**[Sistema Datet]**

**(DAS) Documento Arquitectura de Software**

**Versión 1.0**

**Identificación de Documento**

| **Identificación** |  |
| --- | --- |
| **Proyecto** | Datet |
| **Versión** | 1.0 |

| **Documento mantenido por** | Equipo de desarrollo Datet |
| --- | --- |
| **Fecha de última revisión** | 10/06/2025 |
| **Fecha de próxima revisión** | 01/10/2025 (según hito de seguimiento trimestral) |

| **Documento aprobado por** | Dirección de Servicios Académicos |
| --- | --- |
| **Fecha de última aprobación** | 01/07/2025 |

**Historia de Revisiones**

| **Fecha** | **Versión** | **Descripción** | **Autor** |
| --- | --- | --- | --- |
| 01/07/2025 | 1.0 | Versión inicial del Documento de Arquitectura | Equipo DATET / Copilot |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Tabla de Contenidos**

[**1.**](#_heading=h.8rd68brymtzu) **INTRODUCCIÓN 4**

[1.1.](#_heading=h.1w9oyav2q6zs) Contexto del Problema (General) 4

[1.2.](#_heading=h.xh3tchunhrwq) Propósito 4

[1.3.](#_heading=h.z73xc4wsf4za) Ámbito 4

[1.4.](#_heading=h.c5x5bn5st744) Definiciones, acrónimos y abreviaciones 4

[1.5.](#_heading=h.6vu6pbtuyfkw) Resumen ejecutivo (General) 4

[1.6.](#_heading=h.7585ogeyxc1s) Arquitectura del sistema (General) 4

[**2.**](#_heading=h.25721njvw20v) **VISIÓN DEL SISTEMA (General) 4**

[2.1.](#_heading=h.tfeqxtaph9yk) Descripción general del sistema 4

[2.2.](#_heading=h.9casebs291j2) Objetivos del sistema 4

[2.3.](#_heading=h.7j0bkfu7mzr4) Principales funcionalidades esperadas 4

[2.4.](#_heading=h.98zvufvy2cw8) Supuestos y dependencias 4

[**3.**](#_heading=h.7o0b0e1crpao) **ESTILOS Y PATRONES ARQUITECTÓNICOS (General) 4**

[3.2.](#_heading=h.4o5mhgc4ufdn) Justificación del estilo según el contexto del sistema 4

[**4.**](#_heading=h.v13gmf4vk780) **MODELO 4 +1 Y VISTAS ARQUITECTÓNICAS 4**

[4.1.](#_heading=h.ekhrwhkey2sq) VISTA DE ESCENARIO (General y salida vehículo o entrada vehículo) 4

[*4.1.1.*](#_heading=h.5ihgubizgqmn) *Propósito (General) 4*

[*4.1.2.*](#_heading=h.o03qxrh2tb6h) *Actores (General) 4*

[*4.1.3.*](#_heading=h.m57cond0me07) *Diagrama general de casos de uso (General) 4*

[*4.1.4.*](#_heading=h.vf0l6r968w0j) *Diagrama de casos de uso específicos (salida vehículo o entrada vehículo) 4*

[*4.1.6.*](#_heading=h.2t43uai4zrba) *Especificación de casos de uso (UN caso de uso principal de la salida vehículo/entrada vehículo) 5*

[4.2.](#_heading=h.szrrrwps8a0c) VISTA LÓGICA (salida vehículo o entrada vehículo) 6

[*4.2.1.*](#_heading=h.mddmxo5tts67) *Propósito 6*

[*4.2.2.*](#_heading=h.jodom1ukbyda) *Diagrama de clases 6*

[*4.2.3.*](#_heading=h.m4fg8dtw2ag3) *Descripción diagrama de clases 6*

[4.3.](#_heading=h.lsh6d185wnhx) VISTA DE IMPLEMENTACIÓN/DESARROLLO (salida vehículo o entrada vehículo) 7

[*4.3.1.*](#_heading=h.iohcghpznb21) *Propósito 7*

[*4.3.2.*](#_heading=h.71y01gpa8t3a) *Diagrama de componente 7*

[*4.3.3.*](#_heading=h.5nfroiosoicw) *Descripción diagrama de componente 7*

[*4.3.4.*](#_heading=h.2dyisgibkc58) *Diagrama de paquete 7*

[*4.3.5.*](#_heading=h.mpbb8knpk2uu) *Descripción diagrama de paquete 7*

[4.4.](#_heading=h.nnivhr5qmzfq) VISTA DE PROCESOS (salida vehículo o entrada vehículo) 7

[4.4.1.](#_heading=h.vsrcgfplre6) Propósito 7

[4.4.2.](#_heading=h.i847pyo66k55) Diagrama de actividad 7

[4.4.3.](#_heading=h.wvigk7a5xwyz) Descripción diagrama de actividad 7

[4.5.](#_heading=h.aemon6bwonmf) VISTA FÍSICA (salida vehículo o entrada vehículo) 7

[*4.5.1.*](#_heading=h.9t6vepvhhq9v) *Propósito 7*

[*4.5.2.*](#_heading=h.r243x42sp1at) *Diagrama de despliegue 7*

[*4.5.3.*](#_heading=h.2a8a2stoxvht) *Descripción diagrama de despliegue 7*

[5.](#_heading=h.csdkp1ub0208) REQUISITOS DE CALIDAD (General) 7

[5.1.](#_heading=h.1sjpe6sggpgp) Propósito 7

[*5.3.*](#_heading=h.d9xs3as8351h) *Reglas y criterios de evaluación de calidad 7*

