

**UNIVERSIDAD PRIVADA DE TACNA**

**FACULTAD DE INGENIERIA**

**Escuela Profesional de Ingeniería de Sistemas**

**Proyecto A*PI y Funciones de Lugares***

Curso: *Tópicos de Base de Datos Avanzados*

Docente: *Patrick Jose Cuadros Quiroga*

Alumno:

***Mayner Gonzalo Anahua Coaquira (2020067145)***

**Tacna – Perú**

***2024***

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

Sistema Proyecto API y Funciones de Lugares

Documento de Especificación de Requerimientos 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**

[INTRODUCCION](#_Toc394513795) 4

[I. Generalidades de la Empresa](#_Toc394513799) 5

[1. Nombre de la Empresa 5](#_Toc394513800)

[2. Vision 5](#_Toc394513800)

[3. Mision 5](#_Toc394513800)

[4. Organigrama 5](#_Toc394513800)

[II. Visionamiento de la Empresa](#_Toc394513799) 5

[1. Descripcion del Problema 5](#_Toc394513800)

[2. Objetivos de Negocios 5](#_Toc394513800)

[3. Objetivos de Diseño 5](#_Toc394513800)

[4. Alcance del proyecto 5](#_Toc394513800)

[5. Viabilidad del Sistema 5](#_Toc394513800)

[6. Informacion obtenida del Levantamiento de Informacion](#_Toc394513800) 6

[III. Análisis de Procesos](#_Toc394513799) 6

[a) Diagrama del Proceso Actual – Diagrama de actividades](#_Toc394513800) 6

[b) Diagrama del Proceso Propuesto – Diagrama de actividades Inicial](#_Toc394513800) 7

[IV Especificacion de Requerimientos de Software](#_Toc394513799) 7

[a) Cuadro de Requerimientos funcionales Inicial](#_Toc394513800) 7

[b) Cuadro de Requerimientos No funcionales](#_Toc394513800) 7

[c) Cuadro de Requerimientos funcionales Final](#_Toc394513800) 8

[d) Reglas de Negocio](#_Toc394513800) 9

[V Fase de Desarrollo](#_Toc394513799) 12

[1. Perfiles de Usuario](#_Toc394513800) 12

[2. Modelo Conceptual 5](#_Toc394513800)

[a) Diagrama de Paquetes 5](#_Toc394513800)

[b) Diagrama de Casos de Uso](#_Toc394513800) 12

[c) Escenarios de Caso de Uso (narrativa)](#_Toc394513800) 14

[3. Modelo Logico](#_Toc394513799) 23

[a) Analisis de Objetos](#_Toc394513800) 23

[b) Diagrama de Actividades con objetos](#_Toc394513800) 32

[c) Diagrama de Secuencia](#_Toc394513800) 37

[d) Diagrama de Clases](#_Toc394513800) 42

[CONCLUSIONES](#_Toc394513803) 46

[RECOMENDACIONES](#_Toc394513804) 46

[BIBLIOGRAFIA](#_Toc394513805) 46

[WEBGRAFIA](#_Toc394513806) 46

**INTRODUCCIÓN**

Este informe tiene como objetivo presentar de forma detallada el proyecto API de Lugares, desarrollado como parte del sistema de gestión de lugares, direcciones y categorías. La API ha sido diseñada utilizando FastAPI y CouchDB, proporcionando una interfaz eficiente para realizar operaciones CRUD sobre estos recursos. Además, se abordarán los objetivos, la viabilidad y los requisitos del proyecto, junto con el análisis de los procesos actuales y propuestos.

**I. Generalidades de la Empresa**

**1. Nombre de la Empresa**

Universidad Privada de Tacna (UPT)

**2. Visión**

Ser una universidad líder en el desarrollo y la innovación tecnológica, proporcionando soluciones eficaces y accesibles a las necesidades de la sociedad, especialmente en el ámbito de la gestión de datos y la tecnología aplicada a diversos sectores.

**3. Misión**

Formar profesionales competentes, creativos y responsables, con un alto nivel de calidad en su educación, capaces de generar soluciones tecnológicas que contribuyan al desarrollo sostenible de la región y del país.

**4. Organigrama**

Diagrama

Descripción generada automáticamente

**1. Descripción del Problema**

Actualmente, no existe una solución eficiente y escalable para gestionar la información sobre lugares, direcciones y categorías en un formato digital accesible. Los procesos manuales de gestión de estos datos limitan la accesibilidad y la efectividad en el uso de la información.

