****

**UNIVERSIDAD PRIVADA DE TACNA**

**FACULTAD DE INGENIERÍA**

**Escuela Profesional de Ingeniería de Sistemas**

**“Proyecto *DevFlow”***

**Curso:**

*Patrones de Software*

**Docente:**

*Mag. Patrick Cuadros Quiroga*

**Integrantes:**

*Akhtar Oviedo, Ahmed Hasan - (2022074261)*

*Anampa Pancca, David Jordan - (2022074268)*

*Salas Jimenez, Walter Emmanuel - (2022073896)*

**Tacna – Perú**

*2025*

| CONTROL DE VERSIONES | | | | | |
| --- | --- | --- | --- | --- | --- |
| Versión | Hecha por | Revisada por | Aprobada por | Fecha | Motivo |
| 1.0 | AHAO, DJAP, WESJ | PCQ | - | 17/09/25 | Versión 1.0 |

# 

**Proyecto DevFlow**

**Documento de Arquitectura de Software**

**Versión *1.0***

| CONTROL DE VERSIONES | | | | | |
| --- | --- | --- | --- | --- | --- |
| Versión | Hecha por | Revisada por | Aprobada por | Fecha | Motivo |
| 1.0 | AHAO, DJAP, WESJ | PCQ | - | 17/09/25 | Versión 1.0 |

ÍNDICE GENERAL

**Contenido**

[***1.***](#_heading=h.tjgvbx7ihkut) ***INTRODUCCIÓN 5***

[**1.1.**](#_heading=h.47gjskb673gv) **Propósito (Diagrama 4+1) 5**

[**1.2.**](#_heading=h.u63ijdma55dt) **Alcance 5**

[**1.3.**](#_heading=h.3qfbwx91mir2) **Definición, siglas y abreviaturas 5**

[**1.4.**](#_heading=h.wi1nmmq6qrai) **Organización del documento 5**

[***2.***](#_heading=h.srshy1mzk0rh) ***OBJETIVOS Y RESTRICCIONES ARQUITECTONICAS 5***

[2.1.1.](#_heading=h.u5vosx7rwryj) Requerimientos Funcionales 5

[2.1.2.](#_heading=h.zejh9za72x00) Requerimientos No Funcionales – Atributos de Calidad 5

[***3.***](#_heading=h.3ru7upqjjejm) ***REPRESENTACIÓN DE LA ARQUITECTURA DEL SISTEMA 6***

[**3.1.**](#_heading=h.dw2vbowp2itr) **Vista de Caso de uso 6**

[3.1.1.](#_heading=h.fx6pj1i6crf9) Diagramas de Casos de uso 6

[**3.2.**](#_heading=h.sh2lg66i9paf) **Vista Lógica 6**

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

[3.2.2.](#_heading=h.wfkuhn49m1g) Diagrama de Secuencia (vista de diseño) 7

[3.2.3.](#_heading=h.clb6so2dqazv) Diagrama de Colaboración (vista de diseño) 7

[3.2.4.](#_heading=h.ipkxw38qdofw) Diagrama de Objetos 7

[3.2.5.](#_heading=h.yf7oqnx8f6db) Diagrama de Clases 7

[3.2.6.](#_heading=h.n4wexhqg9jlp) Diagrama de Base de datos (relacional o no relacional) 7

[**3.3.**](#_heading=h.m6eb6sy30d02) **Vista de Implementación (vista de desarrollo) 7**

[3.3.1.](#_heading=h.whlsk9kguh1p) Diagrama de arquitectura software (paquetes) 7

[3.3.2.](#_heading=h.fleswk17xneb) Diagrama de arquitectura del sistema (Diagrama de componentes) 7

[**3.4.**](#_heading=h.fje16axaqov8) **Vista de procesos 7**

[3.4.1.](#_heading=h.fmf5hbm0oilw) Diagrama de Procesos del sistema (diagrama de actividad) 8

[**3.5.**](#_heading=h.omgbzp9xgimi) **Vista de Despliegue (vista física) 8**

[3.5.1.](#_heading=h.7hkc9q5qimyd) Diagrama de despliegue 8

[***4.***](#_heading=h.iiod06kmgyh6) ***ATRIBUTOS DE CALIDAD DEL SOFTWARE 8***

[**Escenario de Funcionalidad 8**](#_heading=h.vic8ny3jb3vf)

[**Escenario de Usabilidad 8**](#_heading=h.87548ebqp1eq)

[**Escenario de confiabilidad 9**](#_heading=h.iflnryim5v2u)

[**Escenario de rendimiento 9**](#_heading=h.jtrgspli7s8i)

[**Escenario de mantenibilidad 9**](#_heading=h.gl03w3njcogs)