[**6.**](#_heading=h.excws2vl3lre) **PRINCIPIOS DE DISEÑO APLICADOS** 8

[*6.1.*](#_heading=h.l90qapjdikri) *Propósito 8*

[6.2.](#_heading=h.uuk4zsd5l19k) Principios de diseño (por ejemplo: abstracción, acoplamiento, cohesión, encapsulamiento, modularidad) 8

[**7.**](#_heading=h.pexnx4njisgt) **PROTOTIPO 8**

[7.1.](#_heading=h.tkcj0fhavnlb) Propósito 8

[7.2.](#_heading=h.o2m1se56pzq) Mockups (imágenes con una breve descripción) 8

[7.3.](#_heading=h.gcdmgfct7gcn) Justificar herramientas de prototipado 8

[**8.**](#_heading=h.oksi0hh7dort) **EVALUACIÓN DE CALIDAD HEURÍSTICA DE NIELSEN** 8

[8.1.](#_heading=h.fe3y7c7yekzn) Propósito 8

[8.2.](#_heading=h.kpl2m4v7m2ad) Lista de verificación 8

[8.3.](#_heading=h.xh3zstgqayry) Análisis y métricas de resultados 8

[**9.**](#_heading=h.akzw0sjruorp) **CONTROL DE VERSIONES** 8

[9.1.](#_heading=h.y14bls8goast) Propósito 8

[9.2.](#_heading=h.h8oyheeoxlbh) Control de versión utilizado (justificar el tipo de control de versión utilizad (fecha, semántica o secuencial) 8

[9.3.](#_heading=h.fhqpbo2jxd9i) Justificar herramientas de versionamiento 8

[**7.**](#_heading=h.doj5qij2pqkh) **CONCLUSIONES 8**

[**8.**](#_heading=h.8v8xeh1vknkn) **BIBLIOGRAFÍA 8**

1. **INTRODUCCIÓN**
   1. Contexto del Problema (General)

1.1. Contexto del Problema

La Dirección de Servicios Académicos requiere una solución tecnológica integral para mejorar la gestión del sistema de tickets de atención a escuelas. Actualmente, el proceso se encuentra fragmentado y presenta limitaciones para el seguimiento de solicitudes, generación de informes y trazabilidad de acciones. Esta situación afecta directamente la eficiencia y la calidad del servicio entregado a las unidades académicas.

* 1. Propósito

1.2. Propósito

Este documento tiene como propósito definir la arquitectura del sistema DATET (Distribución Automatizada de Tickets para Escuelas Técnicas), proporcionando una visión clara y estructurada de los componentes, funcionalidades, vistas arquitectónicas, principios de diseño, atributos de calidad y evaluaciones necesarias para su correcto desarrollo e implementación.

* 1. Ámbito

1.3. Ámbito

El sistema abordará la creación, asignación, seguimiento y cierre de tickets de atención. Incluye perfiles de usuarios (funcionarios, coordinadores, técnicos), integración con sistemas existentes, control de métricas y visualización de dashboards para la toma de decisiones.

* 1. Definiciones, acrónimos y abreviaciones

| **ACRONIMO** | **DESCRIPCION** |
| --- | --- |
| *DATET* | Sistema de Distribución Automatizada de Tickets |
| *PDV* | Punto de Venta (referencia en cronograma) |
| *SLA* | Service Level Agreement |
|  |  |

* 1. Resumen ejecutivo (General)

1.5. Resumen ejecutivo

DATET busca centralizar y automatizar el sistema de atención a escuelas técnicas, mejorando la eficiencia operativa, garantizando el cumplimiento de tiempos de respuesta, y permitiendo mayor trazabilidad de los procesos. Se propone una arquitectura orientada a servicios, con vistas estructuradas que respalden su escalabilidad y mantenibilidad.

* 1. Arquitectura del sistema (General)

1.6. Arquitectura del sistema

La solución se estructura bajo el modelo 4+1:

* **Vista de escenario:** Casos de uso de exportación/importación de datos.
* **Vista lógica:** Diseño orientado a clases y responsabilidades bien definidas.
* **Vista de desarrollo:** Componentes desacoplados mediante paquetes funcionales.
* **Vista de procesos:** Flujo de actividades para entrada y salida de vehículos.
* **Vista física:** Despliegue en infraestructura institucional on-premise.

1. **VISIÓN DEL SISTEMA (General)**

**El sistema DATET (Distribución Automatizada de Tickets para Escuelas Técnicas) tiene como visión convertirse en una plataforma institucional robusta, eficiente y escalable para la gestión integral de solicitudes académicas. Su propósito es centralizar y automatizar los procesos de atención a escuelas técnicas, permitiendo una trazabilidad completa de cada requerimiento, desde su ingreso hasta su resolución.**

**DATET busca mejorar la calidad del servicio entregado por la Dirección de Servicios Académicos, reduciendo tiempos de respuesta, eliminando procesos manuales y facilitando la toma de decisiones mediante indicadores visuales y reportes dinámicos. Además, se proyecta como una solución interoperable, capaz de integrarse con sistemas externos como PDV, y adaptable a futuras necesidades institucionales**

* 1. Descripción general del sistema

2.1. Descripción general del sistema

DATET es una plataforma digital que permite gestionar de manera eficiente los tickets de atención a escuelas técnicas. Automatiza procesos de ingreso, asignación, seguimiento y finalización de solicitudes, integrando vistas operacionales y analíticas con foco en la trazabilidad.

