

**[Nombre del proyecto]**  
**(DAS) Documento Arquitectura de Software**  
**Versión 1.0**

## Identificación de Documento

<b>Identificación</b>	001-002-003
<b>Proyecto</b>	Sistema de Automatización de Procesos Aduaneros
<b>Versión</b>	1.8

<b>Documento mantenido por</b>	Fernando soto,daniel godoy, brayan godoy,patricio maldonado
<b>Fecha de última revisión</b>	27-05-2025
<b>Fecha de próxima revisión</b>	01-07-2025

<b>Documento aprobado por</b>	Mabel Herrera Pino
<b>Fecha de última aprobación</b>	

## Historia de Revisiones

Fecha	Versión	Descripción	Autor
13-05-2025	1.0	Introducción y visión del sistema	Todos los integrantes
23-05-2025	1.2	revisión completa y agregar información al documento	Todos los integrantes
29-05-2025	1.4	completar información faltante y adjuntar diagramas	Todos los integrantes
01-07-2025	1.8	Nueva versión del documento, adjuntando información sugerida en base a lo nuevo que se encuentra, control de versiones-prototipo-evaluación calidad	Todos los integrantes

## Tabla de Contenidos

<b>1. INTRODUCCIÓN</b>	<b>4</b>
1.1. CONTEXTO DEL PROBLEMA	4
1.2. PROPÓSITO	4
1.3. ÁMBITO	4
1.4. DEFINICIONES, ACRÓNIMOS Y ABREVIACIONES	4
1.5. RESUMEN EJECUTIVO	4
1.6. ARQUITECTURA DEL SISTEMA	4
<b>2. VISIÓN DEL SISTEMA</b>	<b>4</b>
2.1. DESCRIPCIÓN GENERAL DEL SISTEMA	5
2.2. OBJETIVOS DEL SISTEMA	5
2.3. PRINCIPALES FUNCIONALIDADES ESPERADAS	5
2.4. SUPUESTOS Y DEPENDENCIAS	5
<b>3. ESTILOS Y PATRONES ARQUITECTÓNICOS</b>	<b>5</b>
3.2. JUSTIFICACIÓN DEL ESTILO SEGÚN EL CONTEXTO DEL SISTEMA	5
<b>4. MODELO 4 +1 Y VISTAS ARQUITECTÓNICAS</b>	<b>5</b>
4.1. VISTA DE ESCENARIO	5
4.1.1. <i>Propósito</i>	5
4.1.2. <i>Actores</i>	5
4.1.3. <i>Diagrama general de casos de uso</i>	5
4.1.4. <i>Diagrama de casos de uso específicos</i>	5
4.1.5. <i>Lista de casos de uso</i>	5
4.1.6. <i>Especificación de casos de uso</i>	5
4.2. VISTA LÓGICA	7
4.2.1. <i>Propósito</i>	7
4.2.2. <i>Diagrama de clases</i>	7
4.2.3. <i>Descripción diagrama de clases</i>	7
4.3. VISTA DE IMPLEMENTACIÓN/DESARROLLO	7
4.3.1. <i>Propósito</i>	7
4.3.2. <i>Diagrama de componente</i>	7
4.3.3. <i>Descripción diagrama de componente</i>	7

4.3.4.	<i>Diagrama de paquete</i>	7
4.3.5.	<i>Descripción diagrama de paquete</i>	7
4.4.	<b>VISTA DE PROCESOS</b>	7
4.4.1.	PROPÓSITO	7
4.4.2.	DIAGRAMA DE ACTIVIDAD	7
4.4.3.	DESCRIPCIÓN DIAGRAMA DE ACTIVIDAD	7
4.5.	<b>VISTA FÍSICA</b>	7
4.5.1.	<i>Propósito</i>	7
4.5.2.	<i>Diagrama de despliegue</i>	7
4.5.3.	<i>Descripción diagrama de despliegue</i>	7
5.	<b>REQUISITOS DE CALIDAD</b>	7
5.1.	PROPÓSITO	7
5.3.	<i>Reglas y criterios de evaluación de calidad</i>	7
6.	<b>PRINCIPIOS DE DISEÑO APLICADOS</b>	8
6.1.	<i>Propósito</i>	8
6.4.	<i>Diseño centrado en el usuario (UX/UI, prototipos, experiencia de usuario)</i>	8
7.	<b>CONCLUSIONES</b>	8
8.	<b>BIBLIOGRAFÍA</b>	8

## INTRODUCCIÓN

### 1.1. Contexto del Problema (General)

El proceso actual para ingresar a Chile en un vehículo particular implica pasar por varios controles manuales y tratar con diferentes agencias gubernamentales, como el Servicio Nacional de Aduanas, la Policía de Investigaciones (PDI) y el Servicio Agrícola y Ganadero (SAG). Todo esto puede causar demoras considerables en los pasos fronterizos y complica la trazabilidad de la información. Es necesario contar con un sistema que digitalice y centralice el proceso de declaración para facilitar el cruce de fronteras.

