INTEGRACÌON CONTINUA

Proyecto

**TUTOR**

Ing. Carlo Iván Rivera

**PRESENTADO POR**

Luis Alfredo Bastidas Plazas cód. 1521020698

William Arturo Rodríguez Salamanca cód. 1421021084

Uriel Alonso Flórez Vargas cód. 1521020480

Diego Armando Pinillo Sánchez cód. 1521021465

Hermes Fernando Amaris Martínez cód. 1921024209

Politécnico Gran Colombiano

2019

**TABLA DE CONTENIDO**

[LISTA DE TABLAS 2](#_Toc24203926)

[OBJETIVOS 5](#_Toc24203927)

[OBJETIVO GENERAL. 5](#_Toc24203928)

[OBJETIVO ESPECÍFICO. 5](#_Toc24203929)

[PROBLEMA DE INVESTIGACIÓN 6](#_Toc24203930)

[JUSTIFICACIÓN 7](#_Toc24203931)

[PATRÓN DE ARQUITECTURA 8](#_Toc24203932)

[REQUERIMIENTOS FUNCIONALES 11](#_Toc24203933)

[REQUERIMIENTOS NO FUNCIONALES 13](#_Toc24203934)

[CASOS DE USOS 15](#_Toc24203935)

[ROLES DE ARQUITECTURA 21](#_Toc24203936)

[GERENTE DEL PROYECTO 21](#_Toc24203937)

[ARTEFACTOS 23](#_Toc24203938)

[GLOSARIO DE TÉRMINOS 31](#_Toc24203939)

[MOCKUPS 32](#_Toc24203940)

[DISEÑO REST 33](#_Toc24203941)

[CONCLUSIONES Y RECOMENDACIONES 34](#_Toc24203942)

[BIBLIOGRAFÍA 36](#_Toc24203943)

# 

# LISTA DE TABLAS

[Tabla 1 Patrón de arquitectura 8](#_Toc22055212)

[Tabla 2 RF crear 11](#_Toc22055213)

[Tabla 3 RF editar 11](#_Toc22055214)

[Tabla 4 RF consultar 12](#_Toc22055215)

[Tabla 5 RF eliminar 12](#_Toc22055216)

[Tabla 6 RF administrar sistema 13](#_Toc22055217)

[Tabla 7 RNF disponibilidad 13](#_Toc22055218)

[Tabla 8 RNF usabilidad 13](#_Toc22055219)

[Tabla 9 RNF seguridad 14](#_Toc22055220)

[Tabla 10 RNF rendimiento 14](#_Toc22055221)

[Tabla 11 RNF mantenibilidad 14](#_Toc22055222)

[Tabla 12 Caso de uso crear usuario 16](#_Toc22055223)

[Tabla 13 Caso de uso crear lista 16](#_Toc22055224)

[Tabla 14 Caso de uso parametrizar listas 17](#_Toc22055225)

[Tabla 15 Caso de uso modificar listas 17](#_Toc22055226)

[Tabla 16 Caso de uso eliminar listas 18](#_Toc22055227)

[Tabla 17 Caso de uso autenticar usuario 19](#_Toc22055228)

[Tabla 18 Caso de uso registro de datos. 19](#_Toc22055229)

[Tabla 19 Caso de uso editar datos. 20](#_Toc22055230)

[Tabla 20 Caso de uso administrar lista 20](#_Toc22055231)

**LISTA DE ILUSTRACIONES**

[Ilustración 1 patrón de arquitectura 9](#_Toc22055253)

[Ilustración 2 cliente 9](#_Toc22055254)

[Ilustración 3 Diagrama casos de usos 15](#_Toc22055255)

[Ilustración 4 Diagrama Entidades 23](#_Toc22055256)

[Ilustración 5 diagrama de objetivos Regional 23](#_Toc22055257)

[Ilustración 6 diagrama de objetivos Zonal 24](#_Toc22055258)

[Ilustración 7 diagrama de objetivos IPS 24](#_Toc22055259)

[Ilustración 8 diagrama de objetivos Estado 24](#_Toc22055260)

[Ilustración 9 diagrama de objetivos Usuario 24](#_Toc22055261)

[Ilustración 10 diagrama de objetivos Perfil 25](#_Toc22055262)

[Ilustración 11 diagrama de objetivos Lista 25](#_Toc22055263)

[Ilustración 12 diagramas de clases 26](#_Toc22055264)

[Ilustración 13 Diagrama de secuencia crear usuarios 26](#_Toc22055265)

[Ilustración 14 diagrama de secuencia crear lista 27](#_Toc22055266)

[Ilustración 15 diagrama de secuencia modificar lista 27](#_Toc22055267)

[Ilustración 16 diagrama de secuencia eliminar lista 28](#_Toc22055268)

[Ilustración 17 diagrama de secuencia auten-usuario 28](#_Toc22055269)

[Ilustración 18 diagrama de secuencia registrar lista 29](#_Toc22055270)

[Ilustración 19 diagrama de secuencia consultar lista 29](#_Toc22055271)

[Ilustración 20 diagrama de actividades 1 30](#_Toc22055272)

[Ilustración 21 diagrama de actividades 2 30](#_Toc22055273)

[Ilustración 22 MOCKUP 1 33](#_Toc22055274)

[Ilustración 23 MACKUPS 2 34](#_Toc22055275)

# OBJETIVOS

## OBJETIVO GENERAL.

* Diseñar e implementar un producto de software con el cual logremos dar respuesta a la necesidad identificada de agilizar el proceso de lista de chequeo de la EPS, con la cual demos una solución óptima y apropiada para las particularidades de la problemática de investigación presentada, empelando métodos de integración continua.

## OBJETIVO ESPECÍFICO.

* Conocer inequívocamente las necesidades del cliente para establecer el tipo de aplicación requerida y así dar la solución óptima a su problemática.
* Determinar el proceso de construcción de software adecuado con relación al tipo de cliente, y las restricciones de costos, tiempo y tecnología, en aras de diseñar correctamente los modelos propuestos y la arquitectura correcta.
* Aplicar correctamente los procesos de modelación de requerimientos y arquitectura de software, con el fin de explicar detalladamente a través de los diagramas correspondientes la estructuración interna y externa de la aplicación.
* Crear las cuentas Github para posterior sincronía de los diferentes participantes del grupo, para posteriormente crear los respectivos contenedores a través del Docker.
* Implementar un producto de software que permita dar una solución eficiente al planteamiento del problema para la EPS.

# PROBLEMA DE INVESTIGACIÓN

Cuando nos referimos a Recursos los definimos como todos aquellos elementos que se requieren para que una empresa u organización puedan alcanzar y lograr sus objetivos, estos son recursos son Humanos, financieros, materiales, técnicos y tecnológicos.

Por ello en la actualidad las empresas y organizaciones deben velar por cuidar y regular sus recursos, esto implica que incorporen procesos y metodologías óptimas para la administración eficientemente del uso adecuado de sus recursos, y es precisamente la forma como se desarrollen estos procesos lo que permitirá alcanzar rápidamente la consecución de sus objetivos.

Nuestro proyecto investigativo se desarrollará en una reconocida EPS, organización con presencia en 1.117 municipios a lo largo y ancho del país y cuya actividad económica se enfoca en asegurar y garantizar la prestación en servicios de salud a millones de colombianos. La necesidad de optimizar el proceso de evaluación y seguimiento a las instituciones prestadoras del servicio de salud (IPS) eficientemente a través de una herramienta de software que permita llevar un control organizado y sistematizado, de la infraestructura de cada una de estas, ya que en la actualidad este proceso se realiza de forma manual, mediante formatos impresos, lo cual dificulta que los supervisores realicen esta tarea de forma óptima y oportuna ocasionando retrasos en la entrega de la información y en la consolidación de los mismos.