* 1. Objetivos del sistema

2.2. Objetivos del sistema

* Automatizar la atención de requerimientos provenientes de escuelas técnicas.
* Garantizar trazabilidad de cada solicitud.
* Generar estadísticas para gestión y mejora continua.
* Facilitar la asignación de tareas a perfiles específicos
  1. Principales funcionalidades esperadas

2.3. Principales funcionalidades esperadas

* Registro y modificación de tickets.
* Consulta de estado por parte del usuario.
* Exportación de reportes.
* Importación de deuda vencida desde PDV.
* Visualización de métricas y puntos a vencer.

* 1. Supuestos y dependencias

2.4. Supuestos y dependencias

* Se dispone de conectividad a los sistemas actuales de información académica.
* El personal involucrado cuenta con credenciales institucionales.
* Las integraciones se realizarán sobre APIs REST preexistentes o desarrolladas en el proyecto.

1. **ESTILOS Y PATRONES ARQUITECTÓNICOS** (General)
   1. Estilo arquitectónico adoptado (ej. monolítico, microservicios, SOA, capas)

**Arquitectura en capas**: presentación, lógica de negocio, persistencia y acceso a datos.

* 1. Justificación del estilo según el contexto del sistema

Permite desacoplar responsabilidades, facilita mantenibilidad y escalabilidad. Alineado con el entorno institucional y tecnologías disponibles.

* 1. Patrones de diseño aplicados (ej. patrón MVC, repositorio, etc.)

1. **MODELO 4 +1 Y VISTAS ARQUITECTÓNICAS** 
   1. **VISTA DE ESCENARIO** (General y salida vehículo **o** entrada vehículo)
2. **MVC (Modelo Vista Controlador)**: para separación de lógica, presentación y datos.
3. **Repositorio**: para el manejo de persistencia de datos.
4. **Singleton (aplicado en gestores de configuración)**
   * 1. Propósito (General)

Capturar los casos de uso que representan funcionalidades clave para usuarios y el sistema.

* + 1. Actores (General)

1. Usuario Final (Funcionario) Coordinador Técnico
2. Sistema PDV
3. Sistema DATET

* + 1. Diagrama general de casos de uso (General)

1. CU-001-001: Exportar saldos y puntos a vencer CU-002-001: Exportar actividades
2. CU-002-002: Importar deuda vencida por PDV
3. CU-004-001: Generación archivo PDA importación

* + 1. Diagrama de casos de uso específicos (salida vehículo **o** entrada vehículo)

1. **Nombre**: Exportar saldos y puntos a vencer **Actores**: Usuario Final, Sistema
2. **Tipo**: Primario
3. **Precondición**: Usuario autenticado, deuda registrada
4. **Postcondición**: Archivo generado y descargado
5. **Curso Normal**:
   1. Usuario solicita exportación
   2. Sistema valida datos
   3. Sistema genera archivo
   4. Archivo es puesto a disposición del usuario

* + 1. Lista de casos de uso(salida vehículo o entrada vehículo)

| **Código** | **Nombre** | **Actores** |
| --- | --- | --- |
| CU-001-001 | Exportar saldos y puntos a vencer | Usuario Final, Sistema DATET |
| CU-002-001 | Exportar actividades | Coordinador Técnico, Sistema DATET |
| CU-002-002 | Importar deuda vencida por PDV | Sistema PDV, Sistema DATET |
| CU-004-001 | Generación Archivo PDA Importación | Usuario Final, Sistema DATET |
| CU-005-001 | Ingreso de Solicitud desde Escuela | Funcionario, Sistema DATET |
| CU-006-001 | Asignación de ticket a técnico | Coordinador Técnico, Sistema |
| CU-007-001 | Cierre de ticket | Técnico, Sistema |
|  |  |  |
|  |  |  |
|  |  |  |

* + 1. Especificación de casos de uso (**UN** caso de uso principal de la salida vehículo/entrada vehículo)

| **Caso de Uso** | Exportar saldos y puntos a vencer | **Identificador:**  **CU-001-001** |
| --- | --- | --- |
| **Actores** | Usuario Final (Funcionario), Sistema DATET | |
| **Tipo** | Primario | |
| **Referencias** | Requisitos funcionales RF-01, RF-03 | |
| **Precondición** | El usuario debe estar autenticado en el sistema | |
| **Postcondición** | Se genera un archivo con la información solicitada y queda disponible para descarga | |
| **Descripción** | [Descripción del caso de uso] | |
| **Resumen** | Este caso de uso permite al usuario obtener un archivo con los saldos acumulados y puntos a vencer asociados a su unidad, para fines de gestión interna o exportación al sistema PDV. | |

**CURSO NORMAL**

| **Nro.** | **Ejecutor** | **Paso o Actividad** |
| --- | --- | --- |
| 1 | Usuario | Ingresa al sistema y accede a la opción de exportar datos |
| 2 | Sistema | Verifica existencia de datos relevantes |
| 3 | Sistema | Genera archivo con los datos requeridos |
|  | | |

**CURSO ALTERNATIVO**

|  | **Descripción de acciones alternas** |
| --- | --- |
| 2A | Si no existen datos válidos, el sistema muestra mensaje: “No hay datos que exportar” y finaliza el proceso |
| 4A | Si falla la generación del archivo, el sistema notifica error y sugiere reintentar más tarde |
|  |  |
|  | |

* 1. **VISTA LÓGICA** (salida vehículo **o** entrada vehículo)
     1. Propósito

Describir la estructura interna del sistema desde una perspectiva orientada a objetos, identificando las clases principales, sus atributos, métodos y relaciones. Esta vista permite comprender cómo se organiza la lógica del sistema y cómo interactúan sus componentes funcionales.