### 1.2. Propósito

Este documento define la arquitectura de software para el "Sistema de Ingreso de Vehículos a Chile". Su finalidad es detallar la estructura, los componentes, las interacciones y las vistas arquitectónicas del sistema, utilizando el modelo 4+1, para asegurarse de que se cumplen las necesidades de todos los interesados.

### 1.3. Ámbito

El alcance de este sistema se limita a la entrada a Chile de un ciudadano en un vehículo particular a través de un paso fronterizo terrestre. Esto abarca la presentación digital del formulario de declaración, la validación por parte de los funcionarios de Aduana y las consultas de interoperabilidad con los sistemas del SAG y la PDI. No se contempla el proceso de salida del país.

### 1.4. Definiciones, acrónimos y abreviaciones

ACRONIMO	DESCRIPCION
DAS	Documento de Arquitectura de Software
PDI	Policía de Investigaciones de Chile
SAG	Servicio Agrícola y Ganadero
CZI	Certificado Zoosanitario de Importación
MVC	Patrón de diseño Modelo-Vista-Controlador
DAO	Objeto de Acceso a Datos (Data Access Object)

### 1.5. Resumen ejecutivo (General)

Este documento detalla cómo está estructurada una aplicación web diseñada para modernizar el ingreso de vehículos a Chile. Gracias a esta solución, los ciudadanos podrán hacer su declaración de manera anticipada a través de un portal en línea. La arquitectura se organiza en un estilo en capas y sigue el patrón MVC, con un servidor central de Aduanas que se comunica con

sistemas externos de otras agencias gubernamentales a través de servicios. A continuación se describen las vistas lógica, de desarrollo, de procesos y física del sistema.

#### 1.6. Arquitectura del sistema (General)

(ej. vista de escenario, vista lógica, vista de desarrollo, vista de proceso, vista física)

## 2. VISIÓN DEL SISTEMA (General)

## 2.1. Descripción general del sistema

Este sistema web está orientado a digitalizar y acelerar el proceso de ingreso de vehículos en los controles fronterizos chilenos. Los ciudadanos podrán realizar en línea la "Declaración Jurada Conjunta" de Aduanas y SAG, lo que permitirá a los funcionarios validar la información de manera más eficaz.

## 2.2. Objetivos del sistema

- Reducir el tiempo de espera en los cruces fronterizos.
- Unificar y digitalizar el proceso de declaración de vehículos y mercancías.
- Fomentar la interoperabilidad entre los sistemas de Aduana, SAG y PDI.
- Mejorar la fiscalización y el control sanitario, previniendo el ingreso de productos peligrosos o ilegales.

## 2.3. Principales funcionalidades esperadas

- Dar la opción a los ciudadanos de completar en línea el formulario de declaración de ingresos.
- Validar la información del vehículo y del propietario.
- Permitir la declaración de mascotas y la inclusión del Certificado Zoosanitario de Importación (CZI).
- Integrar la validación migratoria con los sistemas de la PDI.
- Conectar el control sanitario de productos con los sistemas del SAG.
- Proveer una interfaz para que los fiscalizadores aduaneros registren la revisión del equipaje y del vehículo.

## 2.4. Supuestos y dependencias

Supuestos: Se asume que los ciudadanos cuentan con acceso a internet para completar el formulario en línea. Se asume que los sistemas externos (PDI, SAG) dispondrán de APIs estables para la integración.

Dependencias: El sistema depende críticamente de la disponibilidad y correctitud de los servicios de validación proporcionados por los sistemas de PDI y SAG.

## 3. ESTILOS Y PATRONES ARQUITECTÓNICOS (General)

### 3.1. Estilo arquitectónico adoptado (ej. monolítico, microservicios, SOA, capas)

Se utiliza una arquitectura en capas para la aplicación principal. Para la comunicación con otras agencias del estado, se aplica un enfoque de Arquitectura Orientada a Servicios (SOA), donde el sistema de Aduanas aprovecha los servicios que ofrecen los sistemas de PDI y SAG.

