

**TÍTULO PROYECTO**

Desarrollo web para la administración de las encuestas realizadas en los eventos de la USTA Tunja

**PROPONENTE(S)**

Leidy Melissa Marín Fajardo

1.099.217.312

2251490

Fabián Danilo Rodríguez Ochoa

1.051.476.909

2251163

**DIRECTOR**

Luis Fernando Castellanos Guarín

Tunja

Fecha de presentación (día, mes, año)

**CONTENIDO**

[1. FICHA TÉCNICA DEL PROYECTO 6](#_Toc117009823)

[2. PLANTEAMIENTO DEL PROBLEMA 7](#_Toc117009824)

[3. JUSTIFICACIÓN 8](#_Toc117009825)

[4. OBJETIVOS 9](#_Toc117009826)

[4.1. Objetivo General 9](#_Toc117009827)

[4.2. Objetivos específicos 9](#_Toc117009828)

[5. MARCO TEÓRICO 10](#_Toc117009829)

[5.1. Estado del arte 11](#_Toc117009830)

[5.2. Modelo “4+1” 13](#_Toc117009831)

[5.2.1. Vista lógica 13](#_Toc117009832)

[5.2.2. Vista de procesos 13](#_Toc117009833)

[5.2.3. Vista Física 13](#_Toc117009834)

[5.2.4. Vista de implementación 14](#_Toc117009835)

[5.3. Casos de prueba 14](#_Toc117009836)

[5.3.1. Pruebas de componentes 15](#_Toc117009837)

[5.3.2. Pruebas de integración 15](#_Toc117009838)

[5.3.3. Pruebas del sistema 15](#_Toc117009839)

[5.3.4. Pruebas de aceptación 15](#_Toc117009840)

[5.4. Estándar IEEE 830-1998 15](#_Toc117009841)

[5.4.1. Requerimientos funcionales 16](#_Toc117009842)

[5.4.2. Requerimientos no funcionales 16](#_Toc117009843)

[6. METODOLOGIA 17](#_Toc117009844)

[6.1. Fases de la metodología 17](#_Toc117009845)

[6.1.1. Levantamiento de requerimientos 18](#_Toc117009846)

[6.1.1.1 Elicitación de requerimientos 18](#_Toc117009847)

[6.1.1.2 Especificación 18](#_Toc117009848)

[6.1.1.3 Validación 18](#_Toc117009849)

[6.1.2. Diseño 19](#_Toc117009850)

[6.1.3. Construcción del sistema 19](#_Toc117009851)

[6.1.3.1 Backend 19](#_Toc117009852)

[6.1.3.2 Frontend 19](#_Toc117009853)

[6.1.4. Validaciones 20](#_Toc117009854)

[7. CRONOGRAMA DE ACTIVIDADES 21](#_Toc117009855)

[8. DESARROLLO DEL PROYECTO 22](#_Toc117009856)

[8.1. Fase 1. Levantamiento de requerimientos 22](#_Toc117009857)

[8.1.1. Especificación de requerimientos 22](#_Toc117009858)

[8.1.2. Validación de requerimientos 22](#_Toc117009859)

[8.2. Fase 2. Diseño del sistema 23](#_Toc117009860)

[8.2.1. Diseño de diagramas 23](#_Toc117009861)

[8.2.1.1 Diagramas de casos de uso 23](#_Toc117009862)

[8.2.1.2 Diagrama de clases 24](#_Toc117009863)

[8.2.1.3 Diagrama de actividades 25](#_Toc117009864)

[8.2.2. Diseño de la base de datos 29](#_Toc117009865)

[8.2.3. Diseño de la interfaz 30](#_Toc117009866)

[8.3. Fase 3. Construcción del sistema 38](#_Toc117009867)

[8.3.1. Selección de las tecnologías 38](#_Toc117009868)

[8.3.2. Otras herramientas 39](#_Toc117009869)

[8.3.3. Construcción de la base de datos 41](#_Toc117009870)

[8.3.4. Condiciones necesarias para el funcionamiento de la base de datos 41](#_Toc117009871)

[8.3.5. Construcción del backend 44](#_Toc117009872)

[8.3.5.1 Librerías 44](#_Toc117009873)

[8.3.5.2 Estructura 44](#_Toc117009874)

[8.3.6. Construcción del frontend 46](#_Toc117009875)

[8.4. Fase 4. Pruebas 46](#_Toc117009876)

[8.5. Fase 6. Puesta en producción 47](#_Toc117009877)

[9. CONCLUSIONES Y ANÁLISIS DE RESULTADOS 48](#_Toc117009878)

[10. REFERENCIAS 49](#_Toc117009879)

[11. ANEXOS 51](#_Toc117009880)

[11.1. Anexo 1. Actas de reunión 51](#_Toc117009881)

[11.2. Anexo 2. Documento de especificación de requerimientos 51](#_Toc117009882)

[11.3. Anexo 3. Mockups del diseño de la interfaz 51](#_Toc117009883)

[11.4. Anexo 4. Manual de usuario 51](#_Toc117009884)

**INDICE DE TABLAS**

[*Tabla 1. Objetivos específicos* 9](#_Toc116421773)

[*Tabla 2 Cronograma de actividades* 21](#_Toc116421774)

[*Tabla 3. Fase 1. Levantamiento de requerimientos* 22](#_Toc116421775)

[*Tabla 4. Fase 2 Diseño del sistema* 37](#_Toc116421776)

[*Tabla 5. Fase 3 Selección de las tecnologías* 39](#_Toc116421777)

**INDICE DE ILUSTRACIONES**

[*Ilustración 1 Modelo "4+1"* 13](#_Toc116421851)

[*Ilustración 2 Proceso General Del Testing* 14](#_Toc116421852)

[*Ilustración 3 Modelo de desarrollo por prototipos evolutivos* 17](#_Toc116421853)

[*Ilustración 4 Caso de uso: Administrador* 23](file:///D:\UNIVERSIDAD\proyecto_sistema_encuestas_v2\Requerimientos,%20manual%20e%20informe%20final\Plantilla%20Informe%20Final%20Sistema%20Encuestas.docx#_Toc116421854)

[*Ilustración 5 Caso de uso: Docente, Invitado y Secretaria* 24](#_Toc116421855)

[*Ilustración 6 Caso de uso: Estudiante* 24](#_Toc116421856)

[*Ilustración 7 Diagrama de clases* 25](#_Toc116421857)

[*Ilustración 8 Diagrama de actividades: Registrar, iniciar sesión y crear encuestas* 26](#_Toc116421858)

[*Ilustración 9 Diagrama de actividades: Listar encuestas, ver resultados y responder encuestas* 26](#_Toc116421859)

[*Ilustración 10 Diagrama de actividades: Filtrar búsquedas, editar perfil y agregar usuarios* 27](#_Toc116421860)

[*Ilustración 11 Diagrama de actividades: Listar, modificar usuarios y listar roles* 28](#_Toc116421861)

[*Ilustración 12 Diagrama de actividades: Editar roles* 29](#_Toc116421862)

[*Ilustración 13 Modelo entidad relación de la base de datos* 30](#_Toc116421863)

[*Ilustración 14 Mockup Inicio de sesión* 31](#_Toc116421864)

[*Ilustración 15 Mockup Registrar cuenta* 32](#_Toc116421865)

[*Ilustración 16 Mockup Vista de inicio* 32](#_Toc116421866)

[*Ilustración 17 Mockup Agregar usuario* 33](#_Toc116421867)

[*Ilustración 18 Mockup Listado de usuarios* 33](#_Toc116421868)

[*Ilustración 19 Mockup Modificar usuario* 34](#_Toc116421869)

[*Ilustración 20 Mockup Listar roles* 34](#_Toc116421870)

[*Ilustración 21 Mockup Modificar estado del rol* 35](#_Toc116421871)

[*Ilustración 22 Mockup Crear encuesta* 36](#_Toc116421872)

[*Ilustración 23 Mockup Listado de encuestas* 36](#_Toc116421873)

[*Ilustración 24 Mockup Formulario para responder encuestas* 37](#_Toc116421874)

[*Ilustración 25 Historial de cambios de la rama Origin en GitKraken* 40](#_Toc116421875)

[*Ilustración 26 Repositorio GitHub del proyecto* 40](#_Toc116421876)

[*Ilustración 27 Base de datos PgAdmin* 41](#_Toc116421877)

[*Ilustración 28 Sentencia DML Creación de roles* 42](#_Toc116421878)

[*Ilustración 29 Sentencia DML Inserción de imágenes* 42](#_Toc116421879)

[Ilustración 30 Sentencia DML Creación de usuarios 42](#_Toc116421880)

[Ilustración 31 Sentencia DML Creación de acceso 42](#_Toc116421881)

# FICHA TÉCNICA DEL PROYECTO

|  |  |
| --- | --- |
| Título | Desarrollo web para la administración de las encuestas realizadas en los eventos de la USTA Tunja |
| Autor (es) | Leidy Melissa Marín Fajardo y Fabián Danilo Rodríguez Ochoa |
| Director del proyecto | Luis Fernando Castellanos Guarín |
| Palabras claves | Encuestas, análisis, estadísticas. |
| Descripción | |
| Este sistema de información orientado a la web, buscó satisfacer las necesidades de la Universidad Santo Tomás Seccional Tunja, en torno a la complejidad que se presentaba al evaluar la información obtenida de las encuestas en los eventos realizados por parte de la Universidad. Dicho desarrollo permitió que la información recopilada acerca de estos eventos quedara almacenada y se pudiera manejar de manera sistematizada, de esta forma se logró un mejor control de los resultados que dejaban las encuestas de estos eventos, y así, se pudieron realizar análisis estadísticos, y se generaron sus respectivos reportes. | |

# PLANTEAMIENTO DEL PROBLEMA

Actualmente los sistemas de información son de gran utilidad en pequeñas o grandes empresas, instituciones, u organizaciones, ya que dichos sistemas ayudan a administrar, recolectar, analizar, almacenar y distribuir información relevante para los procesos fundamentales y las particularidades de cada entidad.

Al día de hoy, la Universidad Santo Tomás Seccional Tunja, presenta una complejidad al momento de evaluar la información obtenida de las encuestas que se realizan en los distintos eventos, ya que la información recopilada de estos eventos es manipulada o tratada de forma manual a través de archivos en formato Excel, lo cual genera una gran cantidad de documentación que ocasiona que este proceso sea más confuso.

Es por este motivo, que en muchas ocasiones los reportes de cada evento no están unificados en un solo sistema de información o base de datos, y al momento de comparar y consolidar la información, puede haber varias versiones de estos archivos, puesto que no se cuenta con permisos y roles definidos para que solo el personal autorizado pueda tener acceso a esta información.

Además, esto dificulta realizar búsquedas más específicas, debido a la escasa disponibilidad de información que pueden tener los usuarios en tiempo real, haciendo que se dedique demasiado tiempo del personal de la Universidad para preparar y analizar datos, lo que ocasiona un gasto elevado en tiempo y recursos económicos por parte de la Universidad.

# JUSTIFICACIÓN

La implementación de sistemas de información, brinda la posibilidad de obtener grandes ventajas, puesto que permite que los usuarios puedan acceder a la información de manera rápida y oportuna, ayuda a establecer estrategias y planificaciones operativas, apoya la monitorización de las operaciones, y también favorece la reducción de costos.

Para ello, es necesario un sistema de información orientado a la web que ofrezca múltiples posibilidades, permitiendo acceder a los datos relevantes de manera frecuente y oportuna, donde la información se pueda almacenar y administrar de manera sistemática. Así los usuarios autorizados para la manipulación de la información podrán planificar operaciones y establecer objetivos.

Es por este motivo que el sistema de información orientado a la web, que se propone en este desarrollo, busca satisfacer las necesidades de la Universidad Santo Tomás Seccional Tunja, en torno a la complejidad que se presenta al evaluar la información obtenida de las encuestas en los eventos realizados por parte de la Universidad.

Dicho desarrollo permitiría que la información recopilada acerca de estos eventos quede almacenada y se pueda manejar de manera sistematizada, logrando así, que haya un mejor control de los resultados que dejan las encuestas de estos eventos, y de esta manera, se puedan realizar análisis estadísticos, generando sus respectivos reportes.

Además, con esta solución propuesta, se evitaría la excesiva documentación que actualmente se requiere para analizar y obtener los resultados de cada evento. Por otra parte, se definirían roles para la manipulación de la información, con el fin de que solo el personal autorizado pueda tener acceso y así evitar confusiones al momento de consolidar la información, teniendo en cuenta que además de todo esto, la información podrá ser consultada de manera oportuna y en tiempo real, para que los usuarios autorizados puedan hacer las búsquedas o consultas que necesiten. Y lo más importante, es que la información estará almacenada de manera segura, evitando que esté expuesta a cualquier manipulación por parte de personal no autorizado.

# OBJETIVOS

## Objetivo General

Construir una herramienta informática para la generación de instrumentos de evaluación de eventos realizados en la Universidad Santo Tomás Seccional Tunja, utilizando la arquitectura orientada a servicios para su implementación.