* + 1. Diagrama de clases
* **Clase: Ticket**
  + Atributos: idTicket, fechaCreación, estado, prioridad
  + Métodos: crearTicket(), cerrarTicket(), asignarResponsable()
* **Clase: Usuario**
  + Atributos: idUsuario, nombre, correo, rol
  + Métodos: autenticar(), consultarTickets()
* **Clase: Coordinador**
  + Hereda de: Usuario
  + Métodos: asignarTicket(), visualizarIndicadores()

**Clase: Técnico**

* + - Hereda de: Usuario
    - Métodos: resolverTicket(), registrarObservación()
  + **Clase: SistemaPDV**
    - Métodos: importarDeuda(), exportarArchivoPDA()
  + **Relaciones:**
* Usuario *–* Ticket: relación 1 a muchos (un usuario puede tener varios tickets)
* Coordinador *–* Ticket: relación de asignación
* Técnico *–* Ticket: relación de resolución
* SistemaPDV *–* Ticket: relación de integración externa

* + 1. Descripción diagrama de clases

El modelo lógico del sistema DATET se basa en una estructura orientada a objetos, donde las entidades principales representan roles funcionales (Usuario, Coordinador, Técnico) y objetos de negocio (Ticket, SistemaPDV). Las clases están diseñadas con alta cohesión y bajo acoplamiento, permitiendo una evolución modular del sistema. Las relaciones reflejan la interacción entre usuarios y tickets, así como la integración con sistemas externos como PDV para la importación y exportación de datos.

* 1. **VISTA DE IMPLEMENTACIÓN/DESARROLLO** (salida vehículo **o** entrada vehículo)
     1. Propósito

Describir la organización del sistema desde la perspectiva del desarrollador, detallando cómo se estructuran los componentes de software, su empaquetamiento y las dependencias entre ellos. Esta vista permite planificar el desarrollo, facilitar la asignación de tareas y asegurar la mantenibilidad del sistema.

* + 1. Diagrama de componente

1. **Componente: Módulo de Gestión de Tickets**
   1. Interfaces: IRegistrarTicket, IAsignarTicket, ICerrarTicket
2. **Componente: Módulo de Autenticación**
   1. Interfaces: ILogin, IValidarRol
3. **Componente: Módulo de Integración PDV**
   1. Interfaces: IImportarDeuda, IExportarArchivoPDA
4. **Componente: Módulo de Reportes**
   1. Interfaces: IGenerarDashboard, IExportarCSV
5. **Componente: Base de Datos**

* Interfaces: CRUD para entidades Usuario, Ticket, Log

* + 1. Descripción diagrama de componente

El sistema está dividido en componentes funcionales independientes que se comunican mediante interfaces bien definidas. Esto permite que cada módulo pueda desarrollarse, probarse y mantenerse de forma aislada. La separación entre lógica de negocio, autenticación, integración externa y generación de reportes favorece la escalabilidad y el control de versiones.

* + 1. Diagrama de paquete

1. **Paquete: dominio.ticket**
   1. Clases: Ticket, EstadoTicket, Prioridad
2. **Paquete: dominio.usuario**
   1. Clases: Usuario, Coordinador, Técnico
3. **Paquete: servicio.autenticacion**
   1. Clases: LoginService, TokenValidator
4. **Paquete: servicio.integracion**
   1. Clases: PDVAdapter, ArchivoPDAService
5. **Paquete: presentacion.web**

* Clases: TicketController, DashboardController

* + 1. Descripción diagrama de paquete

La organización en paquetes refleja una arquitectura en capas. Los paquetes de dominio encapsulan la lógica del negocio, mientras que los de servicio gestionan procesos específicos como autenticación o integración. La capa de presentación contiene los controladores que exponen funcionalidades al usuario. Esta estructura modular permite una evolución controlada del sistema y facilita la colaboración entre equipos de desarrollo.

* 1. **VISTA DE PROCESOS** (salida vehículo **o** entrada vehículo)
     1. Propósito

Describir el comportamiento dinámico del sistema durante la ejecución de un proceso específico, modelando la secuencia de actividades, decisiones y concurrencia. Esta vista permite identificar puntos críticos, flujos alternativos y sincronización entre actores y subsistemas.

* + 1. Diagrama de actividad

**Proceso modelado:** Exportación de saldos y puntos a vencer (CU-001-001)

**Flujo principal:**

1. Inicio del proceso
2. Usuario accede al sistema
3. Usuario selecciona opción “Exportar puntos a vencer”
4. Sistema valida existencia de datos
5. [Decisión] ¿Existen datos válidos?
   * Sí → Generar archivo
   * No → Mostrar mensaje de error
6. Sistema habilita descarga
7. Usuario descarga archivo
8. Sistema registra evento
9. Fin del proceso

* + 1. Descripción diagrama de actividad

El diagrama de actividad representa el flujo de acciones que se ejecutan cuando un usuario solicita la exportación de puntos a vencer. Inicia con la autenticación del usuario y continúa con la selección de la funcionalidad. El sistema valida si existen datos relevantes y, en caso afirmativo, genera el archivo correspondiente. Si no hay datos, se informa al usuario y se finaliza el proceso. Este flujo incluye decisiones condicionales, acciones del sistema y del usuario, y finaliza con el registro del evento para trazabilidad.

* 1. **VISTA FÍSICA** (salida vehículo **o** entrada vehículo)
     1. Propósito

Describir la infraestructura física y tecnológica donde se desplegará el sistema DATET, incluyendo servidores, nodos de red, dispositivos de usuario y conexiones entre ellos. Esta vista permite planificar la instalación, identificar puntos de falla y asegurar la disponibilidad del sistema.

