****

**UNIVERSIDAD PRIVADA DE TACNA**

**FACULTAD DE INGENIERÍA**

**Escuela Profesional de Ingeniería de Sistemas**

**Proyecto “DataFiller”**

Curso: *Pruebas y Calidad de Software*

Docente: *Mag. Patrick Cuadros Quiroga*

Integrantes:

[***SEBASTIAN NICOLAS FUENTES AVALOS***](mailto:sf2022073902@virtual.upt.pe) ***(2022073902)***

[***MAYRA FERNANDA CHIRE RAMOS***](mailto:mc2021072620@virtual.upt.pe) ***(2021072620)***

[***GABRIELA LUZKALID GUTIERREZ MAMANI***](mailto:gg2022074263@virtual.upt.pe) ***(2022074263)***

**Tacna – Perú**

***2025***

| CONTROL DE VERSIONES | | | | | |
| --- | --- | --- | --- | --- | --- |
| Versión | Hecha por | Revisada por | Aprobada por | Fecha | Motivo |
| 1.0 | MPV | ELV | ARV | 10/10/2020 | Versión Original |

Sistema *DataFiller*

Documento de Arquitectura de Software

Versión *{1.0}*

| CONTROL DE VERSIONES | | | | | |
| --- | --- | --- | --- | --- | --- |
| Versión | Hecha por | Revisada por | Aprobada por | Fecha | Motivo |
| 1.0 | MPV | ELV | ARV | 10/10/2020 | Versión Original |

INDICE GENERAL

**Contenido**

[***1.***](#_heading=h.bk1x3bb8mvjx) ***INTRODUCCIÓN 5***

[**1.1.**](#_heading=h.d11d7cag5ng4) **Propósito (Diagrama 4+1) 5**

[**1.2.**](#_heading=h.kktm13jgvun1) **Alcance 5**

[**1.3.**](#_heading=h.hk16p6of1f3b) **Definición, siglas y abreviaturas 5**

[**1.4.**](#_heading=h.ucia8wlixxb7) **Organización del documento 5**

[***2.***](#_heading=h.fo2ucd6uaoid) ***OBJETIVOS Y RESTRICCIONES ARQUITECTONICAS 5***

[2.1.1.](#_heading=h.s8v6qa6edi5t) Requerimientos Funcionales 5

[2.1.2.](#_heading=h.vcwcbydfv2c7) Requerimientos No Funcionales – Atributos de Calidad 5

[***3.***](#_heading=h.qzktbdz2yuzy) ***REPRESENTACIÓN DE LA ARQUITECTURA DEL SISTEMA 6***

[**3.1.**](#_heading=h.gcb77gvfzm1w) **Vista de Caso de uso 6**

[3.1.1.](#_heading=h.427znvx0b7rb) Diagramas de Casos de uso 6

[**3.2.**](#_heading=h.n03za3oq0w6w) **Vista Lógica 6**

[3.2.1.](#_heading=h.r1vdhsl8udr) Diagrama de Subsistemas (paquetes) 7

[3.2.2.](#_heading=h.y303pjlyo8y2) Diagrama de Secuencia (vista de diseño) 7

[3.2.3.](#_heading=h.3pq9ebye2e8g) Diagrama de Colaboración (vista de diseño) 7

[3.2.4.](#_heading=h.nhynqaxeyvia) Diagrama de Objetos 7

[3.2.5.](#_heading=h.f26b582a2559) Diagrama de Clases 7

[3.2.6.](#_heading=h.vqb69udrw09d) Diagrama de Base de datos (relacional o no relacional) 7

[**3.3.**](#_heading=h.a1f61lm6g4cx) **Vista de Implementación (vista de desarrollo) 7**

[3.3.1.](#_heading=h.3gw601ce5hya) Diagrama de arquitectura software (paquetes) 7

[3.3.2.](#_heading=h.hrlmi549i9zt) Diagrama de arquitectura del sistema (Diagrama de componentes) 7