Por otro lado, para los administradores al ser un proceso casi que manual se les dificulta llevar un control eficiente. Adicionalmente llevar las estadísticas se convierte en una tarea ardua y dispendiosa.

Por ello nos damos a la tarea de dar respuesta al problema planteado, identificar Cuál sería la alternativa sistemática real para nuestro cliente, desarrollar un producto de software con el cual la organización logre administrar eficientemente el recurso humano y la plataforma tecnológica.

Suplir adicionalmente la necesidad de los usuarios finales, que cuenten con una herramienta en línea que les permita registrar mediante una lista de chequeo las evaluaciones de infraestructura de las IPS para posteriormente llevar las estadísticas en informes de seguimiento pertinentes.

Por ello nos damos a la tarea de dar respuesta al problema planteado, identificar Cuál sería la alternativa sistemática real para nuestro cliente, desarrollar un producto de software con el cual la empresa logre administrar eficientemente el recurso humano y la plataforma tecnológica.

Suplir adicionalmente la necesidad de los usuarios finales, que cuenten con una herramienta en línea que les permita agendar y apartar sus recursos y así mismo conocer los horarios disponibles.

# JUSTIFICACIÓN

Nuestro proyecto se desarrollará en una reconocida organización del sector salud, es una empresa con 11 años de experiencia en el mercado, orientada a cubrir y garantizar la prestación de servicios de salud a millones de colombianos.

La organización ha decidido emprender acciones para mejorar sus servicios a través de su plan de avance y desarrollo, el cual tiene como acción principal ayudar a sus colaboradores, convirtiendo su experiencia de seguimiento y evaluación en la más cómoda y ágil, para lograrlo se ha propuesto poner a su disposición una aplicación de escritorio, con la cual los supervisores logren realizar el seguimiento y auditoría a las IPS de manera ágil y rápida.

En la actualidad el proceso de seguimiento y auditoría a las IPS se realiza de forma manual y a través de correo electrónico y formatos impresos, lo cual lo cual impide que los supervisores realicen el proceso de forma dinámica y eficiente, de igual forma realizar estas evaluaciones en horarios de oficina se convierte en una barrera de acceso al reporte de los seguimientos.

Nuestro proyecto se centrará en crear un producto de software que permita facilitar el proceso de seguimiento y reporte de estas auditorías, que permita en tiempo real realizar los seguimientos de infraestructura a las IPS, así mismo que permita optimizar los tiempos de entrega de los reportes. Adicionalmente esta herramienta debe permitir a los administradores consultar, editar o eliminar la información registrada.

Con este proyecto se pretende simplificar el proceso para el seguimiento a través de una lista de chequeo, esto conlleva a una mejoría en los tiempos de respuesta puesto que se simplificará el tiempo para reportar la información como también el proceso de consolidación, al igual que una eficiente administración y control para la organización.

## PATRÓN DE ARQUITECTURA

|  |  |
| --- | --- |
| **NOMBRE:** | **CHECK LIST** |
| **DEFINICIÓN:** | Producto de software para el registro, consolidación y administración de las estadísticas, auditorías y seguimientos de las IPS. Herramienta web lista de chequeo. |
| **PROBLEMA ESPECÍFICO RECURRENTE** | En la actualidad el proceso se realiza de forma manual a través de formatos impresos, para posterior tabulación en hojas de cálculo, recepción a través de correo electrónico, dificultando el proceso de consolidación. Esto adicionalmente ocasiona demoras en el proceso, aumentando los tiempos de reporte de las auditorías, así como posibles errores en la digitación y captación de la información. |
| **SOLUCIÓN:** | Se plantea desarrollar un producto de software, herramienta web a través de HTML, CSS y PHP. Con esta se pretende crear formularios para el registro y captación de la información, para posterior almacenamiento de estos en una base de datos, con el fin que la entidad pueda realizar el análisis de la información captada. Emplearemos Github como herramienta colaborativa y Docker para los contenedores. |
| **DIAGRAMA DE CLASES:** | Modelo Entidad relación para bases de datos. |

Tabla Patrón de arquitectura

El patrón que aplicamos a la Arquitectura es MVC son las siglas de Model View Controller, es decir, modelo vista controlador. El patrón de diseño MVC organiza el código en base a su función, separa el código en tres capas llamadas modelo, vista y controlador, que tiene como objetivo fundamental, separar el código de una aplicación de la siguiente manera:

* Separación de los datos de la aplicación de la interfaz de usuario, y relacionar las acciones de la interfaz de usuario con un controlador que está en medio entre el usuario y los datos.
* En el controlador recibe la lógica de control que nos permite accionar adecuadamente las acciones que el usuario realice en la aplicación.

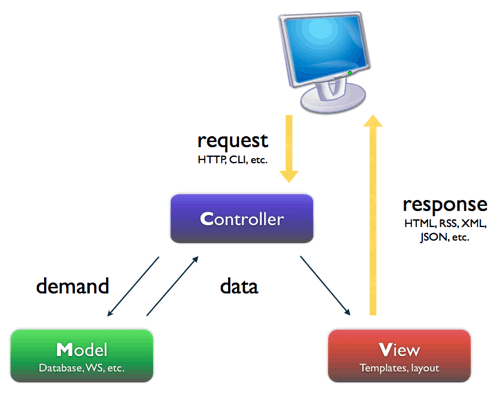


Ilustración patrón de arquitectura

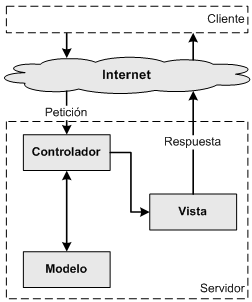


Ilustración cliente

Volviendo a la definición de MVC, el Modelo (M) se encargará de manejar los datos de nuestra aplicación y de persistir los datos. La Vista (V) tendrá como objetivo representar o mostrar los datos, siendo la capa más externa, la interfaz de usuario propiamente dicha, la cual no contiene ninguna lógica. La idea es que las vistas no contengan ningún archivo .CS o. VD enlazadas a las páginas Web, es decir, se utilizarán páginas Web puras sin enlace a código. Y por último el Controlador (C) es como hemos indicado ya, el intermediario o capa intermedia entre la vista y el Modelo.

**¿Cómo funciona la aplicación con MVC?**

**Captura de la petición en el controlador**

La aplicación recibe peticiones que son centralizadas en el Controlador. Éste es el encargado de interpretar, a partir de la URL de la solicitud, el tipo de operación que hay que realizar. Normalmente, esto se hace analizando el valor de algún parámetro que se envía anexando a la URL de la petición y que se utiliza con esta finalidad.

**Procesamiento de la petición**

Una vez que el Controlador determine la operación a realizar, procede a ejecutar las acciones pertinentes, invocando para ello a los diferentes métodos expuestos por el Modelo.

Dependiendo de las acciones a realizar (por ejemplo, un alta de un usuario en el sistema), el Modelo necesitará manejar los datos enviados por el cliente en la petición, datos que le serán proporcionados por el controlador. De la misma manera, los resultados generados por el Modelo (por ejemplo, la información resultante de una búsqueda será entregada directamente al controlador).

Para facilitar este intercambio de datos entre el Controlador y Modelo y, posteriormente, entre Controlador y Vista, las aplicaciones MVC suelen hacer uso de JavaBeans. Un JavaBean no es más que una clase que encapsula un conjunto de datos con métodos de tipo set/get para proporcionar un acceso a los mismos desde el exterior.

**Generación de respuestas**

Los resultados devueltos por el Modelo al Controlador son depositados por éste en una variable de petición, sesión o aplicación, según el alcance que deban tener. A continuación, el Controlador invoca a la página JSP (Java Server Page, Páginas de Servidor Java) que debe encargarse de generar la vista correspondiente, está página accederá a la variable de ámbito donde estén depositados los resultados y los utilizará para generar dinámicamente la respuesta XHTML que será enviada al cliente.