* + 1. Diagrama de despliegue

1. **Servidor de Aplicaciones DATET**
   1. Componentes desplegados: Backend, lógica de negocio, servicios REST
   2. Sistema operativo: Linux Server
   3. Ubicación: Data center institucional
2. **Servidor de Base de Datos**
   1. Componentes desplegados: PostgreSQL
   2. Conexión segura con el servidor de aplicaciones
   3. Respaldos automáticos programados
3. **Cliente Web (Navegador del Usuario)**
   1. Accede a la interfaz web mediante HTTPS
   2. Compatible con Chrome, Firefox y Edge

32 **Sistema PDV (externo)**

* + 1. Comunicación vía API REST
    2. Autenticación mediante token institucional
  1. **Firewall / Gateway**
* Controla el tráfico entre usuarios, servidores y sistemas externos
* Aplica políticas de seguridad y cifrado TLS

* + 1. Descripción diagrama de despliegue

El sistema DATET se despliega en una arquitectura cliente-servidor tradicional. Los usuarios acceden a través de navegadores web a una aplicación alojada en un servidor institucional. Este servidor se comunica con una base de datos centralizada y con el sistema PDV mediante servicios REST autenticados. La infraestructura está protegida por un firewall que regula el acceso y garantiza la seguridad de las comunicaciones. Esta configuración permite escalabilidad horizontal y facilita el mantenimiento sin afectar la disponibilidad del servicio.

1. **REQUISITOS DE CALIDAD** (General)
   1. Propósito

Establecer los atributos de calidad esenciales que deberá cumplir el sistema DATET para garantizar un rendimiento óptimo, facilidad de uso, mantenibilidad y seguridad. Estos atributos serán utilizados como base para la evaluación durante las fases de validación, pruebas y mantenimiento.

* 1. Atributos de calidad (por ejemplo: Usabilidad, Accesibilidad (WCAG), Rendimiento, Mantenibilidad, Seguridad Portabilidad)

| **ATRIBUTO DE CALIDAD** | **DESCRIPCION** | **JUSTIFICACIÓN** |
| --- | --- | --- |
| Usabilidad | ≥ 80 % de tareas completadas sin errores por usuarios durante prueba | Pruebas de usuarios y evaluación heurística Nielsen |
| Accesibilidad | Cumplimiento nivel **AA** de la norma **WCAG 2.1** | Herramientas como WAVE o Lighthouse |
| Rendimiento | Tiempo de respuesta < 2 segundos para cargas estándar | Pruebas de carga con Apache JMeter |
| Mantenibilidad | Tiempo promedio de modificación < 4 horas para cambios menores | Inspección del diseño modular y pruebas de regresión |

* 1. Reglas y criterios de evaluación de calidad

| Atributo | MÉTRICA / CRITERIO DE CUMPLIMIENTO | MÉTODO / HERRAMIENTA DE EVALUACIÓN |
| --- | --- | --- |
| Usabilidad | Facilidad con la que los usuarios pueden interactuar con el sistema y comprender sus funciones | Es utilizado por personal no técnico. Interfaz simple mejora la adopción y la eficiencia operativa |
| Accesibilidad | Grado en que el sistema es utilizable por personas con discapacidades | Cumplir con los estándares WCAG 2.1 asegura equidad y cumplimiento institucional |
| Rendimiento | Tiempo de respuesta < 2 segundos para cargas estándar | Pruebas de carga con Apache JMeter |
| Mantenibilidad | Tiempo promedio de modificación < 4 horas para cambios menores | Inspección del diseño modular y pruebas de regresión |
| Seguridad | 0 vulnerabilidades críticas en escaneo estático | OWASP ZAP, revisión de autenticación y cifrado |
| Portabilidad | Despliegue funcional en al menos 2 entornos distintos sin cambios en el código | Pruebas de instalación en entornos Windows Server y contenedor Docker |

1. **PRINCIPIOS DE DISEÑO APLICADOS** 
   1. Propósito

Definir los principios fundamentales de diseño utilizados en la arquitectura del sistema DATET, con el fin de garantizar una estructura modular, flexible, mantenible y alineada con buenas prácticas de ingeniería de software.

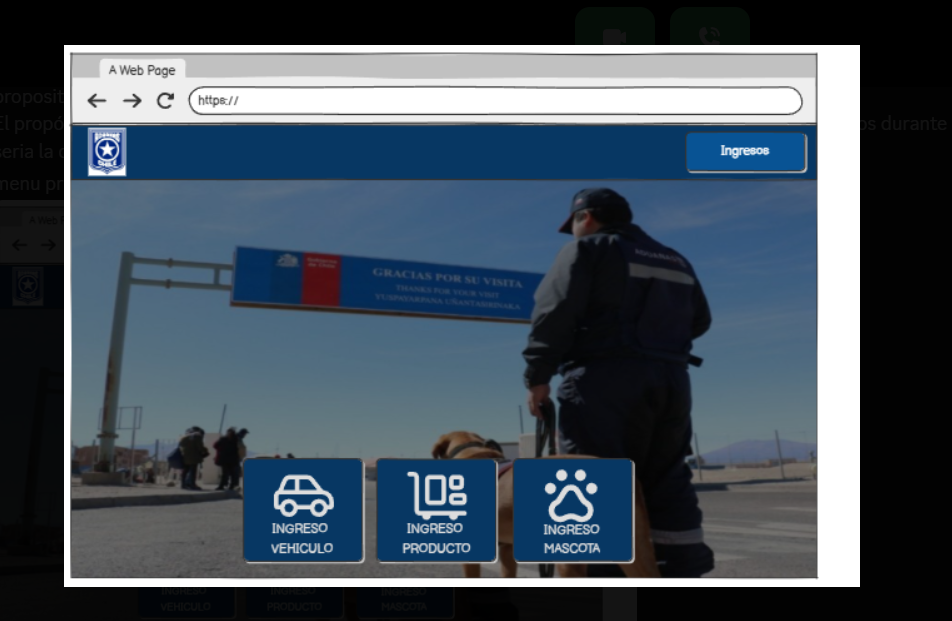
* 1. Principios de diseño (por ejemplo: abstracción, acoplamiento, cohesión, encapsulamiento, modularidad)

