro para los nodos, como se ve en la figura 9, esto permitiría representar más realmente lo que ocurriría en una puesta a producción real. Por otro lado si se contará con otros servicios gratuitos como el anterior Hiroku, algo similar a Google cloud o Amazon, se buscaría hacer el despliegue en diferentes servicios y pasar a algo cien porciento real; pudiendo ver como realmente los nodos externos con unos ajustes de url en sus web services y clientes correspondientes, se conectan a través de internet y pueden comunicarse.

8. Vista de Implementación

La vista de implementación se centra en los componentes en tiempo de ejecución que forman el sistema.

8.1. Estructura de la Aplicación

El sistema se desarrolla como una aplicación Java EE. La capa de cliente tiene dos interfaces, una aplicación Web con páginas dinámicas y una aplicación móvil para Android. La capa de negocio cuenta con dos JAR; cargaUy.jar, cargaUyPDI.jar. El componente carga y PDI.jar depende de Web Services.

8.2. Diagrama de la Aplicación

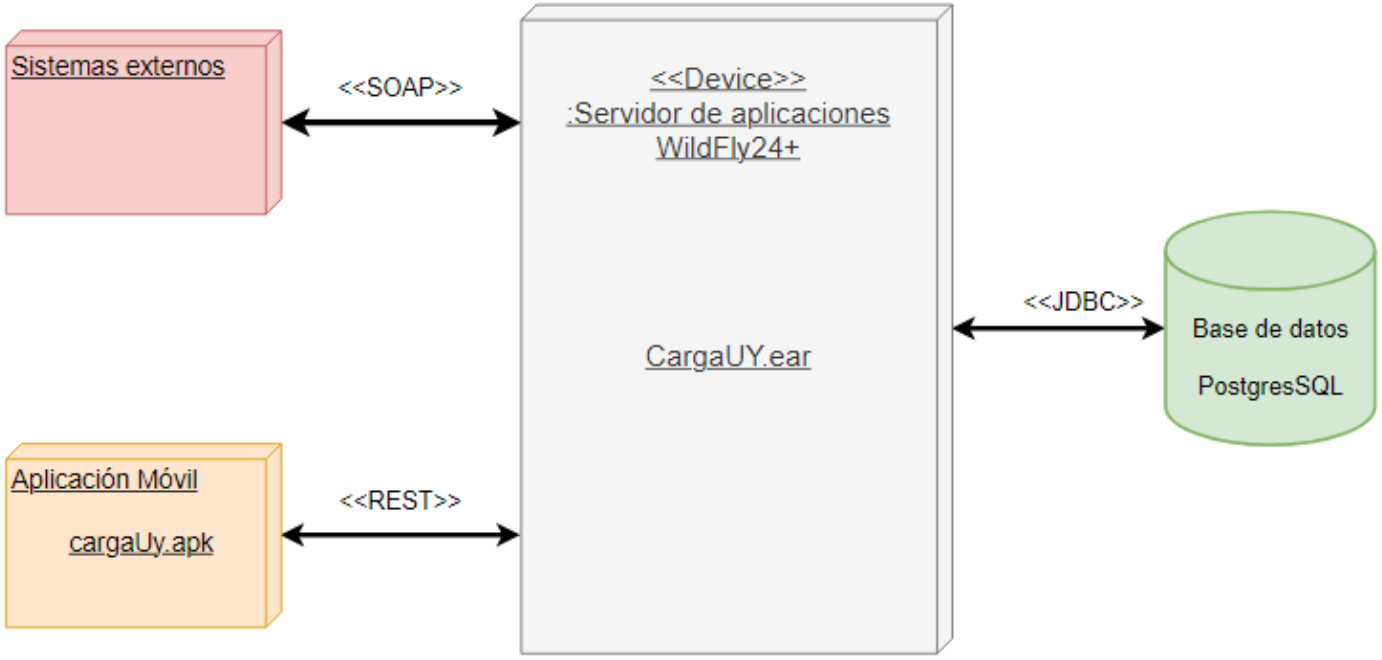


Figura 10: diagrama de implementación resumido

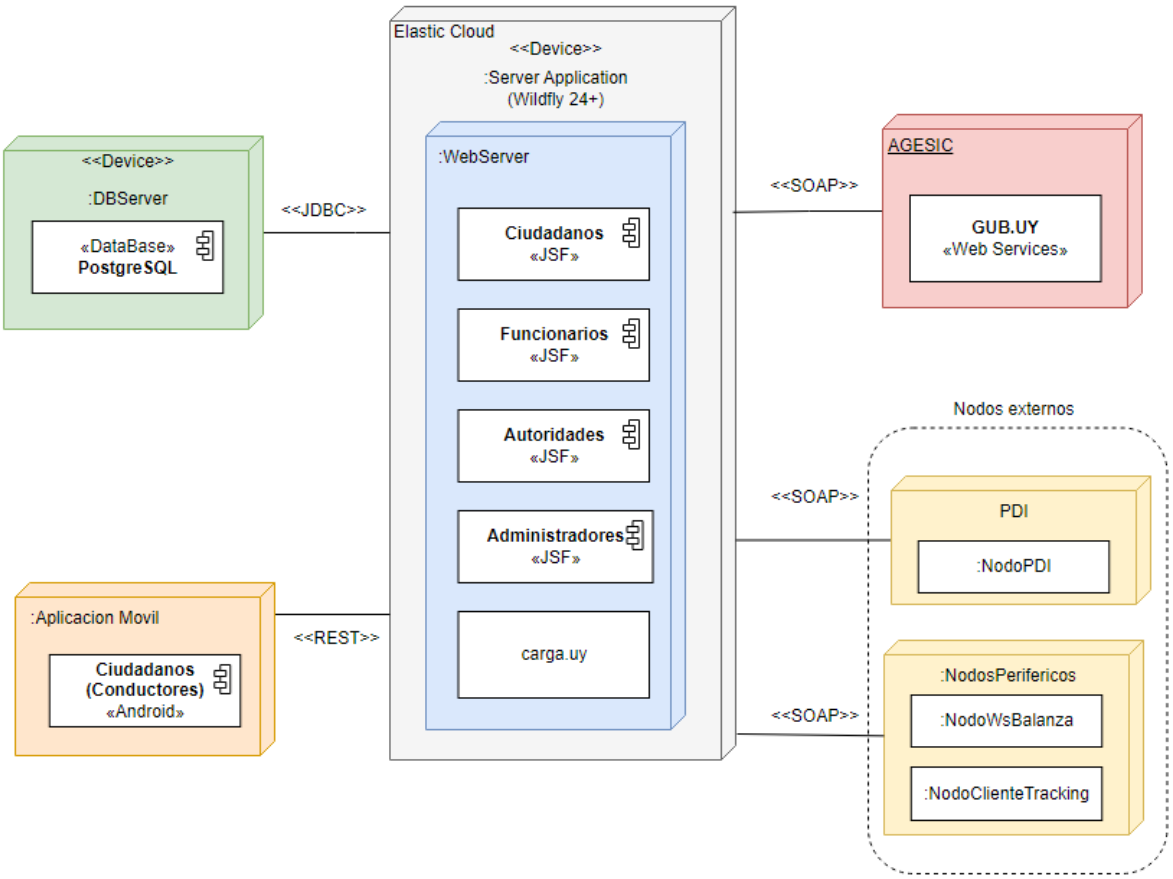


Figura 11: Vista Implementación

9. Vista de Decisiones de Arquitectura

La Vista de Decisiones de Arquitectura presenta y describe las principales decisiones de arquitectura tomadas.

