Documento de Diseño

para

Sistema Web para la Gestión de Accesos por COVID-19

Version 2.0

Presentado por: 1. Juan Manuel Cuellar Borrero
2. Nicolás Ibagón Rivera
3. Santiago Uribe Pastás

Presentado a: Gustavo Salazar Garzón
Lector

Noviembre de 2020

Índice general

1 Introducción

El presente documento de diseño brindará a los lectores una comprensión adecuada de todas las fases de diseño que comprende el sistema web para la gestión de accesos por COVID-19.

1.1. Propósito

Este documento tiene como propósito presentar una descripción sobre la arquitectura del proyecto del software, en base al Modelo de Vistas "4+1" el cual describe la arquitectura del software, la cual se compone de cinco vistas concurrentes, siendo estas respectivamente Vista de casos de Uso, Vista Lógica, Vista de Procesos, Vista Física, Vista de Despliegue. Las vistas son usadas para describir el sistema desde el punto de vista de diferentes Stakeholders tales como desarrolladores, ingenieros de sistemas, gerentes de proyecto y usuarios finales. La arquitectura de software que se presenta corresponde a la definitiva a utilizar en el sistema, proporcionando distintas opciones de arquitectura debido a los diferentes puntos de vista de el equipo de desarrollo. El documento proporciona diagramas y diseños detallados, las cuales brindan una mejor visión de las interacciones entre los diferentes tipos de usuarios y el sistema.

1.2. Alcance

El presente documento pretende mostrar, especificar y definir, de una forma clara y concisa la estructuración y diseño del Sistema Web para la Gestión de Accesos por COVID-19. En este se muestran distintos diagramas los cuales facilitan el entendimiento del sistema tanto a nivel estructural como a nivel de comportamiento, haciendo un seguimiento a las acciones que se pueden realizar en el sistema desde la perspectiva de cada uno de los actores, así como análisis de arquitecturas de software a tomar en cuenta a la hora de desarrollar el sistema.

El sistema en cuestión será desarrollado por un grupo de estudiantes de séptimo semestre de la materia *Procesos Y Diseño de Software* en la Pontificia Universidad Javeriana Cali, razón por la cual, si bien este no será liberado y utilizado a nivel nacional por un público común, sí se creará con la intensión de satisfacer los requisitos propuestos, poniendo en práctica los temas vistos durante todo el semestre, en un desarrollo más cercano a la realidad; esto intentado que el sistema mismo sea capaz de manejar un cantidad razonable de usuarios, todo esto tomando en cuenta limitaciones tales como el no uso de API's de pago o servidores de pago donde montar el sistema.

1.3. Definiciones, acrónimos y abreviaciones

- COVID-19 : Enfermedad infecciosa causada por el virus SARS-CoV-2.
- Elasticidad: Capacidad de un sistema de ampliar o reducir rápidamente los recursos informáticos del sistema para satisfacer demandas variables [1].
- Robustez: Capacidad de un sistema de comportarse en forma razonable aún en circunstancias que no fueron anticipadas en la especificación de requerimientos [2].
- SOA: Arquitectura Orientada a Servicios (por sus siglas en inglés)

2 Objetivos y Restricciones de la Arquitectura

En esta sección se detalla los objetivos de calidad que debe cumplir la arquitectura del sistema, por ende se centrará en dichos atributos para proponer arquitecturas posibles para el sistema. De igual forma se muestran cuales serán las restricciones de la arquitectura.

Los principales objetivos a cumplir por la arquitectura son los siguientes:

- Escalabilidad: El sistema tendrá la capacidad de poder soportar cargas de trabajo mas altas con el paso del tiempo.
- Rendimiento: El sistema tendrá tiempos de respuesta adecuados dependiendo de la acción que se esté realizando.
- Disponibilidad: El sistema tendrá una disponibilidad del 96 % lo que es equivalente a que únicamente 2 semanas (15 días) al año el software no estará disponible.

Como restricciones se tiene que el sistema no tendrá que cumplir con tópicos como adaptabilidad, mantenibilidad, portabilidad, elasticidad [1] y robustez [2], entre otros.