Para el desarrollo web de la aplicación se piensa trabajar bajo el lenguaje HTML5, que es el Lenguaje etiquetado de hipertexto, HTML5 pretende proporcionar una plataforma para desarrollar aplicaciones web más parecidas a las aplicaciones de escritorio, donde su ejecución dentro de un navegador no implique falta de recursos o facilidades para resolver las necesidades reales de los desarrolladores.

Para complementar este lenguaje, se utilizó Java Script es un lenguaje interpretado orientado a las páginas web, con una sintaxis semejante a la del lenguaje Java, se utiliza en páginas web HTML, para realizar tareas y operaciones en el marco de la aplicación cliente. Es un lenguaje pequeño y ligero; no es útil como un lenguaje independiente, más bien está diseñado para una fácil incrustación en otros productos y aplicaciones, tales como los navegadores Web.

Para Java Script posiblemente se utilicen librerías de JQuery que nos sirve para simplificar procesos e interactuar mejor con los documentos de HTML.

# REQUERIMIENTOS FUNCIONALES

|  |  |  |
| --- | --- | --- |
| ID | DESCRIPCIÓN | PRIORIDAD |
| RF1 | CREAR | PRIMARIO |
| R1.1 | El administrador del sistema crea usuario nuevo | PRIMARIO |
| R1.2 | El administrador del sistema crea listas de chequeo. | PRIMARIO |

Tabla RF crear

|  |  |  |
| --- | --- | --- |
| ID | DESCRIPCIÓN | PRIORIDAD |

|  |  |  |
| --- | --- | --- |
| RF2 | EDITAR | PRIMARIO |
| R2.1 | El administrador del sistema ingresa al sistema registrando sus datos | PRIMARIO |
| R2.2 | El administrador del sistema registra datos básicos de usuario nuevo. | PRIMARIO |
| R2.3 | El administrador del sistema registra datos lista. | PRIMARIO |
| R2.4 | El usuario registra recurso utilizado | PRIMARIO |
| R2.5 | El usuario registra sus datos de ingreso al sistema | PRIMARIO |

Tabla RF editar

|  |  |  |
| --- | --- | --- |
| ID | DESCRIPCIÓN | PRIORIDAD |
| RF3 | CONSULTAR | PRIMARIO |
| R3.1 | Consultar listado de chequeo por parte del supervisor. | PRIMARIO |
| R3.2 | Consultar datos usuarios por parte del administrador del sistema | PRIMARIO |
| R3.3 | Consultar lista por parte de cualquier usuario. | PRIMARIO |

Tabla RF consultar

|  |  |  |
| --- | --- | --- |
| ID | DESCRIPCIÓN | PRIORIDAD |

|  |  |  |
| --- | --- | --- |
| RF4 | ELIMINAR | PRIMARIO |
| R4.1 | Eliminar usuario por parte del administrador del sistema | PRIMARIO |
| R4.2 | Elimina listas por parte del administrador del sistema. | PRIMARIO |
| R4.3 | Eliminar listas por parte del administrador del sistema. | PRIMARIO |

Tabla RF eliminar

|  |  |  |
| --- | --- | --- |
| ID | DESCRIPCIÓN | PRIORIDAD |
| RF5 | ADMINISTRAR SISTEMA | PRIMARIO |
| R5.1 | Administrar datos de lista | PRIMARIO |
| R5.2 | Administrar datos usuarios | PRIMARIO |
| R5.3 | Administrar datos de gestión de listas | PRIMARIO |
| R5.4 | Administrar listas | PRIMARIO |

Tabla RF administrar sistema

# REQUERIMIENTOS NO FUNCIONALES

**Disponibilidad**

El sistema debe estar en la capacidad de soportar la carga total de trabajo, sin que su uso permite bloquear ni cerrar sus aplicativos.

|  |  |  |
| --- | --- | --- |
| ID REQUERIMIENTO | NOMBRE | DESCRIPCIÓN |
| RNF-001 | disponibilidad | El sistema debe tener una disponibilidad del 99,99% de las veces en que un usuario intente acceder. |

Tabla RNF disponibilidad

**Usabilidad**

El sistema debe estar en capacidad de ofrecer de forma rápida y efectiva toda la información que los clientes necesitan para ejecutar sus acciones, la creación de los usuarios debe ser una acción supremamente fácil de entender y realizar.

|  |  |  |
| --- | --- | --- |
| ID REQUERIMIENTO | NOMBRE | DESCRIPCIÓN |
| RNF-002 | usabilidad | El tiempo de aprendizaje del sistema por un usuario deberá ser menor a 4 horas. |

Tabla RNF usabilidad

**Seguridad**

El sistema debe guardar rigurosamente los datos del usuario y sus transacciones, además, debe almacenar el tipo de información utilizada y el usuario que realizó cada transacción.

|  |  |  |
| --- | --- | --- |
| ID REQUERIMIENTO | NOMBRE | DESCRIPCIÓN |
| RNF-003 | seguridad | Los permisos de acceso al sistema podrán ser cambiados solamente por el administrador de acceso a datos. |

Tabla RNF seguridad

**Rendimiento**

El sistema debe informar a cada usuario la finalización exitosa de cada acción o en su defecto informar el error presentado y su solución más próxima, una vez ejecutada cada operación.

|  |  |  |
| --- | --- | --- |
| ID REQUERIMIENTO | NOMBRE | DESCRIPCIÓN |
| RNF-004 | rendimiento | La aplicación debe ser compatible con todas las versiones de Windows, desde Windows 95. |

Tabla RNF rendimiento

**Mantenibilidad**

El sistema debe ser creada estrictamente por módulos, con el objetivo principal de facilitar su actualización o reemplazo por versiones nuevas.

|  |  |  |
| --- | --- | --- |
| ID REQUERIMIENTO | NOMBRE | DESCRIPCIÓN |
| RNF-005 | mantenibilidad | El promedio de duración de fallas no podrá ser mayor a 15 minutos |

Tabla RNF mantenibilidad

# CASOS DE USOS

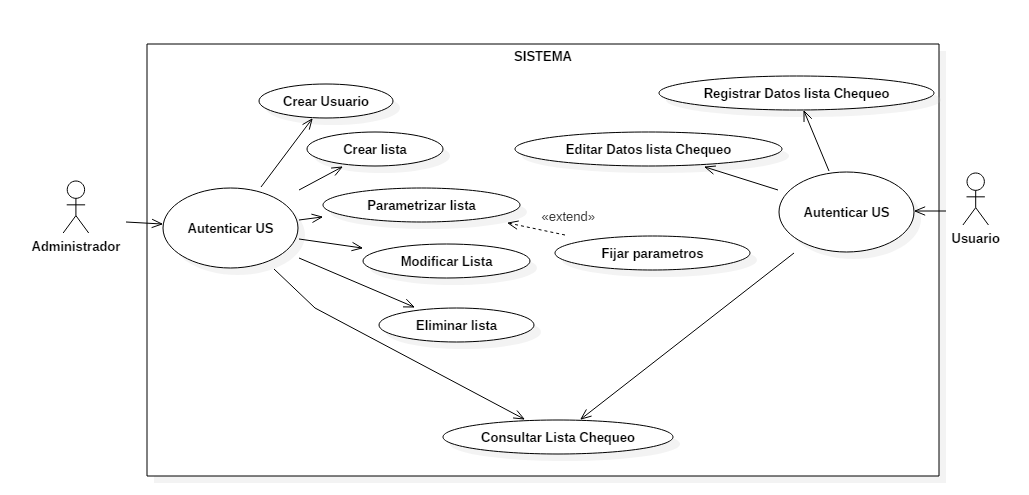


Ilustración Diagrama casos de usos

|  |  |  |
| --- | --- | --- |
| CASO DE USO | CREAR USUARIO | |
| ACTORES | ADMINISTRADOR | |
| OBJETIVO | CREACIÓN Y ALMACENAMIENTO DE NUEVO USUARIO. | |
| PRECONDICIONES | AUTENTICACIÓN DE ADMINISTRADOR EN EL SISTEMA. | |
| POSTCONDICIONES | ALMACENAMIENTO DE DATOS DE REGISTRO Y LOGIN. | |
| FLUJO DE EVENTO | ACTIVADORES DEL ACTOR | 1-Realiza la entrada de datos de usuario nuevo. |
|  | RESPUESTA DEL SISTEMA | 2-El sistema almacena los datos y genera a partir de ellos información de logín. |
| MANEJO DE SITUACIONES EXCEPCIONALES | En caso de presentar datos incompletos o erróneos el sistema lanzará una alerta que ponga al tanto al usuario sobre el campo a corregir. | |

