

APLICACIÓN WEB PARA LA ADMINISTRACIÓN DEL PROCESO DE DESARROLLO EN
TRABAJOS DE GRADO ELABORADOS BAJO LA METODOLOGÍA ÁGIL **MATRAGA**.

JUAN PABLO MOSQUERA COSSÍO
SANTIAGO PRIETO RAMÍREZ

UNIVERSIDAD AUTÓNOMA DE COLOMBIA
FACULTAD DE INGENIERÍA
PROGRAMA DE INGENIERÍA DE SISTEMAS
BOGOTÁ
2021

APLICACIÓN WEB PARA LA ADMINISTRACIÓN DEL PROCESO DE DESARROLLO EN
TRABAJOS DE GRADO ELABORADOS BAJO LA METODOLOGÍA ÁGIL **MATRAGRA**.

JUAN PABLO MOSQUERA COSSÍO
SANTIAGO PRIETO RAMÍREZ

TESIS PROYECTO DE GRADO

Director:
GUSTAVO RIVERA

UNIVERSIDAD AUTÓNOMA DE COLOMBIA
FACULTAD DE INGENIERÍA
PROGRAMA DE INGENIERÍA DE SISTEMAS
BOGOTÁ
2021

TABLA DE CONTENIDO

INTRODUCCIÓN.....	6
1. DATOS GENERALES DEL PROYECTO	7
1.1 TÍTULO DEL PROYECTO	7
1.1.1 ÁREA DE INVESTIGACIÓN	7
1.2 PLANTEAMIENTO DEL PROBLEMA.....	7
1.3 OBJETIVOS	8
1.3.1 OBJETIVO GENERAL.....	8
1.3.2 OBJETIVOS ESPECÍFICO:.....	8
1.4 JUSTIFICACIÓN	8
1.5 ALCANCES	9
1.5.1 LIMITACIONES	9
2. MARCO DE REFERENCIA	10
2.1 ANTECEDENTES	10
2.2 MARCO TEÓRICO.....	10
2.2.1 METODOLOGÍA ÁGIL PARA TRABAJOS DE GRADO MATRAGRA	10
2.2.2 INGENIERÍA DE SOFTWARE.....	15
2.2.3 COMPUTACIÓN EN LA NUBE.....	16
2.2.4 BASES DE DATOS NOSQL	17
2.3 MARCO CONCEPTUAL.....	19
2.3.1 APLICACIÓN WEB.....	19
2.3.2 AWS.....	19
2.3.3 AWS LAMBDA	19
2.3.4 AWS API GATEWAY	20
2.3.5 AWS DYNAMODB	20
2.3.6 AWS S3	21
2.3.7 AWS CLOUDWATCH	21
3. ASPECTOS ADMINISTRATIVOS	22
3.2 RECURSOS	22
3.3 CRONOGRAMA.....	23
4. INGENIERIA DEL PROYECTO.....	25
4.1 MODELO DE NEGOCIO	25
4.1.1 DIAGRAMA HIPO	25
4.2 MODELO CANVAS	26

4.2 METODOLOGIA DE DISEÑO	27
4.2.1 MODELO DE CASOS DE USO	27
4.2.2 DISEÑO DE BASE DE DATOS	35
4.2.3 DISEÑO DE ARQUITECTURA DE LA APLICACIÓN.....	35
4.2.4 DISEÑO DE INTERFACES	36
5. PRUEBAS.....	43
6. ANÁLISIS DE RESULTADOS	46
ANALISIS DE RESULTADO (alcance vs desarrollado en el proyecto)	46
7. CONCLUSIONES Y RECOMENDACIONES.....	47
8. ANEXOS	48
REFERENCIAS.....	49

TABLA DE ILUSTRACIONES

Ilustración 1: registro de actividades MaTraGra, fuente: metodología ágil MaTraGra (Rivera, 2018).....	11
Ilustración 2: ciclo de vida de MaTraGra, fuente: metodología ágil MaTraGra (Rivera, 2018)	12
Ilustración 3: funcionamiento de API Gateway, fuente: AWS Documentation (Amazon, 2021).	20
Ilustración 4: funcionamiento de AWS CloudWatch, fuente: AWS Documentation (Amazon, 2021).	22
Ilustración 5. Tabla recursos humanos.....	22
Ilustración 6. Tabla recursos logísticos.....	22
Ilustración 7. Tabla recursos tecnológicos.....	23
Ilustración 8. Tabla recursos administrativos	23
Ilustración 9. cronograma de actividades.....	24
Ilustración 10. planificación de actividades.....	24
Ilustración 11: Diagrama HIPO, fuente: elaboración propia	25
Ilustración 12 tabla modelo de negocio Canvas, fuente: elaboración propia.....	26
Ilustración 13 Caso de uso interactuar con el proyecto.....	27
Ilustración 14 Caso de uso interactuar en el foro.....	30
Ilustración 15 Caso interactuar con la agenda	32
Ilustración 16 Caso de uso publicar documentos.....	34
Ilustración 17 Diseño de tablas de base de datos	35
Ilustración 18 Diagrama de arquitectura, fuente: elaboración propia.	35
Ilustración 19 Diagrama generado de infraestructura frontend, fuente: elaboración propia.....	36
Ilustración 20 Diagrama de infraestructura backend generado, fuente: elaboración propia.	36
Ilustración 21 Mapa de navegación, fuente: elaboración propia.....	37
Ilustración 22 Boceto Diseño de interfaz de proyecto	37
Ilustración 23 Interfaz landing page.....	38
Ilustración 24 Interfaz Documentación.....	38
Ilustración 25 Interfaz Ingresar	38
Ilustración 26 Interfaz Registrarme.....	39
Ilustración 27 Interfaz Proyectos	39
Ilustración 28 Interfaz Proyecto.....	39
Ilustración 29 Interfaz Foro	40
Ilustración 30 Landing page móvil	40
Ilustración 31 Interfaz Ingresar y Registrarme móvil	41
Ilustración 32 Interfaz Proyectos y Proyecto móvil.....	41
Ilustración 33 Interfaz Foro y Documentación móvil.	42
Ilustración 34 Resultado general de la prueba, fuente: elaboración propia.	43
Ilustración 35 Resultado de performance, fuente: elaboración propia.....	43
Ilustración 36 Resultado de mejores prácticas, fuente: elaboración propia.	44
Ilustración 37 Resultado SEO, fuente: elaboración propia.	44
Ilustración 38 Petición exitosa con autenticación	44
Ilustración 39 Petición fallida por autenticación.....	45
Ilustración 40 Lista de endpoints	45

INTRODUCCIÓN

En el proceso académico de la educación superior los proyectos de grado son una parte fundamental para la aplicación de los conocimientos adquiridos durante este proceso, por esto el adecuado seguimiento y desarrollo de esos proyectos es sumamente importante para la satisfactoria culminación académica.

Por ende, la necesidad de generar un proceso que permita la organización y adecuado seguimiento de la información debido a esto nació **MaTraGra**, la cual es una metodología ágil desarrollada por el ingeniero Gustavo Rivera (2018), la cual permite ambientes contextualizados para el desarrollo y seguimiento de proyectos de grado implementados en las facultades de ingenierías de educación superior.

En ese proyecto se desarrollará una aplicación que emplee la metodología **MaTraGra**, en donde se facilitará la gestión para los estudiantes y tutores, permitiendo el acceso remoto de los diferentes usuarios implicados en cada una de las etapas del proceso, mejorando y agilizando procedimientos de manera ordenada.

1. DATOS GENERALES DEL PROYECTO

1.1 TÍTULO DEL PROYECTO

Aplicación web para la administración del proceso de desarrollo en trabajos de grado elaborados bajo la metodología ágil MaTraGra.

1.1.1 ÁREA DE INVESTIGACIÓN

- Línea: Algoritmos y programación.
- Sub-Línea: Desarrollo de productos.

1.2 PLANTEAMIENTO DEL PROBLEMA

Actualmente no existe un canal oficial especializado que facilite la comunicación entre los alumnos y tutores con relación a los proyectos de grado lo que dificulta el intercambio de información entre los estudiantes que desarrollan los proyectos y los tutores; por lo tanto, el avance y desarrollo de los proyectos de grado que son requisito obligatorio para graduarse se dificulta y extiende.

Lo anterior genera la necesidad de una herramienta y/o canal de comunicación para administración, control y gestión que permita a los estudiantes que desarrollan sus trabajos de grado mantener contacto con sus tutores, especialmente en el contexto social actual en donde por motivos de pandemia, las reuniones presenciales son limitadas y se deben evitar. Por lo tanto, un sistema que conceda la adecuada comunicación en la situación actual permitiría un adecuado desarrollo y seguimiento por parte del tutor a sus estudiantes, también les daría a los estudiantes un espacio en donde solucionar sus dudas y mostrar sus progresos a su tutor.

1.3 OBJETIVOS

1.3.1 OBJETIVO GENERAL

Desarrollar una aplicación web con soporte en plataformas móviles enfocada al desarrollo proyectos de software planteados en trabajos de grado orientados por la metodología MaTraGra.

1.3.2 OBJETIVOS ESPECÍFICO:

- Permitir la comunicación entre estudiante(s) y tutor.
- Administrar la agenda, actividades y desarrollo del proyecto de software.
- Gestionar los procesos orientadores de la metodología MaTraGra.

1.4 JUSTIFICACIÓN

El proceso por el cual se da seguimiento a los proyectos de grado no es eficiente para ninguno de las partes involucradas, ya que no permite una adecuada comunicación entre el estudiante y el tutor, lo cual genera retrasos en su desarrollo y ejecución. Por estos motivos es necesario aplicar un sistema software adaptado e implementado para administrar los proyectos de grado desarrollados por los estudiantes.

Por lo tanto, el sistema software aplica la metodología ágil MaTraGra permitiendo la recopilación de todos los proyectos en una sola plataforma, en donde se generará un proceso metodológico que permita la adecuada retroalimentación de las partes involucradas y el cumplimiento de los compromisos adquiridos en el anteproyecto.

1.5 ALCANCES

El presente proyecto maneja los siguientes alcances:

- Aplicación web con arquitectura en la nube.
- Entregar documentación acorde con la norma ISO 9001.

1.5.1 LIMITACIONES

Dentro de los elementos que limitan el amplio desarrollo del proyecto se encuentran:

TIEMPO

- El proyecto está compuesto por dos estudiantes que tendrán disponibilidad de 10 horas semanales.
- El proyecto cuenta con un tiempo establecido de seis meses para su desarrollo y ejecución.

USUARIOS

- La aplicación está diseñada para ser utilizada, inicialmente, por estudiantes de ingeniería de sistemas de la Universidad Autónoma de Colombia que decidan utilizar la metodología ágil MaTraGra.

PROCEDIMENTAL

- Se basará exclusivamente en el documento explicativo de la metodología MaTraGra, proveída por el profesor Gustavo Rivera (2018).

INGENIERIL

- Debido a que el proyecto es desarrollado en la nube, solo se utilizan los servicios y recursos de Amazon Web Services (Amazon, 2021).
- La aplicación desarrollara el back-end en Node.js 12 y el front-end se desarrollará en React.JS; todo el proceso va a funcionar con los recursos de Amazon Web Services y plataformas web, por lo tanto, la aplicación funcionara en buscadores web como: Google Chrome, Safari, Mozilla Firefox, Edge y sus versiones Mobile.

2. MARCO DE REFERENCIA

2.1 ANTECEDENTES

Como antecedentes a esta propuesta se identifican algunos proyectos que con anterioridad han trabajado sobre temas relacionados como, por ejemplo:

El proyecto de grado “Software para el seguimiento, la gestión y el control de proyectos de grado en el departamento de electrónica” (Rodríguez & Ladino, 2014) en la Pontificia Universidad Javeriana en donde se construyó una herramienta web para la gestión y administración del proceso de trabajos de grado en el departamento de electrónica de la Pontificia Universidad Javeriana.

Por otro lado, el proyecto de grado “Sistema de información y gestión de proyectos de grado” (Sarmiento & Quirós, 2013) en la Universidad Libre, cuyo objetivo principal tal como se menciona es Desarrollar un prototipo de sistema de gestión para controlar y administrar de manera ordenada las entregas de los trabajos de grado.

Aplicación denominada TraGraApp, enfocada a el desarrollo de proyectos de software en trabajos de grado para plataformas móviles sin incluir ambientes Web. Desarrollada como trabajo de grado en la Fundación Universidad Autónoma de Colombia (D’Aleman & Castillo, 2016).

2.2 MARCO TEÓRICO


2.2.1 METODOLOGÍA ÁGIL PARA TRABAJOS DE GRADO MATRAGRA

ROLES: Bajo la premisa de construcción de la metodología ágil en ambientes contextualizados por los trabajos de grado en el espacio académico en Ingeniería, dispuestos por Instituciones de Educación superior, para el cumplimiento de una práctica integral y consecuentes con la generalidad de los trabajos de grado, se disponen de dos roles:

- Estudiante: adquiere la máxima responsabilidad académica en el proceso del trabajo de grado, derivada de la responsabilidad en el cumplimiento de lo pactado en el anteproyecto, relacionada con el desarrollo del aplicativo y la documentación en torno a él y al proyecto abordado. Puede considerarse en forma individual o en equipo como se disponga.
- Director: asume roles de líder encargado de direccionar el proceso integralmente, con la máxima responsabilidad de validar ingenierilmente los productos entregables y la aplicación final, asimismo, la documentación de apoyo al proceso. Implícitamente se advierte en él, el papel de representación informal del usuario de la aplicación, en acciones de requerimientos y validación. Productor de las órdenes o certificaciones finales de cumplimiento.

DOCUMENTACIÓN: se dispone como parte del proceso metodológico y para efectos de sustentación documental del trabajo abordado, el cumplimiento de los compromisos adquiridos aprobados y registrados en el anteproyecto, de dos formas o categorías concurrentes de documentación:

- Registro de actividades: en el cual se consignan, las actividades propuestas y realizadas durante todo el proceso, con el control de fechas correspondientes y las observaciones pertinentes -figura 1-. Además, puede usarse como instrumento de planeación de agenda de actividades y registro sustentable del proceso.



REGISTRO DE ACTIVIDADES

TRABAJO DE GRADO:

ESTUDIANTES: _____

DIRECTOR: _____

No.	FECHA HOY	ACTIVIDADES REALIZADAS	ACTIVIDADES A REALIZAR	FECHA PRÓXIMA	FIRMA ESTUDIANTE	FIRMA DIRECTOR	OBSERVACIONES
1							
2							
3							
4							
5							
6							
7							
8							
9							
10							

Ilustración 1: registro de actividades MaTraGra, fuente: metodología ágil MaTraGra (Rivera, 2018)

- Documento ingenieril: consecuente con las especificaciones y requerimientos del caso solicitadas por la institución, en él se documenta el proceso del trabajo de grado, además de las evidencias del cumplimiento ingenieril del ciclo de desarrollo del aplicativo. Soporte para la sustentación del espacio académico realizado por los estudiantes y certificado por el director. De responsabilidad, en su construcción, del equipo de estudiantes.

COMPONENTES: Se presenta la descripción de los procesos o fases que conforman de la metodología y su explicación fundamentada en el proceder para ser aplicada, en relación con las facetas que determinan los prerrequisitos, funcionalidad y los resultados esperados.

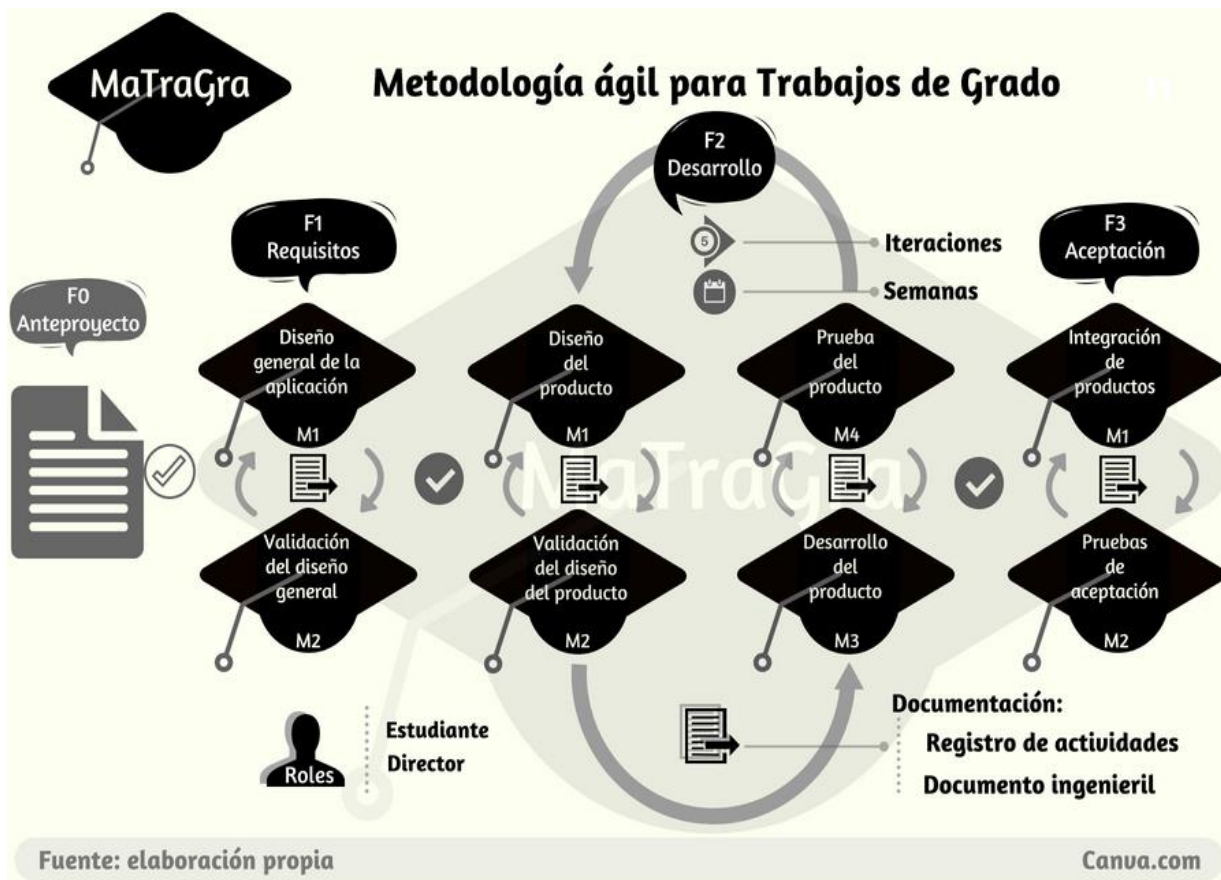


Ilustración 2: ciclo de vida de MaTraGra, fuente: metodología ágil MaTraGra (Rivera, 2018)

FASE 0, Anteproyecto: hace referencia al anteproyecto, constituyéndose en una fase prerequisite no contemplada en la metodología propuesta.

FASE 1, Requisitos: Es la primera fase de exploración, a partir de la concepción de la idea del producto de software, donde se contempla la identificación de requerimientos, el diseño ingenieril de la aplicación, planeación del trabajo sus espacios de validación. Adicionalmente, corresponde a las actividades de planificación de tiempos, recursos, espacios y actividades a desarrollar. Se disponen de dos momentos:

- Momento 1, Diseño general de la aplicación: plantea el análisis de requerimientos y el modelo estructural de la aplicación, se establecen en diseño, los componentes y la arquitectura del software, especificación de la plataforma y herramientas de desarrollo, diseño de la Base de datos, entre otros; todo ello dispuesto en los documentos pertinentes basados en las disposiciones institucionales correspondientes. Asimismo, la planeación funcional y operativa del trabajo a realizar en forma general, espacios de trabajo, responsabilidades y momentos de entrega de compromisos y pruebas, plasmada en el registro de actividades. Momento atenuante y determinado por los integrantes del equipo de estudiantes del trabajo de grado.

- Momento 2, Validación del diseño general: espacio de evaluación y validación ingenieril, del “Diseño General de la Aplicación”, realizada por el director del trabajo de grado, en conjunción del equipo de estudiantes; de donde se puede derivar ajustes al trabajo realizado, por consiguiente, la iteración en la fase 1, hasta el cumplimiento de los objetivos planteados en el anteproyecto en perspectiva.

FASE 2, Desarrollo: dispuestos en ciclos de trabajo supeditado a lo planeado en la fase anterior, se presenta el desarrollo del aplicativo desde la concepción ingenieril en módulos o productos entregables. Conlleva ciclos de trabajo -se recomienda en promedio 5 ciclos-, donde se obtiene al final de cada ciclo general el producto funcional validado. Fase compuesta de iteraciones relacionadas con el diseño y validación del producto objetivo, a la que paralelamente desarrolla la documentación técnica y de planeación de actividades. Se disponen de 4 momentos y 2 iteraciones:

- Momento 1, Diseño del producto: se fundamenta en la realización del diseño modelado ingenieril del producto de software entregable, análisis de requerimientos desde la perspectiva del usuario, especificaciones del producto entregable, se constituye el alcance y su operatividad, su modelo funcional, las pruebas a contemplar, soportada por la documentación correspondiente. Momento de responsabilidad del equipo de estudiantes.
- Momento 2, Validación del diseño del producto: Para el aseguramiento de cumplimiento de especificaciones, de acuerdo a los requisitos de diseño y pruebas planificados en el momento anterior, participa del momento, además del equipo de estudiantes, el director como componente validador del diseño del producto, mediante su retroalimentación permite la iteración de éstos dos momentos –Momento 1 y 2-, se dispone de la documentación actualizada correspondiente.
- Momento 3, Desarrollo de producto: de responsabilidad del equipo de estudiantes, basado en el diseño liberado en el momento anterior, y sustentado en lo dispuesto en la fase 1 de planeación. Se consideran en éste momento, los preceptos ingenieriles sobre desarrollo de software, en los cuales se incluyen las pruebas de consistencia pertinentes. Asimismo, se desarrolla la sustentación documental del trabajo realizado.
- Momento 4, Prueba del producto: espacio de trabajo en el que se incorpora el director conjuntamente al equipo de estudiantes, para realizar la verificación general y específica de la evolución del proceso, asimismo, de los compromisos adquiridos, evaluados a partir del producto de software entregable y el soporte documental correspondiente. De lo cual, se pueden derivar recomendación que sustentan la iteración del proceso de desarrollo del producto y que puede aplicar también al rediseño del mismo.

Se recomienda dar continuidad a la realización de un nuevo producto entregable, al culminar el que se encuentra trabajando; Momento en el cual se cuenta un espacio de transición para la Revisión diagnóstica del proceso, con el fin de determinar el cumplimiento global de la planeación y las dificultades evidenciadas. Adicionalmente, es conveniente sustentar y planear el trabajo en el registro de actividades.

FASE 3, Aceptación: hace referencia a la integración de los productos entregables, para determinar la totalidad de la aplicación propuesta en el anteproyecto, para verificar su funcionalidad y aplicación a estándares de reconocimiento ingenieril. Permite establecer iteraciones hasta determinar el cumplimiento integral de lo pactado. Igualmente, es un proceso documentado en sus dos medios dispuestos –documento ingenieril y registro de actividades-, originando la certificación correspondiente. Hacen parte de ésta fase dos momentos iterables:

- Momento 1, Integración de productos: ocasionado por la terminación de los productos entregables planeados, de participación exclusiva del equipo de estudiantes, de acuerdo con los preceptos ingenieriles demandados, que incluyen las pruebas correspondientes. Momento documentado acorde a los planteamientos ingenieriles.
- Momento 2, Prueba de la Aplicación: Orientado a la validación final del aplicativo y el soporte documental correspondiente, en donde participa el director del trabajo de grado, quien adquiere la responsabilidad de dar fe del proceso y el cumplimiento de los compromisos adquiridos en el anteproyecto. Se cuenta con la posibilidad de iterar estos momentos, de acuerdo a la retroalimentación emitida por el director del trabajo de grado.

Al producto final de software, se pretende realizar las pruebas necesarias para determinar su cumplimiento en términos de funcionalidad y calidad. Al soporte documental, su afinamiento a los preceptos ingenieriles y a la normatividad institucional; hasta obtener el visado del director del cumplimiento a cabalidad de los compromisos expuestos en el trabajo de grado en relación con la aplicación.

Es importante acotar, que el proceso metodológico para el desarrollo de la aplicación se dispone en cumplimiento a lo planteado en el anteproyecto, asimismo, de lo esbozado por las normas institucionales, respecto a la conducta de trabajo y la documentación y entregables correspondientes, todo ello certificado por el director del trabajo de grado, en consenso con el equipo de estudiantes.

ESTRATEGIAS DE TRABAJO: se identifican como fundamentos estratégicos demandados en la metodología MáTraGra, los siguientes:

- Trabajo colaborativo y en equipo: involucrando al director desde su perfil de cumplimiento, con responsabilidades definidas dentro del equipo, cuando se trata de más de un estudiante.
- Asignación de responsabilidades sistémica y holística: de acuerdo a las experiencias del equipo de trabajo.

- Cumplimiento de responsabilidades: determinada por la participación de los actores en la planificación y la validación final de los productos contemplados. Con repercusión académica de acuerdo con la normativa.
- Retroalimentación –feedback-: mediante la validación de cumplimiento por parte del directos, en el espacio de transición entre fases o iteraciones de los momentos de cada fase.
- Documentación concurrente, con el fin de dar cumplimiento a los requerimientos formales académico-administrativos de las instituciones.
- Bitácora del proceso: registro de las actividades planificadas y desarrolladas con la validación del director, como evidencia histórica del proceso.
- Pruebas programáticas –testing-: planeadas y con el soporte ingenieril de estándares de aceptación nacional o internacional, de responsabilidad parcial del equipo de estudiantes y final del director.

Autoría: Gustavo A. Rivera S (2018).

2.2.2 INGENIERÍA DE SOFTWARE

Las mejores definiciones de ingeniería de software pueden ser:

- Según la IEEE: “La aplicación tecnológica y científica de conocimiento, procesos y experiencia para diseñar, implementar, probar y documentar software” (The Bureau of Labor Statistics—IEEE Systems and software engineering)
- Según la IEEE: “La aplicación de métodos sistemáticos, disciplinados y cuantificables para el desarrollo, la operación y el mantenimiento de software” (IEEE Standard Glossary of Software Engineering Terminology)
- Según Ian Sommerville: “Una disciplina ingenieril que se ocupa de todos los aspectos relacionados con producción de software”
- Según Fritz Bauer: “El uso de principios de ingeniería para económicamente obtener software de confianza y que funciona eficientemente en computadores reales”
- Según Merriam Webster: “Una rama de la ciencia de la computación que se encarga del diseño, la implementación y el mantenimiento de complejos programas de computación”
- Según Google: “La ingeniería de software, no se refiere solamente al acto de escribir código, también se refiere a todas las herramientas y procesos que una

organización utiliza para para crear y mantener ese código durante todo su ciclo de vida. Ingeniería de software puede ser pensada como la programación integrada a través del tiempo”.

Para concluir, la ingeniería de software es el proceso de examinar y determinar las necesidades de los usuarios. Esto se hace desarrollando y diseñando, y al final probando el software para saber si suple las necesidades del usuario final. La ingeniería de software es usada para sistemas más grandes y complejos, los cuales son creados por organizaciones y negocios. Hay diferentes tipos de ingeniería de software. Esta el front-end, donde la parte visual de la aplicación es construida y Esta el back-end, en donde los ingenieros crean las partes que se encargan de recibir, almacenar y tratar la información.

2.2.3 COMPUTACIÓN EN LA NUBE

La computación en la nube es la disponibilidad a pedido de los recursos del sistema informático, especialmente el almacenamiento de datos y la capacidad de cómputo, sin una gestión activa directa por parte del usuario. El término se usa generalmente para describir los centros de datos disponibles desde cualquier lugar para muchos usuarios a través de Internet desde cualquier dispositivo móvil o fijo.

A menudo, el término «computación en la nube» se lo relaciona con una reducción de costos, disminución de vulnerabilidades y garantía de disponibilidad. Asimismo, la computación en la nube se la relaciona con un modelo de pago por uso. No obstante, el concepto de pago no puede ser solo relacionado con erogación económica dado que solo aplica en caso de proveedores externos, y en muchos casos hace referencia a poder medir el consumo aplicándose a centros de costos de la propia empresa.

La computación en la nube es un nuevo modelo de prestación de servicios tecnológicos que impacta sin lugar a duda en diversos negocios. Este modelo se apoya en infraestructuras tecnológicas dinámicas, caracterizados por la virtualización de recursos, un alto grado de automatización, una elevada capacidad de adaptación para atender demandas variables.

El concepto de «nube informática» es muy amplio, y abarca casi todos los posibles tipos de servicio en línea, pero cuando las empresas predicen ofrecer un utilitario alojado en la nube, por lo general se refieren a alguna de estas tres modalidades: el software como servicio (SaaS, por sus siglas en inglés), plataforma como servicio (PaaS) e infraestructura como servicio (IaaS).

El software como servicio es un modelo de distribución de software en el que las aplicaciones están alojadas por una compañía o proveedor de servicio y puestas a disposición de los usuarios a través de una red, generalmente internet. La plataforma como servicio es un conjunto de utilitarios para abastecer al usuario de sistemas operativos y servicios asociados a través de Internet sin necesidad de descargas o instalación alguna. Infraestructura como servicio se refiere a la tercerización de los equipos utilizados para apoyar las operaciones, incluido el almacenamiento, hardware, servidores y componentes de red.

2.2.4 BASES DE DATOS NOSQLⁱ

El termino NoSQL se refiere a la denominación en inglés Not Only SQL. Plantea modelos de datos específicos de esquemas flexibles que se adaptan a los requisitos de las aplicaciones más modernas (ACENS, 2021).

Las bases de datos NoSQL son sistemas de almacenamiento de información que no cumplen con el esquema entidad–relación. Tampoco utilizan una estructura de datos en forma de tabla donde se van almacenando los datos, sino que para el almacenamiento hacen uso de otros formatos (Grapheverywhere, 2021).

Esta forma de almacenar la información ofrece ciertas ventajas sobre los modelos relacionales. Entre las ventajas más significativas podemos destacar:

- Se ejecutan en máquinas con pocos recursos: Estos sistemas, a diferencia de los sistemas basados en SQL, no requieren de apenas computación, por lo que se pueden montar en máquinas de un coste más reducido.
- Escalabilidad horizontal: Para mejorar el rendimiento de estos sistemas simplemente se consigue añadiendo más nodos, con la única operación de indicar al sistema cuáles son los nodos que están disponibles.
- Pueden manejar gran cantidad de datos: Esto es debido a que utiliza una estructura distribuida, en muchos casos mediante tablas Hash.
- No genera cuellos de botella: El principal problema de los sistemas SQL es que necesitan transcribir cada sentencia para poder ser ejecutada, y cada sentencia compleja requiere además de un nivel de ejecución aún más complejo, lo que constituye un punto de entrada en común, que ante muchas peticiones puede ralentizar el sistema.

Algunas de las diferencias más destacables que nos podemos encontrar entre los sistemas NoSQL y los sistemas SQL están:

- No utilizan SQL como lenguaje de consultas. La mayoría de las bases de datos NoSQL evitan utilizar este tipo de lenguaje o lo utilizan como un lenguaje de apoyo. Por poner algunos ejemplos, Cassandra utiliza el lenguaje CQL, MongoDB utiliza JSON o BigTable hace uso de GQL.

- No utilizan estructuras fijas como tablas para el almacenamiento de los datos. Permiten hacer uso de otros tipos de modelos de almacenamiento de información como sistemas de clave–valor, objetos o grafos.
- No suelen permitir operaciones JOIN. Al disponer de un volumen de datos tan extremadamente grande suele resultar deseable evitar los JOIN. Esto se debe a que, cuando la operación no es la búsqueda de una clave, la sobrecarga puede llegar a ser muy costosa. Las soluciones más directas consisten en desnormalizar los datos, o bien realizar el JOIN mediante software, en la capa de aplicación.
- Arquitectura distribuida. Las bases de datos relacionales suelen estar centralizadas en una única máquina o bien en una estructura máster–esclavo, sin embargo, en los casos NoSQL la información puede estar compartida en varias máquinas mediante mecanismos de tablas Hash distribuidas.

Dependiendo de la forma en la que almacenen la información, nos podemos encontrar varios tipos distintos de bases de datos NoSQL:

- Bases de datos clave – valor: Son el modelo de base de datos NoSQL más popular, además de ser la más sencilla en cuanto a funcionalidad. En este tipo de sistema, cada elemento está identificado por una llave única, lo que permite la recuperación de la información de forma muy rápida, información que habitualmente está almacenada como un objeto binario (BLOB). Se caracterizan por ser muy eficientes tanto para las lecturas como para las escrituras. Algunos ejemplos de este tipo son Cassandra, BigTable o HBase.
- Bases de datos documentales: Este tipo almacena la información como un documento, generalmente utilizando para ello una estructura simple como JSON o XML y donde se utiliza una clave única para cada registro. Este tipo de implementación permite, además de realizar búsquedas por clave–valor, realizar consultas más avanzadas sobre el contenido del documento.
Son las bases de datos NoSQL más versátiles. Se pueden utilizar en gran cantidad de proyectos, incluyendo muchos que tradicionalmente funcionarían sobre bases de datos relacionales. Algunos ejemplos de este tipo son MongoDB o CouchDB.
- Bases de datos en grafo: En este tipo de bases de datos, la información se representa como nodos de un grafo y sus relaciones con las aristas del mismo, de manera que se puede hacer uso de la teoría de grafos para recorrerla. Para sacar el máximo rendimiento a este tipo de bases de datos, su estructura debe estar totalmente normalizada, de forma que cada tabla tenga una sola columna y cada relación dos.
Este tipo de bases de datos ofrece una navegación más eficiente entre relaciones que en un modelo relacional. Algunos ejemplos de este tipo son Neo4j, InfoGrid o Virtuoso.

- **Bases de datos orientadas a objetos:** En este tipo, la información se representa mediante objetos, de la misma forma que son representados en los lenguajes de programación orientada a objetos (POO) como ocurre en JAVA, C# o Visual Basic .NET. Algunos ejemplos de este tipo de bases de datos son Zope, Gemstone o Db4o.

2.3 MARCO CONCEPTUAL

2.3.1 APLICACIÓN WEB

Una aplicación web, es un programa que está almacenado en un servidor remoto y desplegado en internet a través de la interfaz de los navegadores web. Estas son consideradas como servicios web y la gran mayoría, aunque no todas las páginas web, son aplicaciones web. Para que una página web sea considerada aplicación web, esta debe ejecutar alguna función en tiempo real por su usuario (llenar un formulario, crear información nueva, etc.) Las aplicaciones web son diseñadas para una gran variedad de usos y pueden ser usadas por cualquier persona con acceso a un navegador web. Los tipos de aplicaciones web más comunes son: aplicaciones e-mail (Gmail, Outlook, Yahoo!), calculadoras online o tiendas e-commerce. Algunas aplicaciones web solo pueden ser accedidas desde navegadores web específicos, sin embargo, la mayoría son accesibles desde todos los navegadores web.

2.3.2 AWS

Amazon Web Services (AWS abreviado) es una colección de servicios de computación en la nube pública (también llamados servicios web) que en conjunto forman una plataforma de computación en la nube, ofrecidas a través de Internet por Amazon.com. Es usado en aplicaciones populares como Dropbox, Foursquare, HootSuite. Es una de las ofertas internacionales más importantes de la computación en la nube y compite directamente contra servicios como Microsoft Azure, Google Cloud Platform y IBM Cloud. Es considerado como un pionero en este campo

2.3.3 AWS LAMBDA

Con AWS Lambda, puede ejecutar código sin aprovisionar ni administrar servidores. Solo paga por el tiempo de computación que consume, no se aplica ningún cargo si el código no se encuentra en ejecución. Puede ejecutar código para casi cualquier tipo de aplicación o servicio Backend sin tener que realizar tareas de administración. Solo tiene que cargar el código y Lambda se encargará de todo lo necesario para ejecutar y escalar el código con alta disponibilidad. Puede configurar el código para que se active automáticamente desde otros servicios de AWS o puede llamarlo directamente desde cualquier aplicación web o móvil.

2.3.4 AWS API GATEWAY

Amazon API Gateway es un servicio completamente administrado que facilita a los desarrolladores la creación, la publicación, el mantenimiento, el monitoreo y la protección de API a cualquier escala. Las API actúan como la "puerta de entrada" para que las aplicaciones accedan a los datos, la lógica empresarial o la funcionalidad de sus servicios de Backend.

Con API Gateway, puede crear API RESTful y API WebSocket que permiten aplicaciones de comunicación bidireccional en tiempo real. API Gateway admite cargas de trabajo en contenedores y sin servidor, así como aplicaciones web.

API Gateway gestiona todas las tareas implicadas en la aceptación y el procesamiento de hasta cientos de miles de llamadas a API simultáneas, entre ellas, la administración del tráfico, compatibilidad con CORS, el control de autorizaciones y acceso, la limitación controlada, el monitoreo y la administración de versiones de API. API Gateway no requiere pagos mínimos ni costos iniciales.

Se paga por las llamadas a las API que se reciben y por la cantidad de datos salientes transferidos; además, con el modelo de precios por niveles de API Gateway, puede reducir sus costos a medida que cambie la escala de uso de las API.

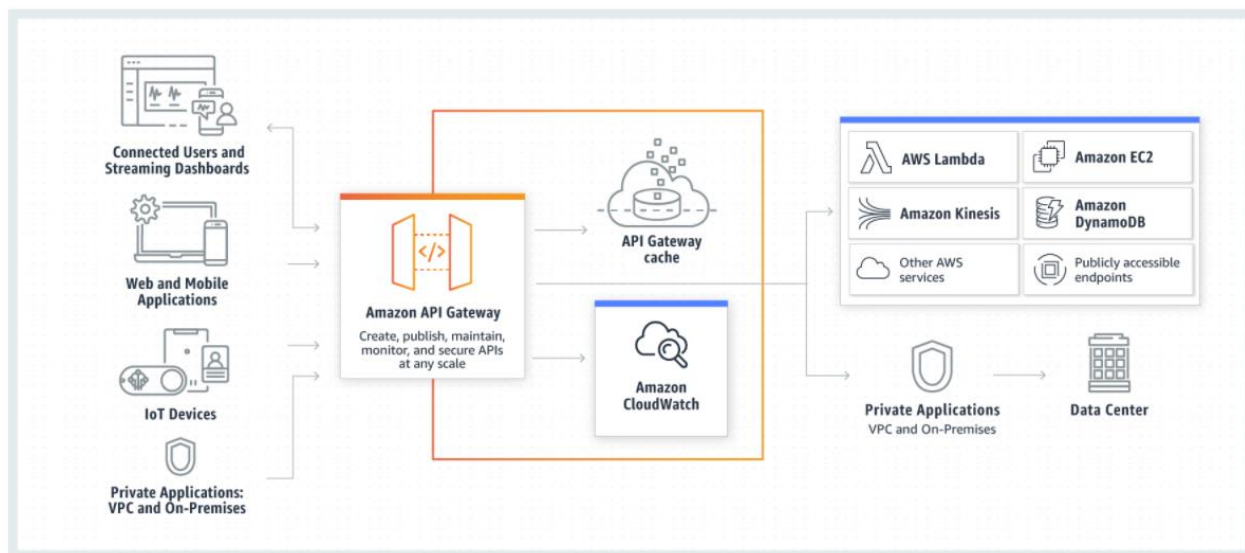


Ilustración 3: funcionamiento de API Gateway, fuente: AWS Documentation (Amazon, 2021).

2.3.5 AWS DYNAMODB

Amazon DynamoDB es una base de datos NoSQL de clave-valor y documentos que ofrece

rendimiento en milisegundos de un solo dígito a cualquier escala. Se trata de una base de datos completamente administrada, duradera, multiactiva y de varias regiones que cuenta con copia de seguridad, restauración y seguridad integradas, así como almacenamiento de caché en memoria para aplicaciones a escala de Internet. DynamoDB puede gestionar más de 10 billones de solicitudes por día y puede admitir picos de más de 20 millones de solicitudes por segundo.

Muchos de los negocios del mundo con un crecimiento más rápido, como Lyft, Airbnb y Redfin, así como compañías como Samsung, Toyota y Capital One, utilizan la escala y el rendimiento de DynamoDB para ofrecer soporte a sus cargas de trabajo fundamentales.

Cientos de miles de clientes de AWS han elegido DynamoDB como su base de datos de claves-valor y documentos para aplicaciones móviles, web, juegos, tecnología publicitaria e IoT, entre otras, que necesitan acceso a datos con baja latencia a cualquier escala.

2.3.6 AWS S3

Amazon Simple Storage Service (Amazon S3) es un servicio de almacenamiento de objetos que ofrece escalabilidad, disponibilidad de datos, seguridad y rendimiento líderes en el sector. Gracias a Amazon S3, clientes de todos los tipos y sectores pueden almacenar y proteger cualquier volumen de datos para los más variados fines, como usarlos en lagos de datos, sitios web, aplicaciones móviles, procesos de copia de seguridad y restauración, operaciones de archivado, aplicaciones empresariales, dispositivos IoT y análisis de big data. Amazon S3 proporciona características de administración fáciles de utilizar que le permiten organizar los datos y configurar sofisticados controles de acceso con objeto de satisfacer sus requisitos empresariales, organizativos y de conformidad. Amazon S3 está diseñado para ofrecer una durabilidad del 99,999999999 % (11 nueves) y almacena datos de millones de aplicaciones para empresas de todo el mundo.

2.3.7 AWS CLOUDWATCH

Amazon CloudWatch es un servicio de monitorización y observación creado para ingenieros de DevOps, desarrolladores, ingenieros de fiabilidad de sitio (SRE) y administradores de TI. CloudWatch ofrece datos e información procesable para monitorizar sus aplicaciones, responder a cambios de rendimiento que afectan a todo el sistema, optimizar el uso de recursos y lograr una vista unificada del estado de las operaciones. CloudWatch recopila datos de monitorización y operaciones en formato de registros, métricas y eventos, lo cual ofrece una vista unificada de los recursos, las aplicaciones y los servicios de AWS que se ejecutan en servidores locales y de AWS. Puede usar CloudWatch para detectar comportamientos anómalos en sus entornos, definir alarmas, comparar registros y métricas, realizar acciones automatizadas, resolver problemas y descubrir información para mantener sus aplicaciones en ejecución sin problemas.

El funcionamiento de CloudWatch recopila datos de monitorización y operaciones en formato de registros, métricas eventos, y permite su visualización mediante paneles automatizados para

obtener una vista unificada de los recursos, las aplicaciones y los servicios de AWS que se ejecutan en servidores locales y de AWS. Estas métricas y registros se pueden correlacionar para comprender mejor el estado y el rendimiento de los recursos. También se pueden crear alarmas según los umbrales de valores de métricas especificados o que detecten comportamientos de métricas anómalos en función de algoritmos de aprendizaje automático. Para responder de forma rápida mediante acciones, es posible configurar acciones automatizadas para que se emita una notificación cuando se active una alarma e iniciar automáticamente Auto Scaling, por ejemplo, para reducir el tiempo medio de resolución. También tiene la posibilidad de profundizar en sus métricas, registros y rastreos y analizarlos para comprender mejor cómo mejorar el rendimiento de las aplicaciones.

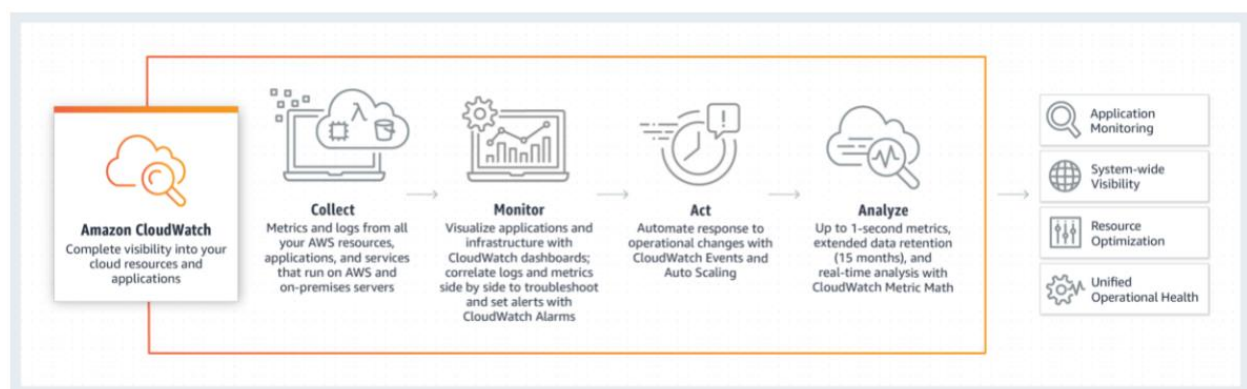


Ilustración 4: funcionamiento de AWS CloudWatch, fuente: AWS Documentation (Amazon, 2021).

3. ASPECTOS ADMINISTRATIVOS

3.2 RECURSOS

RECURSO HUMANO

RUBRO	DESCRIPCIÓN
Horas de trabajo	Tiempo de trabajo de los autores

Ilustración 5. Tabla recursos humanos

RECURSO LOGÍSTICO

RUBRO	DESCRIPCIÓN	COSTE	UNIDAD
AWS	Cuenta de AWS con presupuesto	\$ 30.00	USD

Ilustración 6. Tabla recursos logísticos

RECURSO TECNOLÓGICO

RUBRO	DESCRIPCIÓN	CANTIDAD	COSTE	UNIDAD
Equipos para desarrollo	computadores	2	\$ 1,800,000.00	COP
Total			\$ 3,600,000.00	COP

Ilustración 7. Tabla recursos tecnologicos

RECURSO ADMINISTRATIVO

RUBRO	DESCRIPCIÓN	COSTE	UNIDAD
Papelería	Impresión de documentos	\$ 50,000.00	COP
Papelería	Poster del trabajo de grado	\$ 30,000.00	COP
Total		\$ 80,000.00	COP

Ilustración 8. Tabla recursos administrativos

3.3 CRONOGRAMA

DESCRIPCIÓN	DEPENDENCIAS	PRODUCTO	FECHA DE INICIO	FECHA FIN
- FASE 1 CONTEXTUALIZACIÓN DEL PROYECTO:				
Subtarea 1: Preparación e investigación del levantamiento del arte	N/A	Estado del arte	22/02/2021	24/02/2021
Subtarea 2: Realización del documento del estado del arte	N/A	Estado del arte	24/03/2021	3/3/2021
Subtarea 3: Aprobación del documento del estado del arte	N/A	Estado del arte	3/3/2021	6/3/2021
Subtarea 4: Diseño general de la aplicación	N/A	Estado del arte	6/3/2021	13/03/2021
Subtarea 5: Validación del diseño	N/A	Estado del arte	14/03/2021	17/03/2021
- FASE 2: DESARROLLO				
Subtarea 1: Diseño del software	Validación del diseño	Marco Teórico	18/03/2021	22/03/2021
Subtarea 2: Validación del diseño del software	Aprobación del documento	Marco Teórico	23/03/2021	26/03/2021

Subtarea 3: Desarrollo del software	Validación del diseño del software	Marco Teórico	24/03/2021	11/5/2021
Subtarea 4: Pruebas de uso del software	Desarrollo del software	Aplicación	12/5/2021	19/05/2021
- FASE 3: ACEPTACIÓN				
Subtarea 1: Pruebas de aceptación	Desarrollo del software	Resultados (Documento)	20/05/2021	27/05/2021
- FASE 4: CONCLUSIÓN DEL PROYECTO				
Subtarea 1: Construcción del trabajo de grado	Implementación del protocolo de evaluación	Documento de grado	21/05/2021	28/05/2021
Subtarea 2: Aprobación del trabajo de grado	Construcción del trabajo de grado	Documento de grado	29/05/2021	5/6/2021

Ilustración 9. cronograma de actividades

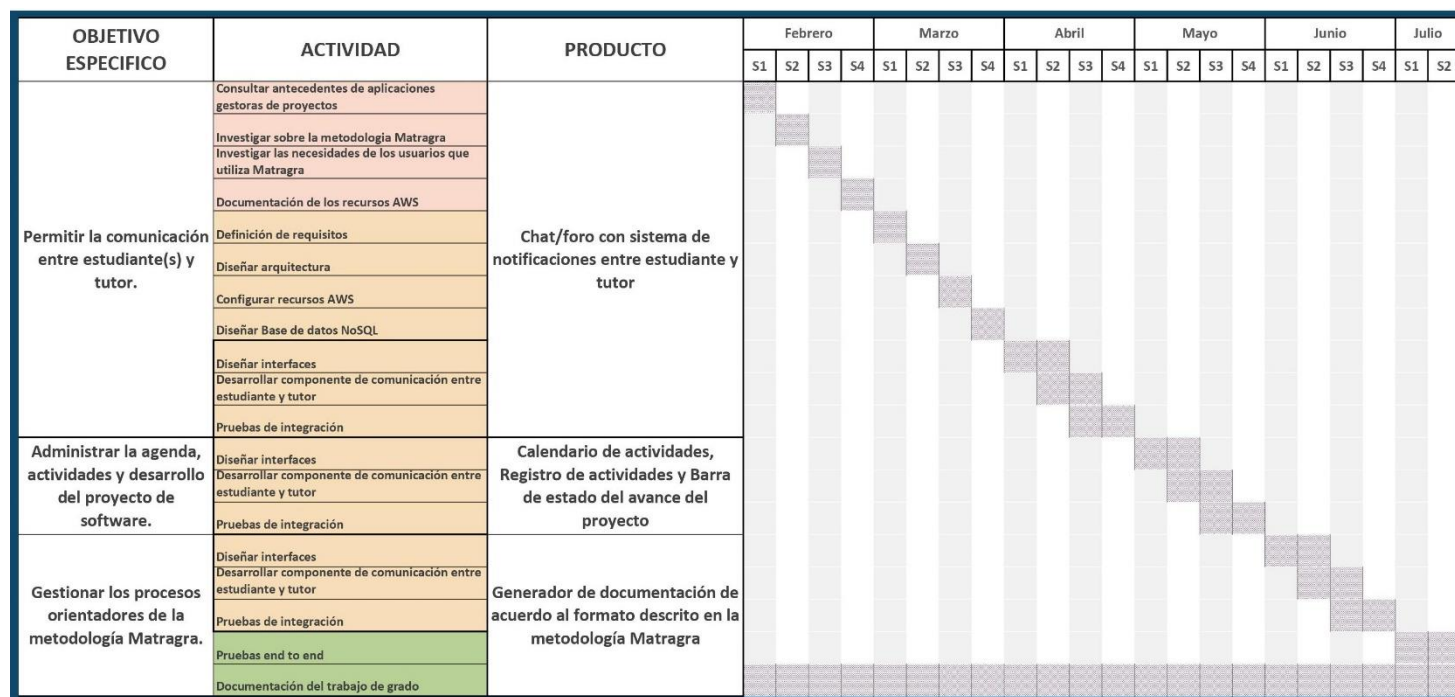


Ilustración 10. planificación de actividades

4. INGENIERIA DEL PROYECTO

4.1 MODELO DE NEGOCIO

4.1.1 DIAGRAMA HIPO

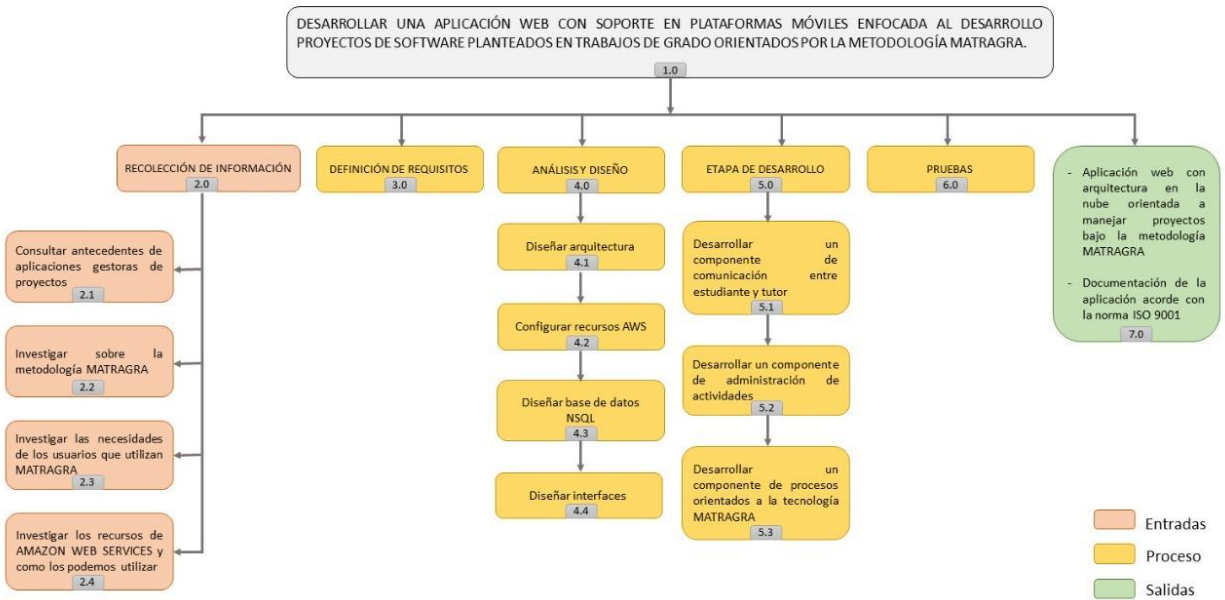


Ilustración 11: Diagrama HIPO, fuente: elaboración propia

4.2 MODELO CANVAS

MODELO DE NEGOCIOS CANVAS

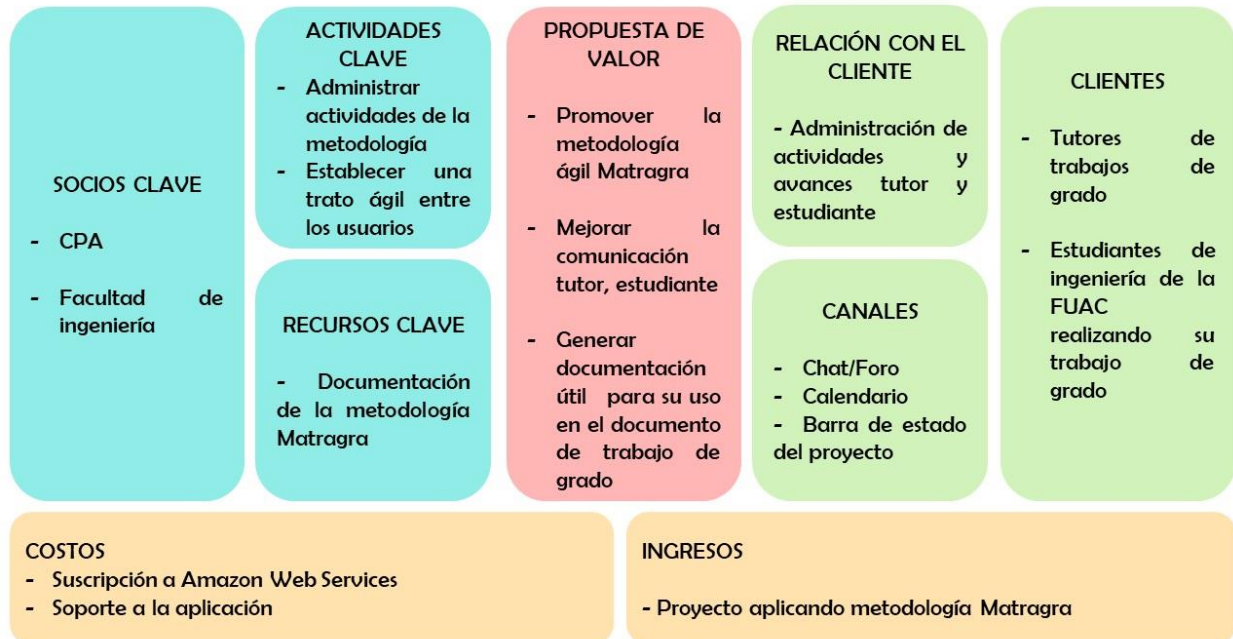


Ilustración 12 tabla modelo de negocio Canvas, fuente: elaboración propia

4.2 METODOLOGIA DE DISEÑO

4.2.1 MODELO DE CASOS DE USO

A continuación se muestran los diagramas de casos de uso y los casos de uso asociados.

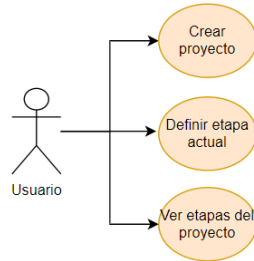
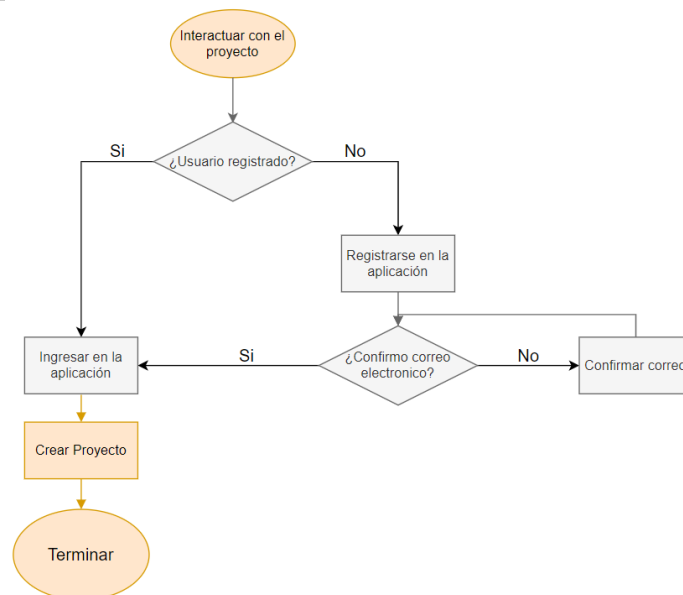


Ilustración 13 Caso de uso interactuar con el proyecto

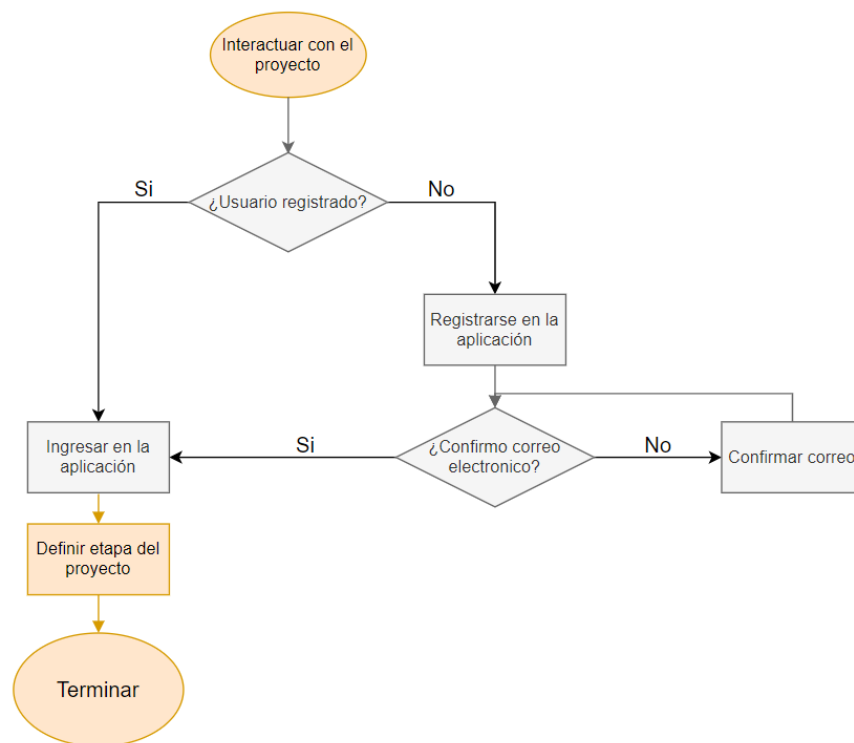
ID:	CU-01	Nombre:	Crear con el proyecto	
Fecha:	10/4/2021	Versión:		1
Descripción:	Permite al usuario Crear un proyecto en la aplicación.			
Actores:	Tutor y estudiantes.			
Pre-Condiciones:	<ul style="list-style-type: none"> • El usuario debe estar previamente registrado. • El usuario debe confirmar su correo electrónico. • Dependiendo la opción el usuario debe contar con un rol valido de lo contrario no se mostrará la opción. 			

Flujo básico de éxito



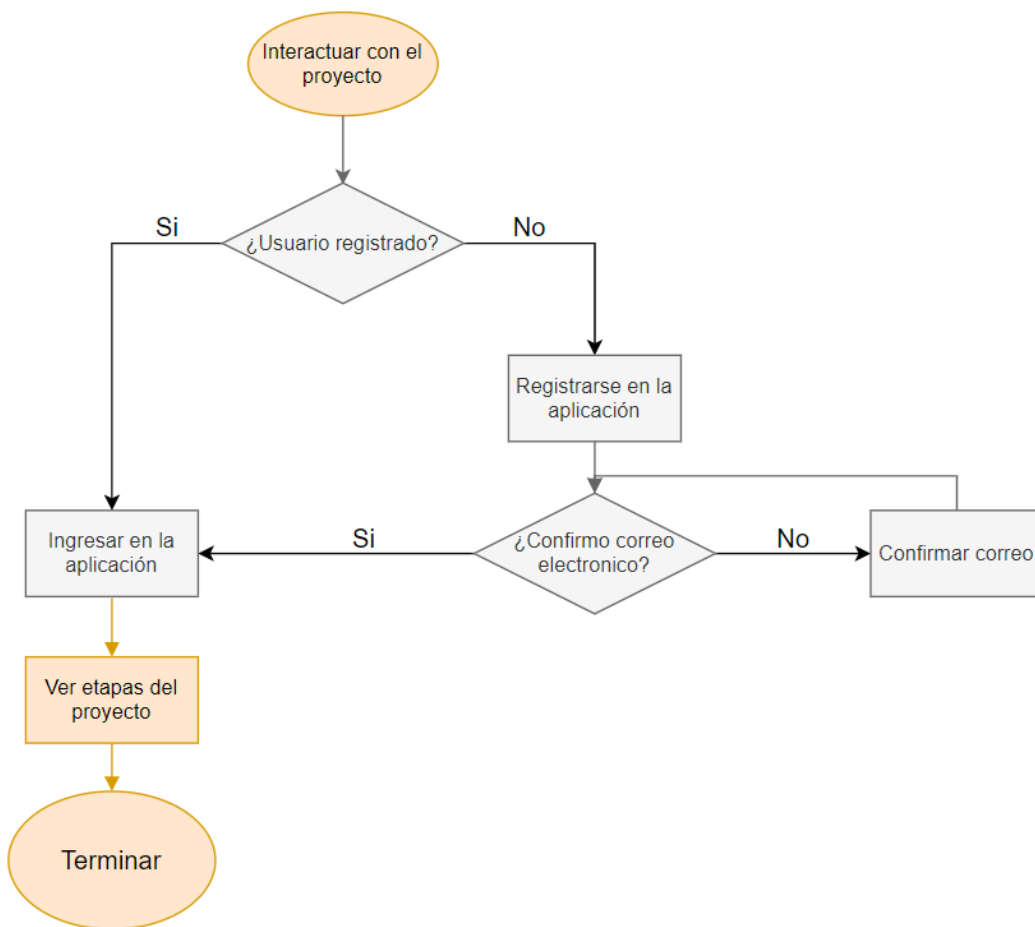
ID:	CU-01.1	Nombre:	Definir etapa del proyecto	
Fecha:	10/4/2021	Versión:		1
Descripción:	Permite al usuario definir etapa del proyecto en la aplicación.			
Actores:	Tutor y estudiantes.			
Pre-Condiciones:	<ul style="list-style-type: none"> • El usuario debe estar previamente registrado. • El usuario debe confirmar su correo electrónico. • Dependiendo la opción el usuario debe contar con un rol valido de lo contrario no se mostrará la opción. 			

Flujo básico de éxito



ID:	CU-01.2	Nombre:	Ver etapas del proyecto	
Fecha:	10/4/2021	Versión:		1
Descripción:	Permite al usuario ver etapas del proyecto en la aplicación.			
Actores:	Tutor y estudiantes.			
Pre-Condiciones:	<ul style="list-style-type: none"> • El usuario debe estar previamente registrado. • El usuario debe confirmar su correo electrónico. • Dependiendo la opción el usuario debe contar con un rol valido de lo contrario no se mostrará la opción. 			

Flujo básico de éxito



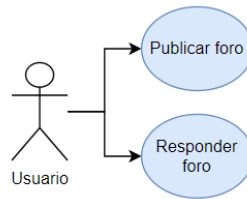
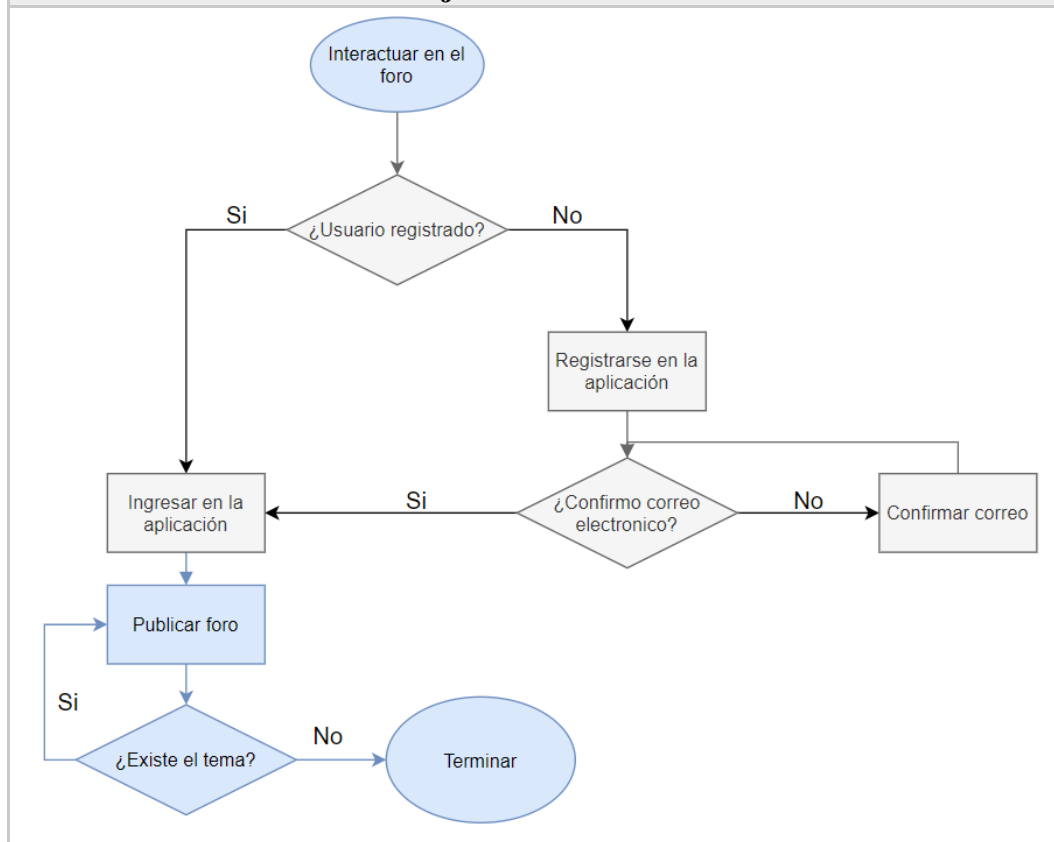


Ilustración 14 Caso de uso interactuar en el foro

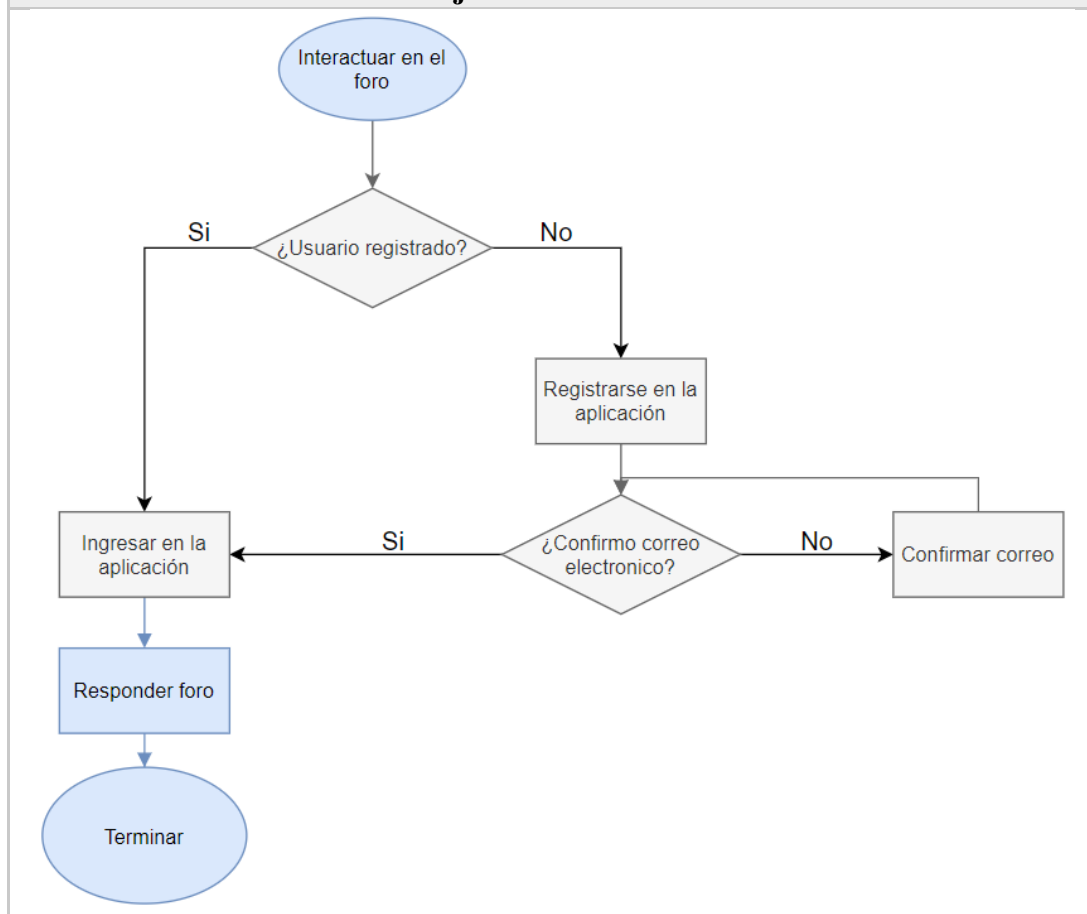
ID:	CU-02	Nombre:	Interactuar en el foro	
Fecha:	10/4/2021	Versión:	1	
Descripción:	Permite al usuario interactuar en el foro de la aplicación.			
Actores:	Tutor y estudiantes.			
Pre-Condiciones:	<ul style="list-style-type: none"> • El usuario debe estar previamente registrado. • El usuario debe confirmar su correo electrónico. 			

Flujo básico de éxito



ID:	CU-02.1	Nombre:	Responder en el foro	
Fecha:	10/4/2021	Versión:	1	
Descripción:	Permite al usuario responder en el foro de la aplicación.			
Actores:	Tutor y estudiantes.			
Pre-Condiciones:	<ul style="list-style-type: none"> • El usuario debe estar previamente registrado. • El usuario debe confirmar su correo electrónico. 			

Flujo básico de éxito



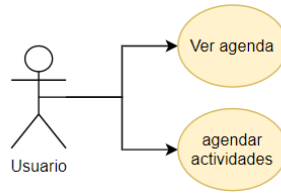
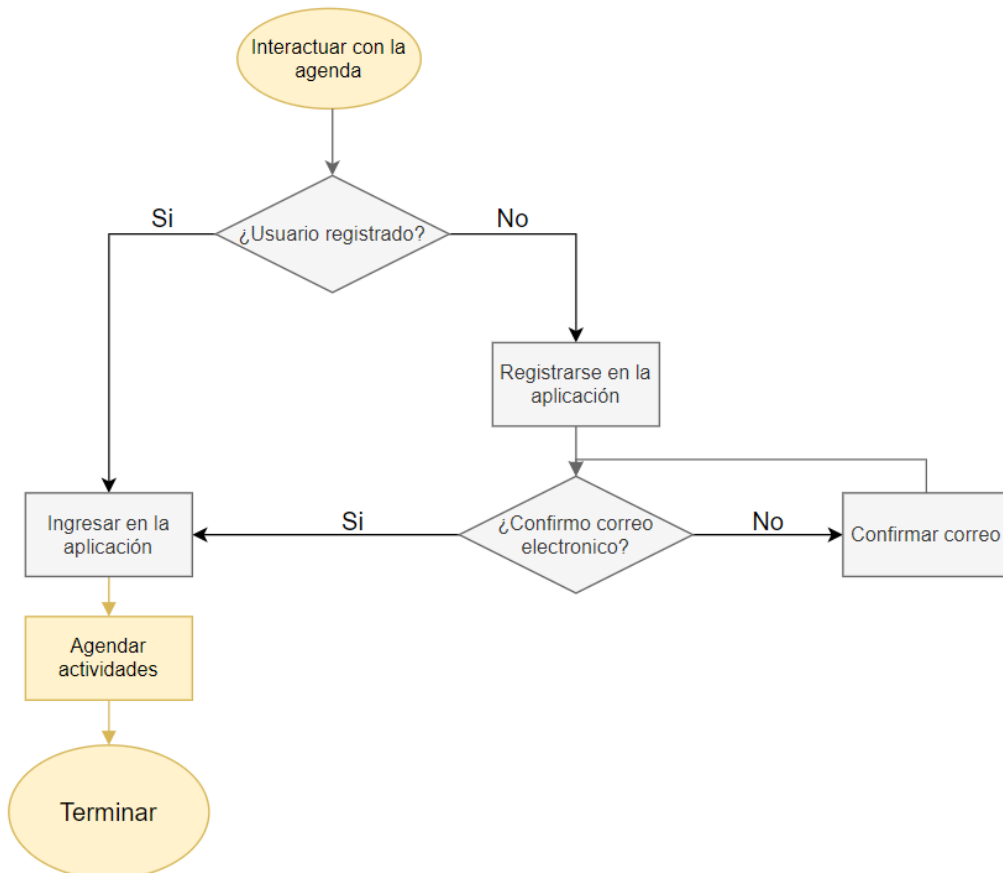


Ilustración 15 Caso interactuar con la agenda

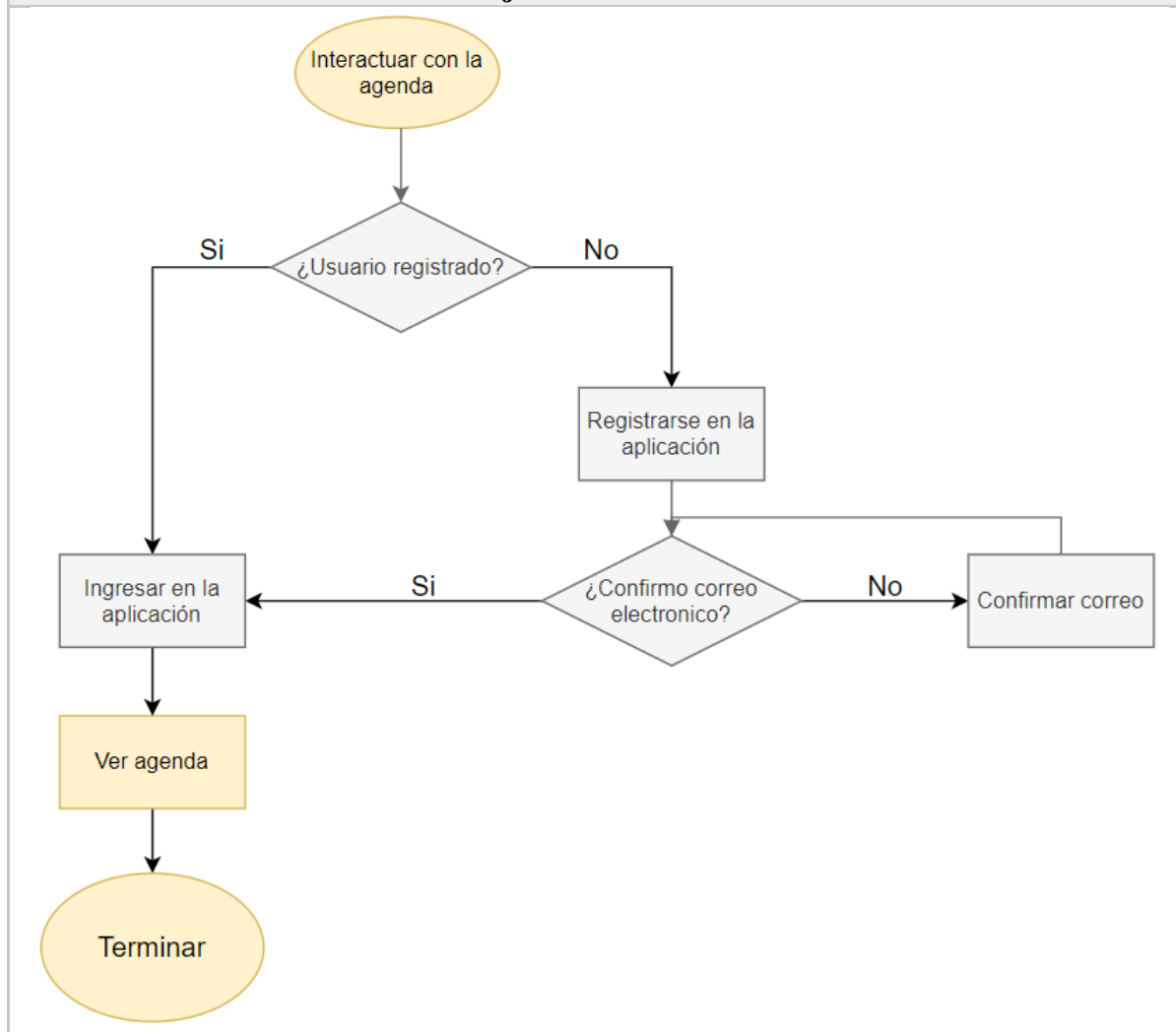
ID:	CU-03	Nombre:	Interactuar con la agenda	
Fecha:	10/4/2021	Versión:		1
Descripción:	Permite al usuario interactuar con la agenda de la aplicación.			
Actores:	Tutor y estudiantes.			
Pre-Condiciones:	<ul style="list-style-type: none"> • El usuario debe estar previamente registrado. • El usuario debe confirmar su correo electrónico. 			

Flujo básico de éxito



ID:	CU-03.1	Nombre:	Ver la agenda	
Fecha:	10/4/2021		Versión:	1
Descripción:	Permite al usuario ver la agenda de la aplicación.			
Actores:	Tutor y estudiantes.			
Pre-Condiciones:	<ul style="list-style-type: none"> • El usuario debe estar previamente registrado. • El usuario debe confirmar su correo electrónico. 			

Flujo básico de éxito



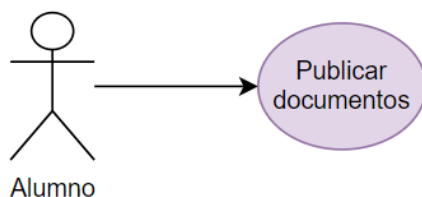
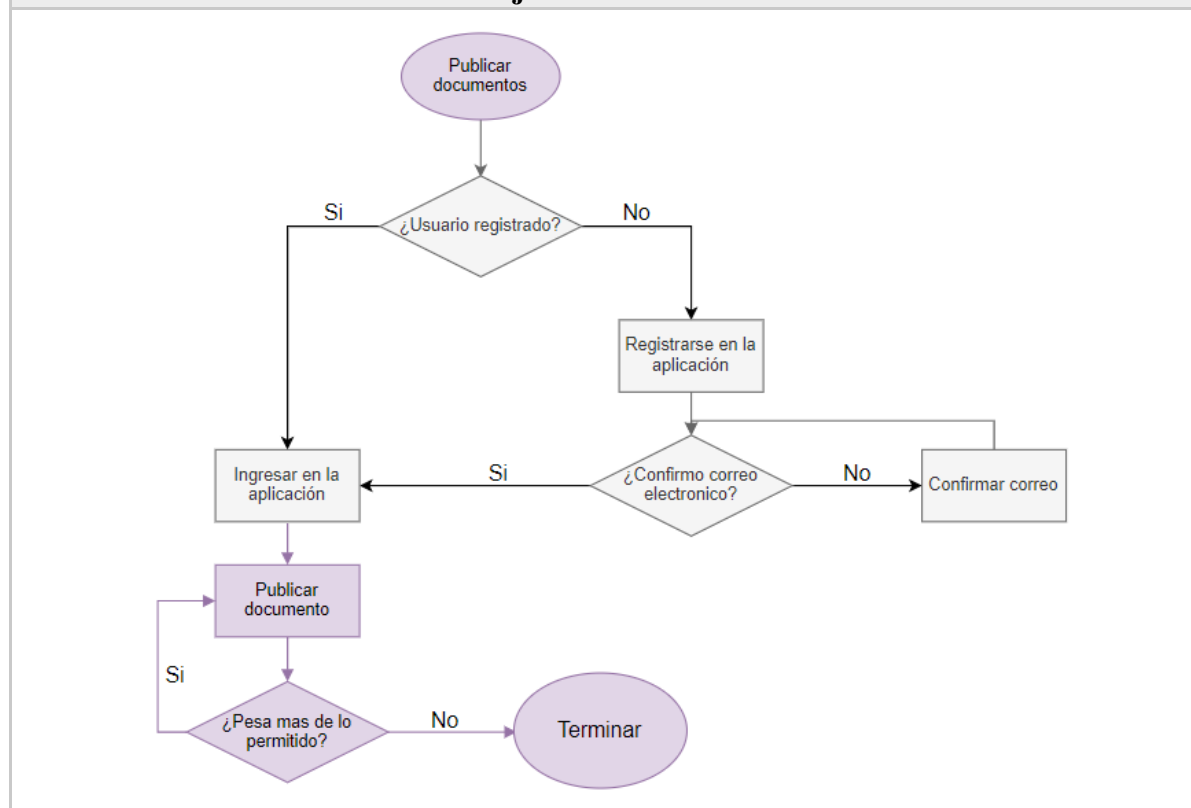


Ilustración 16 Caso de uso publicar documentos

ID:	CU-04	Nombre:	Publicar documentos	
Fecha:	10/4/2021	Versión:		1
Descripción:	Permite al usuario publicar documentos del proyecto o de referencias para el mismo en la aplicación.			
Actores:	Tutor y estudiantes.			
Pre-Condiciones:	<ul style="list-style-type: none"> • El usuario debe estar previamente registrado. • El usuario debe confirmar su correo electrónico. 			

Flujo básico de éxito



4.2.2 DISEÑO DE BASE DE DATOS

Para el diseño de las tablas y los modelos en las mismas se uso como referencia la guía de AWS de mejores prácticas en DynamoDB.

<https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/best-practices.html>

En DynamoDB se utiliza la clave primaria compuesta (primary key), en la cual especificas una clave de partición (pk) y una clave de ordenamiento (sk). La clave de ordenamiento (sk) se usa para ordenar elementos en la misma partición.

A continuación, se va a muestra cómo fueron diseñadas las 4 tablas del proyecto y se especificara los pk y sk utilizados en cada una de ellas:

pk	sk	Attributes	
PROJECT	EMAIL	password	roleName
PROJECT	USER		

pk	sk	Attributes				
PROJECT	EVENT	date	detail	duration	subject	time

pk	sk	Attributes							
PROJECT	FORO	answers	description	files	phase	phase0			
PROJECT	LIST						phase1	phase2	phase3
								general	

pk	sk	Attributes			
PROJECT	STATE	index			
PROJECT	PHASE		objectives	scope	task

Ilustración 17 Diseño de tablas de base de datos

4.2.3 DISEÑO DE ARQUITECTURA DE LA APLICACIÓN

Para el diseño de la arquitectura de la aplicación escogimos una arquitectura cien por ciento cloud, serverless que, aunque inicialmente es pequeña permite una escalabilidad tanto horizontal como vertical.

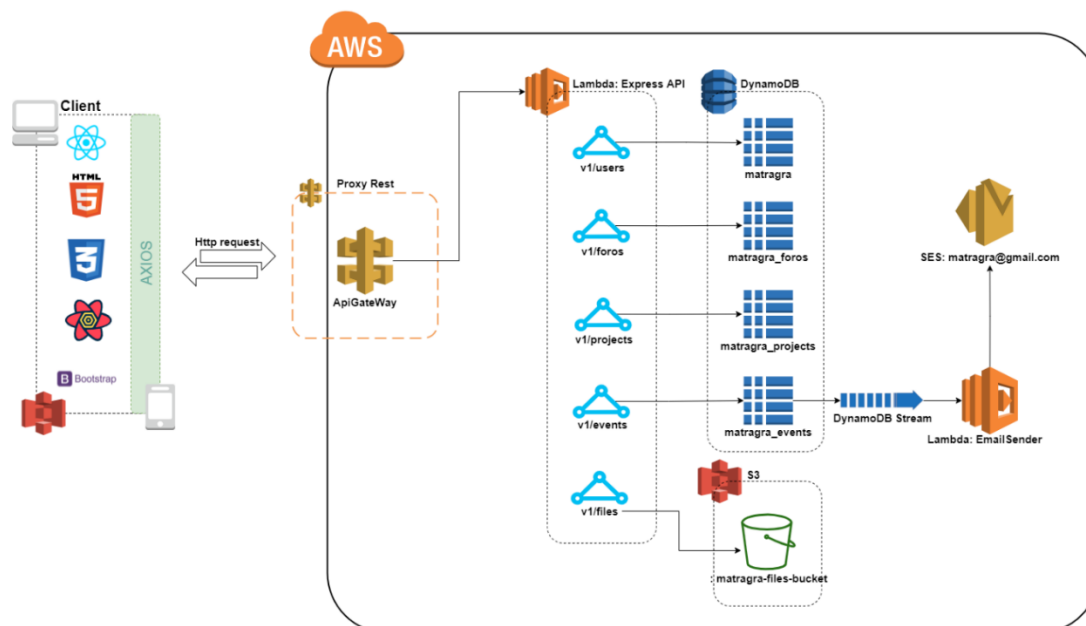


Ilustración 18 Diagrama de arquitectura, fuente: elaboración propia.

Todos los recursos de infraestructura se desplegaron en AWS utilizando IAC (Infrastructure As Code) mediante la definición y posterior carga de los componentes como un archivo en formato YAML utilizando la tecnología de CloudFormation lo que facilita la replicación en diferentes ambientes y la implementación de la infraestructura en flujos de despliegue e integración continua.

A continuación, los diagramas generados en la carga del archivo con la definición de los componentes de infraestructura.

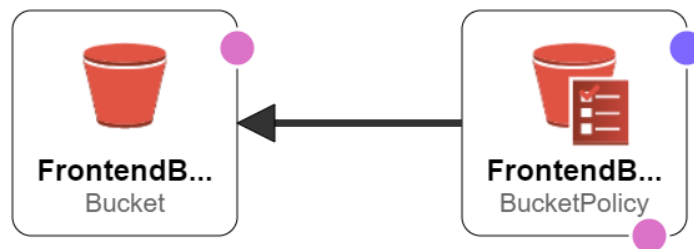


Ilustración 19 Diagrama generado de infraestructura frontend, fuente: elaboración propia.

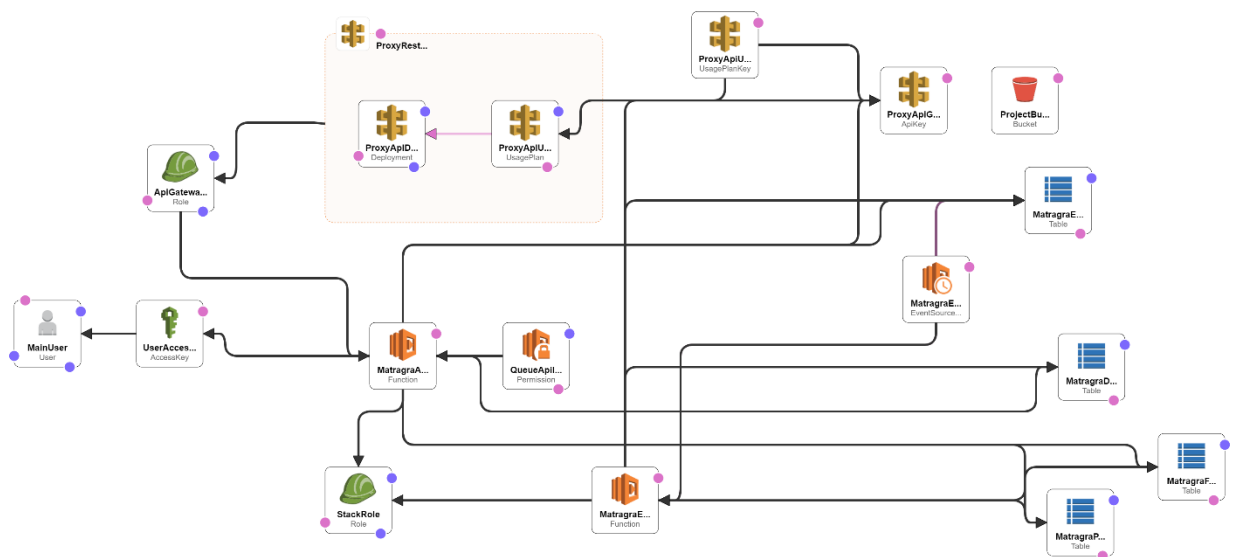


Ilustración 20 Diagrama de infraestructura backend generado, fuente: elaboración propia.

4.2.4 DISEÑO DE INTERFACES

En esta parte se encuentra el diseño de las interfaces de la aplicación tanto para web como móvil; además del mapa de navegación.

La siguiente imagen muestra el mapa de navegación de la aplicación web:

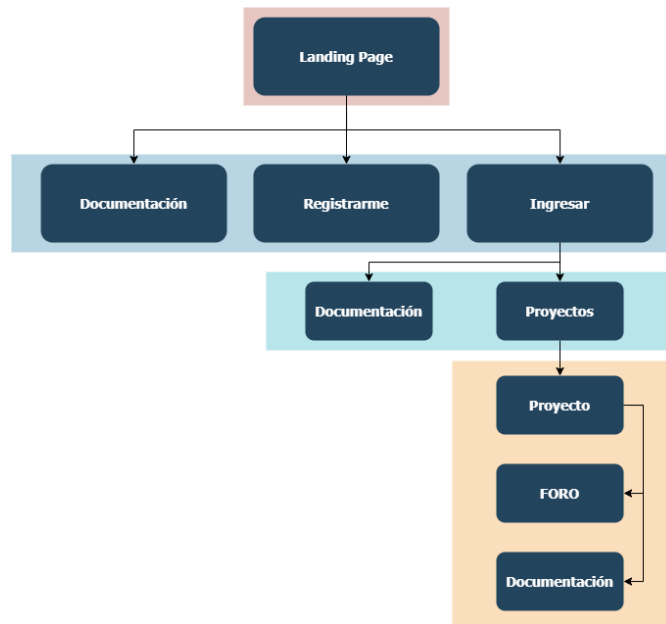


Ilustración 21 Mapa de navegación, fuente: elaboración propia.

Inicialmente el boceto de la funcionalidad era este:

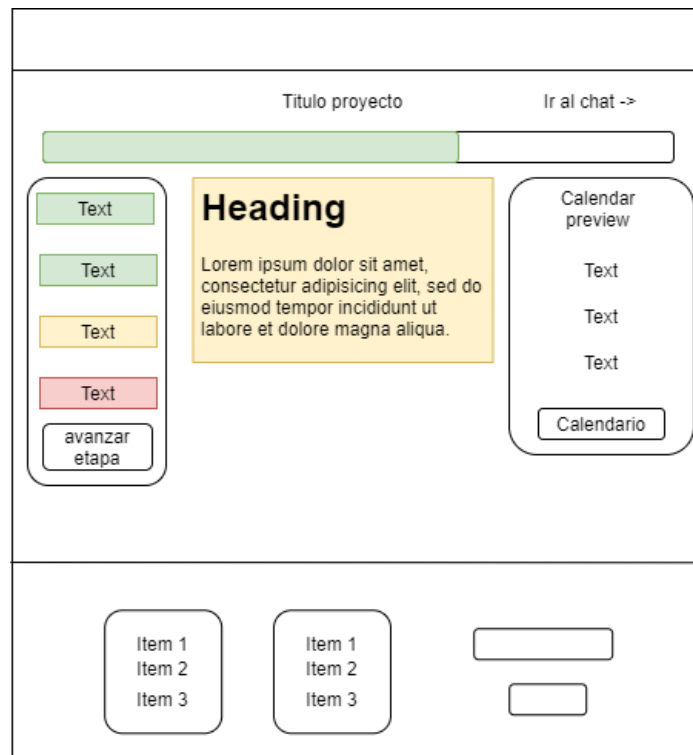


Ilustración 22 Boceto Diseño de interfaz de proyecto

A continuación, se muestran las interfaces web:

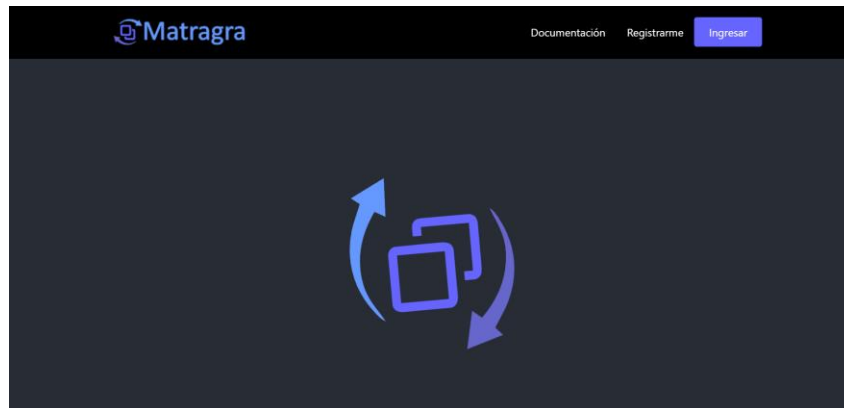


Ilustración 23 Interfaz landing page



Ilustración 24 Interfaz Documentación

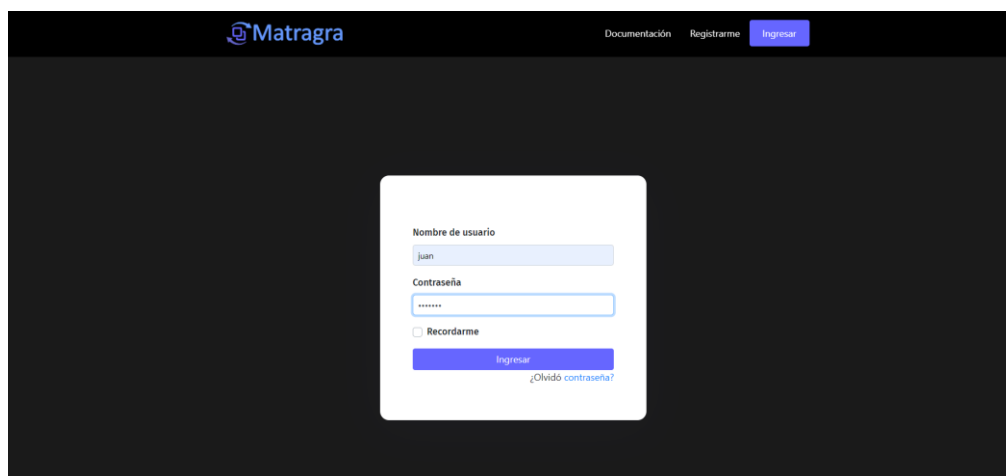


Ilustración 25 Interfaz Ingresar

Ilustración 26 Interfaz Registrarme



Ilustración 27 Interfaz Proyectos



Ilustración 28 Interfaz Proyecto



Ilustración 29 Interfaz Foro

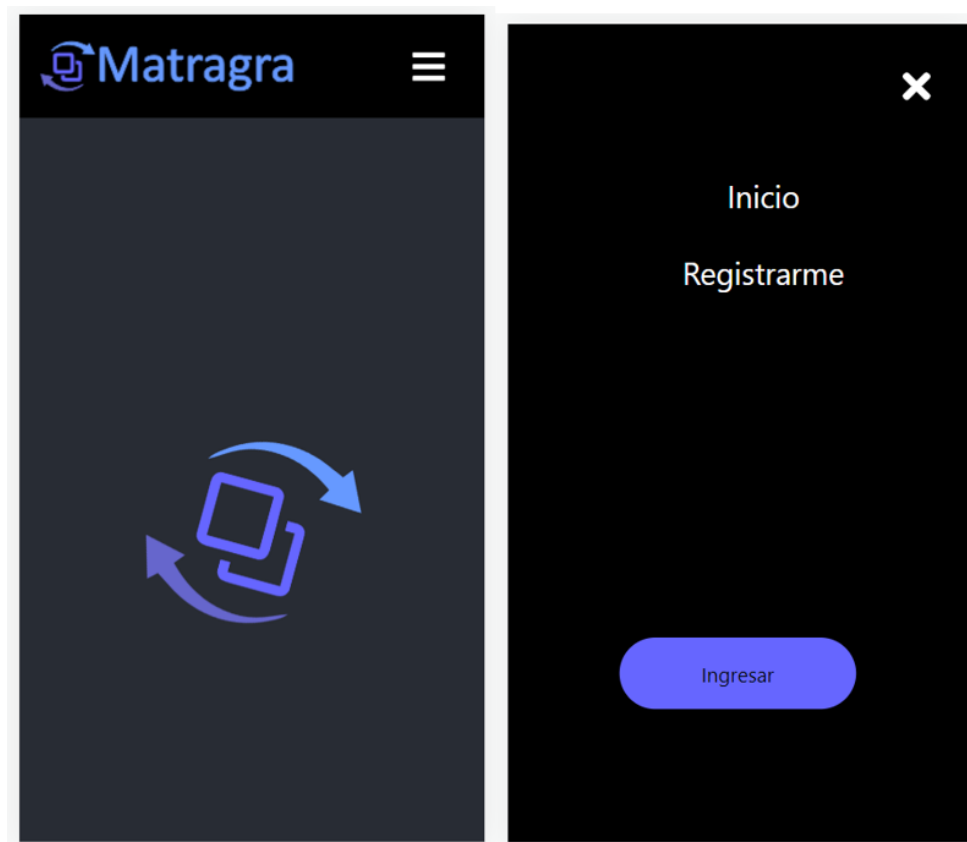


Ilustración 30 Landing page móvil

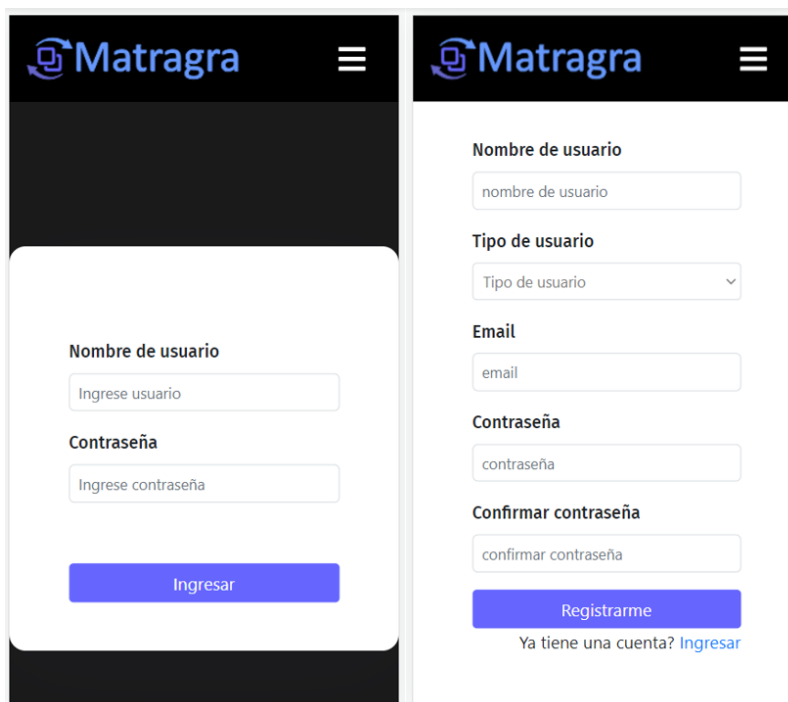


Ilustración 31 Interfaz Ingresar y Registrarme móvil



Ilustración 32 Interfaz Proyectos y Proyecto móvil

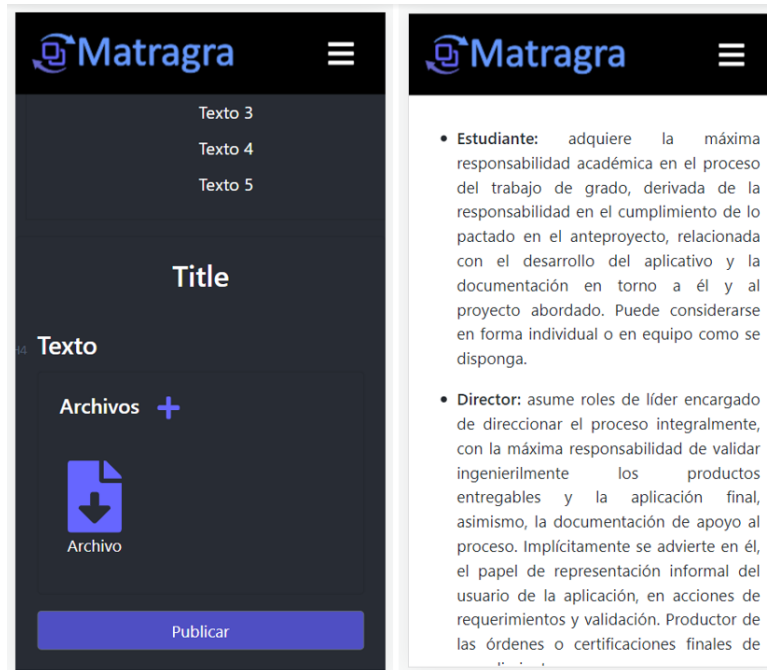


Ilustración 33 Interfaz Foro y Documentación móvil.

5. PRUEBAS

Para las pruebas de la aplicación se utilizaron herramientas automatizadas como Google Lighthouse que generan reporte de la aplicación en diferentes áreas como performance, mejores prácticas y accessibility y seguridad, basado en el OWASP top 10, entre otras. En las siguientes imágenes se muestra el resultado de las pruebas:

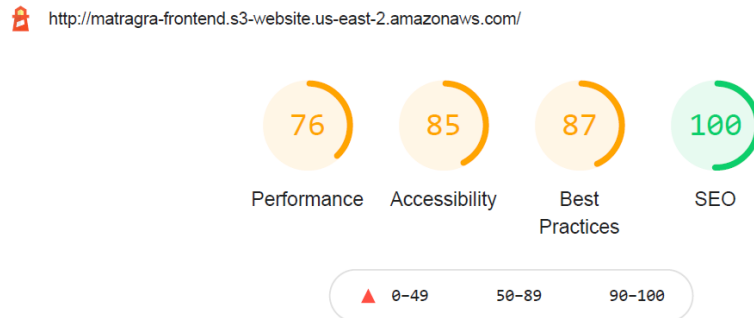


Ilustración 34 Resultado general de la prueba, fuente: elaboración propia.

En estos resultados la calificación no fue mayor debido a que se tienen en cuenta parámetros como el uso de cifrado SSL o protocolo HTTPS que no se implementaron debido a los costos que estos podrían implicar además que no se consideraron necesarios debido a que es un ambiente de desarrollo que sería utilizado en su etapa más relevante para demostraciones.

Aquí se muestran algunos de los parámetros tenidos en cuenta para la prueba:

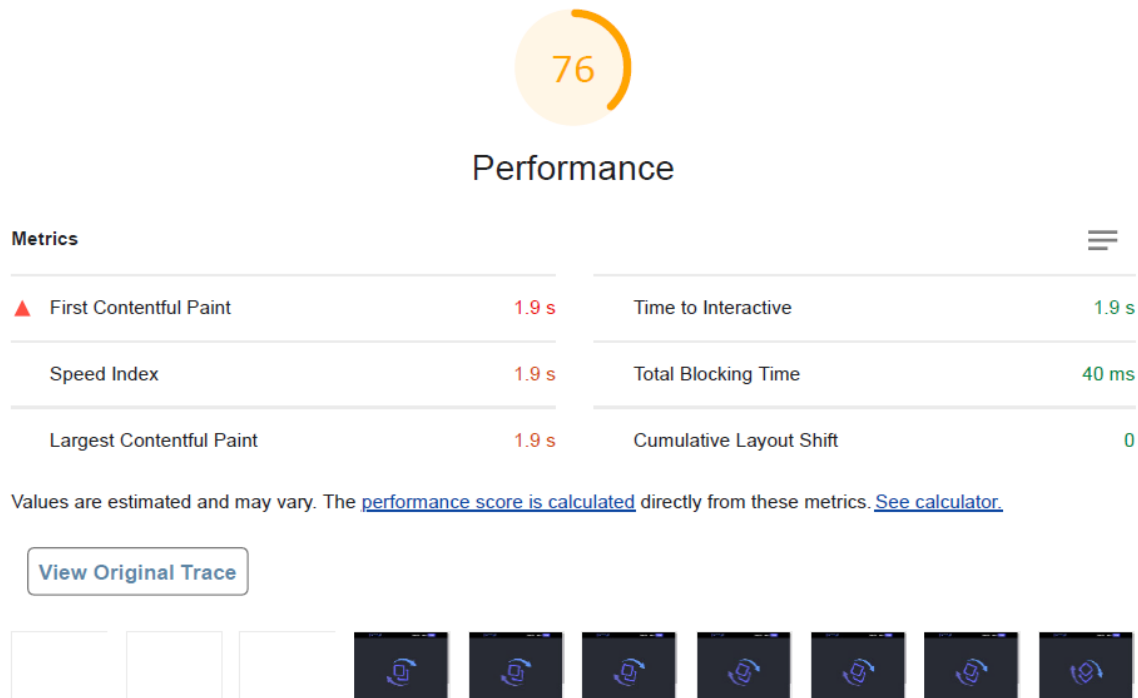


Ilustración 35 Resultado de performance, fuente: elaboración propia.

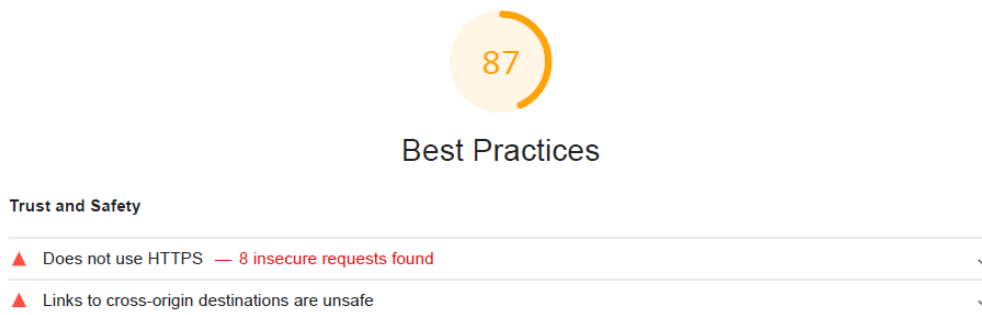


Ilustración 36 Resultado de mejores prácticas, fuente: elaboración propia.

En la imagen anterior se puede ver que algo tenido en cuenta para la puntuación de mejores practicas fue lo mencionado anteriormente, el uso de HTTPS, además de el hecho que se encuentra habilitado el uso del cross-origin, necesario para que el servidor acepte las conexiones de la aplicación web.

Por otro lado, en la siguiente imagen se puede ver que la aplicación está optimizada para búsquedas en la web.

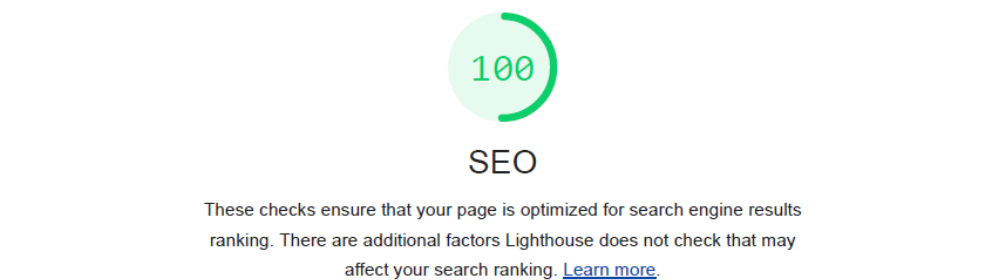


Ilustración 37 Resultado SEO, fuente: elaboración propia.

Para las pruebas del API se usó la aplicación Postman, teniendo en cuenta diferentes casos y para metros como entradas invalidas, y no autorizadas.

En la siguiente imagen se puede ver una petición valida con el header de autorización:

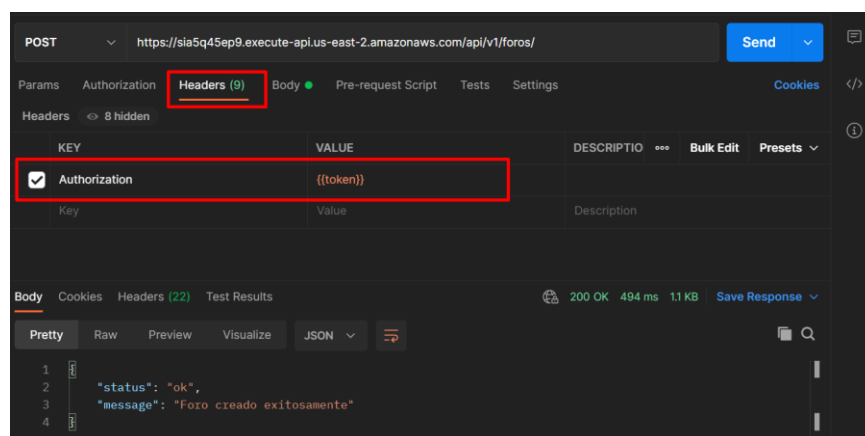


Ilustración 38 Petición exitosa con autenticación

Ahora una petición fallida por falta de autenticación:

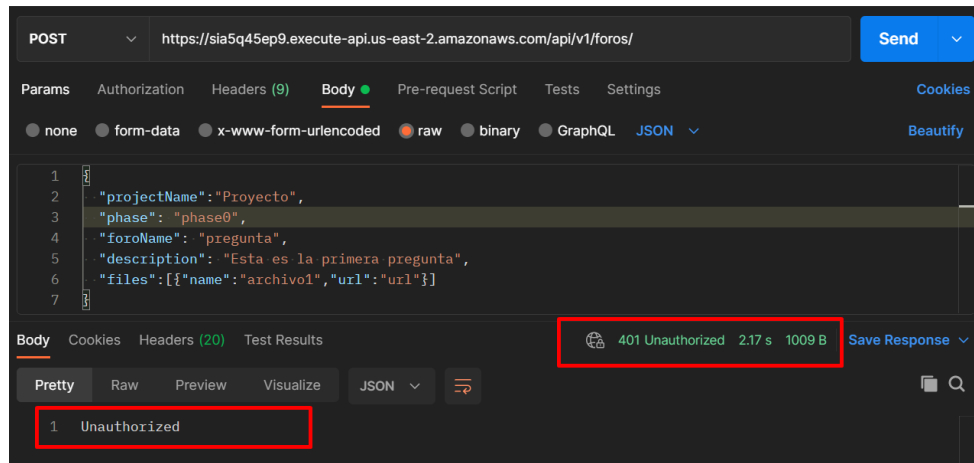


Ilustración 39 Petición fallida por autenticación

Y de la misma forma se definieron y probaron todos los endpoints de la API.

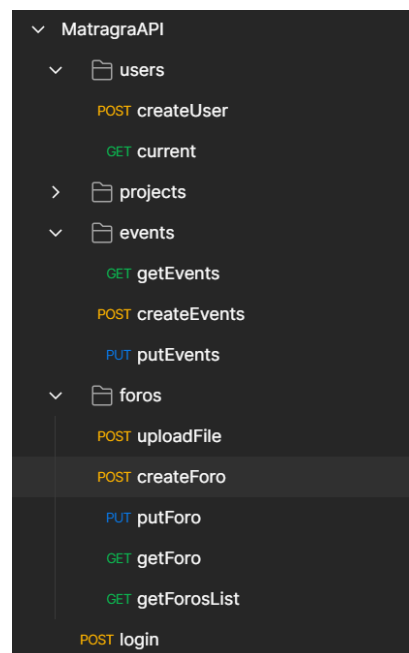


Ilustración 40 Lista de endpoints

Para ver el resultado completo de las pruebas o la colección de postman, estos pueden ser consultados en los Anexos.

6. ANÁLISIS DE RESULTADOS

En el proyecto de grado se logro el desarrollo de la aplicación web responsiva con las funcionalidades objetivo, definidas desde el anteproyecto además de desplegarla en un ambiente cloud real. Adicionalmente usamos últimas tecnologías como infraestructura cloud, Infraestructure as Code (IaC) que permitió desplegar toda la infraestructura de la aplicación web desde un archivo, funciones lambda para una arquitectura serverless y bases de datos no relaciones, entre otras como las tecnologías utilizadas del frontend.

ANALISIS DE RESULTADO (alcance vs desarrollado en el proyecto)

Alcance	Resultado esperado			Estado
	Objetivo específico	Actividades	Producto	
Aplicación web con arquitectura en la nube. Entregar documentación acorde con la norma ISO 9001.	Permitir la comunicación entre estudiante(s) y tutor.	Estado del arte	Marco de referencia	Finalizado
		Levantamiento de información	Definición de requisitos	Finalizado
		Diagramas de la aplicación	Arquitectura	Finalizado
		Configurar recursos AWS		
		Diseñar Base de datos NoSQL		
		Diseñar interfaces	Chat/foro con sistema de notificaciones entre estudiante y tutor	Finalizado
		Desarrollar componente de comunicación entre estudiante y tutor		
		Pruebas funcionales		
	Administrar la agenda, actividades y desarrollo del proyecto de software. MaTraGra.	Diseñar interfaces	Calendario de actividades, Registro de actividades y Barra de estado del avance del proyecto	Finalizado
		Desarrollar componente de comunicación entre estudiante y tutor		
		Pruebas funcionales		
	Gestionar los procesos orientadores de la metodología	Diseñar interfaces	Generador de documentación de acuerdo al formato descrito en la metodología MaTraGra	Finalizado
		Desarrollar componente de comunicación entre estudiante y tutor		
		Pruebas funcionales		
		Pruebas de integración end-to-end	Documento acorde con ISO 9001	Finalizado
		Documentación del trabajo de grado		

7. CONCLUSIONES Y RECOMENDACIONES

Consecuentes con los objetivos específicos, la aplicación concebida el actual proyecto, cumplió las especificaciones definidas para el soporte de aplicación de la metodología ágil MaTraGra, en la administración y seguimiento de los procesos, como en el apoyo a la comunicación entre usuarios.

El proyecto de “Aplicación Web para manejar proyectos según la metodología MaTraGra”, con base en el análisis de resultados realizado y expuesto anteriormente, se concluye:

- El Marco de referencia fue desarrollado dentro del estado del arte, al hacer revisión de la documentación de otras aplicaciones para manejar proyectos de grado y es una ayuda para desarrollar nuestra aplicación web con arquitectura en la nube.
- La definición de requisitos se completó y documento luego de haber realizado el levantamiento de información sobre los componentes necesarios para la aplicación web con arquitectura en la nube.
- El diseño y aplicación de la arquitectura Cloud fue completada con el conjunto de actividades de configurar recursos AWS, Diagramas de la aplicación y el diseño de la base de datos NoSQL.

El aplicativo puede extenderse en su uso a otras Instituciones de Educación Superior, debido a su planteamiento universal a proyecto de software en el ámbito de trabajos de grado.

8. ANEXOS

- Resultado de pruebas Google Lighthouse, disponible en:
<https://github.com/SantiagoPri/Matragra/blob/master/incluido%20en%20doc/performance%20and%20accesability.pdf>
- Colección de Postman, disponible en:
https://github.com/SantiagoPri/Matragra/blob/master/incluido%20en%20doc/MatragraAPI.postman_collection.json

REFERENCIAS

ACENS. 2021. *Bases de datos NoSQL. Qué son y tipos que nos podemos encontrar*. Acens.com. Disponible en: <https://www.acens.com/wp-content/images/2014/02/bbdd-nosql-wp-acens.pdf>. (03/08/2021).

Amazon, 2021. *AWS Documentation*. Docs.aws.amazon.com. Disponible en: <https://docs.aws.amazon.com/>. (03/08/2021).

Atlassian. 2021. *Product Demo - Jira Software | Atlassian*. [online] Available at: <https://www.atlassian.com/es/software/jira>. (03/08/2021).

D'Aleman S., C. S. & Castillo P., F. R. 2016. TraGrApp. Construcción de una Aplicación Móvil para la Administración y Control de los Trabajos de Grado en Ingeniería de Sistemas de la FUAC. FUAC.

Grapheverywhere. 2021. Bases de Datos NoSQL | Qué son, marcas, tipos y ventajas. grapheverywhere.com. Disponible en: <https://www.grapheverywhere.com/bases-de-datos-nosql-marcas-tipos-ventajas>. (03/08/2021).

Rivera S., G. 2018. *Metodología ágil orientada a trabajos de grado - MaTraGra*. UNAM. Disponible en: <https://repositorial.cuaed.unam.mx:8443/xmlui/handle/20.500.12579/5423>. (03/08/2021).

Rodríguez Mateus, J. & Ladino López, A., 2014. *Software para el seguimiento, la gestión y el control de proyectos de grado en el departamento de electrónica*. Pontificia Universidad Javeriana. Disponible en: <https://repository.javeriana.edu.co/handle/10554/16430>, (03/08/2021).

Sarmiento Forero, J. & Quirós Traslaviña, F., 2013. *Sistema de información y gestión de proyectos de grado*. Universidad Libre de Colombia. Disponible en: <https://repository.unilibre.edu.co/bitstream/handle/10901/8875/Sistema%20de%20informacion%20y%20gestion%20de%20proyectos%20de%20grado.pdf?sequence=1>. (03/08/2021).