[**Otros Escenarios 9**](#_heading=h.tk658c5siahw)

1. INTRODUCCIÓN

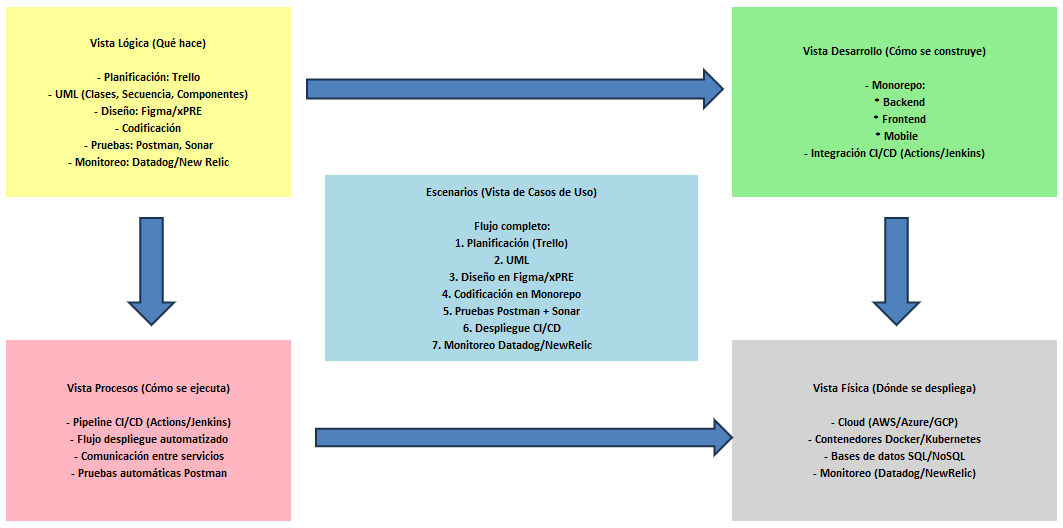
El presente documento describe la visión del proyecto DevFlow, una iniciativa tecnológica orientada a integrar de manera práctica y modular todas las fases del ciclo de vida del desarrollo de software (web, móvil y de escritorio). DevFlow nace como respuesta a una necesidad cada vez más evidente: la falta de un entorno unificado que acompañe a los desarrolladores, desde la concepción de la idea hasta el monitoreo de una aplicación en producción.

En el ámbito universitario, miles de estudiantes y docentes enfrentan el reto de enseñar y aprender metodologías de desarrollo en entornos fragmentados. Las prácticas suelen enfocarse en fases aisladas, dejando de lado la visión integral del ciclo de vida. Esto genera limitaciones para aplicar de manera real y completa las metodologías ágiles, DevOps y los marcos de calidad que exigen el mercado.

En el mundo profesional, los equipos de desarrollo —ya sean startups, freelancers o grandes empresas— suelen invertir un tiempo significativo configurando manualmente tableros, repositorios, pipelines y entornos de monitoreo, duplicando esfuerzos que podrían reutilizarse o estandarizarse. Esto genera costos innecesarios, baja productividad y dificulta mantener un flujo ordenado en proyectos de mediana y gran escala.

DevFlow busca cerrar esa brecha ofreciendo un ecosistema unificado y escalable, donde tanto un estudiante que desarrolla su primer prototipo como un equipo profesional que trabaja en un producto de alto impacto puedan encontrar un marco metodológico, artefactos preconfigurados y ejemplos prácticos listos para usar. El objetivo es claro: transformar la manera en que se aprende, planifica y ejecuta el desarrollo de software, brindando trazabilidad, orden y calidad en cada fase.

* 1. Propósito (Diagrama 4+1)



* 1. Alcance

El alcance del proyecto en su primera fase (MVP) se centra en construir un **flujo mínimo funcional** que demuestre la integración de todas las fases del ciclo de vida:

* **Planificación**: tablero en Trello con backlog, To Do, Doing, Review y Done.
* **Análisis**: creación de diagramas UML exportables (.puml).
* **Diseño**: wireframes y prototipos en Figma y xPRE, con tokens de diseño listos para reutilizar.
* **Codificación**: repositorio monorepo con scaffolds para backend (API REST en Node.js/Express), frontend web (React), móvil (React Native) y desktop (Electron).
* **Pruebas**: colección Postman (mínimo 3 endpoints) y configuración de SonarQube.
* **Despliegue**: Dockerfile, docker-compose.yml, workflow CI/CD (GitHub Actions) y Jenkinsfile declarativo.
* **Monitoreo**: dashboards en Datadog y alertas en New Relic.