3 Representación Arquitectónica

En esta sección se hará un análisis para descartar algunas de las arquitecturas y posteriormente se hará la selección de la arquitectura final para el software. "La arquitectura de software es un conjunto de patrones que proporcionan un marco de referencia necesario para guiar la construcción de un software, permitiendo al grupo de desarrollo compartir una misma línea de trabajo y cubrir todos los objetivos y restricciones de la aplicación"[3]; debido a lo anterior vemos que la elección de la arquitectura de software es muy importante pues la manera en que se estructura un sistema tiene un impacto directo sobre la capacidad de este para permitir o impedir que se satisfagan los atributos de calidad del sistema.

3.1. Arquitecturas Candidatas

Las arquitecturas de software que fueron analizadas y posteriormente descartadas fueron las siguientes:

- **Arquitectura basada en eventos:** Esta arquitectura fue descartada debido a que a que posee poca comprensibilidad, por lo que puede ser difícil prever qué pasará en respuesta a una evento, lo que dificultaría el desarrollo del sistema.
- **Arquitectura SOA:** Esta arquitectura fue descartada debido a que la velocidad de intercambio de información entre cada servicio es más lenta comparada con una conexión directa, por lo que los tiempos de respuesta se verán perjudicados.
- **Arquitectura cliente-servidor:** Al ser un servidor y múltiples clientes si hay algún problema y el servidor se cae, el sistema quedaría inutilizable por lo que su resistencia a fallas es muy mala. De igual forma el rendimiento se vería afectado cuando una gran cantidad de clientes envían peticiones simultáneas al mismo servidor, lo que generaría congestión de trafico.

3.2. Arquitectura Seleccionada

La arquitectura presentada a continuación fue la que después de los análisis y consideraciones mas cumple con los atributos de calidad del sistema.

- **Arquitectura 3 capas:** Esta arquitectura maneja un alto rendimiento (ya que posee pocas capas), por lo que los tiempos de respuesta del sistema serán bastante buenos. De igual forma tenemos que es fácil de desarrollar, no se necesita mucha experiencia para poder implementarla.

4 Vista de Casos de Uso

Los diagramas de caso de uso se encuentran adjuntos en el anexo como un archivo PDF con el nombre *Casos de Uso.pdf* junto con su respectiva documentación en un archivo PDF con el nombre *Documentación Casos de Uso.pdf*.

5 Vista Lógica

5.1. Diagrama de Clases

El diagrama de clases se encuentra en el anexo como *Diagrama de Clases.png*, este diagrama busca ilustrar como están conformadas y relacionadas las entidades que interactúan con el sistema. Cabe resaltar que los metodos correspondientes de cada clase hacen referencia a los casos de uso de cada actor.

5.2. Modelo de Datos

Para el modelo de almacenamiento a implementar en el sistema fue escogida Cassandra, la cual es una base de datos no relacional que utiliza CQL como lenguaje de consultas. Esta al ser una base de datos distribuida puede manejar, de una manera eficiente, la gran cantidad de datos que se manejarían para un sistema desplegado a nivel nacional como lo es el sistema web para la gestión de accesos por COVID-19 que se piensa desarrollar. Además de ello la disponibilidad de esta también es muy alta, esto pues se tiene una alta tasa de redundancia a través de la replicación en diferentes nodos. Cassandra es además una base de datos que en la industria ha dado buenos resultados, [4],[5],[6] pues gigantes de la tecnología como Apple, Netflix, CERN la utilizan en sus sistemas teniendo excelentes resultados.

6 Vista de Procesos

En esta sección se detallan los procesos del sistema. Esto se hace por medio de diagrama de actividades. Únicamente se mostrarán 2 funcionalidades del sistema, las cuales son el registro de visita de un civil en un establecimiento publico (Figura 6.1) y generar historial pruebas COVID-19 desde un establecimiento de salud (Figura 6.2).