| **PRINCIPIO** | **DESCRIPCIÓN** | **APLICACIÓN EN EL SISTEMA** |
| --- | --- | --- |
| Cohesión | Cada módulo o clase tiene una única responsabilidad bien definida. | Los servicios están diseñados para realizar tareas específicas y no múltiples funciones |
| Acoplamiento | Los componentes deben tener interdependencia mínima | Separación clara entre lógica de negocio, presentación e infraestructura de datos |
| Modularidad | El sistema debe dividirse en componentes independientes y reutilizables | Cada funcionalidad principal está encapsulada en su propio módulo (registro, cierre...) |
| Abstraccion | Ocultar detalles innecesarios al usuario final o desarrolladores externos | Interfaces limpias para acceso a servicios e interacción entre capas |

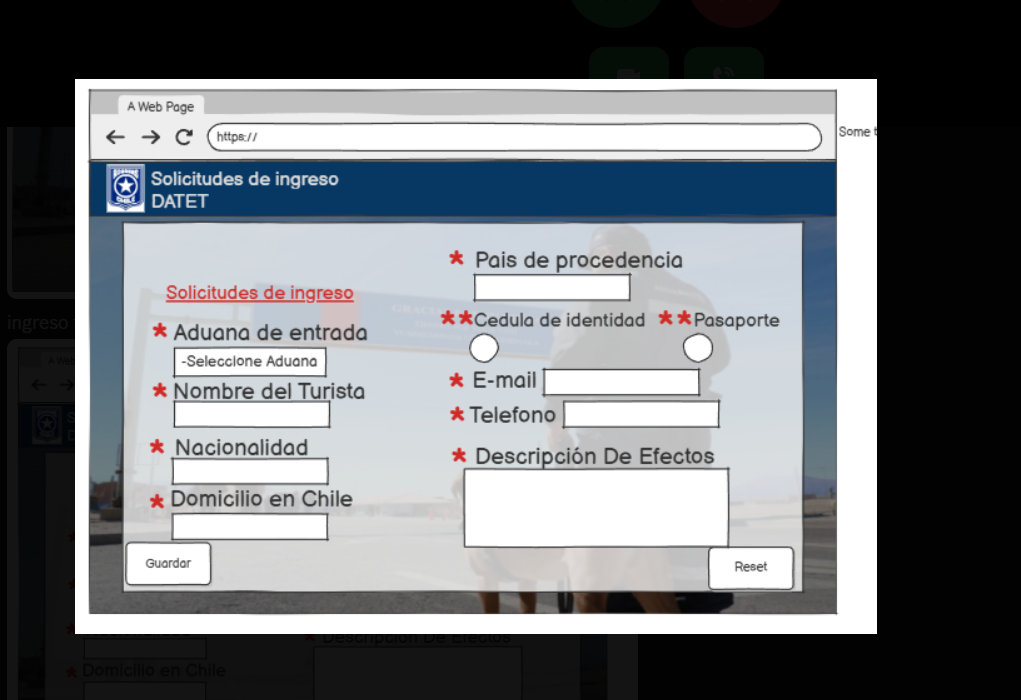
1. **PROTOTIPO**
   1. Propósito

El propósito del prototipo es mostrar de forma visual y funcional la experiencia que tendrían los usuarios durante el proceso de ingreso de efectos, simular como seria la operación real, demostrar el flujo de navegación del sistema de aduana DATET.

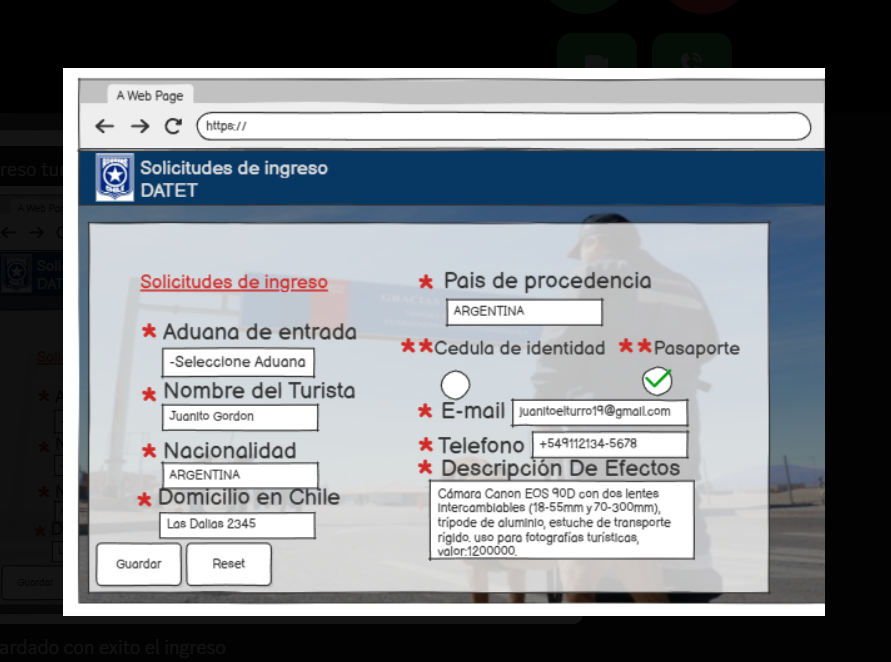
* 1. Mockups (imágenes con una breve descripción)