**2. Objetivos de Negocios**

* Mejorar la eficiencia en la gestión de lugares, direcciones y categorías.
* Proveer una API accesible que permita integrarse con otros sistemas.
* Facilitar el uso de tecnologías modernas para la gestión de datos en tiempo real.

**3. Objetivos de Diseño**

* Crear una API RESTful escalable y segura.
* Utilizar tecnologías como FastAPI y CouchDB para un alto rendimiento y flexibilidad.
* Desarrollar un sistema que pueda ser fácilmente desplegado y mantenido utilizando Docker.

**4. Alcance del proyecto**

El proyecto abarca el diseño y la implementación de la API de Lugares, incluyendo las funcionalidades de: Gestión de lugares, Gestión de direcciones y Gestión de categorías.

Despliegue y mantenimiento de la infraestructura usando Docker.

**5. Viabilidad del Sistema**

El sistema es viable desde el punto de vista técnico, económico y operativo. Se utilizarán tecnologías probadas como FastAPI y CouchDB, y el sistema será accesible mediante Docker, lo que facilita su mantenimiento y escalabilidad. El coste estimado del proyecto es accesible dentro de los recursos disponibles.

**6. Información obtenida del Levantamiento de Información**

Se recopiló información clave de los stakeholders para comprender las necesidades de los usuarios y los requisitos específicos del sistema. Los requerimientos funcionales y no funcionales fueron definidos con base en esta información.

**III. Análisis de Procesos**

**a) Diagrama del Proceso Actual – Diagrama de actividades**

Diagrama

Descripción generada automáticamente

**b) Diagrama del Proceso Propuesto – Diagrama de actividades Inicial**

Diagrama

Descripción generada automáticamente

**IV. Especificación de Requerimientos de Software**

**a) Cuadro de Requerimientos funcionales Inicial**

|  |  |  |
| --- | --- | --- |
| Código | Requerimiento | Descripción |
| RF01 | Gestión de lugares | CRUD para los lugares |
| RF02 | Gestión de direcciones | CRUD para las direcciones asociadas a los lugares |
| RF03 | Gestión de categorías | CRUD para las categorías de los lugares |

**b) Cuadro de Requerimientos No funcionales**

|  |  |  |
| --- | --- | --- |
| Código | Requerimiento | Descripción |
| RNF01 | Escalabilidad | El sistema debe ser capaz de manejar grandes volúmenes de datos |
| RNF02 | Seguridad | Autenticación y autorización para el acceso a la API |
| RNF03 | Rendimiento | Respuestas en menos de 200ms |

**c) Cuadro de Requerimientos funcionales Final**

|  |  |  |
| --- | --- | --- |
| Código | Requerimiento | Descripción |
| RF01 | Gestión de lugares | CRUD completo de lugares |
| RF02 | Gestión de direcciones | CRUD completo de direcciones |
| RF03 | Gestión de categorías | CRUD completo de categorías |
| RF04 | Despliegue | Uso de Docker para el despliegue fácil y escalable |

**Issues del Repositorio 'proyecto-si8811a-2024-ii-u2-api-y-funciones-anahua'**

Tabla 1: Issues del repositorio

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Número | Título | Estado | Creado por | URL |
| #10 | Automatizar pruebas unitarias del componente | closed | MAYnerAC | https://github.com/UPT-FAING-EPIS/proyecto-si8811a-2024-ii-u2-api-y-funciones-anahua/issues/10 |
| #9 | Realizar la ejecucion de pruebas unitarias del componente | closed | MAYnerAC | https://github.com/UPT-FAING-EPIS/proyecto-si8811a-2024-ii-u2-api-y-funciones-anahua/issues/9 |
| #8 | Automatizar para Integración Continua del componente | closed | MAYnerAC | https://github.com/UPT-FAING-EPIS/proyecto-si8811a-2024-ii-u2-api-y-funciones-anahua/issues/8 |
| #7 | Refactorizar de Codigo a Componentes | closed | MAYnerAC | https://github.com/UPT-FAING-EPIS/proyecto-si8811a-2024-ii-u2-api-y-funciones-anahua/issues/7 |
| #6 | FD06-EPIS-PropuestaProyecto | closed | MAYnerAC | https://github.com/UPT-FAING-EPIS/proyecto-si8811a-2024-ii-u2-api-y-funciones-anahua/issues/6 |
| #5 | Crear el Roadmap en la sección de Wiki | closed | MAYnerAC | https://github.com/UPT-FAING-EPIS/proyecto-si8811a-2024-ii-u2-api-y-funciones-anahua/issues/5 |
| #4 | FD04-EPIS-Informe Arquitectura de Software | closed | MAYnerAC | https://github.com/UPT-FAING-EPIS/proyecto-si8811a-2024-ii-u2-api-y-funciones-anahua/issues/4 |
| #3 | FD03-EPIS-Informe Especificación Requerimientos | closed | MAYnerAC | https://github.com/UPT-FAING-EPIS/proyecto-si8811a-2024-ii-u2-api-y-funciones-anahua/issues/3 |
| #2 | FD02-EPIS-Informe Vision | closed | MAYnerAC | https://github.com/UPT-FAING-EPIS/proyecto-si8811a-2024-ii-u2-api-y-funciones-anahua/issues/2 |
| #1 | FD01-EPIS-Informe de Factibilidad | closed | MAYnerAC | https://github.com/UPT-FAING-EPIS/proyecto-si8811a-2024-ii-u2-api-y-funciones-anahua/issues/1 |

Fuente: Elaboración propia

**d) Reglas de Negocio**

Cada lugar debe tener una dirección asociada.

Las categorías deben ser únicas y asociadas a un conjunto específico de lugares.

Solo los administradores tienen permiso para modificar los datos.

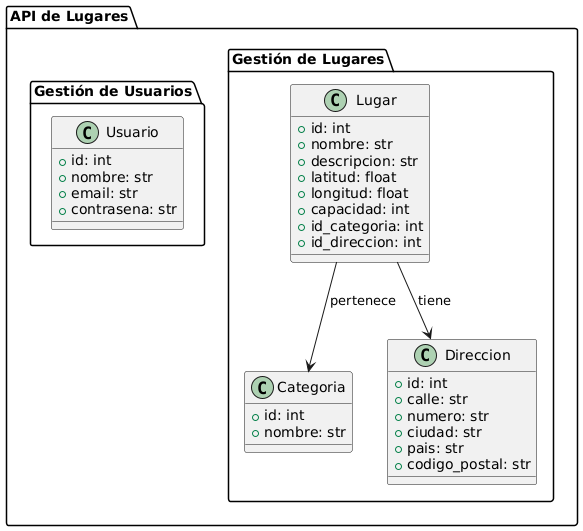
**V. Fase de Desarrollo**

**1. Perfiles de Usuario**

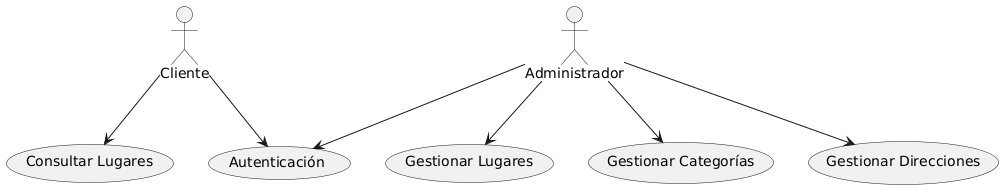
* Administrador: Usuario encargado de gestionar la información de lugares, direcciones y categorías.
* Desarrolladores: Usuarios que implementarán y mantendrán la API.

**2. Modelo Conceptual**

**a) Diagrama de Paquetes**



**b) Diagrama de Casos de Uso**



**c) Escenarios de Caso de Uso (narrativa)**

Caso de uso: Crear Lugar

Actor: Administrador

Descripción: El administrador crea un nuevo lugar proporcionando los detalles necesarios como nombre, dirección, capacidad, etc.

Flujo principal: El administrador ingresa los datos del lugar, la API los valida y los almacena en CouchDB.

**3. Modelo Lógico**

**a) Análisis de Objetos**

Diagrama

Descripción generada automáticamente

**b) Diagrama de Actividades con objetos**

Diagrama

Descripción generada automáticamente

**c) Diagrama de Secuencia**

Diagrama

Descripción generada automáticamente

**d) Diagrama de Clases**

Diagrama

Descripción generada automáticamente

**CONCLUSIONES**

El desarrollo de la API de Lugares es una solución viable que permite la gestión eficiente de información de lugares, direcciones y categorías. La utilización de FastAPI, CouchDB y Docker garantiza un sistema rápido, escalable y seguro.

**RECOMENDACIONES**

Se recomienda proceder con la implementación de la API conforme a los plazos establecidos, asegurando una adecuada configuración de la infraestructura y pruebas exhaustivas para garantizar el funcionamiento óptimo del sistema.

**BIBLIOGRAFÍA**

* FastAPI. <https://fastapi.tiangolo.com/>
* CouchDB. <https://couchdb.apache.org/>

**WEBGRAFÍA**

* Docker. <https://docs.docker.com/>