[**3.4.**](#_heading=h.8yhy8zaao0h7) **Vista de procesos 7**

[3.4.1.](#_heading=h.rlfaro7i7x2a) Diagrama de Procesos del sistema (diagrama de actividad) 8

[**3.5.**](#_heading=h.8olhg3zacns1) **Vista de Despliegue (vista física) 8**

[3.5.1.](#_heading=h.s45dzmmgjdj6) Diagrama de despliegue 8

[***4.***](#_heading=h.6h6lm3bd1t9p) ***ATRIBUTOS DE CALIDAD DEL SOFTWARE 8***

[**Escenario de Funcionalidad 8**](#_heading=h.qna5mvxkd4f0)

[**Escenario de Usabilidad 8**](#_heading=h.5lpe26fvl0im)

[**Escenario de confiabilidad 9**](#_heading=h.uq3qruy7itq9)

[**Escenario de rendimiento 9**](#_heading=h.on2wuwlmzr6e)

[**Escenario de mantenibilidad 9**](#_heading=h.5q0562u7q382)

[**Otros Escenarios 9**](#_heading=h.h1uqva4vmzfu)

# INTRODUCCIÓN

* 1. Propósito (Diagrama 4+1)

Este documento presenta una visión integral y detallada de la arquitectura del sistema DataFiller, una plataforma web diseñada para automatizar la generación de datos de prueba realistas para bases de datos SQL y NoSQL. El propósito principal de DataFiller es proporcionar una arquitectura que cumpla con los requisitos funcionales y no funcionales del sistema, garantizando eficiencia, seguridad y una experiencia de usuario fluida. Dada la naturaleza de la aplicación, se ha elegido el estilo arquitectónico de 3 capas MVC (Modelo-Vista-Controlador), el cual permite una separación clara entre los datos, la interfaz de usuario y la lógica de control. Este documento adopta el modelo de vistas arquitectónicas 4+1, que ofrece diversas perspectivas del sistema, facilitando la comprensión y participación de los diferentes stakeholders según sus intereses. La arquitectura propuesta está diseñada para soportar la generación de datos sintéticos que respeten las relaciones entre tablas y las restricciones de integridad, además de facilitar la implementación del sistema de planes (gratuito y premium).

* 1. Alcance

El proyecto DataFiller abarcará el desarrollo de una plataforma web que permitirá a usuarios generar datos de prueba realistas para bases de datos, con las siguientes funcionalidades e inclusiones:

* Análisis automático de scripts SQL y NoSQL para detectar la estructura de tablas, relaciones y restricciones sin requerir conocimientos técnicos avanzados por parte del usuario.
* Generación de datos sintéticos realistas que respeten las relaciones entre tablas y las restricciones de integridad, con cantidades variables según el plan (10 registros por tabla en plan gratuito, cantidad ampliada en plan premium).
* Implementación de un sistema de planes con limitaciones diferenciadas:
  + Plan gratuito: 3 consultas diarias, 10 registros por tabla, formato SQL
  + Plan premium: Consultas ilimitadas, mayor cantidad de registros, todos los formatos, datos personalizados por industria
* Desarrollo de un sistema de autenticación de usuarios y gestión de suscripciones con integración a pasarela de pagos para el plan premium (S/9.99 mensual).
* Interfaz web intuitiva que permita a los usuarios pegar scripts, visualizar resultados y descargar datos generados.
* Sistema de soporte por correo electrónico para todos los usuarios, con atención prioritaria para usuarios premium.
  1. Definición, siglas y abreviaturas

SQL: Lenguaje de Consulta Estructurado

NoSQL: Not Only SQL

BD: Base de Datos

SCPT: Scripts

IA: Inteligencia Artificial

API: Application Programming Interface

MVC: Modelo-Vista-Controlador

QA: Quality Assurance (Aseguramiento de Calidad)

DBA: Database Administrator (Administrador de Base de Datos)

GDPR: General Data Protection Regulation (Reglamento General de Protección de Datos)