## Objetivos específicos

*Tabla 1. Objetivos específicos*

|  |  |
| --- | --- |
| Nro. | Objetivo específico |
| 1 | Construir la especificación de requerimientos funcionales y no funcionales de la herramienta informática, utilizando el estándar IEEE 830-1998. |
| 2 | Diseñar la herramienta informática para establecer las vistas de usuario que permitan analizar el software a desarrollar, utilizando el modelo de referencia “4+1” propuesto por Kruchten. |
| 3 | Implementar la herramienta informática de generación de instrumentos de evaluación, definiendo dos componentes: backend y frontend, utilizando la arquitectura orientada a servicios |
| 4 | Validar la herramienta informática diseñando casos de prueba, que permitan identificar las funcionalidades de acuerdo con los requerimientos del desarrollo. |

*Fuente: Autor*

# MARCO TEÓRICO

La evaluación de eventos es un proceso de observación detallada, medición absoluta y seguimiento puntualizado del desarrollo de un evento con el fin de valorar sus resultados con exactitud. (CARCAMO et al., 6)

Se trata de un proceso continuo que ocurre a través de todas las etapas del ciclo de vida de un evento. Permite la creación de estrategias que se utilizan para identificar y puntualizar las características básicas, y estadísticas clave de determinado evento. También es una herramienta o ayuda de retroalimentación que se debe proporcionar a los distintos grupos involucrados en el evento, jugando así un papel importante en el proceso de gestión, gracias a su característica de ser una poderosa instrumento de información que facilita el proceso de análisis. (CARCAMO et al., 6)

Las tres etapas básicas del proceso de evaluación de eventos son:

* Estudio de viabilidad o fase de evaluación previa al evento, que debe realizarse antes del evento para determinar, con un mínimo margen de error, si su inicio es viable o no. (CARCAMO et al., 6)
* Proceso de seguimiento y control, que ocurre durante la ejecución del evento para verificar que todo esté funcionando de acuerdo con el cronograma programado y, cuando sea necesario, se puedan tomar las acciones correctivas para modificar las fallas y omisiones del evento que se detecten. (CARCAMO et al., 6)
* La evaluación posterior al evento se enfoca en medir los resultados del evento en términos de impacto, imagen y satisfacción del cliente o usuario. De esta fase se toma información para indicar acciones de mejora continua para versiones posteriores del evento o para nuevos eventos futuros con características similares. (CARCAMO et al., 6)

Una plataforma puede facilitar el proceso de recolección de comentarios y opiniones de los participantes de un evento. Las encuestas de retroalimentación, de experiencia general y los diferentes tipos de preguntas alentarán a los participantes a compartir comentarios de una manera conveniente a través de sus dispositivos. (Wong, 2018)

Estas plataformas cumplen con factores claves para garantizar su buen funcionamiento, algunas de esos factores son:

1. Informes y análisis centralizados: Buscar informes, incluidos los datos de todas las opiniones de los participantes y hacer un análisis valioso determinará la calidad de la tecnología que atraerá a los participantes, además de aprender qué informes están disponibles y cómo pueden ser exportados. (Wong, 2018)
2. Integraciones: Si su equipo quiere crear una integración entre sus sistemas y plataformas de gestión de eventos, pídale al proveedor información acerca de los costos para acceder a una API. (Wong, 2018)
3. Opciones de seguridad y privacidad: Para mantener la seguridad de datos de los eventos, haga preguntas sobre la capacidad de la plataforma, su control de acceso, lista de registros, etc. Esto también puede ser importante para las políticas de privacidad y los Términos de uso, con el objetivo de que los participantes puedan entender cómo se utilizan sus datos antes de completar el registro o participar en una aplicación. (Wong, 2018)
4. Alojamiento de documentos: Pregunte acerca de la capacidad de organizar y compartir presentaciones, documentos, imágenes, mapas y más para mantener a los asistentes informados, educados y comprometidos durante su evento. Conozca los distintos tipos de archivos que se pueden descargar. (Wong, 2018)

## 5.1. Estado del arte

El software de encuestas en línea y las herramientas de cuestionarios aparecieron por primera vez a fines de la década de los 90. Estas son herramientas de encuestas bastante tradicionales que consisten en una larga lista de preguntas y respuestas anónimas. Sin embargo, desde el auge de las opiniones de los sitios web, las herramientas de encuestas en línea han evolucionado. Su continua popularidad también se debe al hecho de que las encuestas en línea son una manera fácil de contactar a una audiencia definida y obtener sus respuestas.

A continuación se presentan algunos softwares, los cuales fueron seleccionados porque ofrecen soluciones similares respecto a la administración de encuestas realizadas en eventos.

* Mopinion: Es una herramienta de análisis para la opinión de los clientes que hace uso de encuestas en línea, y ofrece oportunidades de análisis en profundidad para sus usuarios. Con la visualización de datos en tiempo real, en paneles de control y gráficos personalizados, donde los usuarios pueden manejar grandes cantidades de datos de forma rápida y eficiente. (Gilliam Haije, 2017)

Sitio web: https://mopinion.com/es/

* Typeform: Es un software de encuestas en línea con una interfaz muy fácil de usar que permite a los usuarios construir sus propias encuestas. Como encuestado, el proceso para responder a una de estas encuestas es bastante fluido. Por otra parte, también incluye funciones como preguntas y respuestas ilimitadas, opciones para exportar datos, temas diseñados a petición, así como plantillas listas para utilizar y funciones básicas de generación de informes. (Gilliam Haije, 2017)

Sitio web: www.typeform.com

* Survicate: Permite a los usuarios activar encuestas específicas en función de las visitas a diferentes partes de su sitio web. Los usuarios cuentan con la opción de enviar cuestionarios por correo electrónico. Cuando se trata de crear encuestas, Survicate ofrece una biblioteca de encuestas prefabricadas para que elijas. Esta herramienta de encuesta también incluye funcionalidad de panel, análisis NPS y opciones de exportación CSV y XLS. (Gilliam Haije, 2017)

Sitio web: www.survicate.com

* PollDaddy: Se trata de un software de encuestas en línea que permite a los usuarios investigar dos maneras diferentes: investigan en su página web o invitar a la gente a responder por correo electrónico. Cuando los datos se han recopilado, se pueden establecer filtros para analizar y utilizarlos en los informes. Los resultados de estas encuestas se pueden ver en tiempo real, además de poderse exportar a archivos PDF, Excel, CSV, Google Docs y XML. (Gilliam Haije, 2017)

Sitio web: www.polldaddy.com

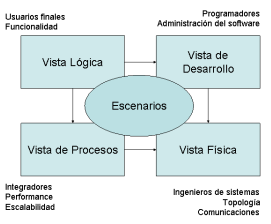
* Qualtricsics: Es un software de encuestas en línea y una herramienta de cuestionario para organizar de manera mejorada las encuestas tradicionales. Proporciona más de 100 tipos de preguntas, algunos modelos de encuestas basados en trabajo, así como funciones lógicas, activistas de correo electrónico y bifurcación lógica. Los usuarios pueden seleccionar entre más de 30 tipos de gráficos diferentes para añadirlos a informes web en tiempo real y exportar resultados en CSV o SPSS. (Gilliam Haije, 2017)

Sitio web: www.qualtrics.com

## 5.2. Modelo “4+1”

Uno de los modelos más reconocidos es el “4+1” de Philippe Kruchten. El cual identifica cuatro vistas diferentes: lógica, proceso, física e implementación.

*Ilustración 1 Modelo "4+1"*



*Fuente: Kruchten, P. (1995).*

### 5.2.1. Vista lógica

El propósito de esta vista es la descomposición orientada a objetos. En esta vista, se representa la implementación de casos de uso, subsistemas, paquetes y clases de los casos de uso más importantes en términos de arquitectura. (Moreno Sabido y Ucán Pech, s. f.)

### 5.2.2. Vista de procesos

El propósito de esta vista es la concurrencia y la coordinación de procesos, la integridad del sistema, la tolerancia a fallas y cómo las abstracciones clave de la vista lógica encajan en la arquitectura. (Moreno Sabido y Ucán Pech, s. f.)

### 5.2.3. Vista Física

Muestra la ubicación del software en el hardware. Los elementos definidos en la vista lógica se copian a los componentes del software (servicios, procesos, etc.) o componentes de hardware, los cuales definen con mayor precisión cómo se debe implementar la solución. (Moreno Sabido y Ucán Pech, s. f.)

### 5.2.4. Vista de implementación

Por último, la vista de implementación, la cual se descompone en subsistemas, describe la estructura del entorno de desarrollo; el software es asociado en pequeños subsistemas que luego pueden ser desarrollados por uno o varios desarrolladores; Estos subsistemas tienen la forma de programas o bibliotecas organizados en capas y jerarquías. (Moreno Sabido y Ucán Pech, s. f.)

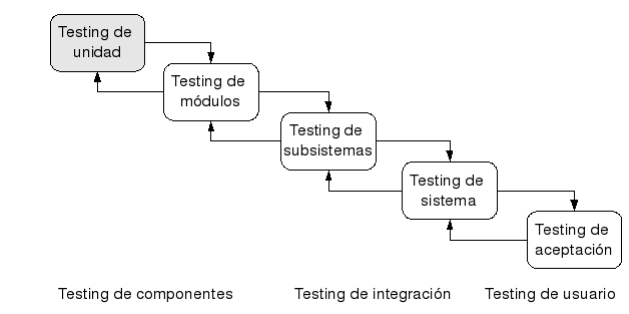
## 5.3. Casos de prueba

Para desarrollar un software de calidad y sin errores, la organización y los casos de prueba son muy importantes.

Según la norma ISO / IEC / IEEE 24765: 2010, al momento de diseñar o establecer los casos de prueba, se debe tener en cuenta lo siguiente:

* Verificación: Proceso de evaluación de un sistema o componente para determinar si un producto en una determinada etapa de desarrollo cumple con las condiciones establecidas al comienzo de esa fase. (Mera Paz, 2016)
* Validación: Proceso de evaluación de un sistema o componente, durante o al final del desarrollo, para establecer cuándo se están cumpliendo los requisitos específicos. (Mera Paz, 2016)

*Ilustración 2 Proceso General Del Testing*



*Fuente: Sommerville, I. (1995)*

### 5.3.1. Pruebas de componentes

Su propósito es identificar fallas y verificar el comportamiento de módulos de software, programas, objetos, clases, etc. Que se pueden probar por separado; es decir, que se pueden ejecutar independientemente del resto del sistema, dependiendo del contexto. (Mera Paz, 2016)

### 5.3.2. Pruebas de integración

Son responsables de probar interfaces entre componentes o módulos; por ejemplo, autenticación de usuario del componente con el sistema operativo, el sistema de archivos integrado con el hardware, etc. (Mera Paz, 2016)

### 5.3.3. Pruebas del sistema

Se refieren a todo el sistema; se debe establecer un plan de prueba de manera clara y bien estructurada. (Mera Paz, 2016)

### 5.3.4. Pruebas de aceptación

Se enfocan en aceptar los criterios proporcionados en el contrato de desarrollo de software, que se acuerda entre el fabricante del software y el cliente. (Mera Paz, 2016)

## 5.4. Estándar IEEE 830-1998

El Comité de Estándares de Ingeniería de Software y Sistemas del IEEE (S2ESC) está constituido por el Consejo de Actividades de Estándares de la Sociedad Informática del IEEE para codificar las normas de las prácticas profesionales de ingeniería de software en estándares, incluyendo la estandarización de procesos, productos, recursos, notaciones, métodos, nomenclaturas, técnicas y soluciones para la ingeniería de software y sistemas dependientes de software. (IEEE Computer Society 2008)

El S2ESC hace referencia al uso de normas de ingeniería del software profesionales educadores y clientes. El S2ESC armoniza el desarrollo de normas nacionales e internacionales de ingeniería del software y promueve la disciplina y la profesionalización de la ingeniería del software. El S2ESC también promueve la coordinación con otras iniciativas del IEEE. (IEEE Computer Society 2008-1)

Sustituido por ISO/IEC/IEEE 29148:2011. Explica el contenido y las cualidades que se debe tener para una buena especificación de requisitos de software (SRS) y se presentan varios ejemplos de esquemas de SRS. Esta práctica recomendada está destinada a especificar los requisitos del software que se va a desarrollar, pero también puede aplicarse para ayudar en la selección de productos de software internos y comerciales. También se proporcionan directrices para el cumplimiento de la norma IEEE/ EIA12207.1- 1997. (IEEE Computer Society 2008-2)

### 5.4.1. Requerimientos funcionales

Los requerimientos funcionales son aquellos que definen las funciones que el sistema deberá realizar, mencionan las transformaciones que el sistema efectúa sobre las entradas para producir salidas. Es significativo que se describa el ¿Qué? y no el ¿Cómo? se debe hacer dichas transformaciones. Estos requerimientos a medida que avanza el proyecto de software, se convierten en algoritmos, la lógica y gran parte del código del sistema. (Arias Chaves, 2005)

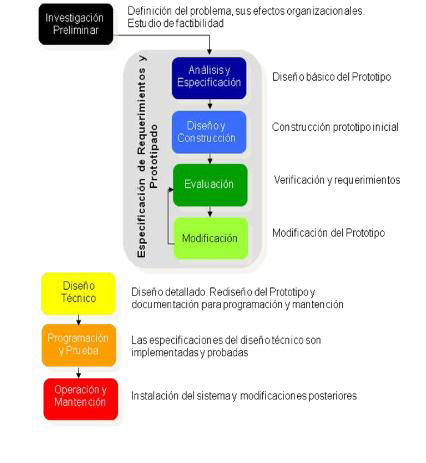
### 5.4.2. Requerimientos no funcionales

Los requerimientos no funcionales hacen referencia a las características que de una u otra manera puedan limitar el sistema, como por ejemplo, el rendimiento, interfaces de usuario, robustez del sistema, disponibilidad de equipo, mantenimiento, seguridad, esquemas, modelos, entre otros. (Arias Chaves, 2005)

# METODOLOGIA

Para el desarrollo del proyecto se propone la metodología de desarrollo Prototipado. La cual se basa en la creación de una implementación parcial del sistema para estudiar los requerimientos del mismo. El prototipo se fabricará lo antes posible. Esto se entrega a los usuarios, clientes o sus representantes para que ellos puedan explorarlo y manipularlo. Luego, estas personas brindan comentarios sobre lo que les gusta y lo que no les gusta del prototipo propuesto, y en seguida, se toman en cuenta los aportes que los usuarios han proporcionado para desarrollar el sistema real. (Repositorio Digital de la Facultad de Ingenieria UNAM, s. f.)

*Ilustración 3 Modelo de desarrollo por prototipos evolutivos*



*Fuente: http://gestionrrhhusm.blogspot.com/2011/05/modelo-de-prototipo.html*

## Fases de la metodología

Este desarrollo de software se realizara mediante las siguientes fases:

### Levantamiento de requerimientos

Esta etapa se desarrollará a través de entrevistas con la facultad de Ingeniería de Sistemas de la Universidad Santo Tomás Seccional Tunja. Dichas entrevistas se van a realizar por medio de tareas previas, como: estudiar el dominio del problema, seleccionar a las personas que se van a entrevistar, determinar el objetivo y el contenido de las reuniones, planificar la entrevista, llevarla a cabo y finalmente, realizar un análisis de la información recolectada en la entrevista.