Tabla Caso de uso crear usuario

|  |  |  |
| --- | --- | --- |
| CASO DE USO | CREAR LISTA | |
| ACTORES | ADMINISTRADOR | |
| OBJETIVO | ALMACENAMIENTO DE DATOS Y CREDENCIALES DE LOGIN. | |
| PRECONDICIONES | AUTENTICACIÓN DE ADMINISTRADOR EN EL SISTEMA. | |
| POSTCONDICIONES | ALMACENAMIENTO DE DATOS DE REGISTRO Y LOGIN. | |
| FLUJO DE EVENTO | ACTIVADORES DEL ACTOR | 1- Realiza la entrada y almacenamiento de datos. |
|  | RESPUESTA DEL SISTEMA | 2- El sistema almacena los datos y genera credenciales de acceso, agenda y recursos. |
| MANEJO DE SITUACIONES EXCEPCIONALES | En caso de fallar con el registro se solicitará el reingreso de datos. | |

Tabla Caso de uso crear lista

|  |  |  |
| --- | --- | --- |
| CASO DE USO | PARAMETRIZAR LISTAS/ FIJAR PARÁMETROS | |
| ACTORES | ADMINISTRADOR | |
| OBJETIVO | ASIGNACIÓN DE RECURSOS. | |
| PRECONDICIONES | AUTENTICACIÓN DE ADMINISTRADOR EN EL SISTEMA. | |
| POSTCONDICIONES | ALMACENAMIENTO DE DATOS DE REGISTRO Y LOGIN. | |
| FLUJO DE EVENTO | ACTIVADORES DEL ACTOR | 1- Realiza el almacenamiento de la información de las listas. |
|  | RESPUESTA DEL SISTEMA | 2- El sistema almacena y guarda la información registrada por el usuario. |
| MANEJO DE SITUACIONES EXCEPCIONALES | En caso de presentar inconvenientes con el almacenamiento de los listados el sistema retorna al menú principal. | |

Tabla Caso de uso parametrizar listas

|  |  |  |
| --- | --- | --- |
| CASO DE USO | MODIFICAR LISTA | |
| ACTORES | ADMINISTRADOR | |
| OBJETIVO | ACTUALIZAR DATOS DE LA LISTA. | |
| PRECONDICIONES | PREVIO REGISTRO DEL USUARIO. | |
| POSTCONDICIONES | ALMACENAMIENTO Y ACTUALIZACIÓN DE NUEVOS DATOS. | |
| FLUJO DE EVENTO | ACTIVADORES DEL ACTOR | 1-Actualización y/o modificación de datos. |
|  | RESPUESTA DEL SISTEMA | 2- El sistema almacena los cambios en los datos de usuario. |
| MANEJO DE SITUACIONES EXCEPCIONALES | En caso de no modificar con éxito los datos el sistema conservará los datos actuales y solicitará reintentar la actualización. | |

Tabla Caso de uso modificar listas

|  |  |  |
| --- | --- | --- |
| CASO DE USO | ELIMINAR LISTAS | |
| ACTORES | ADMINISTRADOR | |
| OBJETIVO | ELIMINAR LISTAS | |
| PRECONDICIONES | EXISTIR INFORMACIÓN DE LA LISTA DE CHEQUEO. | |
| POSTCONDICIONES | ELIMINACIÓN DE LA LISTA DE CHEQUEO. | |
| FLUJO DE EVENTO | ACTIVADORES DEL ACTOR | 1-Realizar la eliminación de los listados solicitados. |
|  | RESPUESTA DEL SISTEMA | 2-El sistema elimina la solicitud de los listados. |
| MANEJO DE SITUACIONES EXCEPCIONALES | En caso de no realizar la eliminación volverá a su estado inicial del listado. | |

Tabla Caso de uso eliminar listas

|  |  |  |
| --- | --- | --- |
| CASO DE USO | AUTENTICAR USUARIO | |
| ACTORES | USUARIO | |
| OBJETIVO | INGRESAR A LOS LISTADOS. | |
| PRECONDICIONES | PREVIO REGISTRO DE USUARIO, ASIGNACIÓN DE RECURSOS. | |
| POSTCONDICIONES | MOSTRAR DATOS ACTUALIZADOS SOBRE LISTAS Y ASIGNACIÓN DE RECURSOS. | |
| FLUJO DE EVENTO | ACTIVADORES DEL ACTOR | 1-Consulta de datos por parte de usuario. |
|  | RESPUESTA DEL SISTEMA | 2-El sistema muestra los datos consultados por el usuario lista, zonal e IPS. |
| MANEJO DE SITUACIONES EXCEPCIONALES | En caso de fallas en el sistema de consulta la aplicación mantendrá informado al usuario sobre los eventos generados. | |

Tabla Caso de uso autenticar usuario

|  |  |  |
| --- | --- | --- |
| CASO DE USO | REGISTRO DE DATOS LISTA DE CHEQUEO. | |
| ACTORES | USUARIO | |
| OBJETIVO | ACTUALIZAR DATOS DE LA LISTA. | |
| PRECONDICIONES | USUARIO PREVIAMENTE REGISTRADO. | |
| POSTCONDICIONES | REGISTRAR DATOS EN LAS LISTAS DE CHEQUEO. | |
| FLUJO DE EVENTO | ACTIVADORES DEL ACTOR | 1-Solicitud de registro de datos en las listas de chequeo. |
|  | RESPUESTA DEL SISTEMA | 2-El sistema registra los datos registrado. |
| MANEJO DE SITUACIONES EXCEPCIONALES | En caso de no registrar los datos este pasará a estar inactivo hasta que se reanude el registro. | |

Tabla Caso de uso registro de datos.

|  |  |  |
| --- | --- | --- |
| CASO DE USO | EDITAR DATOS DE LISTA DE CHEQUEO. | |
| ACTORES | USUARIO. | |
| OBJETIVO | EDICIÓN DE LISTAS Y DATOS ALMACENADOS. | |
| PRECONDICIONES | USUARIO PREVIAMENTE REGISTRADO. | |
| POSTCONDICIONES | EDITAR DATOS DE LISTAS DE CHEQUEO. | |
| FLUJO DE EVENTO | ACTIVADORES DEL ACTOR | 1-Solicitud de edición de datos en los listados. |
|  | RESPUESTA DEL SISTEMA | 2-El sistema edita el registro de datos en los listados. |
| MANEJO DE SITUACIONES EXCEPCIONALES | En caso de no editar los datos este pasará a estar inactivo hasta que se reanude la edición. | |

Tabla Caso de uso editar datos.

|  |  |  |
| --- | --- | --- |
| CASO DE USO | CONSULTAR LISTAS DE CHEQUEO. | |
| ACTORES | ADMINISTRADOR Y USUARIO. | |
| OBJETIVO | CONSULTA DE DATOS EN LOS LISTADOS. | |
| PRECONDICIONES | USUARIO Y ADMINISTRADOR PREVIAMENTE REGISTRADO. | |
| POSTCONDICIONES | LISTAR CONSULTA POR USUARIO O ADMINISTRADOR. | |
| FLUJO DE EVENTO | ACTIVADORES DEL ACTOR | 1-Solicitud de consulta por parte del usuario o administrador. |
|  | RESPUESTA DEL SISTEMA | 2-El sistema entrega la consulta de listados zonal e IPS. |
| MANEJO DE SITUACIONES EXCEPCIONALES | En caso de no generar la consulta de los listados éstos pasarán a estar inactivo hasta que se reanude la consulta. | |

Tabla Caso de uso administrar lista

# ROLES DE ARQUITECTURA

# GERENTE DEL PROYECTO

NOMBRE: Uriel Alonso Flórez Vargas

Es el líder principal del equipo, organiza, lidera, toma decisiones y ayuda a resolver los problemas, tienen relación directamente con cada miembro del equipo.