LPDP: Ley de Protección de Datos Personales

PCI-DSS: Payment Card Industry Data Security Standard

B/C: Beneficio/Costo

VAN: Valor Actual Neto

TIR: Tasa Interna de Retorno

JSON: JavaScript Object Notation

* 1. Organización del documento

La estructura del documento sigue el modelo de vistas arquitectónicas 4+1, describiendo el sistema desde las perspectivas de:

* Casos de uso: Requerimientos y funcionalidades
* Lógica: Estructura del sistema MVC
* Implementación: Organización del código fuente
* Procesos: Aspectos dinámicos
* Despliegue: Infraestructura cloud y servicios en la nube para el hosting y acceso al sistema

# **OBJETIVOS Y RESTRICCIONES ARQUITECTÓNICAS**

* 1. Priorización de requerimientos

| ID | Descripción | Prioridad |
| --- | --- | --- |
| RF01 | Análisis automático de scripts SQL para detectar estructuras | Alta |
| RF02 | Generación de datos sintéticos realistas que respeten relaciones y restricciones | Alta |
| RF03 | Interfaz web intuitiva para pegar scripts, visualizar y descargar datos | Alta |
| RF04 | Sistema de autenticación de usuarios | Media |
| RF05 | Sistema de planes con limitaciones diferenciadas (gratuito/premium) | Media |
| RF06 | Integración con pasarela de pagos para suscripciones premium | Baja |
| RF07 | Sistema de soporte por correo electrónico con prioridad según plan | Baja |
| RF08 | Datos personalizados por industria para usuarios premium | Baja |

### Requerimientos Funcionales

| ID | Descripción | Prioridad |
| --- | --- | --- |
| RF01 | Análisis automático de scripts SQL para detectar estructuras | Alta |
| RF02 | Generación de datos sintéticos realistas que respeten relaciones y restricciones | Alta |
| RF03 | Interfaz web intuitiva para pegar scripts, visualizar y descargar datos | Alta |
| RF04 | Sistema de autenticación de usuarios | Media |
| RF05 | Sistema de planes con limitaciones diferenciadas (gratuito/premium) | Media |
| RF06 | Integración con pasarela de pagos para suscripciones premium | Baja |
| RF07 | Sistema de soporte por correo electrónico con prioridad según plan | Baja |
| RF08 | Datos personalizados por industria para usuarios premium | Baja |

### Requerimientos No Funcionales – Atributos de Calidad

## 

| **ID. Requerimiento** | **Nombre del Requisito** | **Descripción de Requisito** | **Prioridad** |
| --- | --- | --- | --- |
| RF-001 | Seguridad | El sistema debe garantizar la seguridad de los datos mediante encriptación de contraseñas | Alta |
| RF-002 | Rendimiento | El sistema debe garantizar tiempos de respuesta rápidos al generar datos de prueba, incluso cuando los esquemas de base de datos sean grandes o complejos. | Alta |
| RF-003 | Disponibilidad | Mantener la plataforma disponible y accesible la mayor parte del tiempo. | Alta |
| RF-004 | Usabilidad | Diseñar una interfaz intuitiva y fácil de usar para los usuarios, permitiendo cargar esquemas de base de datos, personalizar configuraciones y generar datos con facilidad. | Media |
| RF-005 | Mantenibilidad | Facilitar la mantenibilidad del sistema mediante código limpio, documentación adecuada y pruebas robustas. | Media |