### 3.2. Justificación del estilo según el contexto del sistema

La arquitectura en capas establece una separación clara de responsabilidades, incluyendo la presentación, la lógica y los datos, lo que hace que la mantenibilidad y el desarrollo en paralelo sean mucho más sencillos. En este contexto, el enfoque SOA es el más adecuado, ya que las agencias gubernamentales son entidades independientes con sus propios sistemas. Este desacoplamiento garantiza que si un sistema externo, como el SAG, falla, no se detenga por completo el funcionamiento del sistema de Aduanas.

### 3.3. Patrones de diseño aplicados (ej. patrón MVC, repositorio, etc.)

- Modelo-Vista-Controlador (MVC): Se utiliza para estructurar la aplicación web, como se evidencia en el diagrama de paquetes, separando la interfaz (Vista), la lógica de negocio (Controlador) y los datos (Modelo).
- Data Access Object (DAO): Se aplica para abstraer y encapsular el acceso a la base de datos, gestionando las conexiones y operaciones de persistencia.



## 4. MODELO 4 +1 Y VISTAS ARQUITECTÓNICAS

### 4.1. VISTA DE ESCENARIO (General y salida vehículo o entrada vehículo)

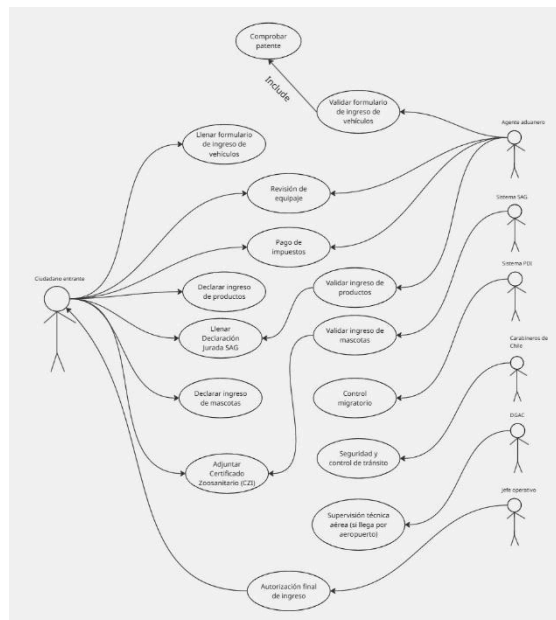
#### 4.1.1. Propósito (General)

Esta vista describe la funcionalidad del sistema desde la perspectiva de los usuarios finales a través de escenarios clave, validando que la arquitectura cumple con los requisitos.

#### 4.1.2. Actores (General)

- Ciudadano Entrante
- Agente Aduanero (Fiscalizador/a aduanero/a)
- Sistema SAG
- Sistema PDI
- Carabineros de Chile
- Jefe Operativo
- DGAC

#### 4.1.3 Diagrama general de casos de uso (General)



#### 4.1.4 Diagrama de casos de uso específicos (salida vehículo o entrada vehículo)

#### 4.1.5 Lista de casos de uso (salida vehículo o entrada vehículo)

Código	Nombre	Actores
CU-001	Realizar declaración de ingreso de vehículo	Ciudadano Entrante
CU-002	Validar formulario de ingreso	Agente Aduanero
CU-003	Validar control migratorio	Sistema PDI
CU-004	Validar control fitozoosanitario	Sistema SAG
CU-005	Registrar revisión de equipaje	Agente Aduanero
CU-006	Procesar pago de impuestos	Agente Aduanero

#### 4.1.6 Especificación de casos de uso (UN caso de uso principal de la salida vehículo/entrada vehículo)

<b>Caso de Uso</b>	"Realizar declaración de ingreso de vehículo"	<b>Identificador:</b> "CU-001"
<b>Actores</b>	Ciudadano Entrante	
<b>Tipo</b>	[Tipo de caso de uso, primario, secundario, opcional]	
<b>Referencias</b>	[Requerimientos o funcionalidades incluidas en este caso de uso. Casos de uso relacionados.]	
<b>Precondición</b>	El ciudadano ha accedido al portal web del sistema de Aduanas.	
<b>Postcondición</b>	Se genera un formulario de ingreso en estado "Pendiente de Revisión"	
<b>Descripción</b>	El ciudadano completa y envía un formulario electrónico con sus datos, los del vehículo, y la declaración de bienes o mascotas que ingresa al país.	
<b>Resumen</b>	El ciudadano llena el formulario en línea para agilizar el trámite en el paso fronterizo.	

#### CURSO NORMAL

Nro.	Ejecutor	Paso o Actividad
1	Ciudadano	Accede a la sección "Ingreso de Vehículo".
2	Sistema	Presenta el formulario de Declaración

		Jurada Conjunta
3	Ciudadano	Ingresa sus datos personales, los del propietario y los del vehículo.
4	Sistema	Pregunta si declara productos de origen vegetal/animal o mascotas
5	Ciudadano	Si declara una mascota, adjunta el Certificado Zoosanitario de Importación (CZI)
6	Ciudadano	Envía la declaración.
7	Sistema	Valida formalmente los datos, genera un código de solicitud y lo deja en estado "Pendiente de Revisión".
<p>[Se describe el proceso o secuencia de pasos ejecutados usando frases cortas]  [Cada paso del proceso puede ser ejecutado por los Actores o por el sistema]  [Se describe la secuencia de acciones realizadas por los actores y la secuencia de actividades realizada por el sistema como respuesta].</p>		

#### CURSO ALTERNATIVO

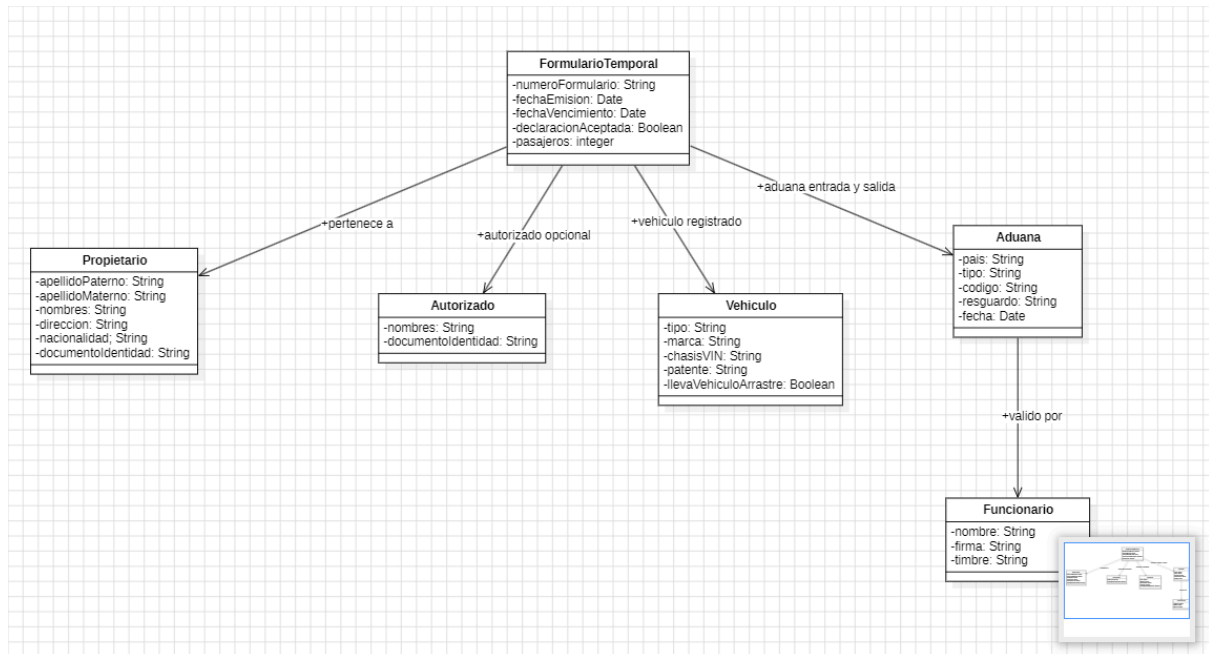
Nro .	Descripción de acciones alternas
7	Si faltan datos obligatorios en el formulario, el sistema muestra un mensaje de error y resalta los campos que deben ser corregidos, no permitiendo el envío hasta que estén completos.
<p>[Cada paso descrito en el curso normal, puede tener actividades alternas, según la distribución de escenarios que ocurra en el flujo de procesos, en esta ficha se completa para cada actividad (haciendo referencia a su número) las posibles secuencias alternas]</p>	

## 4.2. VISTA LÓGICA (salida vehículo o entrada vehículo)

### 4.2.1.1. Propósito

Describir la funcionalidad del sistema en términos de su estructura estática: clases, sus atributos, operaciones y relaciones.

### 4.2.1.2. Diagrama de clases



### 4.2.1.3. Descripción diagrama de clases

El diagrama de clases modela la estructura de datos del sistema, centrando el diseño en la clase **FormularioTemporal**. Esta clase principal agrupa toda la información de una declaración de ingreso, asociándola con un **Propietario**, un **Vehiculo** y, opcionalmente, un conductor **Autorizado**. Además, el diagrama vincula la declaración con la **Aduana** donde ocurre el trámite y el **Funcionario** que lo valida.

## 4.3. VISTA DE IMPLEMENTACIÓN/DESARROLLO (salida vehículo o entrada vehículo)

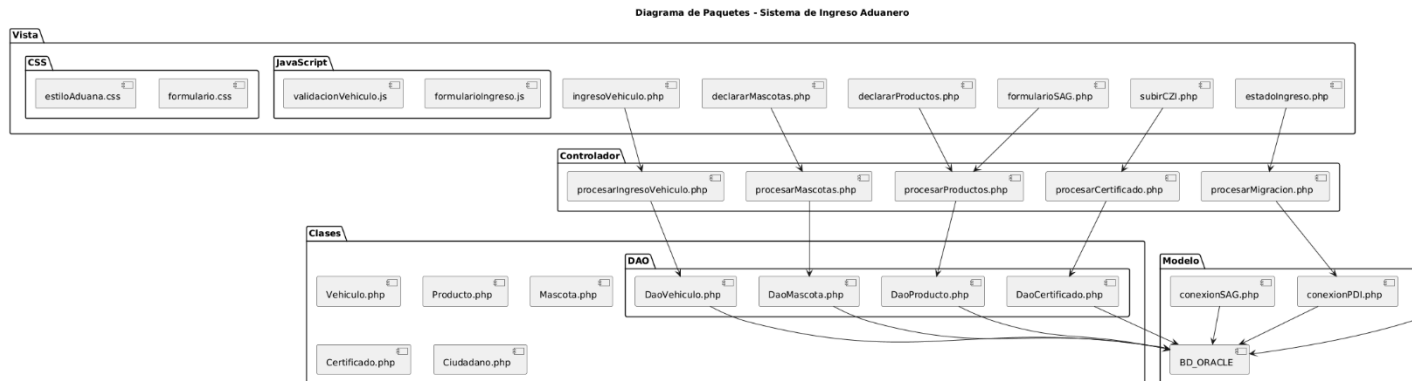
### 4.3.1. Propósito

Describir cómo se organiza el software en módulos dentro del entorno de desarrollo.

### 4.3.2. Diagrama de componente

#### 4.3.3. Descripción diagrama de componente

#### 4.3.4. Diagrama de paquete



#### 4.3.5. Descripción diagrama de paquete

La organización del código fuente sigue el patrón MVC. En el paquete Vista, se almacenan los archivos de presentación, como CSS, JS y plantillas PHP. El Controlador es responsable de la lógica de la aplicación. Los paquetes Clases y DAO forman el modelo, manejando las entidades de negocio y la interacción con la base de datos BD\_ORACLE.

### 4.4. VISTA DE PROCESOS (salida vehículo o entrada vehículo)

#### 4.4.1. Propósito

Describir los aspectos dinámicos del sistema, como los flujos de proceso, la concurrencia y la comunicación entre subprocessos.

#### 4.4.2. Diagrama de actividad

#### 4.4.3. Descripción diagrama de actividad

El proceso comienza cuando el Ciudadano envía el formulario en línea. Esto inicia

flujos de validación que pueden ejecutarse en paralelo:

- 1) el Agente Aduanero recibe una notificación para revisar el formulario
- 2) el sistema consulta al Sistema PDI para el control migratorio
- 3) el sistema consulta al Sistema SAG para el control de productos y mascotas.

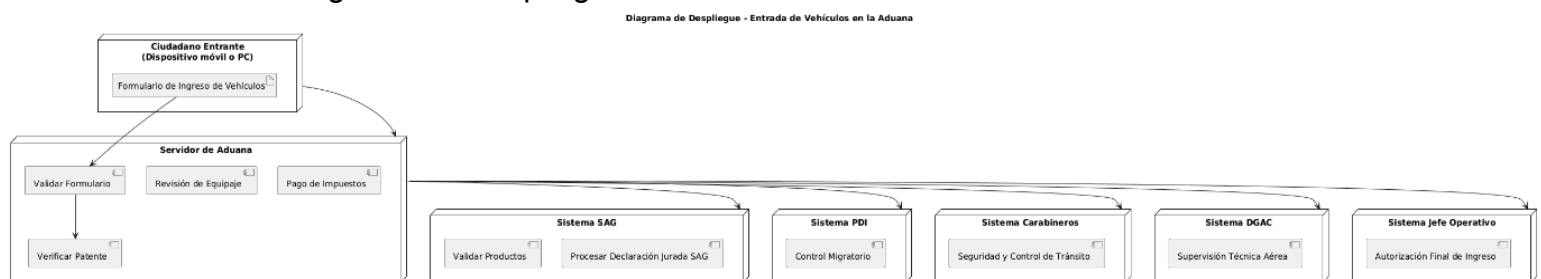
Tras las validaciones remotas, el Agente Aduanero realiza la inspección física si corresponde y finalmente otorga la autorización de ingreso.