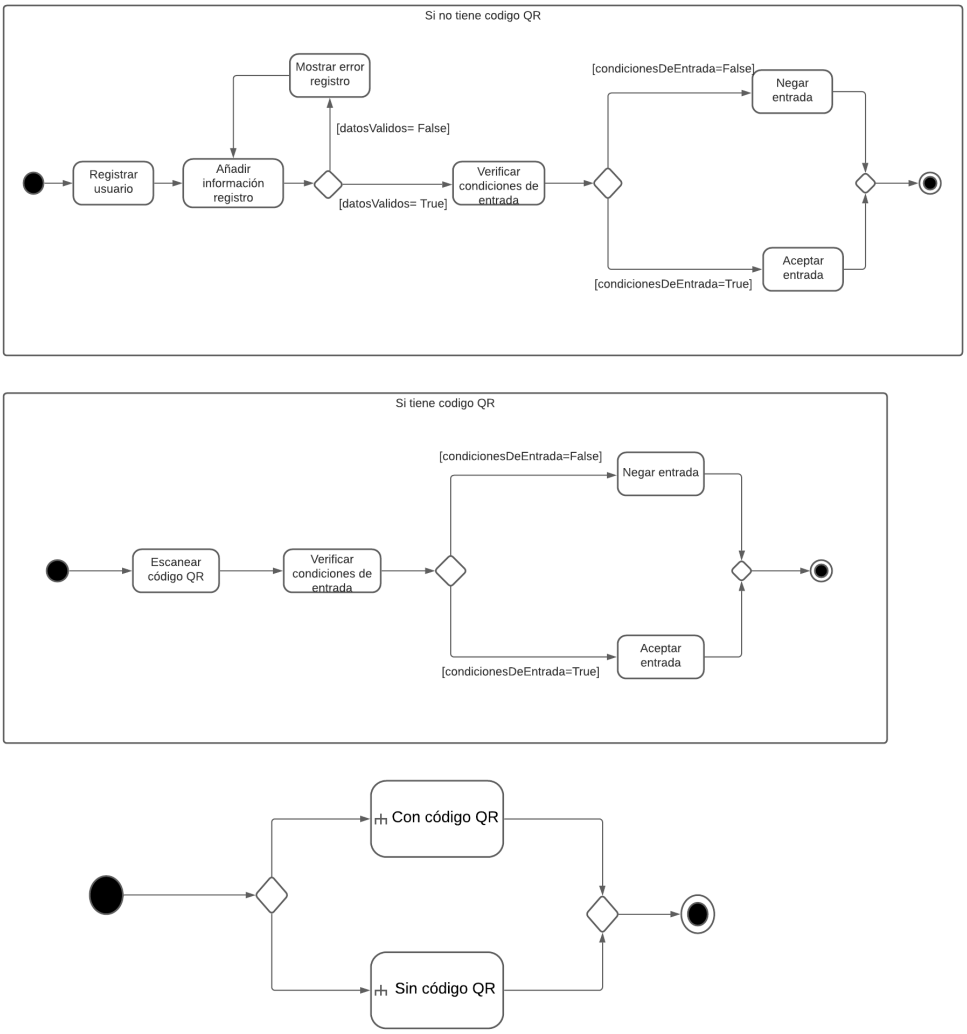

Figura 6.1: Diagrama de actividades funcionalidad 1

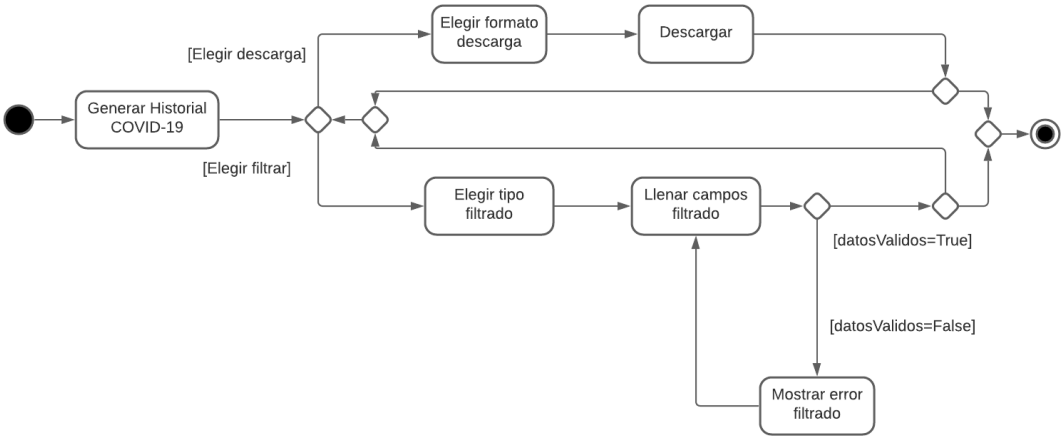

Figura 6.2: Diagrama de actividades funcionalidad 2

7 Vista Física

En esta sección se muestra la arquitectura final del sistema, la cual se puede apreciar en la Figura 7.1. Como se menciono en la sección 2.2 se maneja una arquitectura de 3 capas.

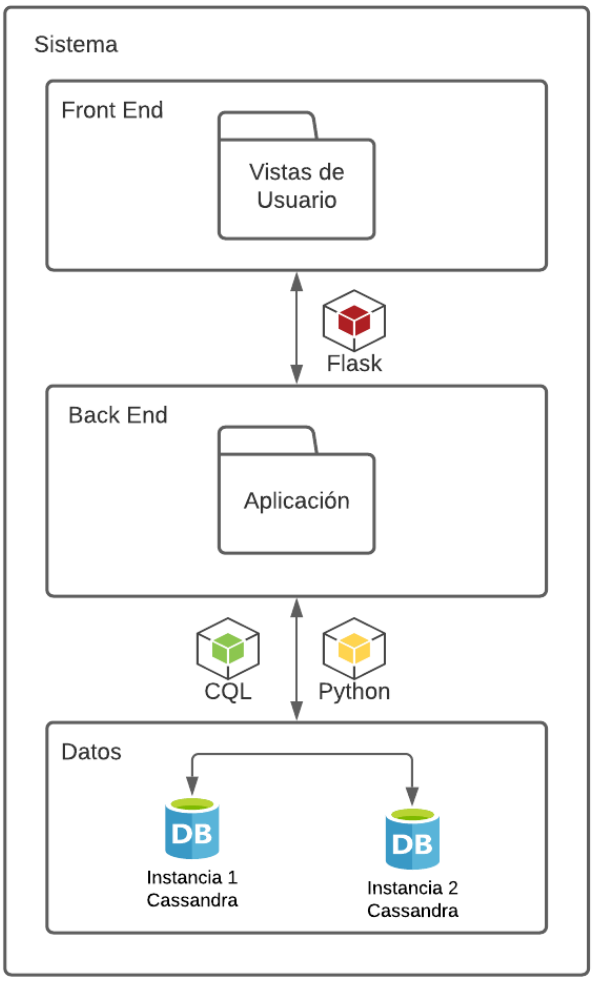

Figura 7.1: Arquitectura del sistema

7.1. Descripción de la Arquitectura

A continuación se describirá cada componente de la arquitectura. Como se maneja una arquitectura por capas se describirán cada una de las capas y como se comunican entre ellas.

- **Capa de Front End:** En esta capa se crea la interfaz del usuario, es decir, la capa con la que interactúa el usuario con el sistema. Esta compuesta por los formularios y las vistas creadas en HTML. Su función es pasarle las peticiones que realice el usuario a la capa de Back End.
- **Capa de Back End:** En esta capa se gestiona la lógica de la aplicación. Es donde se dice qué se hace y cómo se organizan los datos obtenidos de la capa de Front End. Estará conectada con la capa de datos para poder realizar sus funciones.
- **Capa de Datos:** Esta capa se encarga de guardar los datos. Será donde se gestione todo lo relativo a la base de datos y a la creación, edición y borrado de datos que se encuentren en ésta. Recibe solicitudes de almacenamiento o recuperación de información desde la capa de Back End.

La capa de Front End se comunica con la capa de Back End por medio del framework Flask (el cual esta escrito en Python) a través del archivo *app.py*.

La capa de Back End se comunica con la capa de Datos por medio del archivo *database.py*, el cual usa la librería `cassandra.cluster` para conectarse a la base de datos de Cassandra.