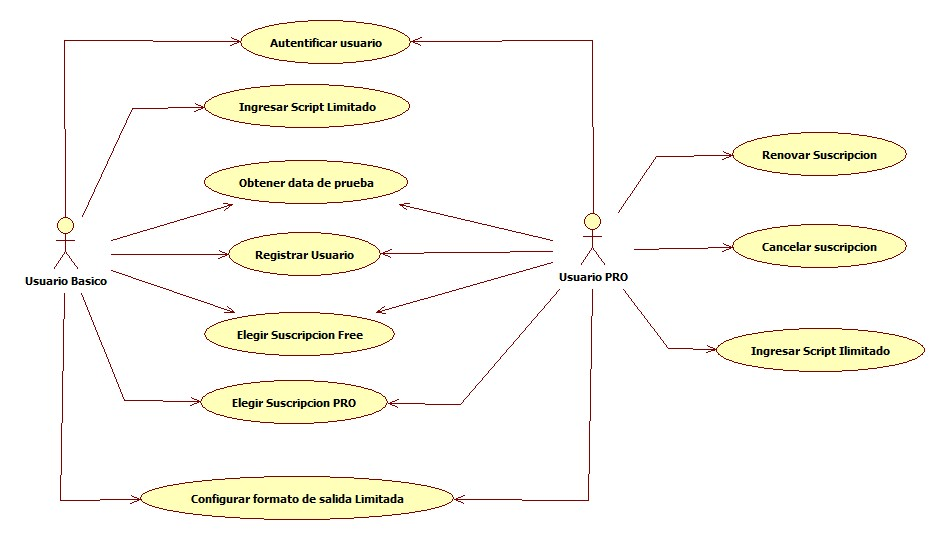
## 

* 1. Restricciones
* **Generación de Datos Sintéticos Realistas**: No se generarán ni utilizarán datos personales reales en la plataforma, cumpliendo con el Reglamento General de Protección de Datos (GDPR) y la Ley de Protección de Datos Personales (LPDP).
* **Limitaciones en el Uso de Datos para Testeo en Entornos Sensibles**: Los usuarios no deben utilizar la plataforma para generar datos sensibles que puedan violar la privacidad en sectores como salud, finanzas o datos de tarjetas de crédito, conforme a leyes como HIPAA y PSD2.
* **Cumplimiento con Normativas de Pago y Transacciones**: El sistema de gestión de pagos para el Plan Premium debe cumplir con la Norma de Seguridad de Datos para la Industria de Tarjetas de Pago (PCI-DSS).
* **Tecnología**: El desarrollo se realizará utilizando PHP versión 8 para el backend, y HTML5, CSS3, JavaScript y Bootstrap para el frontend.
* **Base de Datos**: Se utilizará MySQL 8 para gestionar los datos del sistema.
* **Infraestructura**: El sistema se desplegará en un servidor dedicado con Windows Server (Elastika).

# **REPRESENTACIÓN DE LA ARQUITECTURA DEL SISTEMA**

* 1. Vista de Caso de uso

### Diagramas de Casos de uso



* 1. Vista Lógica

*[La vista lógica se encarga de representar los requerimientos funcionales del sistema. Esta sección describe las partes del diseño del modelo significativas para la arquitectura, tales como subsistemas y paquetes.]*

### 

### Diagrama de Subsistemas (paquetes)

*[Diagrama que define los límites entre el sistema, o parte del sistema, y su ambiente, mostrando las entidades que interactúan con él. ​ Este diagrama es una vista de alto nivel de un sistema.*

*Asimismo, se debe desplegar las partes arquitectónicamente significativas del modelo de diseño, como ser la descomposición en capas, subsistemas o paquetes. Una vez presentadas estas unidades lógicas principales, se profundiza en ellas hasta el nivel que se considere adecuado.]*

### Diagrama de Secuencia (vista de diseño)

### Diagrama de Colaboración (vista de diseño)

### Diagrama de Objetos

### Diagrama de Clases

### Diagrama de Base de datos (relacional o no relacional)

* 1. Vista de Implementación (vista de desarrollo)

*[Se detalla la estructura general del Modelo de Implementación y el mapeo de los subsistemas, paquetes y clases de la Vista Lógica a subsistemas y componentes de implementación de manera más detallada]*

### Diagrama de arquitectura software (paquetes)

*[Se detalla la manera como fue implementado el sistema propuesto, se describe visualmente las capas que tiene el sistema, como están distribuidas y sus principales funciones]*

### Diagrama de arquitectura del sistema (Diagrama de componentes)

*[Se detalla la manera como fue implementado el sistema propuesto, se describe visualmente las capas que tiene el sistema, como están distribuidas y sus principales funciones]*

* 1. Vista de procesos

*[Describe la descomposición del sistema procesos pesados. Indica que procesos o grupos de procesos se comunican o interactúan entre sí y los modos en que estos se comunican.]*