Para el proceso del planteamiento de los requerimientos del software a desarrollar, y con el fin de entregar lo que la Ingeniería de Sistemas de la Universidad Santo Tomás Seccional Tunja desea, se establecen las siguientes etapas [Loucopoulos y Karakostas]: (INGENIERÍA DE REQUERIMIENTOS., s. f.)

#### Elicitación de requerimientos

El objetivo de la elicitación de requisitos es adquirir información relevante sobre el problema que se desea resolver, dicha información se utilizará para crear una especificación formal del software necesario para resolver el problema. Es por esto es que es muy importante que haya una buena comunicación entre el cliente y los desarrolladores, para poder entender las necesidades del interesado. Para llevar a cabo la elicitación de requerimientos se plantea realizar unas entrevistas con el usuario. (INGENIERÍA DE REQUERIMIENTOS., s. f.)

#### Especificación

La especificación de requerimientos puede definirse como un contrato entre el usuario y los desarrolladores del software, donde se describe el comportamiento funcional que se espera del software a desarrollar. Dicha especificación de requerimientos se desarrollaran por medio de casos de uso, puesto que, estos casos de uso son una técnica que permite especificar el comportamiento del sistema, y la secuencia de interacciones entre el sistema y los actores. (INGENIERÍA DE REQUERIMIENTOS., s. f.)

#### Validación

Por medio de la validación de requerimientos, se busca confirmar que el modelo de solicitud coincide con la intención del cliente y del usuario; este es un punto de vista más general, ya que se trata de asegurar que las opiniones y los conceptos que se presentaron en la descripción queden claros y explicados. (INGENIERÍA DE REQUERIMIENTOS., s. f.)

### Diseño

Para el desarrollo de esta fase se definen las siguientes actividades:

* Diseño de diagramas UML (Diagrama de clases y diagrama de actividades)
* Diseño de la base de datos
* Diseño de la interfaz

### Construcción del sistema

Para la construcción del sistema informático, se tienen en cuenta dos componentes: backend y frontend, utilizando la arquitectura orientada a servicios para su implementación.

#### Backend

Se plantea desarrollar el backend con una arquitectura orientada a servicios, utilizando Node como entorno de ejecución y typescript como lenguaje de programación. Es un marco para desarrollar programas concurrentes de alto rendimiento que no se basa en el enfoque convencional de subprocesos múltiples, sino que utiliza E / S asincrónicas con un modelo de programación controlado por eventos. (Tilkov y Vinoski, 2010)

#### Frontend

El desarrollo del frontend se realizara en Angular bajo el leguaje de typescript. Esta herramienta es un marco de diseño de aplicaciones y una plataforma de desarrollo para crear aplicaciones de una sola página eficientes y complejas. (Angular, 2021)

Angular ofrece:

* Marco basado en componentes para crear aplicaciones web escalables. (Angular, 2021)
* Una colección bien integrada de bibliotecas que cubren una amplia gama de funciones que incluyen enrutamiento, administración de formularios, comunicación cliente-servidor, etc. (Angular, 2021)
* Un conjunto de herramientas de desarrollo para ayudarlo a desarrollar, compilar, probar y actualizar su código. (Angular, 2021)

### Validaciones

Para validar la herramienta informática se diseñaran casos de prueba, que permitan identificar las funcionalidades de acuerdo con los requerimientos del desarrollo. Estas validaciones incluyen:

* Prueba de Componentes: Para identificar fallas y verificar el comportamiento de los módulos del sistema
* Prueba de Integración: Para probar interfaces entre componentes o módulos del

Sistema.

* Prueba de Sistema: Para analizar defectos globales o para estudiar aspectos específicos de su comportamiento.
* Prueba de Aceptación: Para aceptar los criterios proporcionados en el contrato de desarrollo de software, que se acuerda entre el o los desarrolladores del software y el cliente-usuario.

# CRONOGRAMA DE ACTIVIDADES

*Tabla 2 Cronograma de actividades*

| Meses | Mes 1 | | | | Mes 2 | | | | Mes 3 | | | | Mes 4 | | | | Mes 5 | | | | Mes 6 | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Semanas | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| ***1. Levantamiento de requerimientos*** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Elicitación de requerimientos | x |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Especificación de requerimientos |  | x | x |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Validación de requerimientos |  | x | x |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| ***2. Diseño*** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Diseño de diagramas UML |  |  | x | x | x |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Diseño de base de datos |  |  |  | x | x | x |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Diseño de interfaz |  |  |  |  |  |  | x | x | x | x |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| ***3. Construcción*** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Construcción del Backend |  |  |  |  |  |  |  |  |  | x | x | x | x | x | x | x |  |  |  |  |  |  |  |  |
| Construcción del Frontend |  |  |  |  |  |  |  |  |  |  |  |  |  |  | x | x | x | x | x | x |  |  |  |  |
| ***4. Validación*** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Casos de prueba |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | x | x | x | x |

*Fuente: Autor*

# DESARROLLO DEL PROYECTO

En esta sección se detallan cada una de las etapas realizadas para completar el proyecto.

## Fase 1. Levantamiento de requerimientos

Esta fase se desarrollara a través de entrevistas con el Stakeholder encargado de la facultad de ingeniería de sistemas de la seccional Tunja. Esto con el fin de recolectar información para determinar los objetivos del proyecto a desarrollar. Esta etapa está compuesta de 2 actividades:

### Especificación de requerimientos

El primer acercamiento se realizó por medio de entrevistas con la decana de la facultad de ingeniería de sistemas, esto con el fin de adquirir información relevante sobre el problema que se desea resolver, esta información se utilizó para crear una especificación formal del software necesario para dar solución al problema. [(Ver anexo: Acta 1)](https://docs.google.com/document/d/1h3G--9bJXTawr-yrqzNbeafuZDVVZSoJ/edit?usp=sharing&ouid=110006067162752201967&rtpof=true&sd=true)

Una vez reunida y consolidada la información obtenida de estas entrevistas, se procedió a crear el documento de especificación de requerimientos con el formato IEEE 830, para especificar el comportamiento del sistema, y la secuencia de interacciones entre el sistema y los actores. [(Ver anexo)](https://docs.google.com/document/d/1hsMAM68KwpP4aY6xSe6scM9H17LSwm67/edit?usp=sharing&ouid=110006067162752201967&rtpof=true&sd=true)

### Validación de requerimientos

Por medio de la validación de requerimientos, se buscó confirmar que el modelo de solicitud coincide con la intención del cliente y del usuario; este es un punto de vista más general, ya que se trata de asegurar que las opiniones y los conceptos que se presentaron en la descripción queden claros y explicados. [(Ver anexo: Acta 12)](https://docs.google.com/document/d/1lIz5-yWosY3FMKUIxYBs4yNPDlIxgw2I/edit?usp=sharing&ouid=110006067162752201967&rtpof=true&sd=true)

*Tabla 3. Fase 1. Levantamiento de requerimientos*

|  |  |
| --- | --- |
| Tipo | Elementos |
| Entradas | * Actas de reunión |
| Instrumentos y/o herramientas | * Conexión a internet * Computador * Programas del computador * Navegador Web * Word |
| Salidas | Documento de especificación de requerimientos |
| Tiempo | 1 semana |

*Fuente: Autor*

## Fase 2. Diseño del sistema

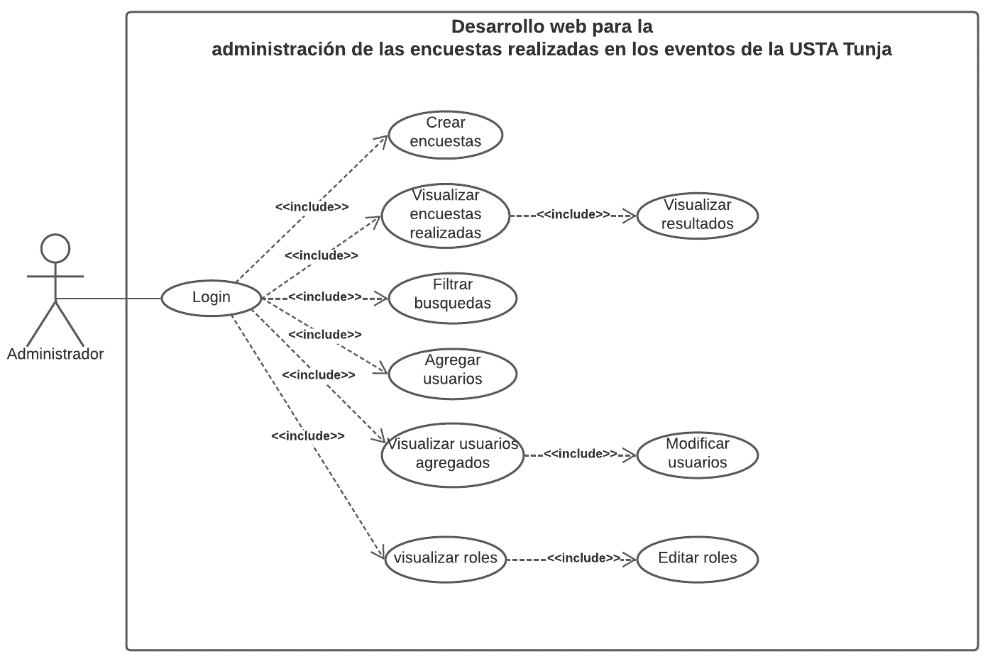
El desarrollo de la fase de diseño estuvo compuesta por tres actividades: el diseño de diagramas, el diseño de la base de datos y el diseño de la interfaz

### Diseño de diagramas

Para esta actividad se diseñaron 3 diagramas: Casos de uso, de clases y de actividades. Puesto que estos diagramas implementan una técnica que permite especificar el comportamiento del sistema, y la secuencia de interacciones entre el sistema y los actores.

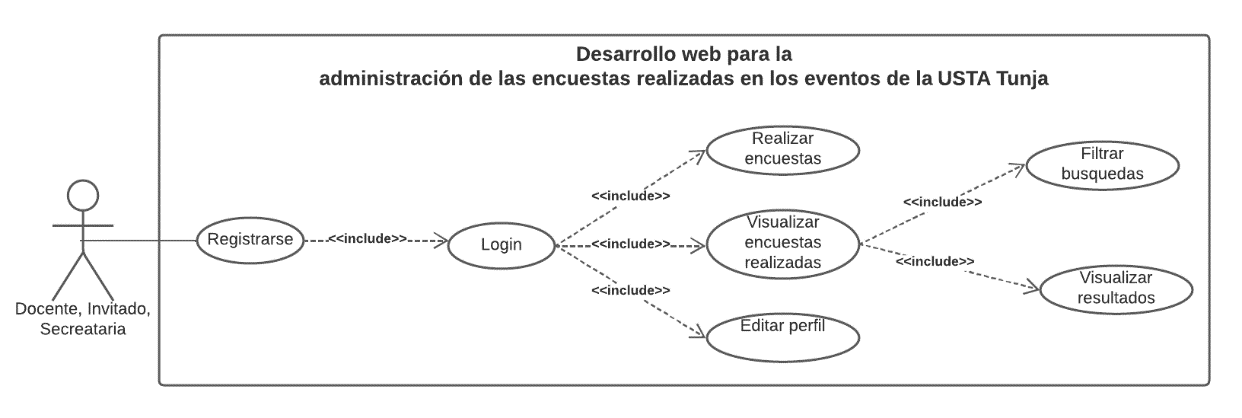
#### Diagramas de casos de uso

*Ilustración 4 Caso de uso: Administrador*



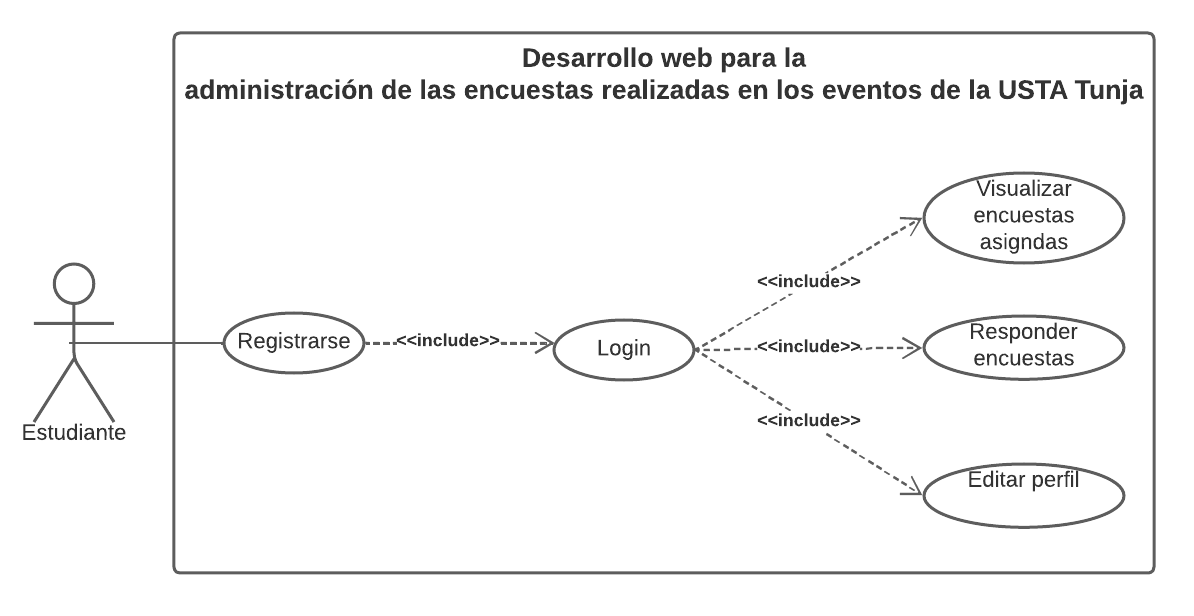
*Fuente: Autor*

*Ilustración 5 Caso de uso: Docente, Invitado y Secretaria*



*Fuente: Autor*

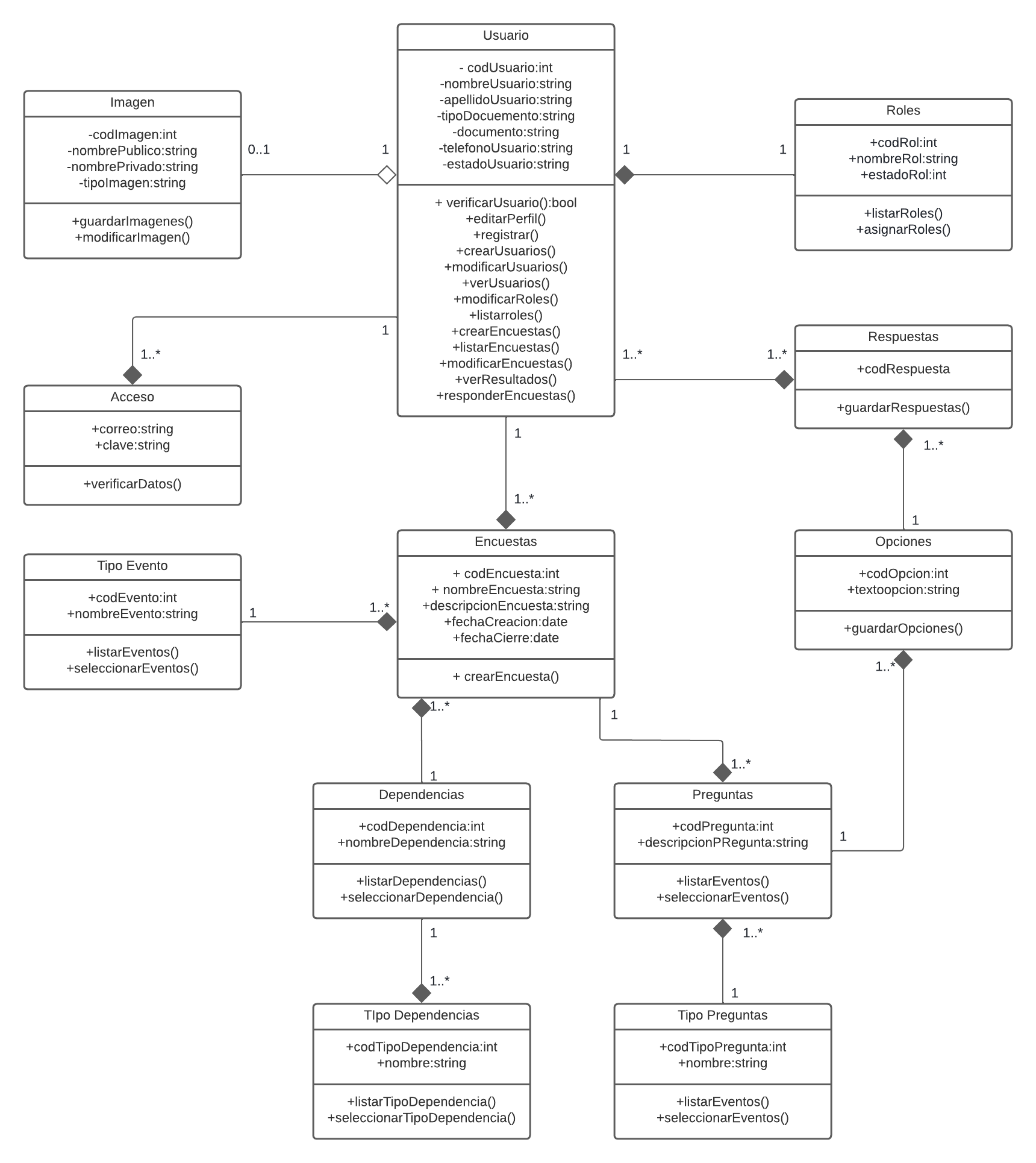
*Ilustración 6 Caso de uso: Estudiante*



*Fuente: Autor*

#### Diagrama de clases

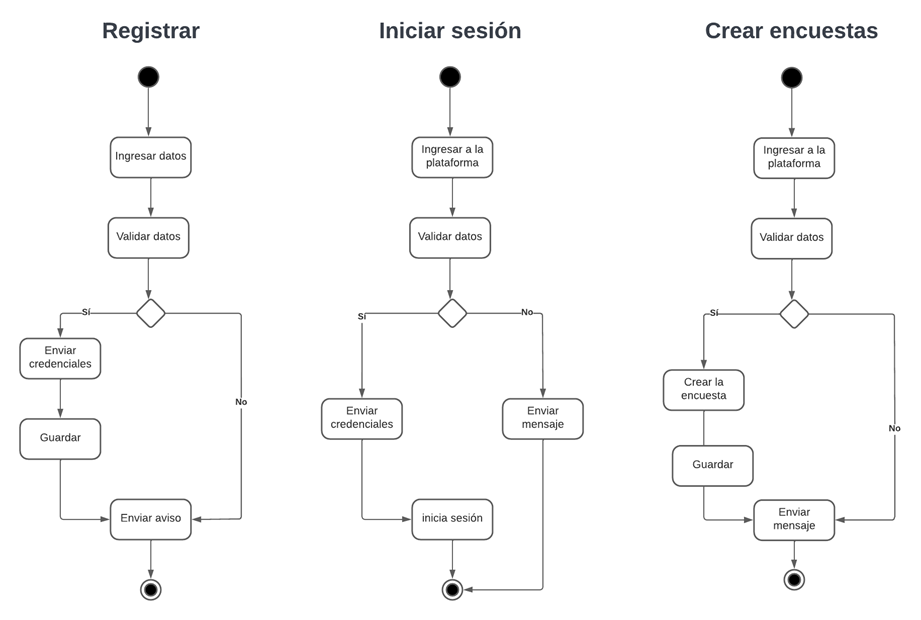
*Ilustración 7 Diagrama de clases*



*Fuente: Autor*

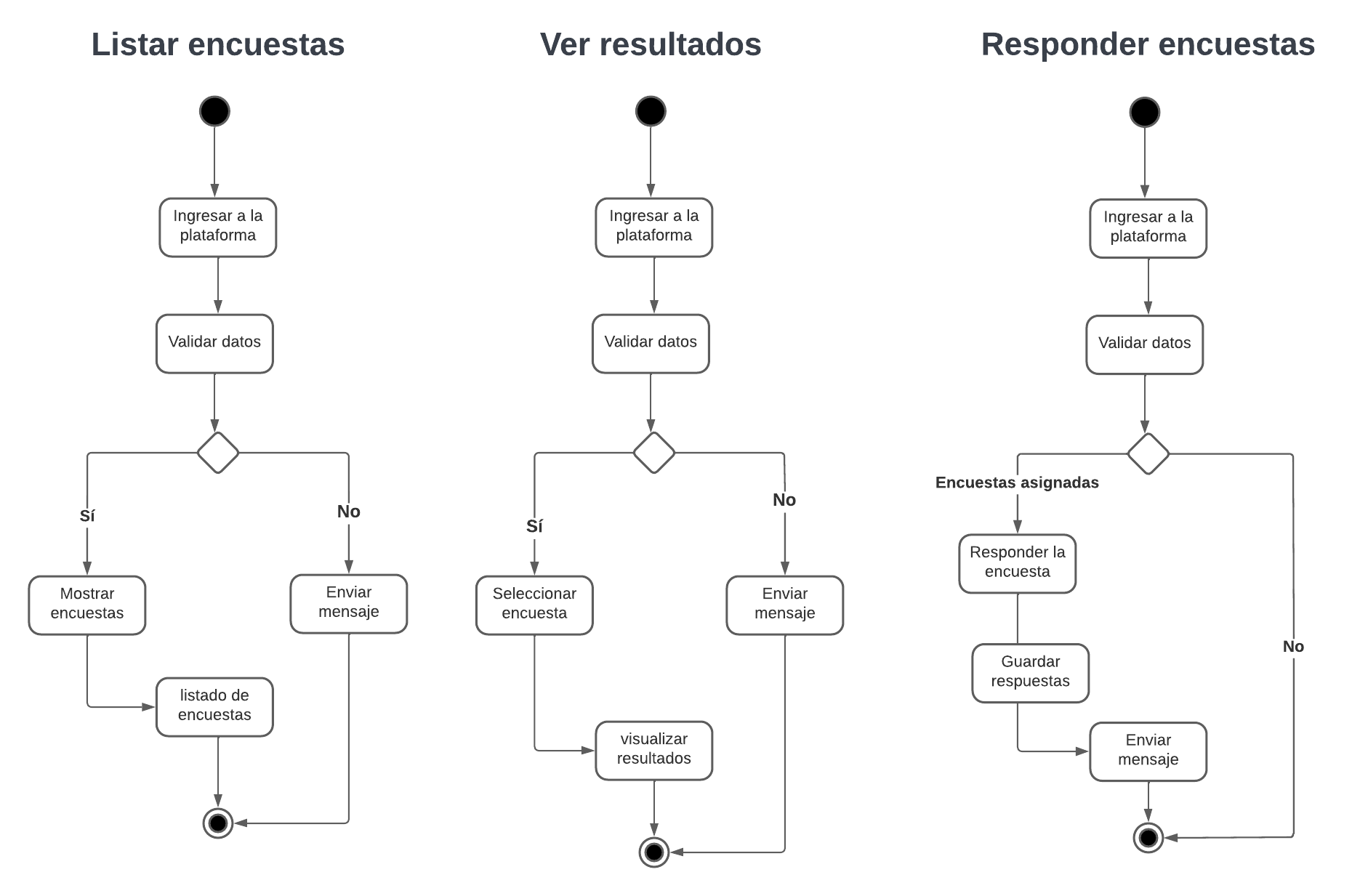
#### Diagrama de actividades

*Ilustración 8 Diagrama de actividades: Registrar, iniciar sesión y crear encuestas*



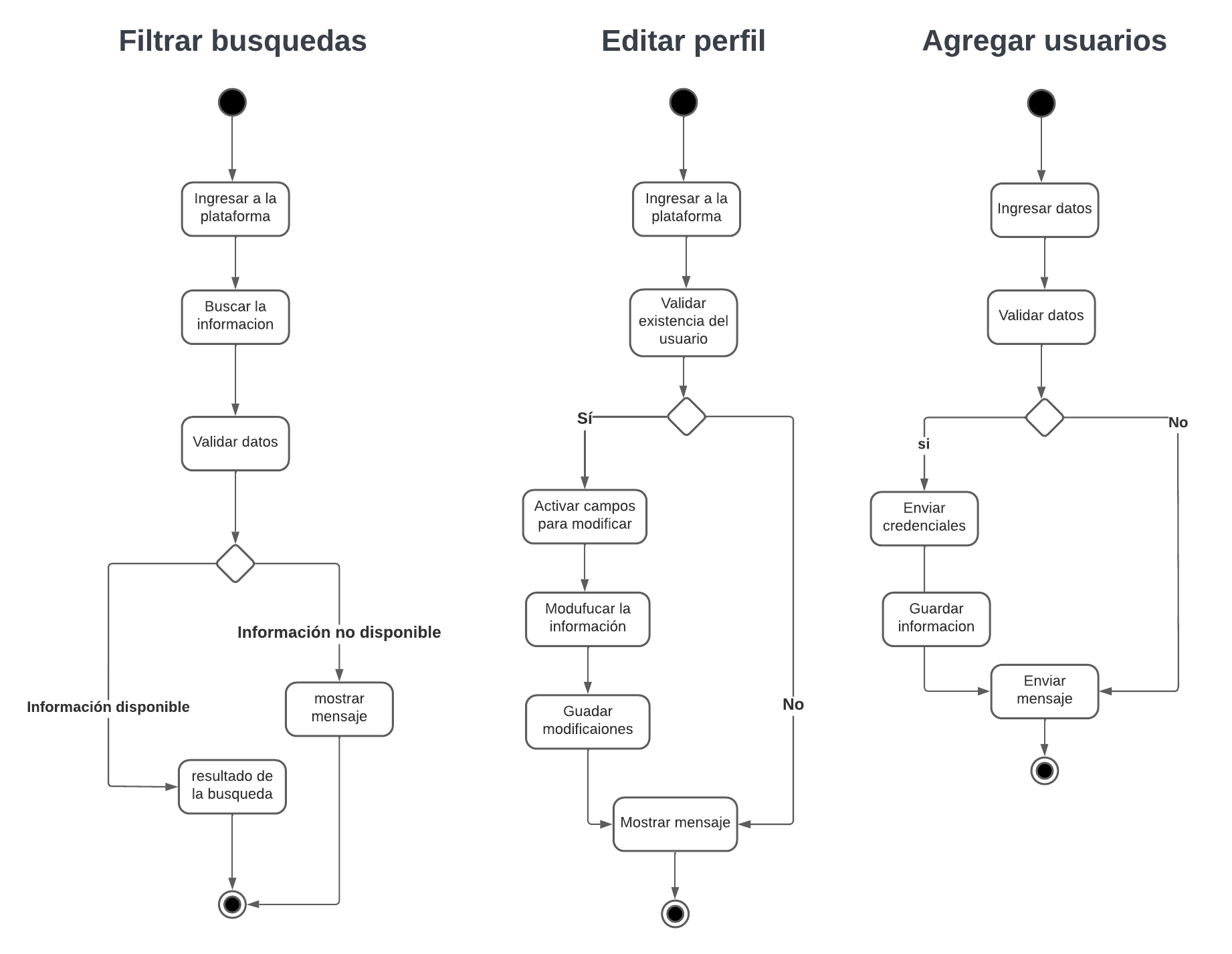
*Fuente: Autor*

*Ilustración 9 Diagrama de actividades: Listar encuestas, ver resultados y responder encuestas*

****

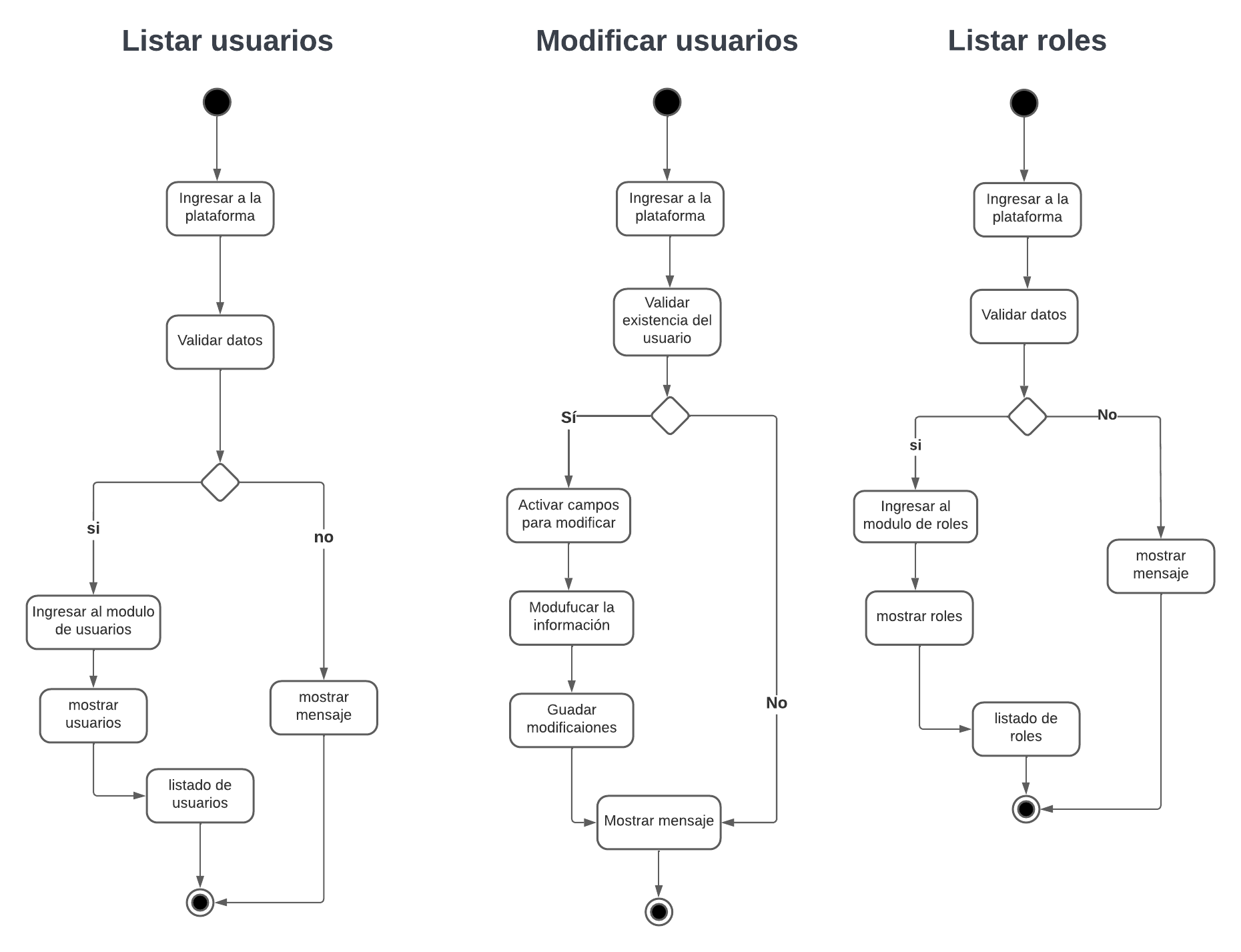
*Fuente: Autor*

*Ilustración 10 Diagrama de actividades: Filtrar búsquedas, editar perfil y agregar usuarios*

****

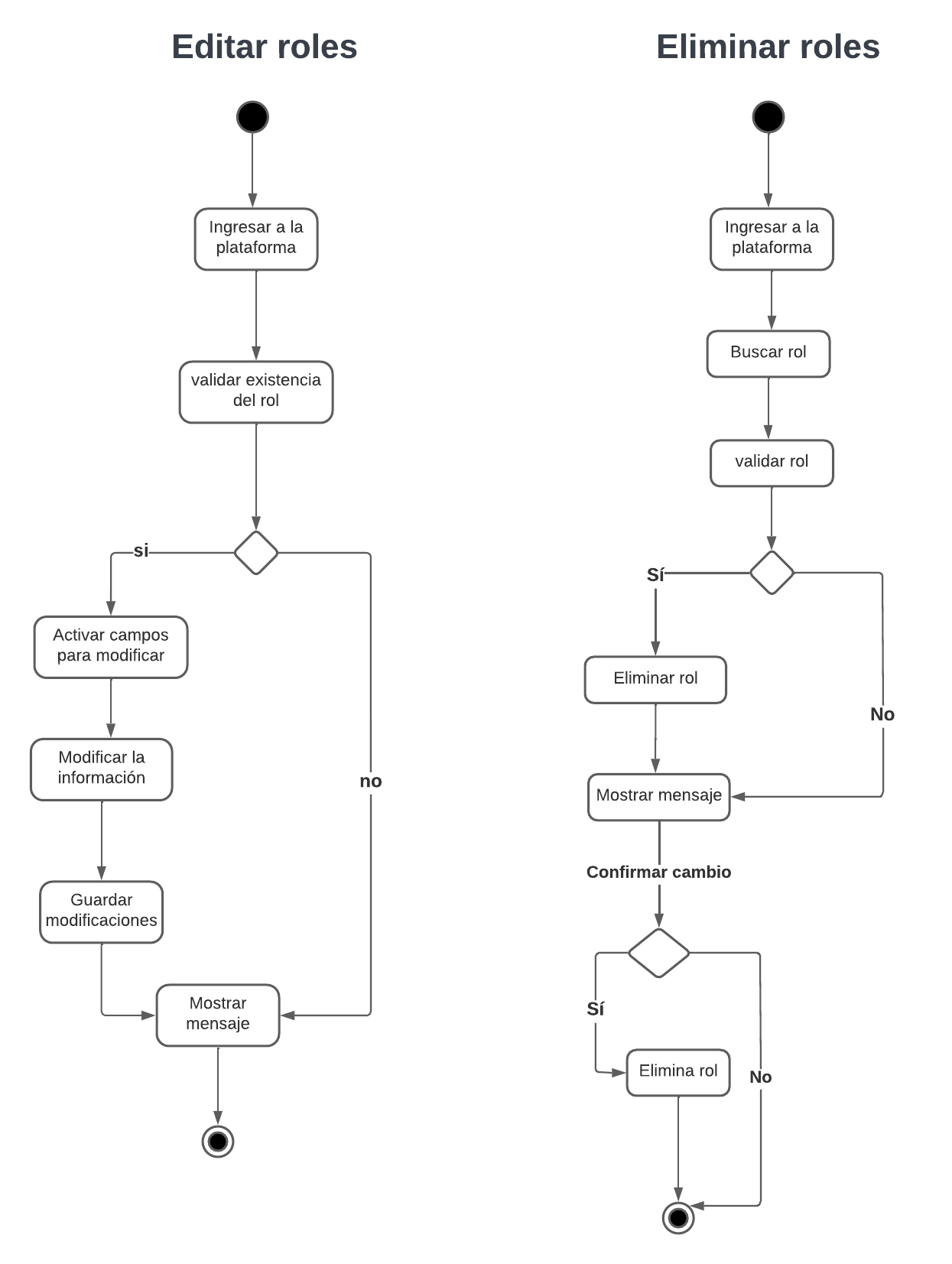
*Fuente: Autor*

*Ilustración 11 Diagrama de actividades: Listar, modificar usuarios y listar roles*

****

*Fuente: Autor*

*Ilustración 12 Diagrama de actividades: Editar roles*

****

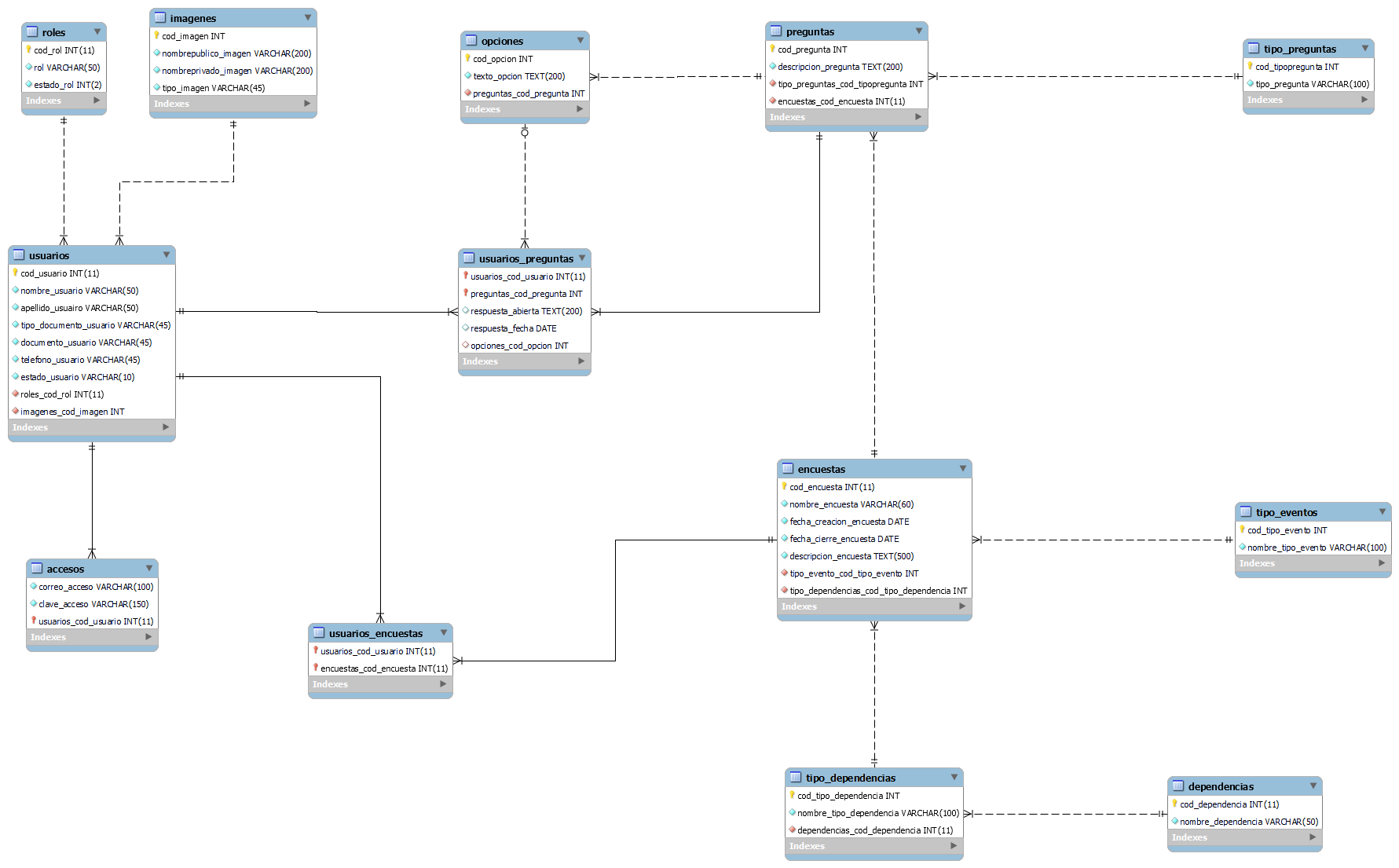
*Fuente: Autor*

### Diseño de la base de datos

A continuación se presenta el diseño entidad-relación de la base de datos, el cual está compuesto por las siguientes tablas:

* roles
* imágenes
* usuarios
* accesos
* usuarios\_encuestas
* encuestas
* tipo\_dependencias
* dependencias
* tipo\_eventos
* preguntas
* tipo\_preguntas
* opciones
* usuarios\_preguntas

*Ilustración 13 Modelo entidad relación de la base de datos*

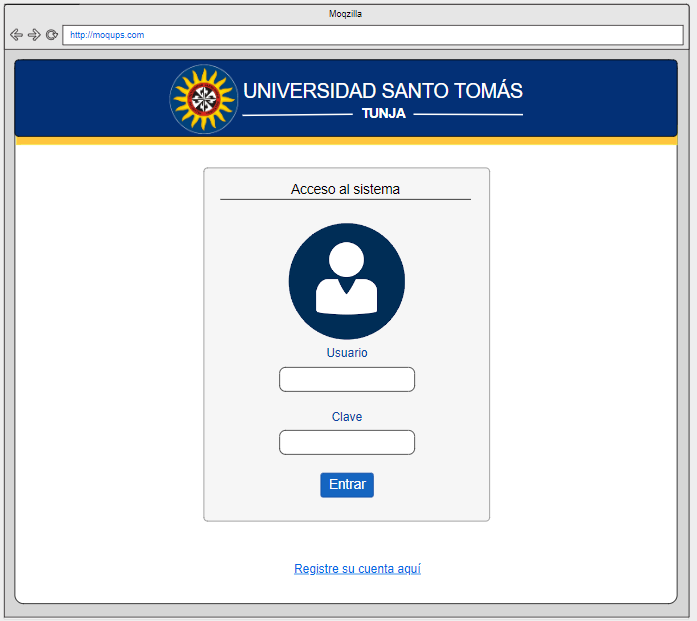


*Fuente: Autor*

### Diseño de la interfaz

Para el diseño de la interfaz del sistema, se crearon unos Mockups del primer prototipo del aplicativo web, con la ayuda de la herramienta *Moqups · Mockups, Wireframes & Prototyping.* Para la creación de estos Mockups se tuvo en cuenta el diseño de un software que fue expuesto por la Stakeholder, esto con el fin de que se pudiera conservar algunos colores, diseño y estructura en el sistema a desarrollar. ([Ver anexo](https://drive.google.com/drive/folders/1-h5UdM-l-DW2SFytgipmV5V_5XbdpA7A?usp=sharing))

*Ilustración 14 Mockup Inicio de sesión*



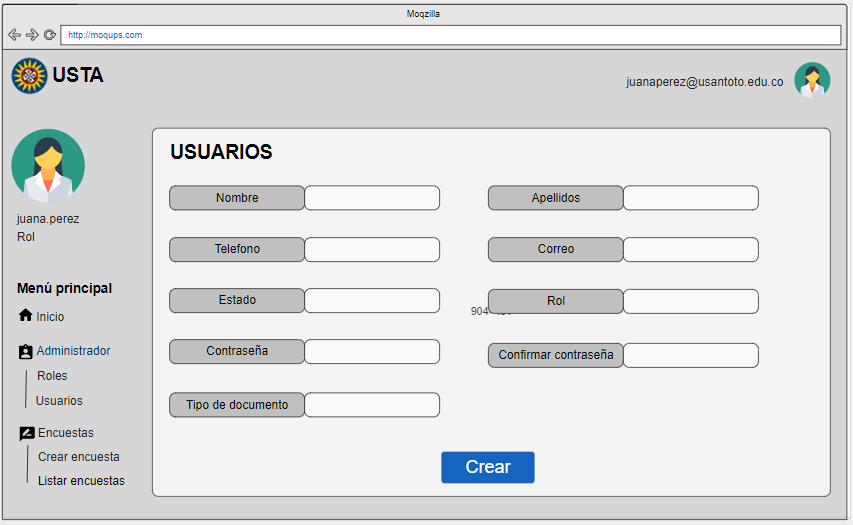
*Ilustración 15 Mockup Registrar cuenta*

**

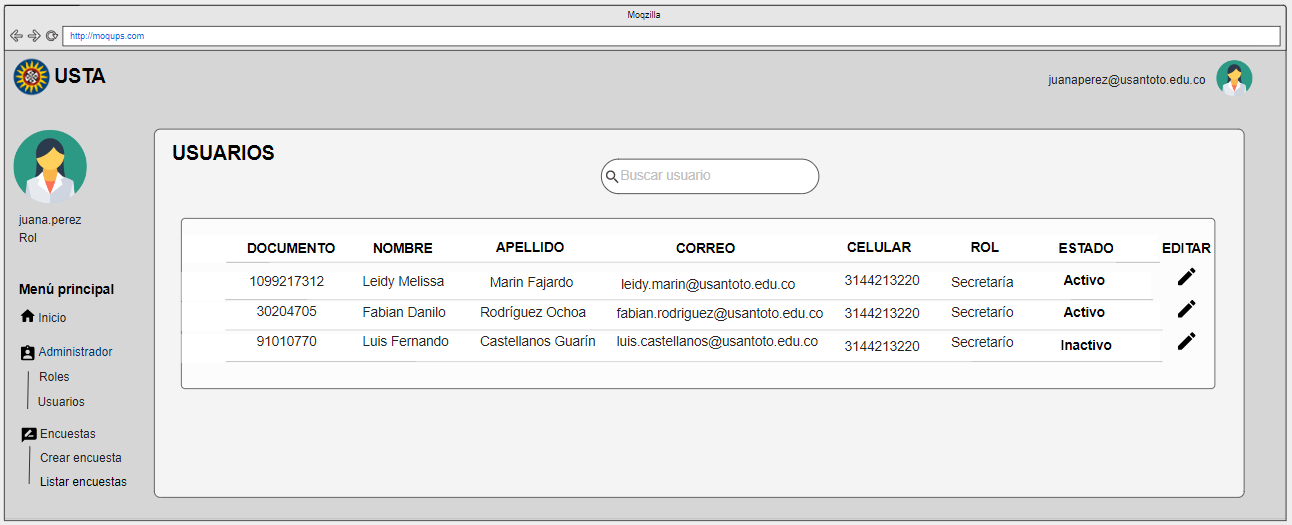
*Ilustración 16 Mockup Vista de inicio*



*Ilustración 17 Mockup Agregar usuario*

**

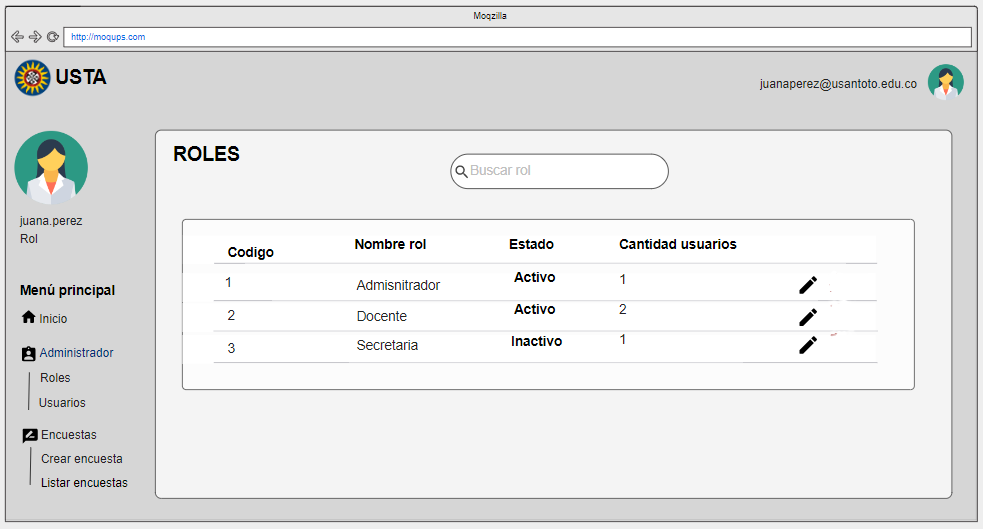
*Ilustración 18 Mockup Listado de usuarios*

**

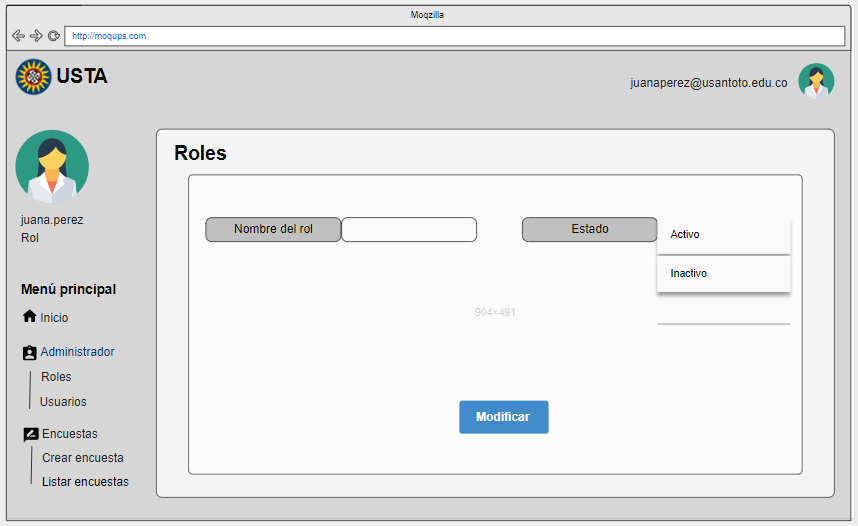
*Ilustración 19 Mockup Modificar usuario*

**

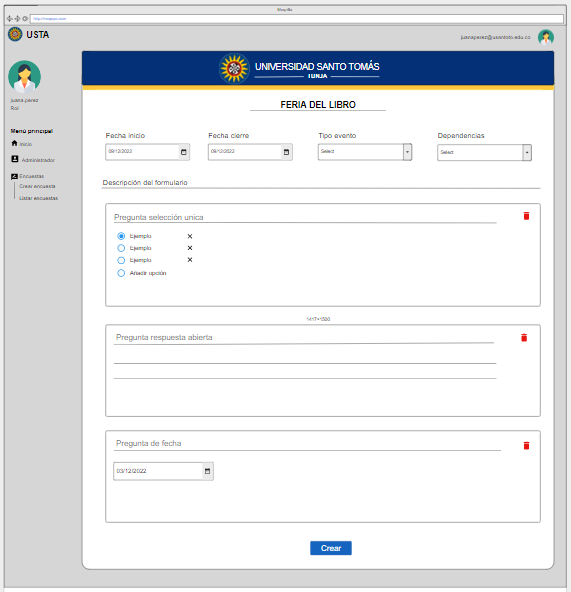
*Ilustración 20 Mockup Listar roles*

**

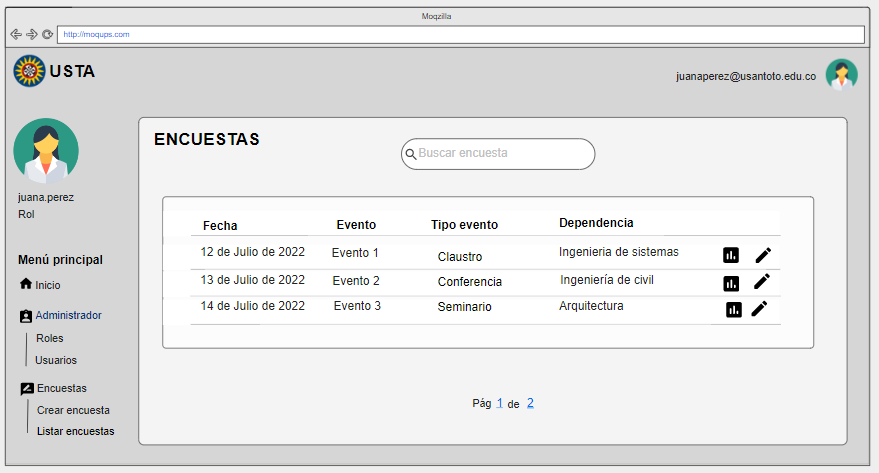
*Ilustración 21 Mockup Modificar estado del rol*

**

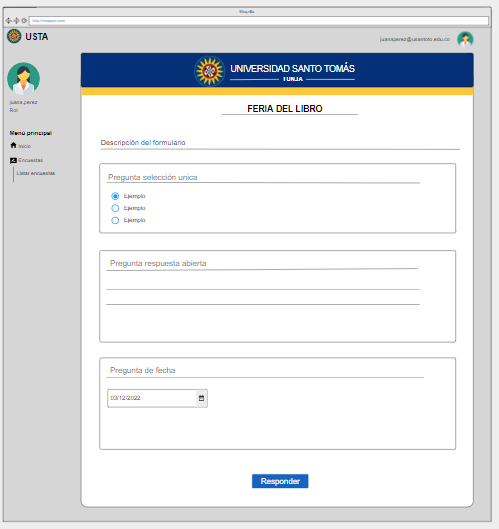
*Ilustración 22 Mockup Crear encuesta*

**

*Ilustración 23 Mockup Listado de encuestas*



*Ilustración 24 Mockup Formulario para responder encuestas*



*Tabla 4. Fase 2 Diseño del sistema*

|  |  |
| --- | --- |
| Tipo | Elementos |
| Entradas | * Documento de requerimientos |
| Instrumentos y/o herramientas | * Conexión a internet * Computador * Programas del computador * Navegador Web * Word * Moqups · Mockups, Wireframes & Prototyping |
| Salidas | Prototipos de los Mockups de la interfaz del sistema, diseño de los diagramas |
| Tiempo | 3 semanas |

*Fuente: Autor*

## Fase 3. Construcción del sistema

Esta arquitectura se compone de dos partes: Backend y Frontend. El desarrollo de la fase de construcción estuvo compuesto de las siguientes actividades:

### Selección de las tecnologías

* Para el motor de base de datos se eligió PostgreSQL. Puesto que este motor de base de datos es relacional, gratuito, seguro y ampliamente usado en el mundo empresarial, además cuenta con buena documentación y soporte. Por otra parte, permite almacenar datos grandes de forma segura.
* Como lenguaje de programación se eligió Typescript, ya que es un lenguaje de programación basado en JavaScript fuertemente tipado que brinda mejores herramientas en cualquier escala. Además, typescript se convierte a JavaScript, el cual se ejecuta en cualquier lugar donde se ejecute JavaScript. Por otra parte, Typescript agrega sintaxis adicional a JavaScript para admitir una integración más estrecha con su editor y de esta manera detectar errores de forma temprana en su editor. (JavaScript With Syntax For Types., s. f.)
* Para el entorno de ejecución se eligió Node js, ya que este entorno de ejecución es controlado por eventos diseñado para crear aplicaciones escalables, permitiendo establecer y gestionar múltiples conexiones al mismo tiempo. Gracias a esta característica, no hay bloqueo de procesos. (Simões, 2021)
* Como editor de código se utilizó Microsoft Visual Studio Code puesto que este editor de código es ligero y además se ejecuta en el escritorio y está disponible para Windows, macOS y Linux. Tiene soporte integrado para JavaScript, Typescript y Node.js, y un rico sistema de complementos para otros lenguajes y tiempos de ejecución.
* Para el framework del front se optó Angular, puesto que este framework ofrece control sobre la escalabilidad. Satisface las demandas masivas de datos mediante la creación de modelos de datos en RxJS, Immutable.js o cualquier otro modelo de inserción. Además este framework ofrece: (Angular, 2021a)
* Velocidad y rendimiento: Angular transforma sus plantillas en código altamente optimizado para las máquinas virtuales de JavaScript, ofreciendo todos los beneficios del código escrito a mano con la productividad de un marco. Además las aplicaciones angulares se cargan rápidamente con un nuevo enrutador de componentes que ofrece distribución automática de código, para que los usuarios carguen solo el código necesario para mostrar la vista que desean. (Angular, 2021a)
* Productividad: Crea rápidamente vistas de interfaz de usuario con una sintaxis de plantillas simple y precisa. Por otra parte obtiene finalización de código inteligente, errores instantáneos y otros comentarios en editores e IDE populares. (Angular, 2021a)

*Tabla 5. Fase 3 Selección de las tecnologías*

|  |  |
| --- | --- |
| Herramienta | Versión |
| Angular | 13.2.2 |
| Visual Studio Code | 1.72.1 |
| Typescript | 4.5.2 |
| Node .js | 16.15.0 |
| Atom | 1.60.0 |
| PostgreSQL | 14.3.1 |
| PG Admin | 6.8 |

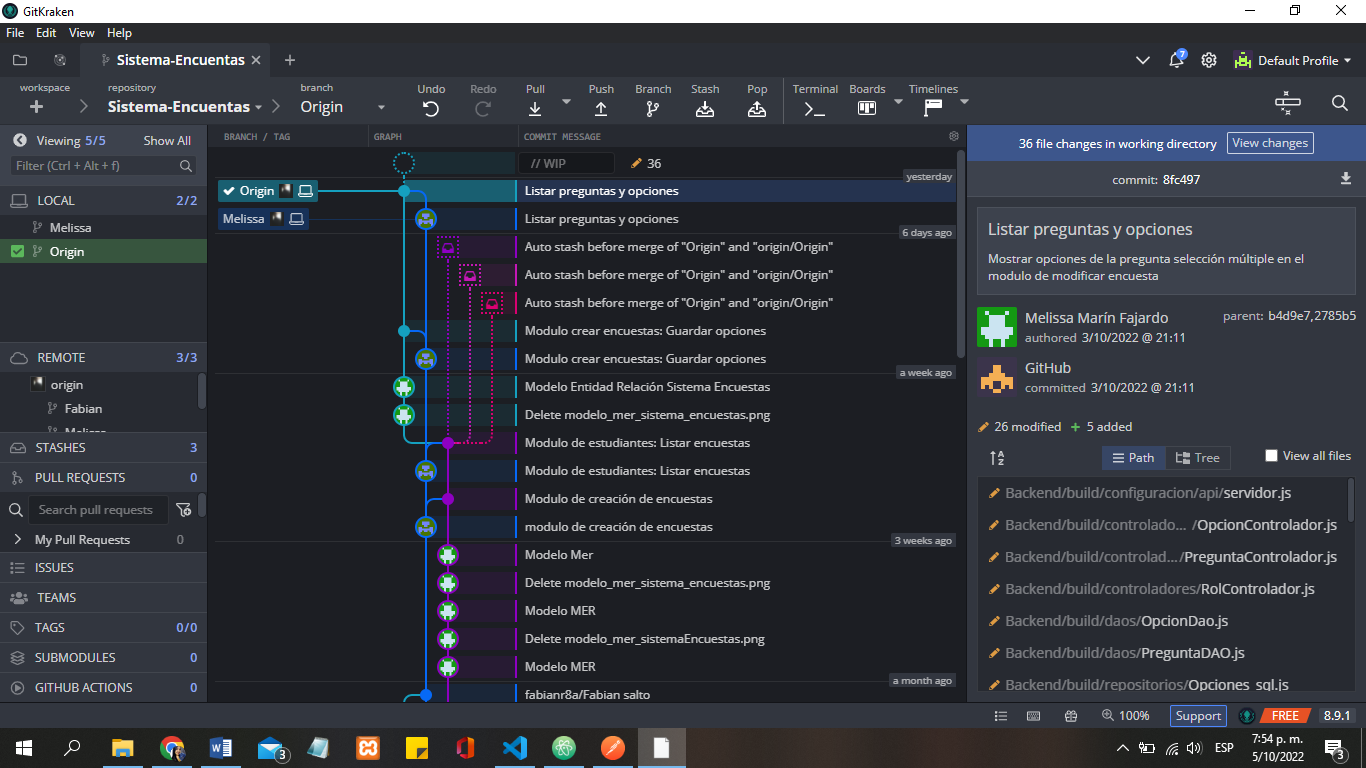
*Fuente: Autor*

### Otras herramientas

Teniendo en cuenta que el desarrollo del proyecto estaba conformado por 2 integrantes, se usó la herramienta GitKraken y GitHub, para ir teniendo un control y conocimiento sobre los cambios que iba haciendo cada integrante del equipo.

* GitKraken: Es un software completo que ofrece una gama de herramientas para ayudar a los desarrolladores en la ejecución de sus proyectos. Además, los equipos de trabajo pueden colaborar más fácilmente, sin importar dónde se encuentren. Por otra parte es fácil de leer y permite visualizar la estructura de la rama y el historial de confirmación. Asimismo, ofrece la posibilidad de visualizar quién ha realizado cambios en el código y cuándo lo ha hecho. También tiene la función de autocompletar los comandos. (Qué es GitKraken | KeepCoding Tech School, s. f.)

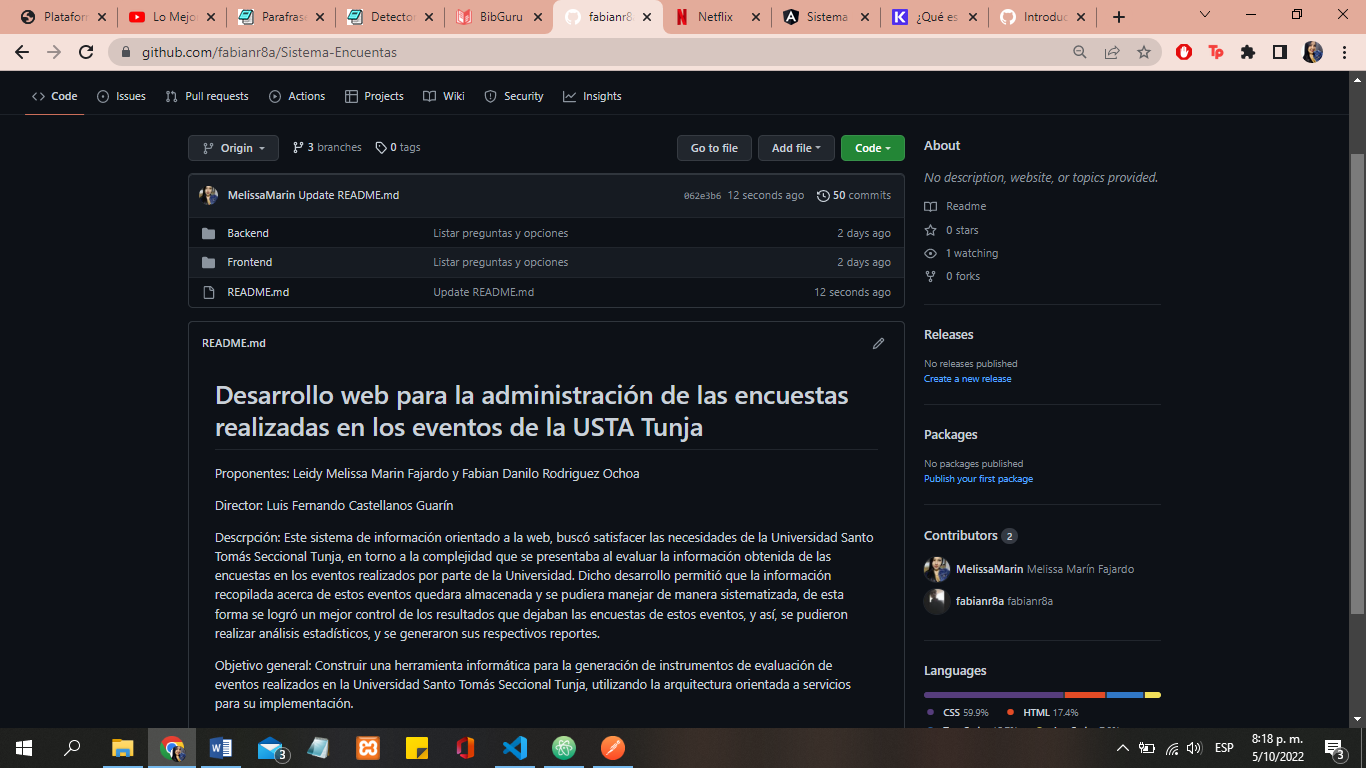
*Ilustración 25 Historial de cambios de la rama Origin en GitKraken*



*Fuente: Autor*

* GitHub: Es una plataforma de desarrollo inspirada en la forma de trabajar de cada desarrollador. Desde el código abierto hasta el comercial, puede alojar y revisar código, administrar proyectos y crear software junto con millones de otros desarrolladores.

*Ilustración 26 Repositorio GitHub del proyecto*

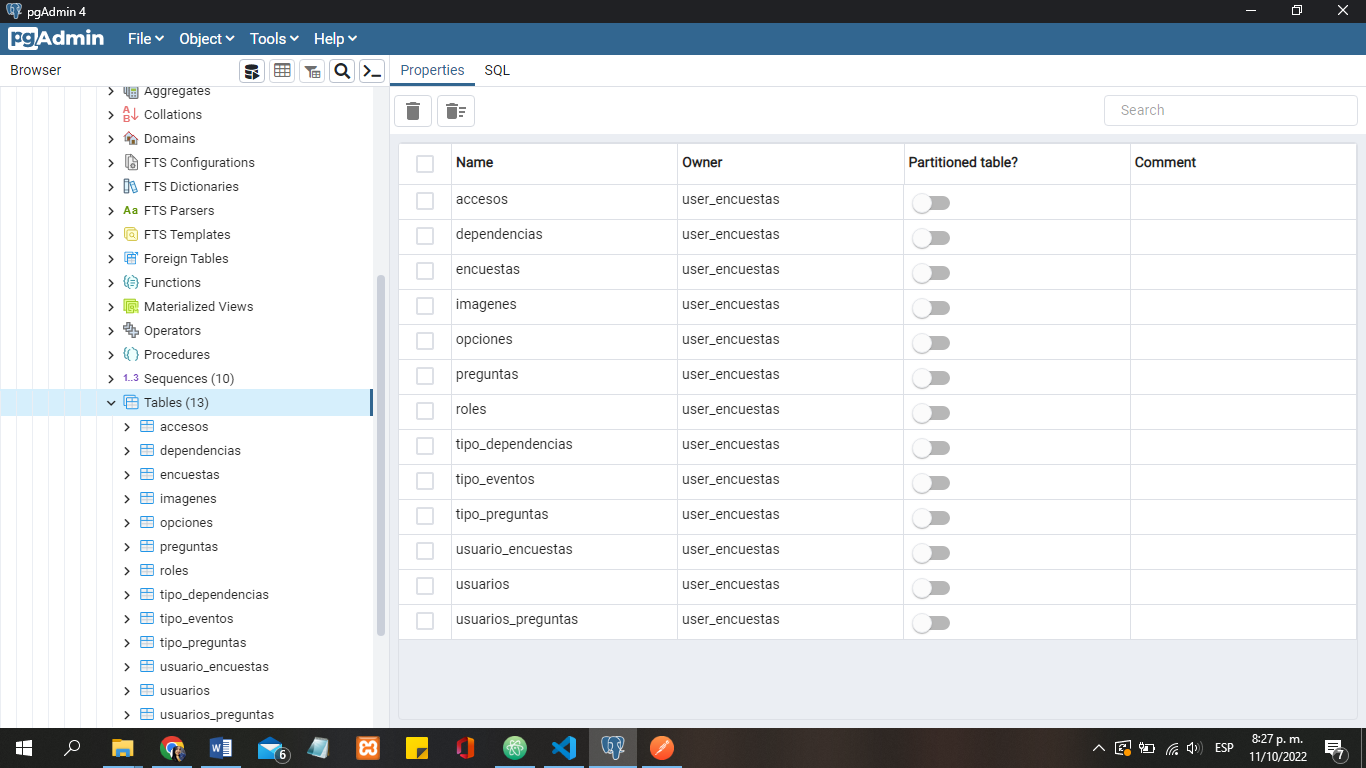


*Fuente: Autor*

### Construcción de la base de datos

Para la construcción de la base de datos se eligió Postgres como ya se había mencionado en la sección de selección de tecnologías. PostgreSQL es un sistema de gestión de base de datos muy estable y robusto que permite la escalabilidad y el mantenimiento de la integridad de los datos gracias a sus políticas internas de seguridad. Además, PostgreSQL ofrece un nivel de consultas bastante amplio, lo cual era necesario para el sistema que se desarrolló, puesto que dicho sistema requiere consultas recursivas para encontrar información de manera eficaz de acuerdo con la lógica proyectada en el diseño de la bases de datos.

*Ilustración 27 Base de datos PgAdmin*

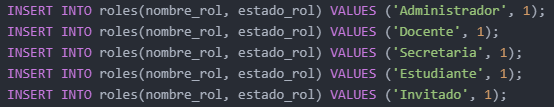


### Condiciones necesarias para el funcionamiento de la base de datos

Para que el funcionamiento de la base de datos se realice de manera exitosa se establece de manera predeterminada unas condiciones en el DML (lenguaje de manipulación de datos). El orden de las condiciones es el siguiente:

1. Los roles ya están establecidos, incluyendo su nombre y estado, donde 1 significa activo y 2 inactivo, estos no se pueden modificar, ni eliminar, además no se crean nuevos roles, esto de acuerdo con los requerimientos establecidos para el proyecto. A continuación se muestra el DML para la correcta inserción de los roles.

*Ilustración 28 Sentencia DML Creación de roles*



1. El aplicativo asignara una imagen por defecto a cada nuevo usuario, la cual puede ser modificada después por el usuario. Un ejemplo de la sentencia para insertar de manera correcta la imagen es la siguiente:

*Ilustración 29 Sentencia DML Creación de imágenes*



1. De manera predeterminada el sistema registra cada nuevo usuario con el rol de **Estudiante,** esto por motivos de seguridad y teniendo en cuenta que son muchos los estudiantes que se pueden registrar en el aplicativo, es más fácil para el administrador cambiar el rol de otros usuarios que asignar el rol a cada estudiante registrado. A continuación se muestra un ejemplo de cómo crear usuarios de manera correcta

Ilustración 30 Sentencia DML Creación de usuarios



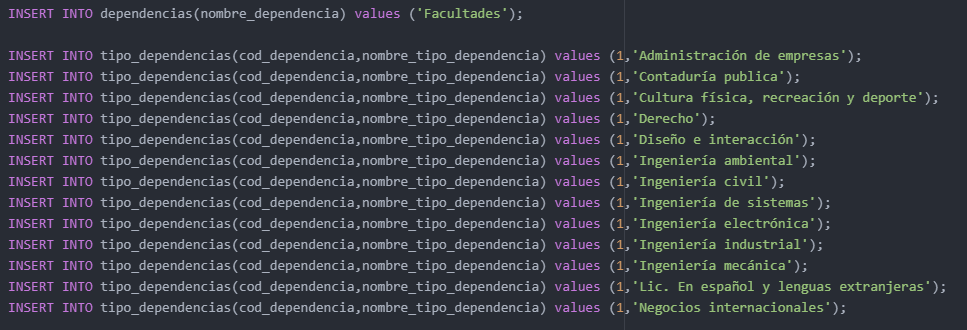
1. Para asignar los accesos al sistema, se debe tener el usuario primeramente creado, por eso la importancia de seguir el orden de las condiciones. Esta inserción debe realizarse como se muestra a continuación en el ejemplo

*Ilustración 31 Sentencia DML Creación de acceso*



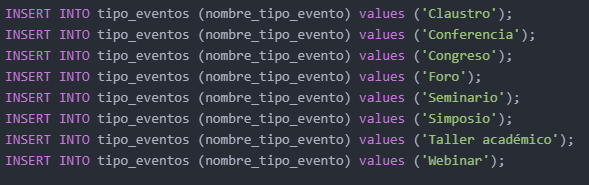
1. Para poder crear los tipos de dependencia, primero se debe crear la dependencia. De acuerdo a los requerimientos del sistema se manejó solo **Facultades** como dependencia, y de allí todas las facultades que hay en la USTA Tunja como tipos de dependencia. A continuación se muestra la sentencia DML en que se debe crear los tipos de dependencia

*Ilustración 32 Sentencia DML Creación de dependencias y tipo dependencias*



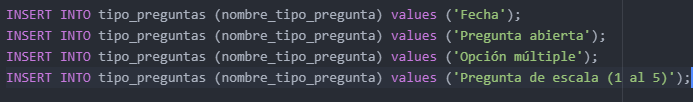
1. Para que las encuestas se puedan crear de manera correcta se debe tener creado los tipos de eventos como lo muestra el siguiente ejemplo

Ilustración 33 Sentencia DML Creación tipos de eventos



1. Seguidamente se crean los tipos de pregunta como lo indica el siguiente ejemplo

*Ilustración 34 Sentencia DML Creación tipo preguntas*



### Construcción del backend

#### Librerías

Las librerías ofrecen muchas funciones estándar para que los desarrolladores no tengan que preocuparse por demasiada funcionalidad. Por lo tanto, pueden usarse para crear fácilmente páginas web con componentes de interfaz de usuario, utilidades de lenguaje, funciones matemáticas y más. Las librerías usadas para la construcción del backend fueron las siguientes:

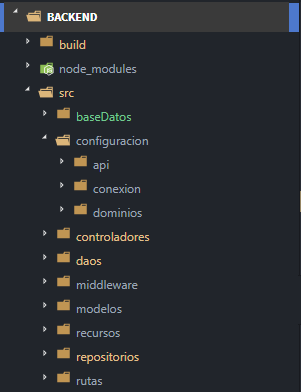
* @types/cors: 2.8.12,
* @types/express: 4.17.13,
* @types/formidable: 2.0.4,
* @types/jsonwebtoken: 8.5.8,
* @types/morgan: 1.9.3

#### Estructura

El framework de Node.js ofrece la posibilidad de estructurar el backend por módulos de acuerdo a la arquitectura que se planteó para el proyecto. Teniendo en cuenta que esto es una aplicación web que consumirá una API, se hace uso de la librería Express, la cual permite hacer las distintas peticiones y proporciona las características necesarias para poder consumir la API.

La estructura backend del sistema está distribuido de la siguiente manera:

*Ilustración 35 Estructura backend*

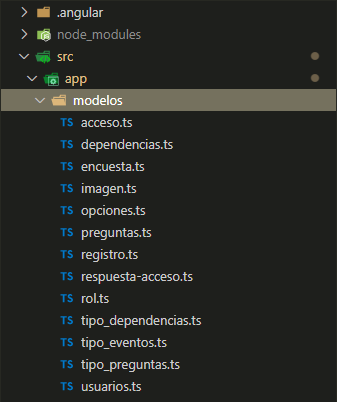


* Carpeta baseDatos: Allí se encuentra el DDL (lenguaje de definición de datos) para la creación de las tablas de la base de datos y el DML (lenguaje de manipulación de datos) con las sentencias para la inserción de los datos a la base de datos.
* Carpeta node\_modules: Esta carpeta se genera automáticamente cuando se crea por primera vez el proyecto, allí se almacenan todos los módulos que se ejecutan cuando se forma el proyecto.
* Carpeta configuración: Se divide en 3 subcarpetas: en la subcarpeta API se encuentran todas las rutas necesarias para poder consumir cada servicio. En la carpeta conexión se encuentran los archivos con las distintas configuraciones como el pool, para que la conexión se pueda realizar de manera exitosa, y finalmente en la subcarpeta dominios se define la lógica de conexión a la base de datos, es decir el usuario, la contraseña, el host, el puerto y el nombre de la base de datos.
* Carpeta controladores: Los archivos allí situados actúan como intermediario y permiten la gestión del flujo de información y las transformaciones para adaptar los datos a las necesidades de cada API.
* Carpeta DAOS: Estos archivos son los encargados de acceder a los datos a través de los métodos allí definidos, además se encargar de ordenar los resultados que arroja la ejecución de estos métodos.
* Carpeta middleware:
* Carpeta modelos: Los archivos que allí se encuentran son utilizados para enviar el objeto completo de cada modelo, en el caso de la imagen se necesita el objeto completo ya que en la búsqueda solo trae los datos de la tabla imágenes y se requiere también enviar el base 64 junto a la imagen
* Carpeta recursos: En esta carpeta se encuentran almacenadas las imágenes utilizadas por el usuario cuando modifica su perfil.
* Carpeta repositorios: Estos archivos son los encargados de acceder a las consultas a la base de datos y obtener la información solicitada por el controlador asignado.
* Carpeta rutas: Allí se encuentran los endpoints de la API, los cuales se encargan de gestionar las peticiones Get, Post y Put hacia cada controlador asignado.

### Construcción del frontend

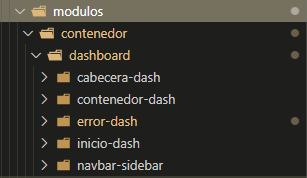
El framework de angular ofrece la posibilidad de modularizar el proyecto de acuerdo a la arquitectura planteada. Para el desarrollo de este proyecto se definieron los siguientes módulos:

*Ilustración 36 Estructura del frontend*



* Modelos: Aquí se define las propiedades para cada modelo, una representación de los datos que maneja el sistema, su lógica de negocio, y sus mecanismos de persistencia.

*Ilustración 37 Estructura del frontend*



* Módulos:

## Fase 4. Pruebas

Describir las pruebas aplicadas y el resultado.

Tabla 6. Fase 5

|  |  |
| --- | --- |
| Tipo | Elementos |
| Entradas | * Bibliotecas virtuales * Documentos y artículos de internet * Libros * Sitios Web * Tutoriales |
| Instrumentos y/o herramientas | * Sitio de trabajo * Conexión a internet * Computador * Programas del computador * Navegador Web * Word * Excel |
| Salidas | Conocimiento y fortalecimiento del paradigma de la programación orientada a objetos, sus características, alcance, técnicas de desarrollo, métodos y funciones entre otros campos vinculados con esta misma. |
| Tiempo | Semanas/meses |

Fuente: Autor

## Fase 6. Puesta en producción

Describir el montaje de la aplicación.

Tabla 7. Fase 6

|  |  |
| --- | --- |
| Tipo | Elementos |
| Entradas | * Bibliotecas virtuales * Documentos y artículos de internet * Libros * Sitios Web * Tutoriales |
| Instrumentos y/o herramientas | * Sitio de trabajo * Conexión a internet * Computador * Programas del computador * Navegador Web * Word * Excel |
| Salidas | Conocimiento y fortalecimiento del paradigma de la programación orientada a objetos, sus características, alcance, técnicas de desarrollo, métodos y funciones entre otros campos vinculados con esta misma. |
| Tiempo | Semanas/meses |

Fuente: Autor

# CONCLUSIONES Y ANÁLISIS DE RESULTADOS

En esta sección se presenta el análisis de los resultados obtenidos y las conclusiones que arroja la investigación desarrollada. Corresponde a argumentos y afirmaciones hechas sobre los resultados o cumplimiento de los objetivos propuestos y/o en general la experiencia del proceso de investigación, teniendo como base el marco de referencia del trabajo. Para ello, puede formularse las siguientes preguntas:

¿Se cumplieron o lograron los objetivos propuestos?

¿Qué dificultades tuvo que enfrentar para realizar o alcanzar dicho(s) objetivo(s)?

Se recomienda un texto no mayor de una hoja.

# REFERENCIAS

[1] CARCAMO, O., CORTES, Y. H. y SALAMANCA, J. C. (2021, julio). PROGRAMACIÓN EVENTO COMERCIAL Y DISEÑO DE LA CAMPAÑA DEDIFUSIÓN. Coursehero.Com. https://www.coursehero.com/file/116166576/evidencia-de-trabajo-de-eventodocx/

[2] Wong, M. (2018, noviembre 12). 21 factors for evaluating event management software. EventMobi. https://www.eventmobi.com/blog/21-factors-for-evaluating-event-management-software/

[3] Gilliam Haije, E. (2017, julio 25). 21 mejores software de encuestas en línea y herramientas de cuestionarios. Mopinion. https://mopinion.com/es/mejores-software-de-encuestas-en-linea-herramientas-de-cuestionarios/

[4] Sabido, M. R. M., & Pech, J. P. U. Proceso Unificado de Desarrollo para Aplicaciones con Arquitectura Orientada a Servicios. https://www.iiis.org/CDs2008/CD2008CSC/CISCI2008/PapersPdf/C316QM.pdf

[5] Mera Paz, J. A. (2016). Análisis del proceso de pruebas de calidad de software. Ingeniería solidaria, 12(20), 163-176. https://revistas.ucc.edu.co/index.php/in/article/view/1482

[6] IEEE Computer Society. (2008). IEEE Standard for Software and System Test Documentation. IEEE Standard 829, Section 3 “Definition”, pp. 8.

[7] IEEE Computer Society. (2008-1). IEEE Standard for Software and System Test Documentation. IEEE Standard 829, Section 10 “Level test design”, pp. 50. 8.

[8] IEEE Computer Society. (2008-2). IEEE Standard for Software and System Test Documentation. IEEE Standard 829.

[9] Chaves, M. A. (2005). La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de software. InterSedes: Revista de las Sedes Regionales, 6(10), 1-13. https://www.redalyc.org/pdf/666/66612870011.pdf

[10] Repositorio Digital de la Facultad de Ingenieria UNAM. (s. f.). Metodologías y procesos de análisis de software. Apache Tomcat/8.5.68. http://www.ptolomeo.unam.mx:8080/xmlui/bitstream/handle/132.248.52.100/175/A5%20Capítulo%202.pdf

[11] INGENIERÍA DE REQUERIMIENTOS. (s. f.). SEDICI - Repositorio de la Universidad Nacional de La Plata. http://sedici.unlp.edu.ar/bitstream/handle/10915/4057/2\_-\_Ingeniería\_de\_requerimientos.pdf?sequence=4&amp;isAllowed=y

[12] Tilkov, S., & Vinoski, S. (2010). Node. js: Using JavaScript to build high-performance network programs. IEEE Internet Computing, 14(6), 80-83. https://ieeexplore.ieee.org/abstract/document/5617064/citations

[13] Angular.io. 2021. Angular. [online] Available at: <https://angular.io/docs> [Accessed 5 December 2021]

[14] JavaScript with syntax for types. (s/f). Typescriptlang.org. Recuperado el 5 de octubre de 2022, de https://www.typescriptlang.org/

[15] Simões, C. (2021, julio 27). ¿Qué es Node.js, y para qué sirve? Blog ITDO - Agencia de desarrollo Web, APPs y Marketing en Barcelona. https://www.itdo.com/blog/que-es-node-js-y-para-que-sirve/

[16] Angular. (2021). Angular.Io. Recuperado el 5 de octubre de 2022, de https://angular.io/

[17] Qué es GitKraken. (2022, mayo 11). KeepCoding Tech School. https://keepcoding.io/blog/que-es-gitkraken/

# ANEXOS

## Anexo 1. Actas de reunión

Enlace: <https://drive.google.com/drive/folders/1EXrGbTJTD6ayAD8-8qkdWXtGFrAI-yvf?usp=sharing>

## Anexo 2. Documento de especificación de requerimientos

Enlace:<https://docs.google.com/document/d/1cujPb_NmvDzpBcD6d_c5X7n2C_GQnTSw/edit?usp=sharing&ouid=110006067162752201967&rtpof=true&sd=true>

## Anexo 3. Mockups del diseño de la interfaz

Enlace: <https://drive.google.com/drive/folders/1-h5UdM-l-DW2SFytgipmV5V_5XbdpA7A?usp=sharing>

## Anexo 4. Manual de usuario

Enlace:<https://docs.google.com/document/d/16ggOrw5AOwRoSM5eoWbipAozfPp9Y6Ro/edit?usp=sharing&ouid=110006067162752201967&rtpof=true&sd=true>

**Anexo 2. Ficha Técnica**

Utilice esta sección para indicar las características técnicas del software, describa las condiciones que requiere un equipo de cómputo para su despliegue.

**Anexo 3. Manual de Usuario**

En esta sección debe realizar un recorrido por las funcionalidades de la aplicación, utilizando pantallazos y descripciones claras que orienten a un usuario en el uso del sistema.