Cabe resaltar que las comunicaciones entre las capas son comunicaciones bidireccionales.

8 Vista de Despliegue

En esta sección se muestra el diagrama de despliegue del sistema. El propósito de este diagrama es proporcionar una vista de la topología del sistema, donde se puedan apreciar las relaciones físicas entre los componentes software y hardware en el sistema.

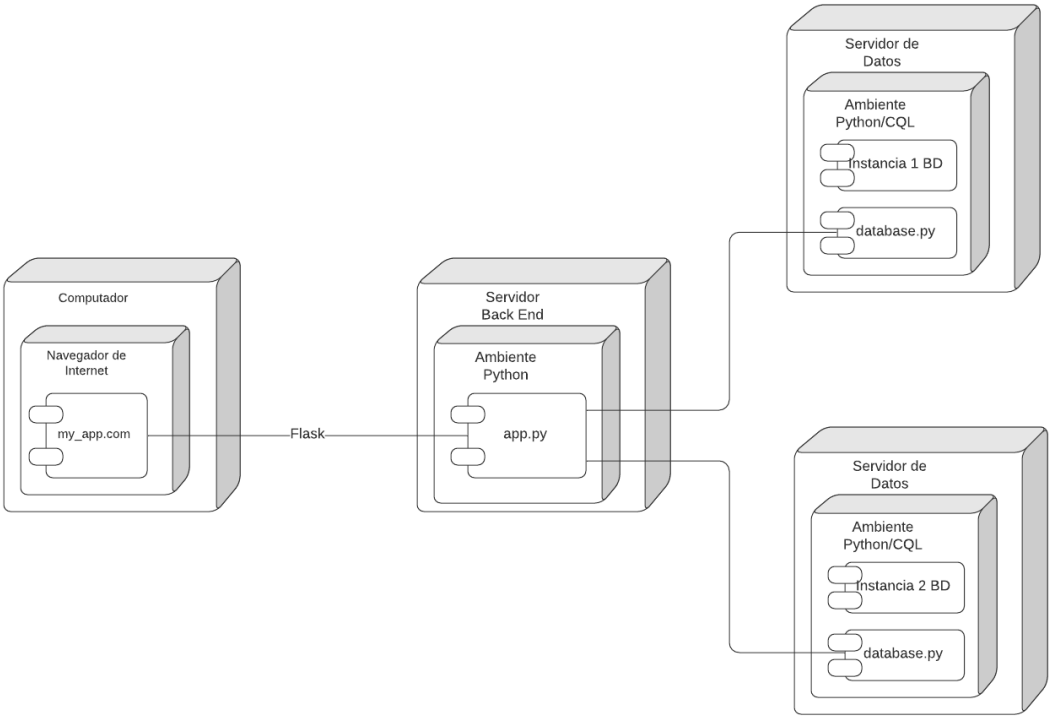

Figura 8.1: Diagrama de despliegue del sistema

Bibliografía

[1] "Qué es informática elástica: definición", Microsoft Azure, 2020. [Online]. Available: https://azure.microsoft.com/es-mx/overview/what-is-elastic-computing/.

[2] "Cualidades del software", fing.edu.uy, 2020. [Online]. Available: https://www.fing.edu.uy/tecnoinf/mvd/cursos/ingsoft/material/teorico/CualidadesSoftware.pdf.

[3] "Arquitectura de Software", ecured.cu. [Online]. Available: https://bit.ly/3mNEyNq

[4] A. Cockcroft, "Migrating Netflix from Datacenter Oracle to Global Cassandra", slideshare.net, 2011. [Online]. Available: https://www.slideshare.net/adrianco/migrating-netflix-from-oracle-to-global-cassandra.

[5] M. Asay, "Apple's secret NoSQL sauce includes a hefty dose of Cassandra", TechRepublic, 2015. [Online]. Available: https://www.techrepublic.com/article/apples-secret-nosql-sauce-includes-a-hefty-dose-of-cassandra/.

[6] "A Persistent Back-End for the ATLAS Online Information Service (P-BEAST)", CERN Document Server, 2012. [Online]. Available: https://cdsweb.cern.ch/record/1432912.