### Diagrama de Procesos del sistema (diagrama de actividad)

*[Se realizará un diagrama del o los procesos del sistema donde se exponga las actividades donde interviene el sistema propuesto, adicionando diagramas que definan el detalle la descomposición del sistema en procesos pesados. Indica que procesos o grupos de procesos se comunican o interactúan entre sí y los modos en que estos se comunican]*

* 1. Vista de Despliegue (vista física)

*[Se despliega uno o más escenarios de distribución física del sistema sobre los cuales se ejecutará y hará el despliegue del mismo. Muestra la comunicación entre los diferentes nodos que componen los escenarios antes mencionados, así como el mapeo de los elementos de la Vista de Procesos en dichos nodos]*

### Diagrama de despliegue

*[un diagrama de despliegue, amplía el sistema de software y muestra los contenedores (aplicaciones, almacenamiento de datos, microservicios, etc.) que componen este sistema de software]*

# **ATRIBUTOS DE CALIDAD DEL SOFTWARE**

*[Los Atributos de Calidad (QAs) son propiedades medibles y evaluables de un sistema, estas propiedades son usadas para indicar el grado en que el sistema satisface las necesidades de los stakeholders [Wojcik 2013].*

*Los QAs además son concebidos como aquellos requerimientos que no son funcionales. De hecho, la funcionalidad es mayormente ortogonal a los QAs; un diseño puede cumplir con la funcionalidad deseada y fallar a la hora de satisfacer sus requerimientos de calidad. De esta manera, se entiende a la funcionalidad como la capacidad del sistema para hacer el trabajo para el cual fue pensado, independientemente de la estructura. Existen QAs mayormente usados que se suelen identificar en numerosos sistemas y se tienen que describir, aunque la lista no es fina ya que muy a menudo hay situaciones en que podrían identificarse y proponerse nuevas propiedades para las diversas necesidades de stakeholders.]*

*.*

**Escenario de Funcionalidad**

*[se califica de acuerdo con el conjunto de características y capacidades del programa, la generalidad de las funciones que se entregan y la seguridad general del sistema.]*

**Escenario de Usabilidad**

*[Este atributo de calidad se refiere a la facilidad con la que un usuario puede aprender a utilizar e interpretar los resultados producidos por un sistema [Barbacci 1995]. Para este atributo de calidad, se suelen considerar diversos aspectos de la interacción humano computadora, tales como: aprendizaje del sistema, utilización eficiente del sistema, minimización del impacto de errores, adaptación del sistema a las necesidades del usuario, confianza y satisfacción, entre otros.]*

**Escenario de confiabilidad**

*[Es el equilibrio entre la confidencialidad, la integridad, la irrefutabilidad y la disponibilidad de la información y datos manipulados por el sistema. Se trata del estado de un sistema, el cual puede ser transitorio y volátil. La seguridad de un sistema se caracteriza por mecanismos y técnicas empleados para intentar reducir los más posible el impacto provocado por un ataque, y las amenazas (entendidas como los caminos mediante los cuales se pueden provocar un ataque).*

*Abarca los planos de observación físico, lógico y humanos. Posee tres tipos de enfoque: prevención, precaución y reacción.]*

**Escenario de rendimiento**

*[Se mide con base en la velocidad de procesamiento, el tiempo de respuesta, el uso de recursos, el conjunto y la eficiencia.] (Pressman 2010, pág. 187)*

**Escenario de mantenibilidad**

*[Combina la capacidad del programa para ser ampliable (extensibilidad), adaptable y servicial. (Pressman 2010, pág. 187)*

**Otros Escenarios**

*[“Otros escenarios como por ejemplo: Performance”*

***Performance****: El atributo de calidad Performance se refiere a la capacidad de responder, ya sea el tiempo requerido para responder a eventos determinados, o bien, la cantidad de eventos procesados en un intervalo de tiempo dado. La Performance caracteriza la proyección en el tiempo de los servicios entregados por el sistema.]*