Lo que quedará fuera del alcance inicial, pero se planificará para fases posteriores, incluye:

* Escalado automático en la nube.
* Analítica predictiva para métricas de calidad.
* Automatización avanzada de reportes personalizados.
* Integraciones multiusuario con permisos diferenciados.
  1. Definición, siglas y abreviaturas

*[Este apartado proporciona las definiciones de todos los términos, acrónimos y abreviaturas utilizadas a lo largo del documento y que permiten una interpretación correcta del mismo. Se han de incluir los términos técnicos, caso de uso por ejemplo, y los específicos del entorno del sistema, lector de bandas por ejemplo. Es conveniente ordenarlos alfabéticamente]*

* 1. Organización del documento

*[Aquí va la organización del proyecto]*

# **OBJETIVOS Y RESTRICCIONES ARQUITECTONICAS**

[Establezca las prioridades de los requerimientos y las restricciones del proyecto)

* 1. Priorización de requerimientos

### Requerimientos Funcionales

| RF | Requerimientos | Concepto | NIVEL |
| --- | --- | --- | --- |
| RF1 | Ingreso de Categorías | El sistema debe permitir registrar y actualizar categorías de contrataciones. | Alto |
| RF2 | Extracción de datos SEACE | El sistema debe conectarse a la fuente oficial del SEACE para recopilar información de contrataciones públicas. | Alto |
| RF3 | Clasificación por Categorías | El sistema debe organizar automáticamente los procesos de contratación según las categorías definidas. | Alto |
| RF4 | Búsqueda Avanzada | El sistema debe permitir búsquedas filtradas por: categoría, entidad contratante, monto, fecha, estado del proceso y proveedor. | Medio |
| RF5 | Visualización de Resultados | El sistema debe mostrar los resultados de las contrataciones en una tabla interactiva con paginación y filtros dinámicos. | Alto |
| RF6 | Exportación de Reportes | El sistema debe permitir exportar los resultados en formatos Excel, CSV y PDF. | Bajo |
| RF7 | Dashboards BI | El sistema debe generar gráficos y paneles dinámicos con indicadores como: Contrataciones por categoría , Contrataciones por entidad, Contrataciones por año/mes y Monto total por categoría?? | Alto |
| RF8 | Gestión de Usuarios | El sistema debe contar con roles de usuario: administrador, analista y público general, con diferentes niveles de acceso. | Medio |
| RF9 | Histórico de Contrataciones | El sistema debe almacenar un histórico de contrataciones para análisis comparativo en el tiempo. | Bajo |
| RF10 | Alertas y Notificaciones | El sistema debe permitir configurar alertas sobre nuevas contrataciones en una categoría de interés. | Bajo |

### Requerimientos No Funcionales – Atributos de Calidad

| **ID** | **Requerimiento** | **Concepto** | **Nivel** |
| --- | --- | --- | --- |
| RF1 | Usabilidad | Fácil de usar, con interfaz intuitiva y clara. | Alto |
| RF2 | Rendimiento | Respuesta rápida (<3s) y soporte para 100+ usuarios. | Alto |
| RF3 | Seguridad | Acceso seguro con roles, permisos y cifrado. | Alto |
| RF4 | Escalabilidad | Crece fácilmente con más recursos en la nube. | Medio |
| RF5 | Compatibilidad | Funciona en web, móvil, desktop y varias BD. | Alto |
| RF6 | Disponibilidad | Siempre activo (99.5%) con monitoreo en tiempo real. | Alto |
| RF7 | Mantenibilidad | Código modular y fácil de actualizar con CI/CD. | Alto |

* 1. Restricciones

*[Aquí van las restricciones del proyecto]*

# **REPRESENTACIÓN DE LA ARQUITECTURA DEL SISTEMA**

* 1. Vista de Caso de uso

*[En esta sección se describen los casos de uso del sistema (nombre de la aplicación), donde se abarcan todas las funcionalidades del sistema, se muestran los actores que interactúan en el sistema y las funcionalidades asociadas; asimismo se listará los casos de uso o escenarios del modelo de casos de uso que representen funcionalidades centrales del sistema final, que requieran una gran cobertura arquitectónica o aquellos que impliquen algún punto especialmente delicado de la arquitectura.*

*La documentación a incluir en esta sección corresponde a la obtenida como consecuencia de la actividad “Realización de casos de uso”:*

- *Flujos de eventos- Diseño: descripción textual de cómo se realiza el caso de uso en términos de los objetos que colaboran. Resumen de los diagramas conectados con el caso de uso y explicación de sus relaciones.*

- *Diagramas de interacción: Diagramas de secuencia, Diagramas de colaboración, objetos participantes, Diagramas de clases.*

- *Requisitos derivados: Descripción textual que recoge todos los requisitos, normalmente los no funcionales, de la realización del caso de uso no que han de tenerse en cuenta durante la implementación]*