Dirige tanto hacia el interior del proyecto como hacia el exterior de este. Lidia con las situaciones que se presenten entre los miembros del equipo, y con las dificultades y decisiones propias de la gestión del proyecto. Se comunica con el cliente para asegurarse de que los recursos necesarios para el proyecto estén disponibles y para trabajar en aras de que el ambiente organizacional sea adecuado para el buen desarrollo del proyecto dentro de la empresa.

ARQUITECTO JEFE.

NOMBRE Luis Alfredo Bastidas Plazas

Definió, artículo, conceptualizo y exploro diferentes alternativas técnicas para dicha visión, creará modelos, componentes, documentará dichas creaciones y, validar la arquitectura contra los requerimientos definidos en conjunto con el cliente.

Logró comprometer al equipo y al cliente con la visión que se tiene, a través de un proceso claro de documentación y “venta” de dicha aproximación.

Está disponible para solucionar las dudas que se puedan presentar por parte del equipo de desarrollo en torno a la arquitectura planteada.

INGENIERO DE REQUERIMIENTOS

NOMBRE Diego Armando Pinillo Sánchez

Es la persona encargada de un conjunto de labores que incluye en primera instancia levantar los requerimientos, actividad que puede tomar múltiples formas, desde encuestas, entrevistas, revisión de documentos, observación natural, elaboración de prototipos, y en general actividades que requieren un alto grado de interacción con los clientes, y claridad respecto a los objetivos y estrategias generales del proyecto. Un buen analista de requerimientos debe estar familiarizado con el ciclo de vida escogido para el proyecto y comprender dónde encaja su labor en dicho ciclo.

INGENIERO PROCESOS DE CALIDAD.

NOMBRE William Arturo Rodríguez Salamanca

Es el encargado de que el proyecto, los procesos en él, y los productos resultantes, cumplan con los estándares definidos para medir la calidad de los mismos. De esta manera, está bajo su responsabilidad la definición de estrategias para que se alcancen los objetivos de calidad propuestos, así como la planeación, y revisión de mecanismos y actividades puntuales orientadas a la gestión de la calidad dentro del proyecto.

Participa en actividades de planeación, medición y mejora de los procesos técnicos y administrativos de la compañía, incluyendo la selección y capacitación del personal, la generación de documentos y artefactos, y en general, la verificación de que todas las fases del proceso de desarrollo se den de la mejor manera posible, y de acuerdo con lo definido dentro de la organización.

Participa en la prueba y manejo de calidad, revisando que los requerimientos del cliente sean los resultados de los requerimientos del cliente.

Desde el inicio del proyecto aparte de mi función principal he colaborado en el análisis de la propuesta, elaboración de la propuesta, planificación, análisis y diseño del proyecto.

INGENIERO PROGRAMADOR

NOMBRE Hermes Fernando Amaris Martínez

Es el encargado de realizar la implementación en código de la herramienta, aplicando las metodologías de programación, contara con la ayuda de los de más roles teniendo en cuenta la aplicación del trabajo colaborativo.