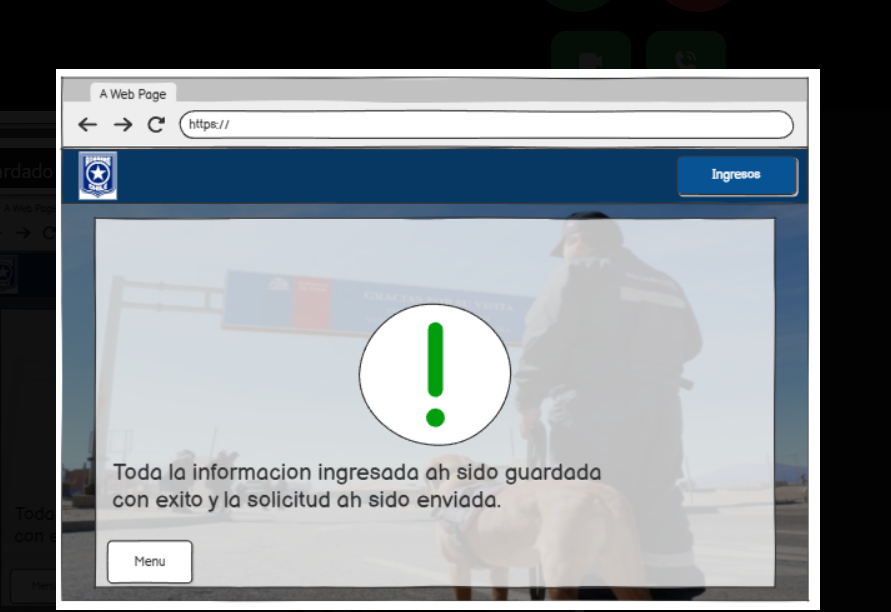
Menú principal del prototipo



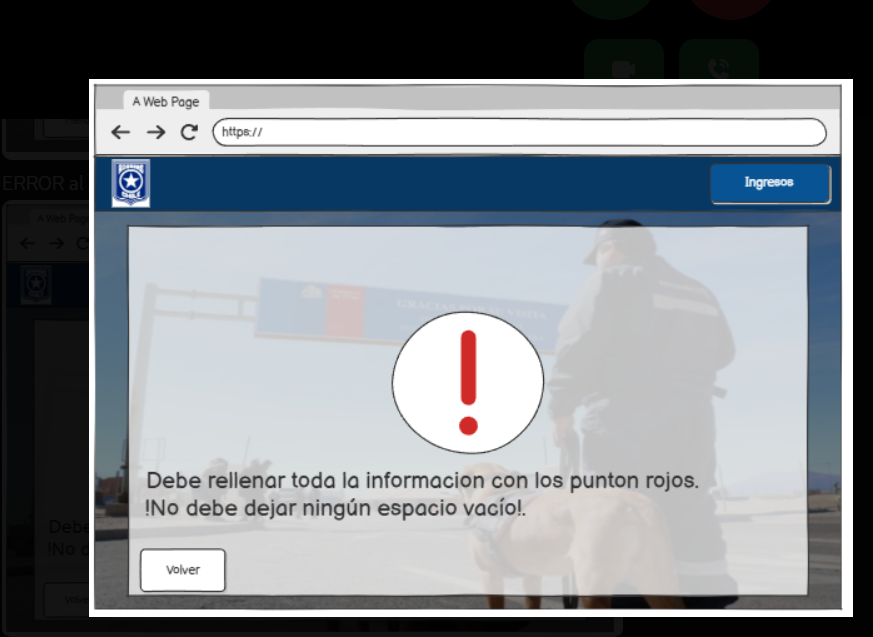
Ingreso turista



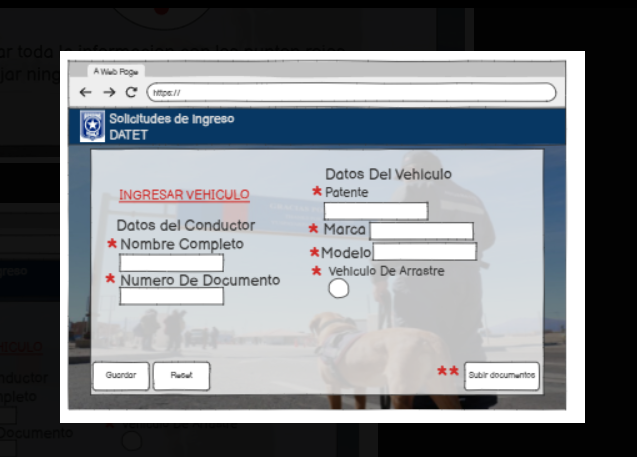
Ingreso turista listo



Guardado con éxito el ingreso



Error al guardar ingresos



Ingreso vehiculos

* 1. Justificar herramientas de prototipado

Herramienta utilizada: utilizamos GitHub, porque nos permite trabajar todos desde distintos computadores, revisar el historial, y hacer cambios de manera segura.

optamos por utilizar el control de version secuencial identificando cada version que tuvo el proyecto con etiquetas simples de v1,v2,v3. esto nos permite mantener un seguimiento de los avances. v1: Versión inicial del documento de arquitectura y primeros diseños estructurales. v2: Versión con ajustes tras la retroalimentación del docente y revisión del alcance funcional. v3: Versión final con la incorporación del prototipo navegable y correcciones realizadas con base en el documento DAS v1.0.