## 4.5. VISTA FÍSICA (salida vehículo o entrada vehículo)

### 4.5.1. Propósito

Mostrar la topología del hardware sobre la cual se despliega el software y las conexiones de red entre los nodos.

### 4.5.2. Diagrama de despliegue



### 4.5.3. Descripción diagrama de despliegue

El Ciudadano Entrante accede al formulario web desde su dispositivo, ya sea una PC o un móvil. Este cliente se comunica con el Servidor de Aduana, donde se ejecuta la aplicación principal. A su vez, este servidor se conecta a través de la red con nodos de sistemas externos como el Sistema SAG, el Sistema PDI, el Sistema Carabineros y otros, para consolidar toda la información necesaria para la autorización del ingreso.

## 5. REQUISITOS DE CALIDAD (General)

### 5.1. Propósito

El propósito de esta sección es aclarar los atributos de calidad, también conocidos como requisitos no funcionales. Estos son los criterios que se emplearán para juzgar cómo funciona el sistema, en lugar de centrarse en sus

comportamientos específicos. Estos requisitos son clave para orientar las decisiones de diseño arquitectónico y garantizar que el producto final sea fácil de usar, eficiente, seguro y mantenible.

## 5.2. Atributos de calidad (por ejemplo: Usabilidad, Accesibilidad (WCAG),

ATRIBUTO DE CALIDAD	DESCRIPCION	JUSTIFICACIÓN
Usabilidad	La interfaz debe ser intuitiva y fácil de usar para ciudadanos con distintos niveles de alfabetización digital.	Para asegurar una alta tasa de adopción y reducir errores en la completitud de los formularios.
Rendimiento	El tiempo de carga de la página y el envío del formulario no debe exceder los 3 segundos.	Para evitar la frustración del usuario y agilizar el proceso en el paso fronterizo.
Fiabilidad	El sistema debe tener una disponibilidad del 99.5% durante el horario de operación de los pasos fronterizos	Una caída del sistema generaría un retorno al proceso manual, causando grandes demoras.
Seguridad	Se debe encriptar toda la comunicación y proteger los datos personales de los ciudadanos.	Para cumplir con las leyes de protección de datos y prevenir fraudes.
Mantenibilidad	La arquitectura debe facilitar la corrección de errores y la adición de nuevas funcionalidades.	Para permitir que el sistema evolucione a bajo costo y con bajo riesgo. Exportar a Hojas de cálculo

Rendimiento, Mantenibilidad, Seguridad Portabilidad)

## 5.3. Reglas y criterios de evaluación de calidad

- **Rendimiento:** El tiempo de carga de la página principal debe ser de 3 segundos. Las consultas a sistemas externos no deben superar los 5 segundos de espera para el usuario.
- **Usabilidad:** Se realizarán pruebas heurísticas y con usuarios para obtener una puntuación de satisfacción > 80%.
- **Seguridad:** Se realizarán análisis de vulnerabilidades periódicos para prevenir ataques comunes (inyección SQL, XSS, etc.).

## 6. PRINCIPIOS DE DISEÑO APLICADOS

### 6.1. Propósito

El propósito de esta sección es definir y explicar los principios esenciales del diseño de software que rigen la arquitectura del sistema. Estos principios actúan como un conjunto de reglas y orientaciones para el equipo de desarrollo, asegurando que las decisiones técnicas que se tomen a lo largo del proyecto sean coherentes, impulsen la calidad y faciliten la mantenibilidad y evolución futura del software.

## 6.2. Principios de diseño (por ejemplo: abstracción, acoplamiento, cohesión, encapsulamiento, modularidad)

PRINCIPIO	DESCRIPCIÓN	APLICACIÓN EN EL SISTEMA
Alta Cohesión	Cada módulo o clase tiene una única responsabilidad bien definida.	Los servicios están diseñados para realizar tareas específicas, por ejemplo, un servicio para validar mascotas y otro para validar productos
Separación de preocupaciones	Dividir el sistema en partes distintas con funcionalidad específica.	El patrón MVC separa la lógica de la interfaz y los datos. El enfoque SOA aísla las integraciones externas.
Bajo Acoplamiento	Los módulos son independientes entre sí, minimizando el impacto de los cambios.	El sistema principal puede operar con funcionalidades básicas incluso si un sistema externo (ej. SAG) no responde.
Encapsulamiento	Ocultar los detalles internos de un componente y exponer solo una interfaz pública.	Los datos internos de las clases están ocultos y solo se puede acceder a ellos a través de métodos públicos (getters/setters).