# 

# ARTEFACTOS

DIAGRAMA ENTIDAD RELACIÓN

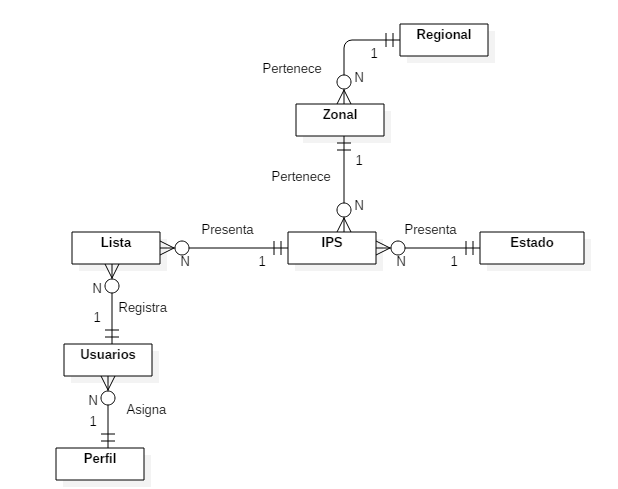


Ilustración Diagrama Entidades

DIAGRAMA OBJETOS

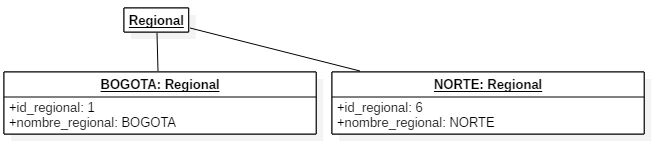


Ilustración diagrama de objetivos Regional

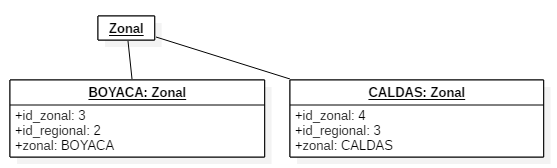


Ilustración diagrama de objetivos Zonal

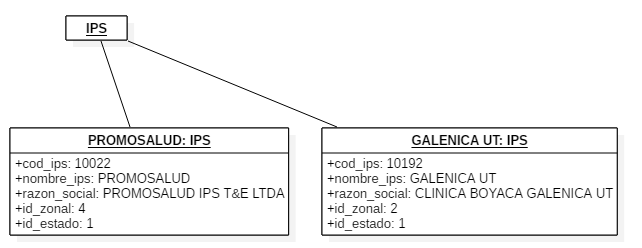


Ilustración diagrama de objetivos IPS

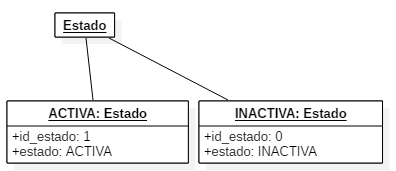


Ilustración diagrama de objetivos Estado

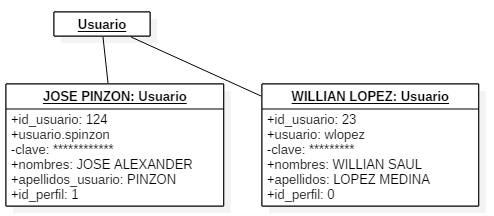


Ilustración diagrama de objetivos Usuario

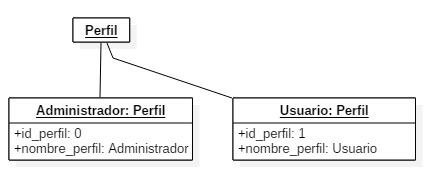


Ilustración diagrama de objetivos Perfil

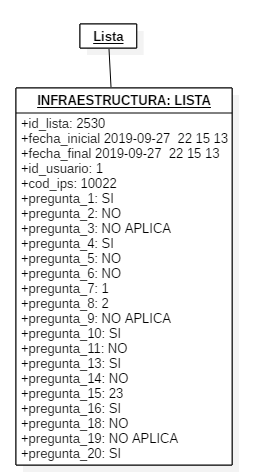


Ilustración diagrama de objetivos Lista

DIAGRAMA DE CLASES

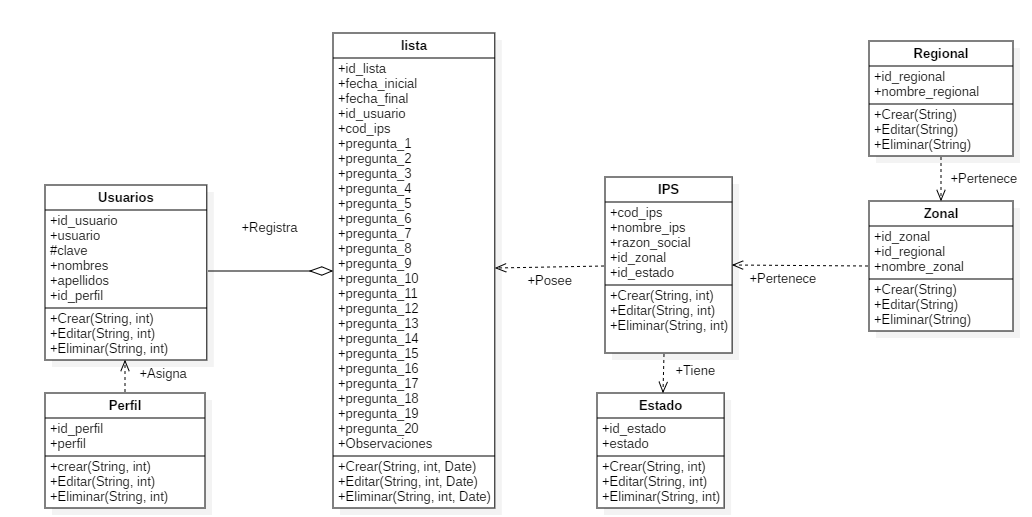


Ilustración diagramas de clases

DIAGRAMAS DE SECUENCIA

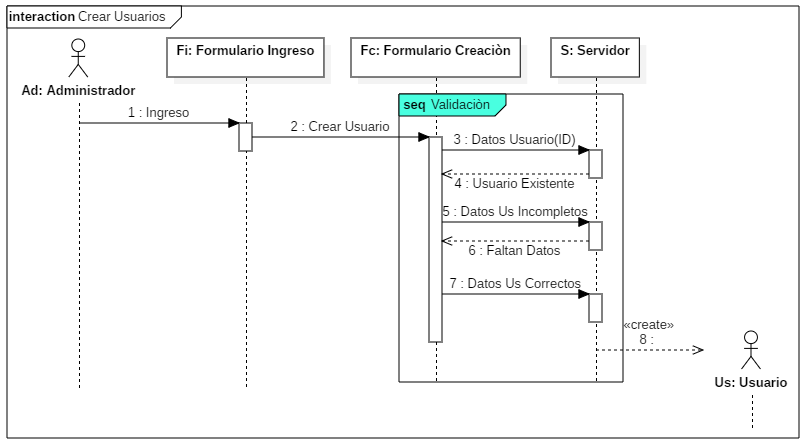


Ilustración Diagrama de secuencia crear usuarios

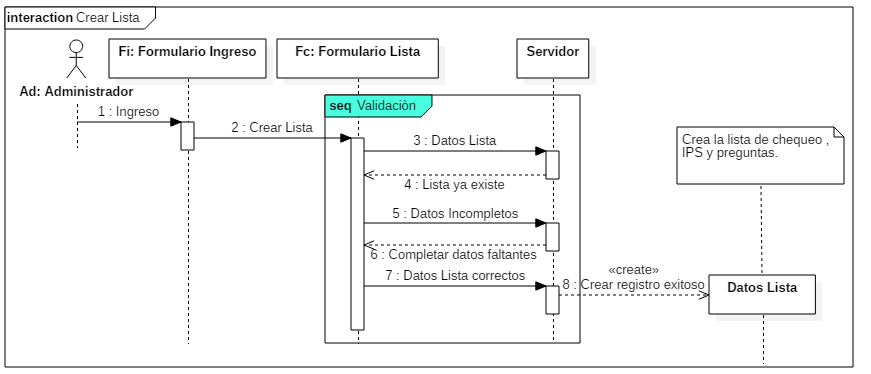


Ilustración diagrama de secuencia crear lista

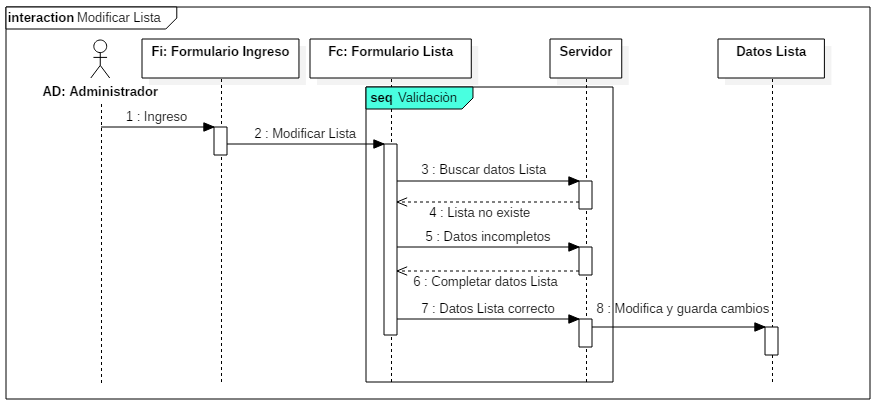


Ilustración diagrama de secuencia modificar lista

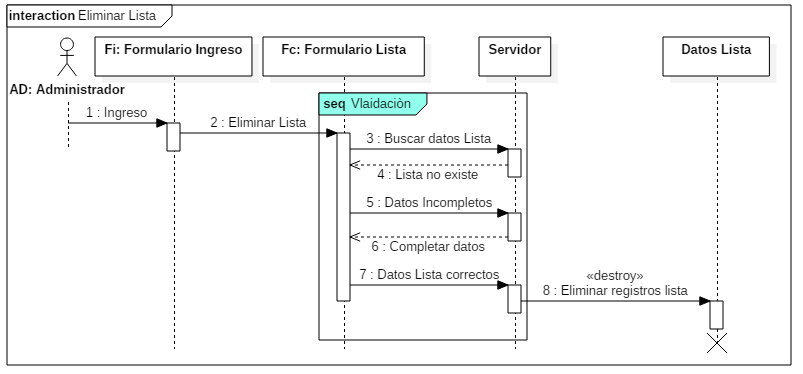


Ilustración diagrama de secuencia eliminar lista

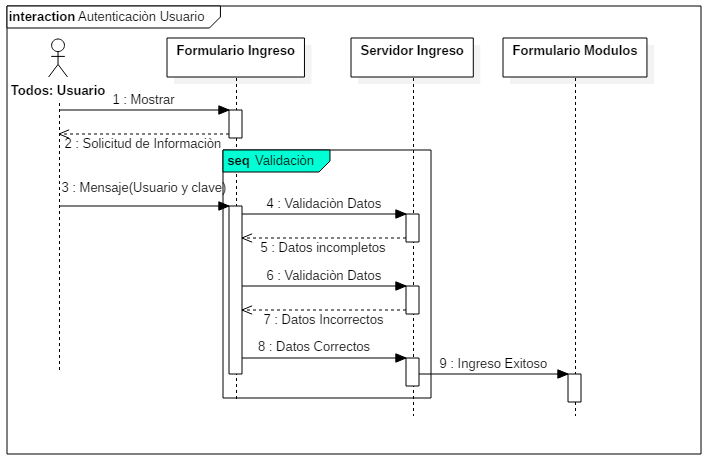


Ilustración diagrama de secuencia auten-usuario

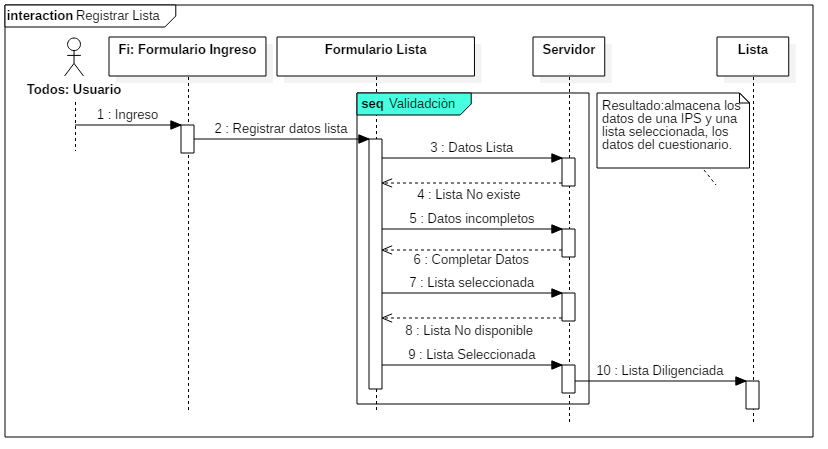


Ilustración diagrama de secuencia registrar lista

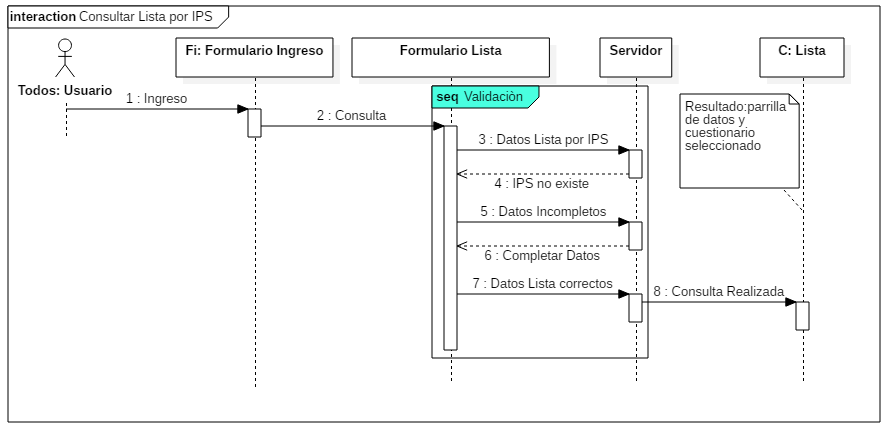


Ilustración diagrama de secuencia consultar lista

DIAGRAMA DE ACTIVIDADES

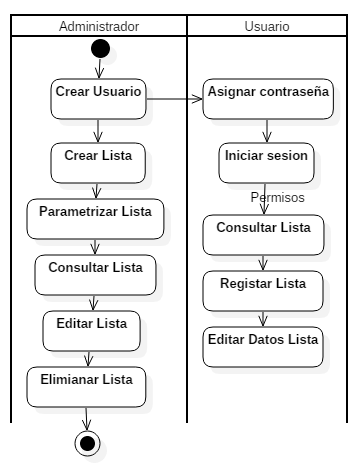


Ilustración diagrama de actividades 1

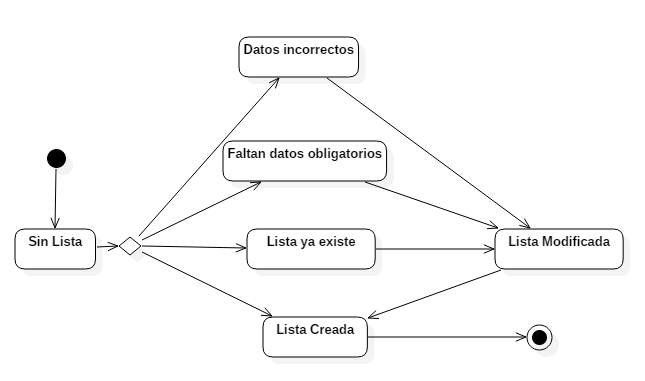


Ilustración diagrama de actividades 2

# GLOSARIO DE TÉRMINOS

**HARDWARE:** Partes físicas y manipulables del computador.

**SOFTWARE:** Programas internos del computador que le permiten realizar varias tareas determinadas.

**ARQUITECTURA:** Es el arte y técnica de diseñar. Proyectar, definir y construir elementos relacionados entre ellos.

**DISEÑAR**: Es la operación mediante la cual se materializa una idea a través de un documento, modelo, etc.

**PUNTO DE VISTA**: Es la postura, consideración u opinión respecto a un tema.

**ROL**: Es la actividad que una persona desempeña en un lugar o en una situación.

**ESTILO**: Conjunto de rasgos particulares de un diseño, patrón. etc.

**FRAMEWORK**: Es un grupo reutilizable de bibliotecas o clase de software.

**DRIVERS**: Son los encargados de guiar el diseño de la arquitectura del sistema.

**PATRONES DE ARQUITECTURA:** El que expresa un esquema de organización estructural esencial para un sistema de software que consta de subsistemas, responsabilidades e interacciones.

**ARQUITECTURA DE REFERENCIA:** Se refiere a una estructura prediseñada que mejora el trabajo, aumenta la manejabilidad y se implementa eficazmente en todos los dominios de la arquitectura, dando soluciones con mejoras actualizadas.

**MODELO**: Es un prototipo que sirve de referencia y ejemplo para todos los que diseñan y confeccionan productos de la misma naturaleza.

**METAMODELO**: Es el modelo de un modelo.

**ADL**: Architecture Description Language. (Lenguaje de la descripción de la arquitectura).

**CASE**: Computer Aided Software Engineering (Ingeniería de software asistida por computadora.

**SOA**: Arquitectura de software orientada a objetos.

**SERVICE INVENTORY DESIGN**: Diseño de inventario del servicio.

**SERVICE DESIGN PATROL**: Patrones de diseño de servicio.

**SERVICE COMPOSITION PATTERNS**: Patrones de diseño de composición de servicio.

**COMMON COMPOUND DESIGN PATTERNS:** Patrones de diseño compuestos comunes.

**LISTA DE CHEQUEO:** Conjunto de preguntas, para una auditoría o seguimiento.

# MOCKUPS

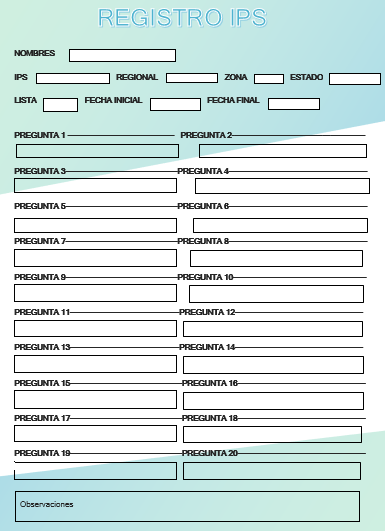


Ilustración MOCKUP 1

**Ilustración cronograma**

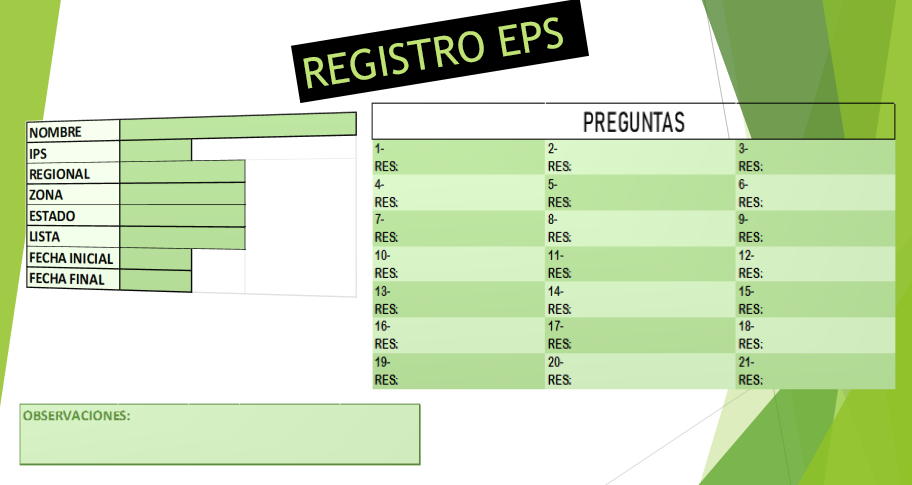


Ilustración MACKUPS 2

El método REST lo vamos a utilizar de la siguiente manera:

# DISEÑO REST

El problema planteado fue la necesidad de optimizar el proceso de evaluación y seguimiento a las instituciones prestadoras del servicio de salud (IPS)  eficientemente a través de una herramienta de software que permita llevar un control organizado y sistematizado, de la infraestructura de cada una de estas, ya que en la actualidad este proceso se realiza de forma manual, mediante formatos impresos, lo cual dificulta que los supervisores realicen esta tarea de forma óptima y oportuna  ocasionando retrasos en la entrega de la información y en la consolidación de los mismos.

Las operaciones nos permiten manipular el estado público del recurso. En nuestro sistema REST típico se definirán cuatro operaciones:

**CREATE**. En esta operación el usuario manda al servidor una petición para crear una nueva pregunta. Opcionalmente el usuario puede mandar una. El servidor responde con el identificador global de nueva pregunta.

**DELETE**. En esta operación el servidor elimina una pregunta de la base de datos. El usuario necesita saber el identificador de la pregunta.

**READ**. Con esta operación el usuario puede leer una representación del estado de una pregunta, identificado con su identificador global o número de la misma. Aquí lo que ocurre realmente es que se copia el estado del recurso en el servidor y se pega en le encuesta. Ambas copias del estado no se mantienen sincronizadas. El servidor puede cambiar el estado real del recurso y el usuario, de forma independiente, puede modificar su copia local del estado del recurso.

**UPDATE**. Como el servidor y el usuario tienen una copia diferente del estado, el usuario puede usar esta operación para sobrescribir o grabar su copia del estado en el servidor. De esta manera se puede actualizar el estado del recurso con las modificaciones hechas en el usuario.

En nuestro diagrama de clases de la base de datos llevamos la siguiente información en la cual empezamos a ingresar nuestras componentes:

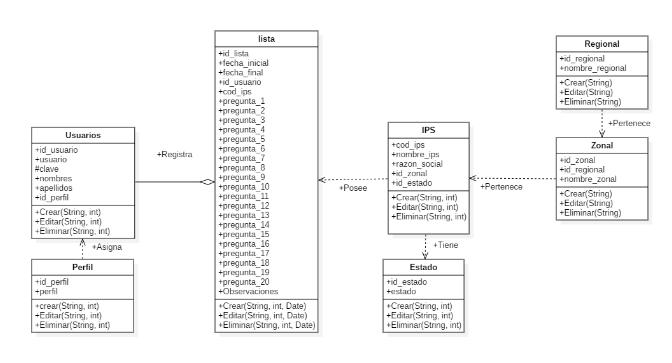


Ilustración diseño clases base de datos

RESULTADOS DE LA METODOLOGÍA ESCOGIDA

La metodología escogida para realizar el análisis e implementar el servicio escogido fue el método REST (REPRESENTATIONAL STATE TRANSFER).

En el cual se apoyó y se probó los métodos básicos de HTTP, como son:

# CONCLUSIONES Y RECOMENDACIONES

# 

Para el problema planteado por la EPS la mejor solución es una implementación de software práctico para que sus colaboradores sean más eficientes en el momento de la recolección de datos y los procesos sean más ágiles y queden los registros requeridos por la EPS.

Para la etapa inicial del proyecto se debe recomendar que el desarrollo de la aplicación se realice en secciones, en aras de facilitar la escalabilidad del software y la implementación de actualizaciones posteriores.

Es fundamental acordar con el cliente el tipo de aplicación solicitada, los requerimientos funcionales y no funcionales, el tiempo de duración del proyecto, el número de personas encargadas de cada fase, el tipo de verificación de cada avance y la puesta en marcha de la aplicación.

Una de las ventajas obtenidas es que nos permitió emplear github es la de realizar un proyecto colaborativo en el cual todo los integrantes del grupo , lograron realizar los aportes.

Es totalmente independiente de la plataforma, así que lo podemos utilizar tanto en Windows, Linux.

Cada mensaje HTTP contiene los datos necesarios para comprender la petición, como conclusión ni el cliente ni el servidor necesitan recordar ningún estado en las comunicaciones entre los mensajes.

Fue necesario conocer bien el proceso de chequeo para poder establecer los requerimientos funcionales del sistema para así realizar los diagramas correspondientes que nos llevara a la realización del proyecto.

De igual manera lo que se busca con el desarrollo de este proyecto es poder automatizar y agilizar todo el proceso de chequeo que tiene actualmente la empresa, creación de usuarios, captura de información, encuentras y de más procesos manuales que conllevan bastante tiempo al momento de un levantamiento de información.

Al momento de continuar con el presente proyecto se recomienda aprovechar las funcionalidades ya desarrolladas y expandirlas de tal forma que los servicios Web no solo proporcionen métodos de consulta sino también de almacenamiento de los datos desde la presentación, este beneficio abre la puerta a muchas ideas innovadoras que pueden ser aprovechadas por el desarrollador de la presentación ya que puede almacenar cualquier tipo de información suministrada por el visitante del sitio.

Implementar nuevos módulos a las funcionalidades que provee la integración de la aplicación Web, módulos tales como administradores de encuestas, graficadores, foros y demás utilidades características de los sitios Web, orientándose a un entorno gráfico más rico en dinamismo presentación.

# BIBLIOGRAFÍA

* Avgeriou, Paris; Uwe Zdun (2005). «Architectural patterns revisited:a pattern language». *10th European Conference on Pattern Languages of Programs (EuroPlop 2005), July* (Irsee, Germany).
* Bass L., Clements P., Kazman R. (2005). *Software Architecture in Practice: Second Edition*. [Addison-Wesley](https://es.wikipedia.org/wiki/Addison-Wesley).
* Buschmann F., Meunier R., Rohnert H. & Sommerlad P. & Stal M. (1996). [*Pattern-Oriented Software Architecture: A System of Patterns*](http://www.wiley.com/WileyCDA/WileyTitle/productCd-0471958697.html). [John Wiley & Sons](https://es.wikipedia.org/wiki/John_Wiley_%26_Sons).
* Pressman, Roger S. (10 de noviembre de 2011). *Software Engineering, a practitioner's approach.* (7ª edición). McGraw-Hill. [ISBN](https://es.wikipedia.org/wiki/ISBN) [9780071267823](https://es.wikipedia.org/wiki/Especial:FuentesDeLibros/9780071267823).
* Universi Web,año 2016,La arquitectura MVC.españa

**Ilustración cronograma**

[https://uniwebsidad.com/libros/jobeet-1-4/capitulo-4/la-arquitectura- mvc?from=librosweb](https://uniwebsidad.com/libros/jobeet-1-4/capitulo-4/la-arquitectura-%20%20%20%20%20%20%20mvc?from=librosweb).