Decisión de Arquitectura 1

Identificador	ID01
Nombre	Plataforma
Categorías	Desarrollo
Problema	La solución planteada cuenta con un componente central y Un componente móvil, el cual interactúa con nodos periféricos y externos. Estos elementos permiten brindar funcionalidades a cuatro tipos de usuarios: Ciudadanos, Autoridades, Funcionarios y Administradores.
Alternativas	- - - -
Decisión	Se implementará con JEE y Android.
Justificación	Solicitud del cliente.
Decisiones Relacionadas	ID02

Decisión de Arquitectura 2

Identificador	ID02
Nombre	Software
Categorías	Plataforma
Problema	Se define el software que cumpla los requerimientos del sistema a desarrollar.
Alternativas	JakartaEE, PostgreSQL, PostGIS, Swift(Android), Java(Android).
Decisión	JakartaEE, PostgreSQL, Java(Android).
Justificación	Solicitud del cliente y conocimiento del equipo de desarrollo.
Decisiones Relacionadas	ID03, ID04, ID05

Decisión de Arquitectura 3

Identificador	ID03
Nombre	Distribución del sistema
Categorías	Plataforma
Problema	Solución
Alternativas	Distribuida, no distribuida
Decisión	Se desarrollara un sistema en capas, tanto lógicas como físicas
Justificación	Segun documentacion, un sistema distribuido favorece la tolerancia a fallos futuros.
Decisiones Relacionadas	- - - -

Decisión de Arquitectura 5

Identificador	ID05
Nombre	Servidor de aplicaciones
Categorías	Plataforma
Problema	Se debe determinar qué servidor de aplicaciones es mas optimo.
Alternativas	GlassFish, WildFly, Apache Tomcat.
Decisión	WildFly
Justificación	Solicitud del cliente.
Decisiones Relacionadas	- - - -

Decisión de Arquitectura 6

Identificador	ID06
Nombre	Autenticación
Categorías	Restricciones
Problema	Es necesario autenticar a los funcionarios y ciudadanos a través de ”gub.uy”.
Alternativas	- - - -
Decisión	API de autenticación de ”gub.uy”.
Justificación	Solicitud del cliente.
Decisiones Relacionadas	- - - -

Decisión de Arquitectura 7

Identificador	ID07
Nombre	Frontend (Framkework)
Categorías	Desarrollo
Problema	Es necesario definir en qué entorno se va a desarrollar el frontend del sistema.
Alternativas	React, Flutter, Angular, Plantillas externas.
Decisión	JSF, Android, Primefaces(plantilla externa).
Justificación	Oportunidad de aprendizaje y experiencia en el equipo de desarrollo, además de agilizar el avance.
Decisiones Relacionadas	- - - -

Referencias

[1] Philippe B Kruchten. ‹ The 4+ 1 view model of architecture› . En: *IEEE software* 12.6 (1995), págs. 42-50.

[2] D Perovich y A Vignaga. *SAD del Subsistema de Reservas del Sistema de Gestión Hotelera*. Inf. téc. Thechnical Report RT03-15, InCo Pedeciba, Montevideo, Uruguay, 2003.

[3] PrimeFaces, documentación. 2023 URL: https://www.primefaces.org/

[4] Todos los diagramas UML. URL: https://diagramasuml.com/

[5] EVA - Taller de sistemas empresariales. URL: https://eva.fing.edu.uy/enrol/index.php

[6] Mi nube Antel. URL: https://minubeantel.uy/index.php?NAME_PATH=Elastic_Cloud

[7] Agencia de gobierno electrónico y sociedad de la información. 2023 URL: https://www.gub.uy/agencia-gobierno-electronico-sociedad-informacion-conocimiento/

[8] Crear un diagrama de implementación UML. URL: https://support.microsoft.com/es-es/office/crear-un-diagrama-de-implementaci%C3%B3n-uml-60811688-d811-4387-9715-c1f02f30b125

[9] La guía sencilla para la diagramación de UML y el modelado de la base de datos. septiembre 24, 2019. URL: https://www.microsoft.com/es-ww/microsoft-365/business-insights-ideas/resources/guide-to-uml-diagramming-and-database-modeling