### Diagramas de Casos de uso

*La descripción de la estructura se ilustra utilizando un conjunto de casos de uso escenarios lo que genera una nueva vista. Los escenarios describen secuencia de iteraciones entre objetos y entre procesos. Se utilizan para identificar y validar el diseño de arquitectura.*

* 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)

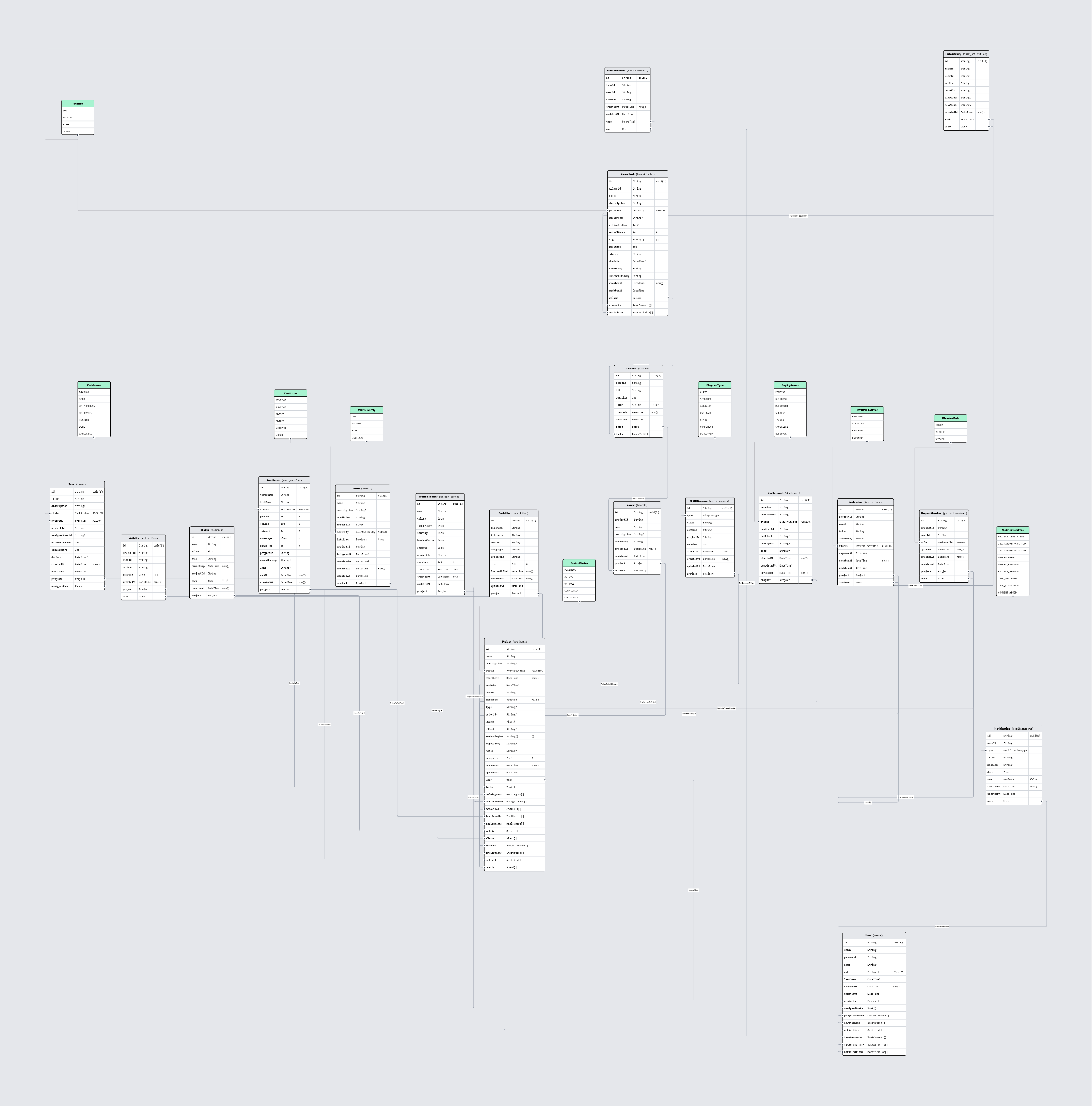
### Diagrama de Secuencia (vista de diseño)

### Diagrama de Colaboración (vista de diseño)

### Diagrama de Objetos

### Diagrama de Clases

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



Link de la Base de Datos (Mejor Calidad): https://drive.google.com/file/d/1Dq3GZXMRUfHN81btI3uv24WYLvsrNf2-/view?usp=sharing

* 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.]*