## 6.3. Diseño centrado en el usuario (UX/UI, prototipos, experiencia de usuario)

El diseño de la interfaz de usuario (UI) y la experiencia de usuario (UX) se centrará en la simplicidad y la claridad, considerando que los usuarios (ciudadanos) pueden tener diferentes niveles de habilidad digital. Se realizarán prototipos de bajo y alto nivel para validar los flujos de interacción con usuarios finales antes del desarrollo, asegurando que el proceso de declaración sea lo más intuitivo posible.

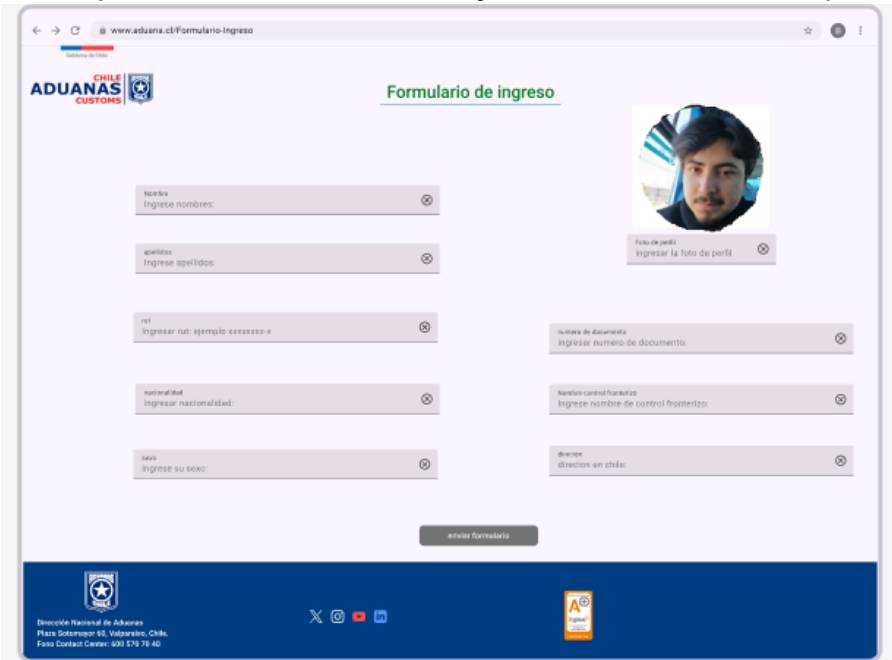
## 7. PROTOTIPO

Este prototipo sirve para mostrar, de forma visual y simple, cómo se verá y funcionará la plataforma para cada tipo de usuario que la utilizará. La idea es que todos puedan entender qué opciones tendrán disponibles y qué podrán hacer dentro del sistema. Además, el prototipo nos ayuda a identificar con anticipación posibles errores o problemas que podrían surgir, para poder corregirlos antes de que se use en la vida real.

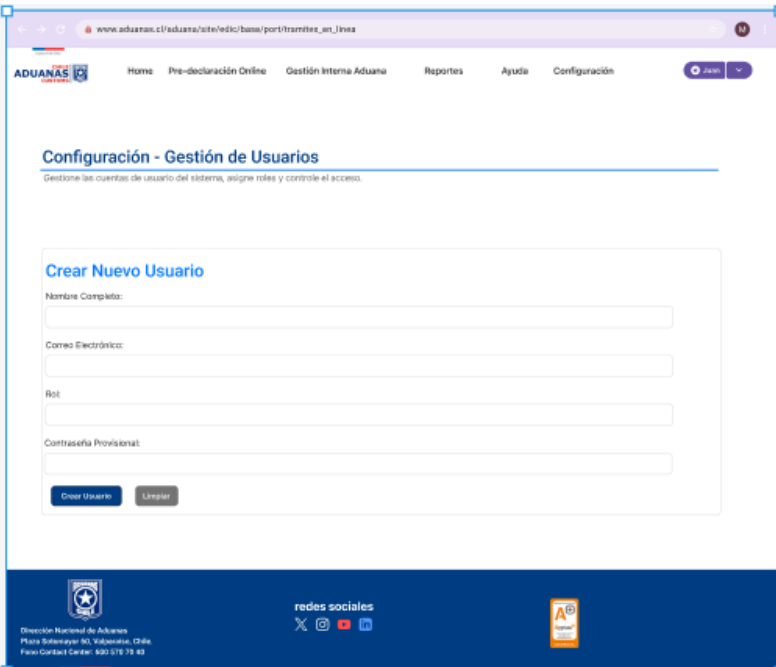


Mockups (imágenes con una breve descripción)

vista formulario de ingreso: verifica de que si paso a otro país de forma legal y saber quien eres en el sentido de justificación de entrar al país



vista gestión de usuario: controlar quién puede entrar a un sistema, qué puede hacer dentro y cómo se organiza la información de esas personas



vista apartado de declaración: puede declarar lo que lleva en su viaje tanto en tren como en auto, puede decir cuantas personas van y menores de edad, decir cuantas mascotas tiene dentro de su viaje y producto no equipaje como eje: parilla, generador, herramientas etc...



### Justificar herramientas de prototipado

se usó figma, ya que es más completo para realizar trabajo de prototipo y tiene una facilidad en usar y es bastante más acercado a lo que podemos usar más adelante como para crear el diseño de una página web

8.

### EVALUACIÓN DE CALIDAD HEURÍSTICA DE NIELSEN

#### Propósito

El objetivo de esta evaluación es entender cómo se encuentra actualmente el sistema, analizando en detalle cada uno de los criterios establecidos para comprobar si se están cumpliendo. También se toma en cuenta la gravedad de los problemas detectados y el impacto que estos tienen en la experiencia del usuario, siempre en relación con lo que se propuso en el prototipo.

Lista de verificación:

Nº	Principio de Usabilidad	¿Qué se evalúa?	¿Se cumple?	Comentarios / Evidencia	Nivel del problema
1	<b>Visibilidad del estado del sistema</b>	¿El sistema informa claramente qué está pasando? (cargas, acciones)	✓	El sistema es claro. Muestra lo que hace en cada momento, al subir archivos o llenar formularios.	Baja
2	<b>Lenguaje claro y cercano al usuario</b>	¿Usa palabras y formas que entienden los usuarios?	✓	Usa palabras comunes y dar ejemplos en vez de enseñar cómo se ocupa	Baja
3	<b>Control y libertad</b>	¿Permite cancelar o deshacer acciones fácilmente?	✓	Siempre se puede volver atrás, cargar de nuevo o corregir. Está diseñado para que sea sencillo.	Baja
4	<b>Coherencia y estándares</b>	¿Tiene un diseño coherente en colores, botones y pantallas?	✓	Se mantiene igual en todas las pantallas y dispositivos.	Baja
5	<b>Prevención de errores</b>	¿Evita errores antes de que pasen?	X	no hay prevención de errores	Media
6	<b>Reconocimiento en lugar de memoria</b>	¿Las funciones importantes siempre están visibles?	✓	Las opciones siempre están a la vista y no hay que complicarse tanto	Baja
7	<b>Flexibilidad y eficiencia</b>	¿Tiene atajos o personalización para usuarios avanzados?	✓	Hay teclas rápidas y opciones avanzadas disponibles.	Baja
8	<b>Diseño simple y claro</b>	¿Evita mostrar cosas innecesarias?	✓	Tiene un diseño bastante común y no es tan difícil de entender.	Baja
9	<b>Ayuda en errores</b>	¿Los mensajes de error son claros y útiles?	X	no salen mensaje de errores (tratar de mejorar)	Media
10	<b>Ayuda y soporte</b>	¿Hay ayuda disponible cuando	✓	hay un soporte donde hay pregunta frecuentes	Media

Análisis y métricas de resultados

## **9. CONTROL DE VERSIONES -> GITHUB REPOSITORIO, SE SUBEN LA DOCUMENTACIÓN DE EVALUACIONES ANTERIORES**

Propósito = dar respaldo de la información y documentación dentro de las redes sociales que están puestas debajo de la aplicación web

Control de versión utilizado (justificar el tipo de control de versión utilizado (fecha, semántica o secuencial) > se ve basado en la versión 1.0, 1.2, 1.6. cada versión va mejor la estructura y errores y cosas faltantes que no nos damos cuenta

Justificar herramientas de versionamiento > github repositorio

## **10. CONCLUSIONES**

El "Sistema de Automatización de Procesos Aduaneros" ha progresado significativamente. El prototipo ha sido clave para visualizar la funcionalidad y mejorar la experiencia del usuario, mientras que la evaluación heurística de Nielsen ha permitido identificar áreas de mejora, aunque el sistema ya es intuitivo y fácil de usar

## **11. BIBLIOGRAFÍA**