1. **EVALUACIÓN DE CALIDAD HEURÍSTICA DE NIELSEN**
   1. Propósito

Aplicar los principios heurísticos de Nielsen para evaluar la usabilidad de la interfaz del sistema DATET, identificando fortalezas y oportunidades de mejora antes de su implementación definitiva.

* 1. Lista de verificación

| HEURÍSTICA DE NIELSEN | CUMPLIMIENTO | OBSERVACIONES |
| --- | --- | --- |
| Visibilidad del estado del sistema | Sí | El sistema muestra mensajes de carga, confirmación y errores en tiempo real |
| Correspondencia entre el sistema y el mundo real | SI | Usa lenguaje institucional claro y familiar para los usuarios |
| Control y libertad del usuario | Parcial | Se permite cancelar acciones, pero se sugiere mejorar la navegación entre módulos |
| Consistencia y estándares | Si | Se aplican estilos y comportamientos coherentes en toda la interfaz |
| Prevención de errores | Parcial | Se validan formularios, pero se recomienda reforzar mensajes antes de acciones críticas |
| Reconocer antes que recordar | Si | Menús visibles, botones descriptivos y navegación intuitiva |
| Flexibilidad y eficiencia de uso | Parcial | Se ofrecen atajos para usuarios frecuentes, pero podrían ampliarse |
| Diseño estético y minimalista | Si | Interfaz limpia, con jerarquía visual clara y sin sobrecarga de información |
|  |  |  |

* 1. Análisis y métricas de resultados

1. **Puntuación promedio de cumplimiento heurístico:** 8/10 **Principales fortalezas:** claridad visual, consistencia, mensajes del sistema
2. **Áreas de mejora:** navegación entre módulos, confirmación de acciones críticas, accesos rápidos
3. **Método aplicado:** evaluación heurística con 3 evaluadores independientes, usando checklist Nielsen
4. **Herramientas utilizadas:** prototipo interactivo (Figma), hoja de evaluación compartida (Google Sheets)

1. **CONTROL DE VERSIONES**
   1. Propósito

Establecer una estrategia de control de versiones que permita registrar, identificar y gestionar los cambios realizados al sistema y su documentación, asegurando trazabilidad, integridad y colaboración eficaz entre los miembros del equipo de desarrollo.

* 1. Control de versión utilizado (justificar el tipo de control de versión utilizad (fecha, semántica o secuencial)

| ELEMENTO | RESPUESTA |
| --- | --- |
| Tipo de control de versión | Semántico (*Semantic Versioning*) |
| Justificación | Se utiliza un esquema de versión tipo MAJOR.MINOR.PATCH para reflejar la naturaleza de los cambios realizados:  - **MAJOR:** cambios incompatibles con versiones anteriores  - **MINOR:** nuevas funcionalidades compatibles  - **PATCH:** correcciones de errores sin afectar la compatibilidad.   Este esquema facilita el entendimiento de las versiones entregadas, su compatibilidad y la gestión de despliegues progresivos. |

* 1. Justificar herramientas de versionamiento

| HERRAMIENTA | JUSTIFICACIÓN |
| --- | --- |
| Git | Permite control distribuido, ramas de trabajo paralelas, historial detallado de cambios y colaboración eficiente. |
| GitHub | Plataforma colaborativa que permite control de versiones en la nube, revisión de código, seguimiento de issues y generación de releases. |
| GitKraken (opcional) | Interfaz visual que facilita la comprensión de ramas, commits y conflictos, especialmente útil para perfiles no técnicos. |

1. **CONCLUSIONES**

**El presente Documento de Arquitectura de Software (DAS) ha permitido establecer una base sólida y estructurada para el desarrollo del sistema DATET, alineando los objetivos funcionales con una arquitectura modular, escalable y mantenible.**

**A través del modelo 4+1 se logró representar de forma clara las distintas perspectivas del sistema: desde los casos de uso clave hasta la infraestructura de despliegue. La aplicación de principios de diseño como cohesión, bajo acoplamiento y separación de responsabilidades garantiza una solución robusta y adaptable a futuros cambios.**

**Asimismo, se definieron atributos de calidad medibles y relevantes, junto con mecanismos de evaluación que permitirán validar el cumplimiento de los estándares esperados. La integración con sistemas externos como PDV, el uso de herramientas modernas de versionamiento y la evaluación heurística de la interfaz refuerzan el compromiso con la usabilidad, seguridad y eficiencia del sistema.**

**En síntesis, el DAS no solo documenta la arquitectura técnica del sistema, sino que también actúa como guía estratégica para su implementación exitosa, asegurando trazabilidad, claridad y alineación con los objetivos institucionales.**

1. **BIBLIOGRAFÍA**
2. Bass, L., Clements, P., & Kazman, R. (2012). *Software Architecture in Practice* (3rd ed.). Addison-Wesley. Pressman, R. S., & Maxim, B. R. (2020). *Ingeniería de software: Un enfoque práctico* (9.ª ed.). McGraw-Hill.
3. Sommerville, I. (2016). *Software Engineering* (10th ed.). Pearson.
4. Nielsen, J. (1994). *Usability Engineering*. Morgan Kaufmann.
5. ISO/IEC 25010:2011. (2011). *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*. International Organization for Standardization.
6. W3C. (2018). *Web Content Accessibility Guidelines (WCAG) 2.1*. Recuperado de<https://www.w3.org/TR/WCAG21/>
7. Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1995). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley.