SISTEMA DE REGISTRO Y CONTROL DE ACCESO Y SALIDA DE MOTOCICLETAS EN LA FACULTAD TECNOLÓGICA DE LA UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS BASADO EN API RESTFUL

SANTIAGO LOPEZ GOMEZ
JEISSON CAMILO ROJAS SÁNCHEZ

UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS
FACULTAD TECNOLÓGICA
INGENIERÍA EN TELEMÁTICA
BOGOTÁ D.C.
2022

SISTEMA DE REGISTRO Y CONTROL DE ACCESO Y SALIDA DE MOTOCICLETAS EN LA FACULTAD TECNOLÓGICA DE LA UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS BASADO EN API RESTFUL

SANTIAGO LOPEZ GOMEZ
JEISSON CAMILO ROJAS SÁNCHEZ

INFORME DE MONOGRAFÍA PARA OPTAR AL TÍTULO DE INGENIERO EN TELEMÁTICA

DIRECTOR, ING. LUIS FELIPE WANUMEN SILVA

UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS
FACULTAD TECNOLÓGICA
INGENIERÍA EN TELEMÁTICA
BOGOTÁ D.C.
2022

Nota de aceptación:

_____
_____
_____
_____
_____
_____

_____

Asesor: Luis Felipe Wanumen Silva

_____

Jurado: William Bautista Herrera

AGRADECIMIENTOS

Agradecemos principalmente a nuestras familias por el apoyo brindado durante el transcurso de toda la carrera, a nuestros docentes que ayudaron a desarrollar un pensamiento crítico y a la adquisición de conocimientos que podremos utilizar en nuestras vidas laborales y personales, en especial a nuestro director de tesis Luis Felipe Wanumen Silva por guiar este proyecto y formar parte del objetivo alcanzado y finalmente a nuestro jurado de tesis William Bautista Herrera quien nos ha ofrecido la oportunidad retroalimentación, encaminando de mejor manera el proyecto.

RESUMEN

Actualmente la Facultad Tecnológica de la Universidad Distrital Francisco José de Caldas presenta un problema en el control del flujo de salida y entrada de motocicletas, dando como resultado retrasos en los tiempos e inseguridades asociadas a la mezcla de información del estudiante y motocicletas.

El presente proyecto está pensado principalmente para facilitar la tarea realizada por el personal de seguridad de la Facultad Tecnológica de la Universidad Distrital, a la hora de conceder el acceso o salida de un estudiante junto su motocicleta.

Basado en lo anteriormente mencionado, es evidente la necesidad de disminuir tiempos y riesgos, por dicha razón se escoge la metodología SCRUM para así planificar el proyecto en su totalidad, pasando por una fase de planeación en donde se describen los objetivos a cumplir, los alcances y limitaciones, una fase de análisis, donde se analicen los requerimientos del usuario mediante historias de usuario, una fase de diseño, donde se realiza la distribución de tareas por medios de Sprint, una fase de implementación donde se realicen los diagramas y modelos obtenidos en la fase anterior, y finalmente la fase de pruebas en donde se compruebe el funcionamiento del proyecto realizado.

El desarrollo de software será realizado con el Stack de tecnologías, Mern Stack, el cual unifica tecnologías basadas en el lenguaje de programación JavaScript, como lo son MongoDB, Express JS, React JS y Node JS, adicionalmente se utilizará la integración de un lector de Código de barras con el fin de leer la información del código del estudiante.

ABSTRACT

Currently, the Technological Faculty of the Francisco José de Caldas District University has a problem in controlling the flow of motorcycles out and in, resulting in time delays and insecurities associated with the mixture of student information and motorcycles.

This project is mainly intended to facilitate the task carried out by the security personnel of the Technological Faculty of the District University, when granting the access or exit of a student along with his motorcycle.

Based on the aforementioned, the need to reduce time and risks is evident, for this reason the SCRUM methodology is chosen to plan the project in its entirety, going through a planning phase where the objectives to be met are described, the scope and limitations, an analysis phase, where the user requirements are analyzed through user stories, a design phase, where the distribution of tasks is carried out by means of Sprint, an implementation phase where the diagrams and models obtained in the previous phase, and finally the testing phase where the operation of the project carried out is verified.

Software development will be carried out with the technology stack, Mern Stack, which unifies technologies based on the JavaScript programming language, such as MongoDB, Express JS, React JS and Node JS, additionally the integration of a script reader will be used. Barcode to read the student code information.

CONTENIDO

AGRADECIMIENTOS 4
RESUMEN 5
ABSTRACT 6
INTRODUCCIÓN 14
CAPÍTULO 1 15
1. FASE DE PLANEACIÓN 15
1.1. TÍTULO 15
1.2. TEMA 15
1.3. PLANTEAMIENTO DEL PROBLEMA 15
1.3.1. DESCRIPCIÓN DEL PROBLEMA 15
1.3.2. FORMULACIÓN DEL PROBLEMA 16
1.4. ESTADO DEL ARTE 17
1.5. JUSTIFICACIÓN 18
1.6. OBJETIVOS 19
1.6.1. OBJETIVO GENERAL 19
1.6.2. OBJETIVOS ESPECÍFICOS 19
1.7. SOLUCIÓN TECNOLÓGICA 20
1.8. METODOLOGÍA DE DESARROLLO (SCRUM) 21
1.8.1. TERMINOLOGÍA SCRUM 21
1.8.2. SCRUM 22
1.8.3. ROLES 24
1.9. RESULTADOS ESPERADOS 25
1.10. IMPACTOS 25
1.11. POSIBLES AFECTACIONES EN EL SISTEMA 26
1.11. MARCO TEÓRICO 27
1.11.1. MARCO HISTÓRICO 27
1.11.2. MARCO CONCEPTUAL 28
1.11.3. MARCO LEGAL 33
1.11.4. MARCO REFERENCIAL 34
1.12. FACTIBILIDAD 34
1.12.1. FACTIBILIDAD OPERATIVA 35

1.12.2. FACTIBILIDAD TÉCNICA 35
1.12.3. FACTIBILIDAD ECONÓMICA 35
1.12.4. FACTIBILIDAD LEGAL 37
1.13. CRONOGRAMA 37
CAPÍTULO 2 41
2. ANÁLISIS 41
2.1. PRODUCT BACKLOG 41
2.1.1. PRODUCT BACKLOG 41
2.1.2. DEFINICIÓN DE USUARIOS 51
CAPÍTULO 3 53
3. DISEÑO 53
3.1. DEFINICIÓN DE SPRINTS 53
3.1.1. PLANEACIÓN DEL SPRINT 53
3.1.2. REFINAMIENTO DEL BACKLOG 53
3.1.3. REVISIÓN DEL SPRINT 53
3.1.4. RETROSPECTIVA DEL SPRINT 53
3.2. SPRINT 1 54
3.2.1. PLANEACIÓN DEL SPRINT 54
3.2.2. REFINAMIENTO DEL BACKLOG 54
3.2.3. REVISIÓN DEL SPRINT 54
3.2.4. RETROSPECTIVA DEL SPRINT 57
3.3. SPRINT 2 57
3.3.1. PLANEACIÓN DEL SPRINT 57
3.3.2. REFINAMIENTO DEL BACKLOG 57
3.3.3. REVISIÓN DEL SPRINT 58
3.3.4. RETROSPECTIVA DEL SPRINT 60
3.4. SPRINT 3 60
3.4.1. PLANEACIÓN DEL SPRINT 60
3.4.2. REFINAMIENTO DEL BACKLOG 61
3.4.3. REVISIÓN DEL SPRINT 61
3.4.4. RETROSPECTIVA DEL SPRINT 64
3.5. SPRINT 4 64
3.5.1. PLANEACIÓN DEL SPRINT 64

3.5.2. REFINAMIENTO DEL BACKLOG 64
3.5.3. REVISIÓN DEL SPRINT 64
3.5.4. RETROSPECTIVA DEL SPRINT 66
3.6. SPRINT 5 66
3.6.1. PLANEACIÓN DEL SPRINT 66
3.6.2. REFINAMIENTO DEL BACKLOG 67
3.6.3. REVISIÓN DEL SPRINT 67
3.6.4. RETROSPECTIVA DEL SPRINT 71
3.7. SPRINT 6 71
3.7.1. PLANEACIÓN DEL SPRINT 71
3.7.2. REFINAMIENTO DEL BACKLOG 71
3.7.3. REVISIÓN DEL SPRINT 72
3.7.4. RETROSPECTIVA DEL SPRINT 75
3.9. DIAGRAMA DE DESPLIEGUE 75
3.10. DIAGRAMA DE PROCESOS 75
3.10.1. DIAGRAMA DE INGRESO DE MOTOCICLETAS 75
3.10.1. DIAGRAMA DE SALIDA DE MOTICICLETAS 76
CAPÍTULO 4 77
4. IMPLEMENTACIÓN 77
4.1. MODELO DE BASE DE DATOS NO RELACIONAL 77
4.2. DICCIONARIO DE DATOS 77
4.2.1 DOCUMENTO MONGODB STUDENT 77
4.2.2 DOCUMENTO MONGODB CREDENTIAL 78
4.2.3 DOCUMENTO MONGODB PARKING 79
4.2.4 DOCUMENTO MONGODB HISTORIAL 80
4.3. ESTRUCTURA DE ARCHIVOS 81
4.3.1 DESARROLLO BACKEND Y FRONTEND 81
4.3.2 ORGANIZACIÓN DEL FRONTEND 82
4.3.2 ORGANIZACIÓN DEL BACKEND 85
4.3.3 CONTROL DE VERSIONAMIENTO POR GIT 87
CAPÍTULO 5 89
5. PRUEBAS 89
5.1. PRUEBAS SPRINT 1 89

5.1.1 PRUEBA 1 DEL SPRINT 1 89
5.1.2 PRUEBA 2 DEL SPRINT 1 90
5.1.3 PRUEBA 3 DEL SPRINT 1 91
5.1.4 PRUEBA 4 DEL SPRINT 1 92
5.2. PRUEBAS SPRINT 2 93
5.2.2 PRUEBA 1 DEL SPRINT 2 93
5.1.2 PRUEBA 2 DEL SPRINT 2 94
5.3. PRUEBAS SPRINT 3 94
5.3.1 PRUEBA 1 DEL SPRINT 3 94
5.3.2 PRUEBA 2 DEL SPRINT 3 95
5.4. PRUEBAS SPRINT 4 97
5.4.1 PRUEBA 1 DEL SPRINT 4 97
5.5. PRUEBAS SPRINT 5 97
5.5.1 PRUEBA 1 DEL SPRINT 5 97
5.5.2 PRUEBA 2 DEL SPRINT 5 98
5.6. PRUEBAS SPRINT 6 100
5.6.1 PRUEBA 1 DEL SPRINT 6 100
CONCLUSIONES 101
REFERENCIAS BIBLIOGRÁFICAS 102

LISTA DE TABLAS

Tabla 1. Roles metodología scrum 24
Tabla 2. Integrantes del proyecto 35
Tabla 3. Productos para utilizar en el desarrollo del proyecto 35
Tabla 4. Recursos de Hardware 36
Tabla 5. Recursos de Software 37
Tabla 6. Costos Varios 37
Tabla 7. Costos totales del proyecto 37
Tabla 8. Actividades principales para realizar en el desarrollo del proyecto 41
Tabla 9. Product Backlog 42
Tabla 10. Historia de usuario numero 1 43
Tabla 11. Historia de usuario numero 2 44
Tabla 12. Historia de usuario numero 3 45
Tabla 13. Historia de usuario numero 4 45
Tabla 14. Historia de usuario numero 5 46
Tabla 15. Historia de usuario numero 6 47
Tabla 16. Historia de usuario numero 7 47

Tabla 17. Historia de usuario numero 8 48
Tabla 18. Historia de usuario numero 9 49
Tabla 19. Historia de usuario numero 10 49
Tabla 20. Priorización y estimación del Product Backlog 50
Tabla 21. Priorización y estimación del Product Backlog según Planning Poker 51
Tabla 22. Priorización y estimación del Product Backlog según Planning Poker ordenado por estimación 51
Tabla 23. Definición de usuarios 52
Tabla 24. Planeación Sprint 1 54
Tabla 25. Refinamiento del Backlog Sprint 1 54
Tabla 26. Planeación del Sprint 2 57
Tabla 27. Refinamiento del Backlog Sprint 2 58
Tabla 28. Planeación de Sprint 3 61
Tabla 29. Refinamiento del Backlog Sprint 3 61
Tabla 30. Planeación de Sprint 4 64
Tabla 31. Refinamiento del Backlog Sprint 4 64
Tabla 32. Planeación del Sprint 5 66
Tabla 33. Refinamiento del Backlog Sprint 5 67
Tabla 34. Planeación Sprint 6 71
Tabla 35. Refinamiento Backlog Sprint 6 72
Tabla 36. Documento MongoDB Student 78
Tabla 37. Documento MongoDB Credential 79
Tabla 38. Documento MongoDB Parking 80
Tabla 39. Documento MongoDB Historial 81
Tabla 40. Prueba 1 del Sprint 1 89
Tabla 41. Prueba 2 del Sprint 1 90
Tabla 42. Prueba 3 del Sprint 1 91
Tabla 43. Prueba 4 del Sprint 1 92
Tabla 44. Prueba 1 del Sprint 2 93
Tabla 45. Prueba 2 del Sprint 2 94
Tabla 46. Prueba 1 Sprint 3 94
Tabla 47. Prueba 2 Sprint 3 96
Tabla 48. Prueba 1 Sprint 4 97
Tabla 49. Prueba 1 Sprint 5 98
Tabla 50. Prueba 2 del Sprint 5 - Verificación de información actualizada sobre motocicletas 99
Tabla 51. Prueba 1 del Sprint 6 100

LISTA DE ILUSTRACIONES

Ilustración 1. Vista simple de un Stack tecnológico 28
Ilustración 2. Composición del Framework Mern Stack 29
Ilustración 3. Cronograma 40

Ilustración 4. Logotipo del aplicativo..... 55
Ilustración 5. Diseño del formulario de registro de estudiantes 55
Ilustración 6. Implementación de la vista para el registro de estudiantes..... 56
Ilustración 7. Vista de validación de campos..... 56
Ilustración 8.Validación de campos del formulario de registro de estudiantes 56
Ilustración 9. Diseño del formulario de consulta de estudiante 59
Ilustración 10. Implementación del formulario de consulta de estudiante 59
Ilustración 11. Diseño de la vista con los datos del estudiante 59
Ilustración 12. Implementación de la vista con los datos del estudiante 60
Ilustración 13. Validación de campos del formulario de consulta del estudiante..... 60
Ilustración 14.Diseño del formulario de edición de estudiantes..... 62
Ilustración 15. Implementación del formulario de edición de estudiantes..... 63
Ilustración 16. Creación de alertas para el formulario de edición de estudiantes..... 63
Ilustración 17. Diseño del formulario de registro de motocicletas 65
Ilustración 18. Implementación de la vista para el registro de motocicletas 65
Ilustración 19. Alertas en el formulario de registro de motocicletas 66
Ilustración 20. Validación de campos del formulario de registro de motocicletas..... 66
Ilustración 21. Diseño del formulario de edición de motocicletas 69
Ilustración 22. Diseño de la funcionalidad de eliminación de motocicletas 69
Ilustración 23. Implementación del formulario de edición de motocicletas 70
Ilustración 24. Implementación de la funcionalidad de eliminación de motocicletas 70
Ilustración 25. Creación de alertas para el formulario de edición de motocicletas ... 70
Ilustración 26. Creación de alertas para la funcionalidad eliminación de motocicletas 71
Ilustración 27. Diseño del formulario de consulta del listado actual 73
Ilustración 28. Diseño del formulario de consulta del listado histórico 73
Ilustración 29. Implementación del formulario de consulta del listado actual 74
Ilustración 30. Implementación del formulario de consulta del listado histórico 74
Ilustración 31. Configuración del buscador de los formularios de búsqueda del listado actual y listado histórico 74
Ilustración 32. Diagrama de despliegue 75
Ilustración 33. Diagrama de entrada de motocicletas..... 76
Ilustración 34. Diagrama de salida de motocicletas 76
Ilustración 35. Modelo de base de datos..... 77
Ilustración 36. Distribución Backend y Frontend 82
Ilustración 37. Distribución Frontend..... 82
Ilustración 38. Carpeta assets - Frontend 83
Ilustración 39. Carpeta views - Frontend..... 83
Ilustración 40. Componentes en el Frontend..... 84
Ilustración 41. Distribución de los componentes en el Frontend 85
Ilustración 42. Carpeta controllers - Backend..... 86
Ilustración 43. Carpeta models - Backend..... 86
Ilustración 44. Carpeta routes - Backend 87
Ilustración 45. Distribución Backend 87

Ilustración 46. Ramas utilizadas en Git 88
Ilustración 47. Prueba 1 del Sprint 1 – Diligenciamiento de información en registro de estudiante..... 89
Ilustración 48. Prueba 1 del Sprint 1 - Validación de campos formulario registro de estudiante..... 90
Ilustración 49. Prueba 2 del Sprint 1 - Validación de tipos de datos en el formulario registro de estudiante..... 90
Ilustración 50. Prueba 3 del Sprint 1 - Validación de guardado de información 91
Ilustración 51. Prueba 3 del Sprint 1 - Validación de guardado de información del estudiante en Postman..... 92
Ilustración 52. Prueba 3 del Sprint 1 - Validación de guardado de información del estudiante en MongoDB..... 92
Ilustración 53. Prueba 4 del Sprint 1 - Validación de duplicidad de estudiantes 93
Ilustración 54. Prueba 1 Sprint 2 - Validación de la búsqueda de un estudiante..... 93
Ilustración 55. Prueba 2 Sprint 2 - Validación de la búsqueda de un estudiante..... 94
Ilustración 56. Información del estudiante a editar 95
Ilustración 57. Prueba 1 Sprint 3 - Validación de alerta en la edición del estudiante 95
Ilustración 58. Prueba 2 Sprint 3 - Validación de la información editada del estudiante en Postman..... 96
Ilustración 59. Prueba 2 Sprint 3 - Validación de la información editada del estudiante en MongoDB..... 96
Ilustración 60. Prueba 1 del Sprint 4 - Validación de campos formulario registro de motocicletas 97
Ilustración 61. Prueba 1 del Sprint 5 - Formulario edición de motocicletas..... 98
Ilustración 62. Prueba 1 del Sprint 5 - Alerta de edición exitosa 98
Ilustración 63. Prueba 2 del Sprint 5 - Verificación de la actualización de motocicleta en el aplicativo 99
Ilustración 64. Prueba 2 del Sprint 5 - Verificación de la actualización de motocicleta en Postman..... 99
Ilustración 65. Prueba 2 del Sprint 5 - Verificación de la actualización de motocicleta en MongoDB 100
Ilustración 66. Prueba 1 del Sprint 6 - Listado Histórico de entradas y salidas de motocicletas. 100
Ilustración 67. Prueba 1 del Sprint 6 - Búsqueda exitosa del código ingresado..... 101

INTRODUCCIÓN

El desarrollo de una aplicación web, suele basarse en las necesidades de una persona o grupo de personas; donde su implementación debe mejorar el resultado de las tareas asociadas a estas. Por este motivo es importante analizar los requerimientos del cliente, pudiendo ser una persona, una organización, un estado, entre otros; antes de poner en ejecución cualquier desarrollo web. Una vez realizado el análisis de los requerimientos del cliente, se deberá escoger las herramientas apropiadas para su construcción, para finalmente realizar pruebas y poner en funcionamiento la solución creada.

En el caso de la Facultad Tecnológica de la Universidad Distrital Francisco José de Caldas se evidencia la necesidad de mejorar el proceso de ingreso y salida de motocicletas pertenecientes a los estudiantes. Por lo anteriormente expuesto se plantea un aplicativo web el cual facilite el registro y verificación de las motocicletas pertenecientes a los estudiantes, reduciendo tiempos de ingreso o salida, aumentado el control, organización y seguridad de los vehículos.

El aplicativo para satisfacer la problemática con la que actualmente cuenta la Facultad Tecnológica de la Universidad Distrital, es desarrollado con Mern Stack, un Framework que consta de un conjunto de herramientas, las cuales utilizan como base el lenguaje de programación JavaScript, lo que genera un producto potente y altamente escalable, este Stack se compone de las siguientes aplicaciones MongoDB, Express JS, React JS y NodeJS, todas creadas para la creación de aplicaciones con alta posibilidad de crecimiento. Además de utilizar dicho conjunto de aplicaciones se utilizará un lector de código de barras el cual ayudará al personal de seguridad a realizar una verificación de identidad mucho más sencilla.

Finalmente, la implementación de dicho proyecto ayudará a la comunidad académica a mantener un mejor flujo de entrada o salida de sus motocicletas y por otra parte ayudará al personal de seguridad de la Facultad a realizar su trabajo.

CAPÍTULO 1

1. FASE DE PLANEACIÓN

1.1. TÍTULO

SISTEMA DE REGISTRO Y CONTROL DE ACCESO Y SALIDA DE MOTOCICLETAS EN LA FACULTAD TECNOLÓGICA DE LA UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS BASADO EN API RESTFUL

1.2. TEMA

El presente proyecto está orientado al desarrollo de un sistema que facilite el ingreso y salida de las motocicletas en la Facultad Tecnológica de la Universidad Distrital Francisco José de Caldas, la solución se implementa con el conjunto de tecnologías Mern Stack, el cual hace la integración de MongoDB, React JS, Express JS y NodeJS.

1.3. PLANTEAMIENTO DEL PROBLEMA

1.3.1. DESCRIPCIÓN DEL PROBLEMA

Actualmente en la Universidad Distrital Francisco José de Caldas sede Tecnológica, se están presentando algunos inconvenientes de control y acceso al parqueadero de motocicletas, debido a que se está haciendo por medio de un proceso manual, en el cual los guardias de seguridad al momento del ingreso del estudiante solicitan el carné estudiantil y la tarjeta de propiedad del vehículo, allí el personal de vigilancia otorga una ficha física con un número específico para que el estudiante la conserve, los documentos solicitados quedan guardados en un estante numerado de menor a mayor, el cual corresponde al número de cada ficha, de acuerdo a lo anterior, en este proceso manual se han identificado algunas problemáticas relevantes que generan una afectación al control y acceso de dichos vehículos a la facultad.

En primer lugar, este proceso manual es susceptible al error humano, tanto del personal de seguridad como del estudiante, en el momento que se otorga la ficha numerada al estudiante puede que los documentos no se almacenen en la casilla correspondiente, lo que puede llevar a una pérdida de los documentos personales de la persona, o que se entreguen a otro estudiante que no sea el propietario. Por otra parte, si el estudiante es quien extravía la ficha otorgada desconociendo el número que esta tenía y procede a salir de la universidad, se genera un inconveniente debido a que esto lleva al guardia de seguridad a

realizar un reproceso en donde deberá buscar entre varias casillas los documentos de la persona, obligándolo a realizar una validación adicional donde confirme que realmente es el propietario tanto de los documentos, como de la motocicleta, en este lapso de tiempo que ocupa este subproceso se puede generar un colapso en el flujo de entrada y salida de los estudiantes, adicionalmente si el guardia de seguridad no hace una verificación precisa de la identidad de la persona cuya tarjeta extravío, puede que un tercero sea quien salga con una motocicleta de la universidad, lo cual se convertiría en un hurto.

En segundo lugar, dicho proceso manual hace el trabajo engorroso para el personal de seguridad, en el momento en que se acaban las casillas del estante numeradas por el alto flujo de vehículos que ingresan a la facultad hace que estos almacenen los documentos fuera del estante, lo que conlleva a que haya una vulnerabilidad a la pérdida de los documentos debido al limitante de almacenamiento físico que existe, por otra parte en el momento que hay solo un guarda de seguridad quien realiza el proceso de control de acceso y salida, por la cantidad de persona que ingresan hace que se congestione tanto la salida como la entrada de dichos vehículos.

En tercer lugar, el proceso manual implica que el estudiante tenga que dejar su carné estudiantil guardado y a cambio la única forma de identificarse dentro de la universidad es con la ficha numérica, lo que genera inconvenientes en el momento en que este quiera realizar algún tipo de trámite dentro de la facultad, ya sea solicitar un espacio en un laboratorio, solicitar un computador, los diferentes trámites de biblioteca y en general algunos procesos internos en donde se necesita el carné para poder acceder a dichos servicios. De la misma forma la ficha numérica al ser el único medio de identificación de un estudiante dentro de la facultad permite acceder a las personas a la facultad, lo cual puede conllevar a que una persona ajena a la universidad pueda ingresar vía peatonal sin ningún inconveniente, solamente mostrando dicha ficha numerada.

Por último, el punto físico de acceso y salida de las motocicletas se encuentra hacia el extremo de la facultad lo que conlleva a que la señal Wifi sea casi nula en estos sitios, por lo que una implementación de una aplicación web que dependa de internet no ayudaría a resolver la problemática del control y acceso de vehículos.

1.3.2. FORMULACIÓN DEL PROBLEMA

¿Cómo desarrollar un sistema utilizando el conjunto de tecnologías MERN Stack, el cual integra MongoDB, React JS, Express JS y NodeJS

que permita el registro y control de acceso y salida de motocicletas en la Facultad Tecnológica de la Universidad Distrital Francisco José de Caldas?

1.4. ESTADO DEL ARTE

La importancia de mantener controlados los procesos de entrada y salida de medios de transporte dentro de una organización se ha vuelto una tarea indispensable, ya que, gracias a su correcto funcionamiento, se logra aumentar la seguridad, la fluidez de los procesos y se disminuye notablemente el error humano. Para satisfacer lo anteriormente mencionado se hace uso de la tecnología específicamente en sistemas o herramientas que logren ayudar al encargado a realizar las tareas de manera correcta, reduciendo tiempos y posibles falencias.

Actualmente, dentro de la Facultada Tecnológica de la Universidad Distrital Francisco José de caldas, no se cuenta con ningún sistema que facilite el proceso de entrada y salida de las motocicletas pertenecientes a los estudiantes, este proceso se realiza de manera manual, en donde se entrega una tarjeta al estudiante en cambio de su carnet estudiantil, con esta misma deberá realizar la entrega al personal de seguridad del parqueadero para poder retirar la motocicleta de las instalaciones de la Facultad.

Con el fin de fortalecer las bases de dicho proyecto es de vital importancia tener referencia de aquellos proyectos realizados en la sede Tecnológica de la Universidad Distrital. Por esta razón se realizó la búsqueda de algunos de los proyectos realizados como modalidad de grado por parte de estudiantes de la facultad que han intentado satisfacer dicha problemática, pero no se encuentran en funcionamiento, entre ellos resaltan los siguientes, “Software de control de acceso y registro de vehículos para los parqueaderos de la Facultad Tecnológica en la Universidad Distrital Francisco José de Caldas, con lector de huella digital” y “Sistema telemático para la gestión del parqueadero de motocicletas de la facultad tecnológica”, los cuales fueron ubicados en el repositorio institucional de la Universidad Distrital y se relacionan con el desarrollo a implementar.

El primer proyecto encontrado “Software de control de acceso y registro de vehículos para los parqueaderos de la Facultad Tecnológica en la Universidad Distrital Francisco José de Caldas, con lector de huella digital”, utiliza un lector de huella digital para la recolección de información junto al lenguaje de programación Java y PostgreSQL como base de datos. Su objetivo principal es desarrollar un software que permita tener un control de acceso de vehículos y registro de los propietarios de estos mediante la lectura de la huella digital, adicionalmente guardará la información recogida para facilitar la reportería y consulta de los movimientos realizados.

Por otra parte, el proyecto “Sistema telemático para la gestión del parqueadero de motocicletas de la facultad tecnológica”, utiliza un lector de código QR el cual escanea la información del estudiante mediante el código generado en el celular del estudiante, esté basado en API RESTful, Java, Spring Boot e IÓNIC. Su objetivo principal es el desarrollo de una aplicación móvil en dispositivos Android, que facilite el control y asignación de espacios de parqueo, esto con el fin de salvaguardar los vehículos de los estudiantes dentro de la Facultad.

Con la implementación del proyecto “Sistema de registro y control de acceso y salida de motocicletas en la Facultad Tecnológica de la Universidad Distrital Francisco José de Caldas Basado en API RESTful”, el cual se desarrollará en el presente documento, se tiene como objetivo principal tener un control del flujo de motocicletas de los estudiantes de la Facultad Tecnológica de la Universidad Distrital, ya que, se piensa en un sistema de escritorio manejado por el personal de seguridad, el cual es el encargado de autorizar o no el ingreso de los vehículos a la Facultad. Con dicho sistema se recolecta la información del estudiante utilizando un lector de código de barras y adicionalmente la información de los vehículos asociados al estudiante de manera manual. La información guardada por primera vez servirá para comparar un nuevo ingreso o salida del estudiante. Esto permitirá mayor agilidad en el flujo de vehículos que entran y salen de la Universidad aumentando el control, organización y seguridad.

Adicionalmente se debe tener en cuenta que el sistema a implementar esta pensado para facilitar el trabajo del personal de seguridad de la facultad, a la hora de dar ingreso o retiro a una motocicleta perteneciente al estudiante, por eso sus funcionales son concisas y fáciles de usar.

Finalmente, el sistema será de mayor utilidad para el personal de soporte de la facultad, comparándolo con los mencionados anteriormente, ya que esta desarrollado en tecnologías actuales, manejando dos bloques de desarrollo como lo son el Backend y Frontend, haciendo que se puedan agregar funcionabilidades, mantenimiento del sistema y la corrección de errores sea sencilla.

1.5. JUSTIFICACIÓN

La automatización de procesos con ayuda de las tecnologías de la información es una de las tareas con gran acogida en todas las organizaciones del mundo, además generan grandes beneficios en los procesos, como por ejemplo la fluidez, el aumento de la productividad, la disponibilidad, la confiabilidad y una mejor escalabilidad económica en poco tiempo. Por dicha razón la Universidad Distrital Francisco José de Caldas al igual que diversas organizaciones del país debe automatizar procesos que no cuentan con la ayuda de la tecnología para su ejecución.

Por estas razones se busca implementar un sistema para llevar un registro y control de las motocicletas que entren y salgan de la Facultad Tecnológica de la Universidad Distrital, usando tecnologías actuales, flexibles y escalables, ya que, actualmente se cuenta con un sistema manual, lo que hace que no sea totalmente segura la verificación del propietario del vehículo, generando problemas de inmediato. Con dicha implementación se busca mejorar el flujo de entrada y salida de motocicletas y la seguridad respecto a la verificación de identidad del estudiante y el vehículo.

Es importante destacar que la implementación del sistema facilitará el trabajo realizado por los guardias de seguridad ubicados en la zona de parqueaderos de facultad, ya que se les brindará un sistema que ayude a mantener un control de las motocicletas, reduciendo los reprocesos de verificación de identidad.

1.6. OBJETIVOS

1.6.1. OBJETIVO GENERAL

Desarrollar un sistema de gestión de ingreso y salida de las motocicletas en la Facultad Tecnológica de la Universidad Distrital Francisco José de Caldas.

1.6.2. OBJETIVOS ESPECÍFICOS

- Identificar los requerimientos del sistema, a partir de la información recolectada del personal de seguridad de la Facultad Tecnológica.
- Implementar una interfaz de programación de aplicaciones web RESTful, la cual permita una interacción de los servicios web.
- Desarrollar un subsistema de almacenamiento que permita persistir la información sobre las motocicletas que ingresan al parqueadero a partir de un lector de código de barras.
- Desarrollar un subsistema de búsqueda, que permita ubicar la información de la motocicleta y su propietario.
- Diseñar un subsistema de visualización que permita obtener la información consolidada sobre los movimientos de las motocicletas.
- Diseñar un subsistema que permita generar los Backups de los consolidados de información de la aplicación web.
- Realizar las pruebas necesarias para la verificación de la correcta funcionalidad del sistema.
- Implementar el sistema en la Facultad Tecnológica de la Universidad Distrital.

1.7. SOLUCIÓN TECNOLÓGICA

Es importante destacar que para la elaboración de la aplicación web que se llevará a cabo, para el flujo de acceso y salida de motos de la Universidad Francisco José de Caldas sede Tecnológica, se opta por elegir un conjunto de herramientas tecnológicas Open Source en base a varios factores, en primer lugar, debido a que estas han tenido un auge en el último tiempo brindando facilidades en el desarrollo de software, en segundo lugar debido a que se puede desarrollar de una manera descentralizada y colaborativa, lo que optimiza el trabajo en equipo, en tercer lugar, estas llegan a ser más económicas, flexibles y duraderas gracias a que son comunidades las que las que hacen parte de su desarrollo y no depende de un único autor, en quinto lugar, al ser abiertas tienen la particularidad de aceptar alguna integración con diferentes software o herramientas provenientes de desarrollos externos, con los cuales tiene una buena adaptabilidad.

En este sentido, todo el desarrollo de la aplicación web utiliza un conjunto de herramientas Open Source, las cuales vienen dadas por el Framework Stack MERN, esto apoya al desarrollo de la aplicación web dado que brinda un conjunto de marcos basados en un solo lenguaje de programación interpretado el cual es JavaScript, esté trabaja tanto en el lado del cliente, como en el lado del servidor. De este modo, gracias a las herramientas otorgadas por el Stack se pueden clasificar en dos grupos de trabajo en la parte de Frontend y Backend.

En primer lugar, en el lado del Backend se trabajará con la gran mayoría del Stack MERN, puesto que allí se requiere de NodeJS, este es un entorno de ejecución para la capa del servidor Backend, el cual permite la escalabilidad en la elaboración del producto, por otra parte, se hace uso de uno de los Framework de NodeJS, el cual es Express, en esencia le brinda un conjunto robusto de características a dicha aplicación. Por último, se utilizará para la base de datos MongoDB, esto indica que será la base de datos a utilizar, la cual es no relacional, brindará a la aplicación web de una escalabilidad que ayudará al almacenamiento de la múltiple información que se almacenará día tras día, a su vez MongoDB proporciona practicidad de conexión, velocidad de recolección de información y muestreo de esta. Además, es una base de datos pensada para el futuro lo que permite que se puedan realizar ajustes o repotenciar si es necesario sin afectar el rendimiento y funcionalidad del sistema. En definitiva, se utilizarán herramientas basadas en JavaScript lo que permite que la curva de aprendizaje sea más amena para el desarrollo de la aplicación web.

En cuanto a la parte del Frontend, se utilizará la última de las herramientas del Stack MERN, siendo ReactJS la faltante, esta permite versatilidad en la

implementación debido a que como al igual que las otras herramientas del Backend es basada en JavaScript, siendo esta una librería del lenguaje, la cual permite crear de interfaces más llamativas, el complemento del componente visual se obtiene con la utilización de la tecnologías HTML, CSS y Bootstrap que harán mucho más amigable la página web para el usuario.

Por último, el contar con un lector de código de barras permitirá la facilidad de la interacción del usuario con la aplicación web, permitiendo que el control de accesos y salida de motocicletas de la facultad sea óptimo.

1.8. METODOLOGÍA DE DESARROLLO (SCRUM)

1.8.1. TERMINOLOGÍA SCRUM

- **Product Owner:** Es la persona encargada de asegurar que el proyecto genere valor para los interesados. En pocas palabras es la persona líder en un proyecto Scrum, siendo el canal de comunicación principal entre el equipo de desarrollo e interesados.
- **Stakeholders:** También conocido como el grupo de interés o interesados del proyecto, este hace referencia a todas las personas, grupos de personas, organizaciones o empresas que interactúan e intervienen en el proyecto. Aunque no realizan ningún tipo de desarrollo, son de vital importancia para el proyecto ya que dan la información necesaria para que el proceso realizado por el equipo Scrum tome mejor rumbo.
- **Scrum Master:** Es la persona encargada del grupo de desarrollo del proyecto, se encarga de verificar los requerimientos, analizar su viabilidad para después ejecutar.
- **Equipo Scrum:** Este es el grupo de desarrollo del proyecto, se compone normalmente entre 3 a 9 personas, se encarga de crear productos según los requerimientos de los interesados. 1
- **Backlog:** Es una lista de trabajo ordenado por prioridades que servirá de base para que el equipo de desarrollo del proyecto siga la ruta según los requerimientos. 2
- **Sprint:** En la metodología Scrum, un Sprint es el nombre que recibe cada uno de los ciclos o iteraciones que se realizan en el desarrollo del proyecto. Este se maneja un tiempo entre dos semanas y máximo de dos meses, aunque puede ser ajustable según el avance del mismo Sprint.

[1] CREATIVERÍA, Los roles del equipo de Scrum, [Consultado el 17, septiembre, 2022], Disponible en Internet: <https://www.lacreativeria.com/blog/los-roles-del-equipo-de-scrum>

[2] RADIGAN, De Dan, Atlanssian, El Backlog del producto: la lista de tareas pendientes definitiva, [Consultado el 17, septiembre, 2022], Disponible en Internet: <https://www.atlassian.com/es/agile/scrum/backlogs>

Se debe tener en cuenta que por cada Sprint realizado se debe generar un entregable, lo que aporta valor a los interesados del proyecto, dando entender en qué estado se encuentra el proyecto.

- **Sprint backlog:** Es la lista específica del Backlog que afectan a un Sprint en específico.
- **Daily Standup:** Son reuniones que se realizan al día con una duración máxima de 15 minutos, en donde deben asistir todos los miembros del equipo de desarrollo del proyecto junto al moderados Scrum Master. En estas reuniones se resuelven dudas, se encaminan objetivos y se contemplan variaciones.
- **Planning Poker:** Es una técnica utilizada en la metodología Scrum basada en cartas de juego la cual sirve para poder estimar y planificar dificultades entre historias de usuario, aunque no es principalmente utilizada para estimar tiempos, sirve al organizar el listado de actividades por estimación en horas.

1.8.2. SCRUM

Para la realización del proyecto se definió la metodología SCRUM, debido a que esta permite que la producción de este se realice con calidad, en tiempo, acorde al presupuesto y de manera adecuada para los grupos de interés. La creación del proyecto se desarrolla en las siguientes fases: (DEEMER, BENEFIELD, LARMAR y VODDE, 2012)

1. **Fase de Planeación**

   En primera instancia, se procede a definir los requisitos y el alcance que tendrá el proyecto a realizar con los Stakeholders, se genera el planeamiento de las fases y el de las próximas iteraciones posteriores, se detalla de manera general la viabilidad que este trae consigo. Entre los procesos fundamentales de esta fase se encuentran:

   - Crear la visión del proyecto, evaluando alcances y limitaciones.
   - Identificar al Scrum Master y los interesados del proyecto.
   - Conformación de los equipos Scrum.
   - Crear el listado de Backlog del producto.

2. **Fase de Análisis**

   En la fase de análisis se realiza un estudio de los requisitos recogidos en la fase de planeación y además se plantean las actividades a realizar. Entre los procesos fundamentales de esta fase se encuentran:

- Crear las historias de usuario.
- Estimar las historias de usuario.
- Análisis de las historias de usuario.
- Comprometer las historias de usuario.
- Identificar tareas con su respectiva estimación.
- Crear el Sprint Backlog.

3. **Fase de Diseño**

   En esta fase se diseña la solución a implementar, esto con la creación de los diferentes bocetos, casos de uso, documentación y entre otras herramientas que permitan definir la arquitectura base del sistema y camino a seguir por medio de los Sprint planificados en la fase anterior. Entre los procesos fundamentales de esta fase se encuentran:

   - Crear entregables.
   - Realizar Daily Standup
   - Realizar Backlog del producto.

4. **Fase de Implementación**

   En la fase de desarrollo se elabora la construcción del producto, con la ayuda de los Stakeholders se complementan y refuerzan los requisitos que se obviaron en la fase de planeación a partir de la implementación de los Sprint utilizados en la fase de diseño. Entre los procesos fundamentales de esta fase se encuentran:

   - Demostración y validación del Sprint
   - Retroalimentación del Sprint.

5. **Fase de transición**

   La fase de transición es también conocida como la fase de cierre, en la cual se evalúan las funcionalidades a modificar a raíz de las pruebas realizadas por los usuarios de la aplicación, lo que conlleva a traer mejoras en el proyecto, para finalmente proceder a entregar el proyecto para el uso de los interesados. Entre los procesos fundamentales de esta fase se encuentran:

   - Enviar entregables.
   - Retroalimentación del proyecto.

Además de lo anteriormente mencionado se opta por utilizar esta metodología, ya que, es un proceso en el cual se aplican un conjunto de mejores prácticas para trabajar de manera colaborativa, entregando

entregas parciales y regulares del producto según el rol a desempeñar. (ATLASSIAN, 2022).

1.8.3. ROLES

Basados en la Terminología Scrum descrita en el inciso 1.8.1. a continuación se describen las actividades principales de cada uno de los roles que interactúan con el proyecto.

Rol	Actividades para realizar
Product Owner	- Definir la visión del proyecto - Definir los alcances y limitaciones. - Definir los objetivos o metas a alcanzar. - Definir la estrategia o ruta a seguir. - Mantener el Product backlog. - Comunicación directa con los interesados del proyecto. - Definir los Sprint junto el Scrum Master. - Realizar Daily Standup.
Scrum Master	- Coordinar el Equipo Scrum. - Resolución de los posibles problemas del Equipo Scrum. - Comunicación directa con el Product Owner. - Comunicación directa con el Equipo Scrum. - Promueve mejoras a los Sprint. - Reportar, archivar y llevar registro de los movimientos realizados. - Realizar Daily Standup.
Equipo Scrum	- Desarrollo de los Sprint basados en Backlog. - Entrega del producto asociado al Sprint. - Realizar Daily Standup.
Stakeholders o interesados	- Definir los requerimientos o necesidades. - Resolver dudas que pueda tener el Equipo Scrum mediante el Product Owner - Definir los criterios de aceptación de cada tarea realizada.

Tabla 1. Roles metodología scrum

1.9. RESULTADOS ESPERADOS

Con la implementación del sistema para el registro y control de acceso y salida de motocicletas en la Facultad Tecnológica de la Universidad Distrital se esperan los siguientes resultados:

- Entrega de una aplicación web capaz de realizar el registro correcto de los estudiantes que requieran el servicio de motocicletas por primera vez.
- Dado que se utilizará la base de datos no relacional MongoDB, no existirá peligro de saturación de consultas o datos, otorgando escalabilidad en el almacenamiento de la información de los usuarios que interactúen con el sistema a diario.
- Poder ejecutar la búsqueda de un registro en específico, es decir, la información de una motocicleta o de su propietario.
- Obtener información en tiempo real de las motocicletas que se encuentran dentro de la Facultad, en donde se pueda visualizar el tiempo de entrada y salida de esta.
- Mantener un histórico de las motocicletas que entran y salen de la Facultad Tecnológica.
- Aumento de la seguridad respecto al ingreso y salida de las motocicletas, teniendo en cuenta que solo se le debe permitir el ingreso o retiro estudiantes pertenecientes a la Universidad y con un vehículo de su propiedad.
- Mejora en el flujo de entrada y salida de las motocicletas, ya que al ser un proceso que se realiza de manera manual tiende a gastar mayor tiempo.

1.10. IMPACTOS

Como primer punto, la implementación de dicho sistema ayudaría al personal encargado de la seguridad en los parqueaderos de la Facultad Tecnológica, ya que al contar con un control en el registro de las motocicletas que entran y salen, se aumentará la seguridad dentro de las instalaciones gracias al registro de entrada y salida, cupos disponibles e información del propietario de la motocicleta.

Como segundo punto, se disminuirá el error humano, que posiblemente pueda comprometer al personal de seguridad encargado de los parqueaderos de las motocicletas, ya que, al realizar un proceso totalmente manual y con el gran flujo con el que se cuenta, este podría autorizar la salida de un estudiante con una motocicleta que no es de su propiedad.

Como tercer punto, se contaría con una recopilación de bases de datos con la información relevante de los propietarios, como nombres completos, código

estudiantil, identificación y datos de contacto, pero además se contaría con información sobre el vehículo que ingrese a la facultad, como la identificación de la tarjeta de propiedad, placas, modelo y una descripción de la motocicleta, cuestión que ayudaría notablemente a verificar la identidad del estudiante y la propiedad del vehículo.

Como cuarto punto, se vería beneficiado el personal de seguridad encargados de los parqueaderos y los estudiantes de la facultad, ya que el flujo de ingreso y salida reduciría tiempos, se haría de una manera controlada y organizada.

1.11. POSIBLES AFECTACIONES EN EL SISTEMA

El personal de soporte de la Facultad Tecnológica de la Universidad Distrital Francisco José de Caldas será la responsable de restablecer el sistema en caso de alguna interrupción, según el manual expuesto en el ítem 6.1.

Se debe tener en cuenta los siguientes lineamientos:

- Al ser un aplicativo que se encuentra alojado en un computador, cualquier fallo en el mismo afectara el funcionamiento del sistema.
- Si el sistema se quiere migrar del computador donde se tiene todo el sistema junto su información, se deben seguir los pasos descritos en el ítem 6.1.
- La tecnología Mern Stack da la certeza de ser estable, dando como resultado una aplicación con bajo nivel de errores, claro está que esto depende de la construcción de la aplicación, la cual se ha probado de diversas manejas para minimizar al máximo las afectaciones. Sin embargo, el sistema puede tener caídas en los servicios asociados a NodeJS y ReactJS, en donde solo se deben restablecer, esto se describe en el ítem 6.1
- Si se habla del almacenamiento asociado al sistema, se debe tener en cuenta que el sistema consume el espacio de la máquina, si se tienen complicaciones con dicho tema se debe ajustar el almacenamiento del equipo.
- Dado que existe un gran nivel de fallas en los sistemas por mal uso del usuario final, se realiza el manual de usuario, el cual se debe tener en cuenta para minimizar complicaciones o mal uso de este. Además, este servirá de base de estudio cuando el personal de vigilancia cambie. Este se describe en el ítem 6.2.

1.11. MARCO TEÓRICO

1.11.1. MARCO HISTÓRICO

El control del ingreso y salida de los vehículos en una organización ha estado a cargo del personal de seguridad del parqueadero, el cual se encarga de realizar la verificación de la persona a ingresar o retirar un vehículo, permitiéndole o no el acceso o retiro de este.

Dentro de la Universidad Distrital se han llevado a cabo ciertas soluciones tecnológicas, para que el personal de seguridad ubicado en el parqueadero de una organización realice su trabajo apoyado en las tecnologías de la información. A continuación, se listan algunos de los proyectos realizados por estudiantes para gestionar y administrar los parqueaderos de la Universidad:

- SOFTWARE DE CONTROL DE ACCESO Y REGISTRO DE VEHÍCULOS PARA LOS PARQUEADEROS DE LA FACULTAD TECNOLÓGICA EN LA UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS, CON LECTOR DE HUELLA DIGITAL

El ingreso de vehículos de los estudiantes, docentes y administrativos a la Facultad Tecnológica de la Universidad Distrital Francisco José de Caldas se realiza de manera manual. Por este motivo se quiere implementar un sistema seguro y eficiente usando la tecnología biométrica específicamente el lector de huella dactilar, ya que es el método de identidad biométrica más usado, también es fácil de adquirir, y fácil de usar.
Esta será una aplicación de escritorio en java que tenga conexión a una base de datos donde se guardaran todos los datos de los usuarios (incluyendo su huella dactilar) y sus vehículos, esto permitirá hacer una relación entre el usuario y el vehículo garantizando que este sea el propietario de este y sea posible verificar su identificación e impedir posibles robos. (ARANGO GRISALES, APARICIO MARTINEZ, 2019)

- SISTEMA TELEMÁTICO PARA LA GESTIÓN DEL PARQUEADERO DE MOTOCICLETAS DE LA FACULTAD TECNOLÓGICA

Actualmente la sede tecnológica de la Universidad Distrital Francisco José de Caldas presenta un problema de espacios para el estacionamiento de vehículos por la gran cantidad de funcionarios, estudiantes y profesores que usan el parqueadero, adicionalmente no

se ven proyectos o soluciones para mejorar esta situación, por lo que se decide crear un aplicativo que ayude al control y manejo del parqueadero de la sede tecnología, mejorando los tiempos de espera y el orden que se tiene.
Este sistema se realizará para dispositivos móviles utilizando el Framework de Java Spring Boot. (CAMARGO RUIZ, TORREZ MENDOZA, 2020)

1.11.2. MARCO CONCEPTUAL

Para el desarrollo del sistema, se hará uso de herramientas de código abierto y de programas libres, es decir sin ningún tipo de recargo económico. Lo que garantiza que una vez se encuentre en funcionamiento las funciones realizadas no quedarán limitadas por algún tipo de licenciamiento. Las herramientas y conceptos claves a utilizar se describen a continuación:

- **Stack tecnológico**
  Un Stack tecnológico, es una lista de todas las herramientas o servicios dispuestos por la tecnología utilizados para la construcción y ejecución de una sola aplicación. (Mixpanel, 2022).
  Este conjunto de herramientas se divide entre el Backend, Frontend y bases de datos, en donde las tecnologías Backend se componen de todos los Frameworks web, lenguajes de programación, servidores y sistemas operativos, por otra parte, las tecnologías Frontend se refiere a la interfaz de usuario y por último las bases de datos contienen todo el flujo de información dentro de la aplicación.

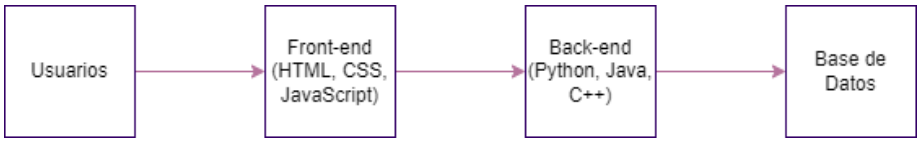


Ilustración 1. Vista simple de un Stack tecnológico

- **Mern Stack**
  Mern Stack es un Framework el cual como su nombre lo indica, es un Stack que consta de un conjunto de herramientas las cuales utilizan de JavaScript tanto en el cliente como también en el servidor, lo que genera una construcción de producto más sencilla al utilizarse en ambos frentes el mismo lenguaje de programación interpretado, Mern Stack se compone de las siguientes tecnologías:

- MongoDB
- Express JS
- React JS
- NodeJS

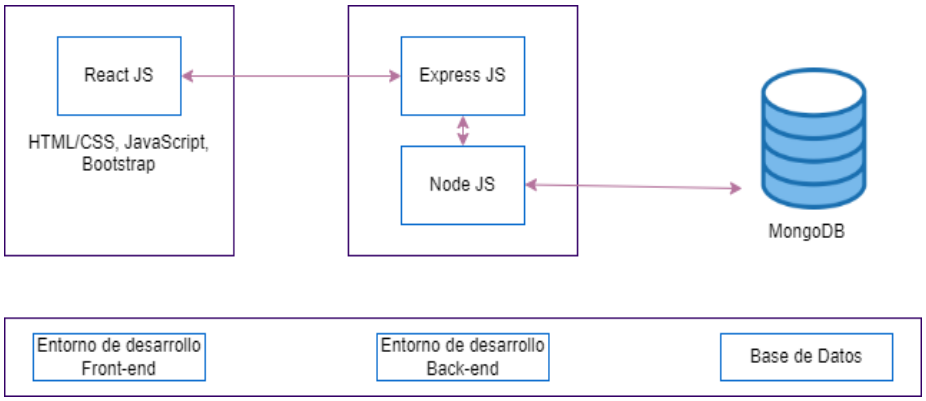


Ilustración 2. Composición del Framework Mern Stack

- **HTML**
  HTML (HyperText Markup Language, en español Lenguaje de marcado de hipertexto), es un lenguaje de marcado que se utiliza para realizar la estructura de la página web a desarrollar. (Editorial Etece, 2021)

- **CSS**
  CSS (Cascading Style Sheets, en español Hojas de estilo en cascada), es un lenguaje de estilos el cual se utiliza para dar diseño, abarcando colores, tipos de fuente, márgenes, entre otros aspectos de lo planteado en el HTML de la página web. (Peiró, 2017)

- **JavaScript**
  JavaScript es un lenguaje de programación, el cual permite crear contenido dinámico dentro de aplicaciones, en conjunto con HTML y CSS. Se caracteriza por ser útil, práctico y lo más importante se encuentra disponible para cualquier navegador web.

- **MongoDB**
  MongoDB es una base de datos distribuida NoSQL, es decir, que maneja el almacenamiento de datos en un formato diferente a las tablas relacionales, teniendo como característica principal la orientación a documentos.

Lo anteriormente mencionado hace referencia a cómo se guardan los datos, a diferencia de las bases de datos relacionales que guardan la información en tablas, estas lo hacen mediante estructuras BSON (representación binaria de JSON), mejorando el rendimiento de los datos en las aplicaciones en que se implemente. (Rubenfa, 2014).

- **Express.JS**
  Express es un Framework o marco de trabajo, el cual se desarrolla dentro de NodeJS y está basado en JavaScript que se ejecuta del lado del servidor, lo que lo lleva a crear sitios web de una manera sencilla (Parada, 2020).

- **React JS**
  React JS es una biblioteca JavaScript para crear interfaces de usuario. (React, 2022). Esta es de código abierto consta de una arquitectura la cual está basada en componentes, las cuales son fragmentos de código en donde utiliza HTML, CSS y JavaScript. Lo que permite una creación de interfaces más llamativas en las diversas aplicaciones web o donde se utilice dicha librería.

- **NodeJS**
  NodeJS es un entorno en el cual en tiempo de ejecución multiplataforma para la capa del servidor (en el lado del servidor), este tiene como objetivo construir productos los cuales sean escalables gracias a que es un entorno controlado por eventos, le permite a su vez establecer y gestionar múltiples conexiones simultáneamente, por otra parte, está diseñado para facilitar la comunicación. (Simoes, 2021)

- **API RESTful**
  Una Api de REST es una interfaz de programación de aplicaciones, que permite interactuar entre servicios para poder obtener información o ejecutar funciones asociadas a esta, de manera que se comprenda la solicitud realizada y se pueda dar una respuesta. (RED HAT, 2020)

- **Framework**
  Framework hace referencia al marco o esquema de trabajo utilizado para el desarrollo de software, ayuda a la simplificación de código e incita a usar buenas prácticas, aumentando productividad, alcance de nuevos componentes, módulos y herramientas, reutilización en otros proyectos y escalabilidad a futuro. (Edix, 2022)

- **Bootstrap**
  Bootstrap es un Framework CSS, desarrollado por Twitter en 2021, para la estandarización de las herramientas de su propia compañía. (Guest, 2022). Esta herramienta es una integración de CSS y JavaScript para dar estilo y además agregar interactividad a las funciones de la página; se usa comúnmente en menús de usuario, controles de página, barras de estado y progreso, entre otras.
  Entre las virtudes que brinda este Framework, se caracteriza el diseño responsive de un sitio web, que hace referencia a cómo se ajusta un contenido en la pantalla del navegador, otra característica son las alertas, la cual es la manera de mostrar mensajes a los usuarios cuando se necesita advertir, informar o comunicar un proceso y por último la barra de navegación, que en pocas palabras son los menús que ayudarán al usuario a moverse por la página web. (Deyimar, 2022)

- **JSON**
  JSON (JavaScript Object Notation, en español Notación de Objetos de JavaScript) es un formato de datos, fácil de leer para los humanos y de interpretar por la máquina, dando como resultado un intercambio de datos más sencillo, rápido y confiable. (JSON, 2021)

  Una de las principales ventajas de utilizar JSON, es que puede ser utilizado, prácticamente en cualquier lenguaje de programación, como por ejemplo Java, JavaScript, PHP, C, C++, Python, entre muchos más. (Oracle, 2022)

- **Protocolo HTTP**
  HTTP (Hypertext Transfer Protocol, en español Protocolo de transferencia de hipertexto), es el protocolo de internet que permite las transferencias de información en la World Wide Web (WWW). Los diferentes modelos de comunicación funcionan gracias al protocolo HTTP el cual la hace posible por medio de transacciones entre las partes, generando peticiones o recibiendo respuestas que se pueden visualizar en el navegador. (El Comercio, 2022)

- **Middleware**
  El middleware es un software el cual permite que distintas aplicaciones logren comunicarse entre ellas, tiene la facultad de conectarlas de una manera inteligente y eficiente, lo que conlleva

a que se innove de una forma más ágil y rápida. La funcionalidad del middleware es actuar como un puente entre tecnologías, herramientas y bases de datos diversas para que pueda integrarse sin dificultad en un único sistema. Debido a este sistema único se puede proveer de un servicio unificado a los usuarios finales. Por ejemplo, una aplicación Frontend de Windows envía y recibe datos desde un servidor Backend de Linux, pero los usuarios de la aplicación no están al tanto de la diferencia. (AWS, 2022).

- **Postman**
  Postman es una aplicación con una versión gratuita y de pago, la cual permite realizar peticiones HTTP a una API (Application programming interface, en español Interfaz de programación de aplicaciones) sin importar si es propia o de terceros. Se utiliza principalmente para ejecutar pruebas de funcionalidad, aunque también permite consumir y depurar la API. Además de lo anteriormente mencionado, permite generar documentación, gestionar el ciclo de vida, es decir, definición, desarrollo, monitoreo y mantenimiento del API evaluada. (Marius Chistensen, 2019).
  Actualmente el uso de Postman se ajusta a los diferentes sistemas operativos del mercado como lo son Windows, Linux y Mac, lo que permite que su uso no se limite a un solo entorno. (Muradas, 2019)

- **Visual Studio Code**
  Visual Studio Code es un editor de código fuente ligero pero potente que se ejecuta en el escritorio y está actualmente disponible para Windows, macOS y Linux. Viene con soporte integrado para JavaScript, TypeScript y Node.js y tiene un rico ecosistema de extensiones para otros lenguajes y tiempos de ejecución (como C++, C#, Java, Python, PHP, Golang, .NET). (Microsoft, 2021).

- **Arquitectura basada en capas**
  En esta arquitectura las capas son una forma de separar responsabilidades y administrar dependencias. Se establece que cada capa tiene una responsabilidad específica. Un aspecto a tener en cuenta es que una capa superior puede utilizar los servicios de una capa inferior, pero no al revés. Los niveles están físicamente separados y se ejecutan en máquinas diferentes. (Azure, 2022).

- **Lector de código de barras**
  Es un dispositivo de lectura de datos que utiliza un láser incorporado para leer un código de barras y emitir la información que representa el código. (BRADY, 2022)

1.11.3. MARCO LEGAL

Para la realización del proyecto “Sistema De Registro Y Control De Acceso Y Salida De Motocicletas En La Facultad Tecnológica De La Universidad Distrital Francisco José De Caldas basado en API RESTful”, se basa en la construcción política de Colombia del año 1991 y la reglamentación realizada por la Universidad Distrital Francisco José de Caldas. Donde se toma como referencia el acuerdo 004 de octubre 11 de 2012, artículo 6 titulado “Acceso a los recursos genéticos con el acuerdo de propiedad intelectual” (Distrital, 2012) en el cual se mencionan la intervención que realiza los diferentes autores en un proyecto como el autor y el director de proyecto de grado, por lo cual el desarrollo de un trabajo de grado hace parte de la Universidad Distrital Francisco José de Caldas, debido a que se utilizan de sus recursos para su ejecución. De acuerdo con lo anterior, se dispone a la firma de un documento, el cual es un acta de compromiso de propiedad intelectual.

Dentro de la realización de la aplicación web se llevará a cabo una recopilación de datos personales los cuales serán tratados de acuerdo a la ley estatutaria 1581 de 2012 - Ley de Protección de Datos Personales, en donde la presente ley tiene por objeto desarrollar el derecho constitucional que tienen todas las personas a conocer, actualizar y rectificar las informaciones que se hayan recogido sobre ellas en bases de datos o archivos, y los demás derechos, libertades y garantías constitucionales a que se refiere el artículo 15 de la Constitución Política; así como el derecho a la información consagrado en el artículo 20 de la misma (Gov, 2012).

Según el decreto 2851 de 2013, el cual hace referencia a la educación vial en el país, se toma como referencia el Capítulo IV, Planes Estratégicos de las Entidades, Organizaciones o empresas en materia de Seguridad Vial, Acción c. Vehículos Seguros. este con el fin de garantizar que las motocicletas que entren a la Facultad tecnológica se encuentren a salvo y sea solo el propietario el autorizado para realizar su retiro. (Gov, 2013)

En términos de derechos de autor se basa en la ley 23 de 1982 de Colombia sobre los derechos de autor, artículo 2, en la cual dice que “Los derechos de autor recaen sobre las obras científicas, literarias y

artísticas, las cuales comprenden todas las creaciones del espíritu en el campo científico, literario y artístico, cualquiera que sea el modo o forma de expresión y cualquiera que sea su destinación…", (Gov, 1982) protegerá al proyecto en el desarrollo de software bajo el aspecto de derechos de autor.

En cuanto a los derechos económicos y de propiedad intelectual utilizados dentro de la aplicación web a implementar, se ceden a la Universidad Distrital Francisco José de Caldas, para que ésta haga uso de este recurso, según convenga. Por ser una universidad pública, el software a realizar será un bien público, lo cual será gratuito para el personal que lo utilice.

1.11.4. MARCO REFERENCIAL

La implementación la aplicación web, que realiza un control de acceso y salida de motos en la Facultad tecnológica, ayuda a prever riesgos de hurto y a su vez solucionar problemas de congestiones en el ingreso a la facultad haciendo que este sea de una manera óptima, esta aplicación permitirá monitorear de manera local el flujo de motos que están dentro de la facultad, lo que ayudará en el desempeño de las labores del personal de seguridad. Esta automatización de procesos a través de aplicaciones web potencializan las actividades con las que se relacionan, como se pudo observar en el marco histórico con las aplicaciones implementadas dentro de la universidad.

Frente a una situación donde se requiera conocer el horario en que más se presenta un alto flujo de motos dentro de la facultad, con el consolidado de información se pueden obtener estos datos, lo que puede permitir llevar a cabo la ejecución de planes y estrategias del personal encargado para tener un mejor control en estos periodos de tiempo.

En este sentido, la capacitación sobre la aplicación web a las personas que interactúan con ella, es necesaria para que se pueda aprovechar al máximo la automatización del proceso el cual actualmente es manual, lo que está generando una problemática en la facultad.

1.12. FACTIBILIDAD

En el siguiente apartado, se describen los factores que influyen en el desarrollo del proyecto, en términos técnicos, operativos, económicos y legales.

1.12.1. FACTIBILIDAD OPERATIVA

Para que el proyecto pueda llevarse a cabo de la mejor manera, se necesita de recurso humano capacitado, el cual realice cada una de las etapas necesarias y finalmente entregue consigo un proyecto a la altura de la Facultad Tecnológica de la Universidad Distrital.

Integrantes	Función
Santiago Lopez Gomez	Estudiante y realizador del proyecto
Jeisson Camilo Rojas Sánchez	Estudiante y realizador del proyecto
Luis Felipe Wanumen Silva	Tutor y Encargado del seguimiento del proyecto

Tabla 2. Integrantes del proyecto

1.12.2. FACTIBILIDAD TÉCNICA

En relación con las herramientas necesarias para la culminación del proyecto son accesibles, desde la cantidad de herramientas hasta el precio de cada una de estas. Lo que significa que el desarrollo del proyecto resulta ser factible técnicamente.

Producto	Cantidad	Descripción
Computador de escritorio	1	Computador All in One HP 21.5" Pulgadas dd1506la Intel Core i3 - RAM 8GB - Disco SSD 256 GB - Negro
Windows 10	1	Sistema Operativo Windows Pro
Lector de código de barras	1	Lector De Códigos De Barras Láser Sat Ld101r Plus Con Base

Tabla 3. Productos para utilizar en el desarrollo del proyecto

1.12.3. FACTIBILIDAD ECONÓMICA

Es esencial para el desarrollo del proyecto dar a conocer los costos que traen consigo todos y cada uno de los implementos que se utilizaran dentro de la elaboración del producto, de esta forma, se pueden encontrar tanto los dispositivos de hardware, como los de software, los cuales fueron utilizados para el planteamiento, desarrollo y entrega de la aplicación web, por otra parte, se tiene los costos ligados a los

recursos humanos y por último se tienen los costos varios que son igualmente necesarios. Lo anteriormente mencionado dará a conocer el coste total del proyecto dando a entender la factibilidad económica del proyecto.

1.12.3.1. RECURSOS DE HARDWARE

Recurso	Descripción	Cant	Valor Unitario	Total
Computador de escritorio	Computador All in One HP 21.5" Pulgadas dd1506la Intel Core i3 - RAM 8GB - Disco SSD 256 GB - Negro	1	$1’899.000	$1’899.000
Lector de código de barras	Lector De Códigos De Barras Láser Sat Ld101r Plus Con Base	1	$ 125.000	$125.000
				Total: $2’024.000

Tabla 4. Recursos de Hardware

1.12.3.2. RECURSOS DE SOFTWARE

Recurso	Descripción	Cant	Valor Unitario	Total
Windows 10 Pro	Sistema Operativo	1	$ 1.099.999	$1.099.999
Visual Studio Code	Editor de código	1	$0	$0
MongoDB	Base de datos	1	$0	$0
MongoDB Compass	Interfaz base de datos	1	$0	$0
Servidor	Servidor	2	$0	$0

	Local			
				Total: $1.099.999

Tabla 5. Recursos de Software

1.12.3.4. COSTOS VARIOS

Nombre	Horas	Valor - Hora	Total
Energía	720 horas	720 * $2.000	$ 1.440.000
Servicio de Internet	6 meses	Valor mes $65.900	$ 395.400
			Total: $1.835.400

Tabla 6. Costos Varios

1.12.3.5. COSTOS TOTALES

Nombre	Total
Recursos de Hardware	$2’024.000
Recursos de Software	$1.099.999
Costos varios	$1.835.400
	Total: $4’959.399

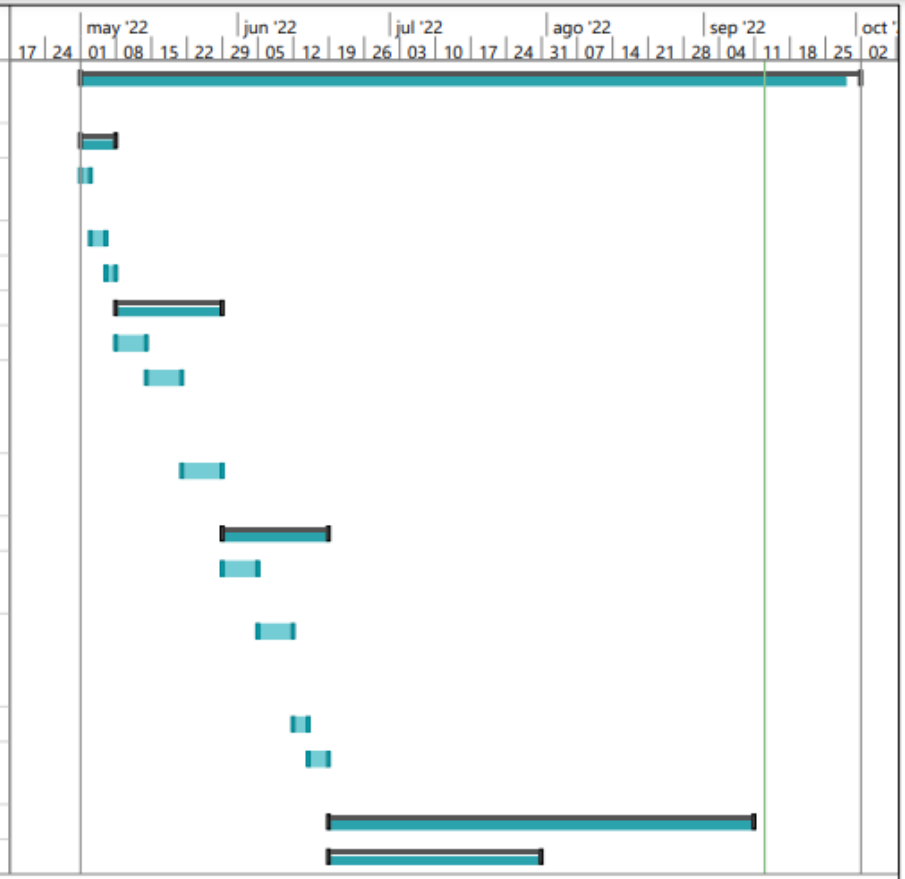
Tabla 7. Costos totales del proyecto

1.12.4. FACTIBILIDAD LEGAL

Es importante destacar que, para el desarrollo del proyecto, se utilizará una pila de herramientas, las cuales vienen dadas por el Framework MERN Stack, el cual utiliza la integración de varias tecnologías, entre ellas se encuentran MongoDB, ExpressJS, ReactJS y NodeJS, por lo tanto, al ser basadas en el lenguaje de programación JavaScript son Open Source, es decir, sin ningún tipo de costo asociado a su uso. Por otra parte, los entornos de desarrollo con los cuales se lleva a cabo la elaboración del producto, los cuales fueron, Visual Studio Code y Postman también son Open Source y en consecuencia no existe ningún trámite de tipo legal, debido a que se trabaja bajo la licencia de uso público GNU GPL (General Public License).

1.13. CRONOGRAMA

Id	Nombre de tarea	Comienzo	Fin
1	**Proyecto Motociclates Factultad Tecnologica**	**dom 01/05/22**	**sáb 01/10/22**
2	**Fase de planeación**	**dom 01/05/22**	**sáb 07/05/22**
3	Planteamiento del problema	dom 01/05/22	lun 02/05/22
4	Delimitación	mar 03/05/22	jue 05/05/22
5	Justificación	vie 06/05/22	sáb 07/05/22
6	**Fase de análisis**	**dom 08/05/22**	**sáb 28/05/22**
7	Modelado del negocio	dom 08/05/22	vie 13/05/22
8	Definición de la arquitectura de software	sáb 14/05/22	vie 20/05/22
9	Parametrización proyecto	sáb 21/05/22	sáb 28/05/22
10	**Fase de diseño**	**dom 29/05/22**	**sáb 18/06/22**
11	Estructuración del sistema	dom 29/05/22	sáb 04/06/22
12	Diseño de microservicios (Back-end y Front-end)	dom 05/06/22	sáb 11/06/22
13	Plantemiento de vistas	dom 12/06/22	mar 14/06/22
14	Estructura de parametros	mié 15/06/22	sáb 18/06/22
15	**Fase de implementación**	**dom 19/06/22**	**sáb 10/09/22**
16	**Back-end**	**dom 19/06/22**	**sáb 30/07/22**

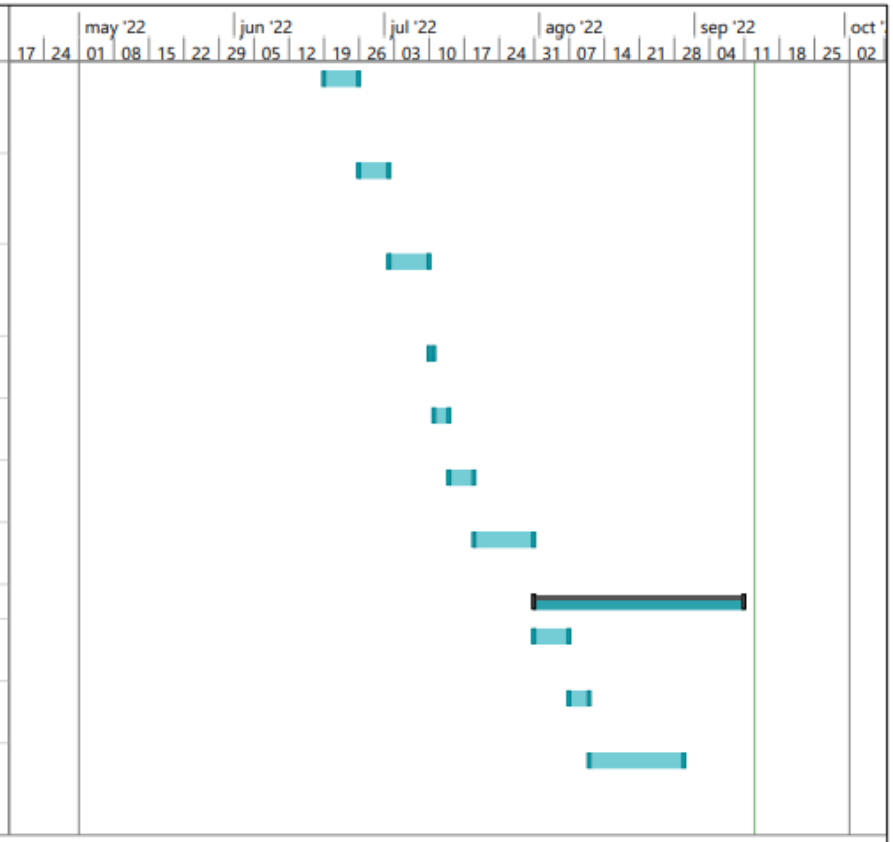


Proyecto: Proyecto1
Fecha: mar 13/09/22

Tarea		Resumen inactivo		Tareas externas	
División		Tarea manual		Hito externo	
Hito		solo duración		Fecha límite	
Resumen		Informe de resumen manual		Progreso	
Resumen del proyecto		Resumen manual		Progreso manual	
Tarea inactiva		solo el comienzo			
Hito inactivo		solo fin			

Página 1

Id	Nombre de tarea	Comienzo	Fin
17	Estructuración proyecto NodeJS con ExpressJS	dom 19/06/22	sáb 25/06/22
18	Instalación de modulos principales (Dependencias)	dom 26/06/22	vie 01/07/22
19	Estructuración de base de datos MongoDB	sáb 02/07/22	sáb 09/07/22
20	Definición del puerto del servidor	dom 10/07/22	dom 10/07/22
21	Implementacíon de Middleware	lun 11/07/22	mié 13/07/22
22	Definición del enrutamiento	jue 14/07/22	lun 18/07/22
23	Creación de servicios REST	mar 19/07/22	sáb 30/07/22
24	**Font-end**	**dom 31/07/22**	**sáb 10/09/22**
25	Estructuración proyecto React	dom 31/07/22	sáb 06/08/22
26	Definición de enrutamiento	dom 07/08/22	mié 10/08/22
27	Implementación del diseño (HTML, CSS, Bootstrap)	jue 11/08/22	lun 29/08/22



Proyecto: Proyecto1
Fecha: mar 13/09/22

Tarea		Resumen inactivo		Tareas externas	
División		Tarea manual		Hito externo	
Hito		solo duración		Fecha límite	
Resumen		Informe de resumen manual		Progreso	
Resumen del proyecto		Resumen manual		Progreso manual	
Tarea inactiva		solo el comienzo			
Hito inactivo		solo fin			

Página 2

Id	Nombre de tarea	Comienzo	Fin
28	Consumo de servicios REST	mar 30/08/22	mar 06/09/22
29	Diagramas funcionales	mié 07/09/22	sáb 10/09/22
30	**Fase de transición**	**dom 11/09/22**	**mié 28/09/22**
31	Creación de manuales	dom 11/09/22	mar 20/09/22
32	Capacitación al personal	mié 21/09/22	vie 23/09/22
33	Entrega proyecto	sáb 24/09/22	mié 28/09/22

Proyecto: Proyecto1
Fecha: mar 13/09/22

Tarea		Resumen inactivo		Tareas externas	
División		Tarea manual		Hito externo	
Hito		solo duración		Fecha límite	
Resumen		Informe de resumen manual		Progreso	
Resumen del proyecto		Resumen manual		Progreso manual	
Tarea inactiva		solo el comienzo			
Hito inactivo		solo fin			

Página 3

Ilustración 3. Cronograma

CAPÍTULO 2

2. ANÁLISIS

2.1. PRODUCT BACKLOG

La lista del producto o Product Backlog es el consolidado que ordena toda actividad que podría ser necesaria para la ejecución del producto, y es la única fuente de requisitos para poder realizar cambios en el producto. En esta lista se enumeran todas las características, funcionalidades, requerimientos, requisitos, mejoras y correcciones que pueden ser solicitadas en entregas posteriores.

En el transcurso del proyecto se desarrollarán las siguientes actividades basadas en el marco de la metodología Scrum:

ITEM	ACTIVIDAD
1	El sistema debe permitir la creación de nuevos estudiantes.
2	El sistema debe permitir la verificación de los estudiantes ya creados.
3	El sistema debe permitir al administrador crear una cantidad ilimitada de motocicletas asociadas a un estudiante.
4	El sistema debe permitir la edición de un estudiante.
5	El sistema no debe permitir la eliminación de un estudiante
6	El sistema debe permitir la visualización de los estudiantes dentro de la facultad
7	El sistema debe permitir la búsqueda de estudiantes registrados

Tabla 8. Actividades principales para realizar en el desarrollo del proyecto

2.1.1. PRODUCT BACKLOG

En la siguiente tabla se muestra el listado de actividades que se desarrollaran durante el transcurso del proyecto:

ITEM	ACTIVIDAD

1	Consulta de estudiante
2	Consulta de estudiante con lector de código de barras
3	Registro de estudiante
4	Edición de estudiante
5	Eliminación de estudiantes
6	Consulta de motocicletas
7	Registro de motocicletas
8	Edición de motocicletas
9	Eliminación de motocicletas
10	Consulta de listado actual y listado histórico

Tabla 9. Product Backlog

2.1.1.1. HISTORIAS DE USUARIO

Las historias de usuario son la descripción de los requerimientos de los interesados del proyecto de una manera sencilla. Se deben tener en cuenta la relación de los roles, funcionalidad y el resultado esperado con cada uno de los requerimientos.

Entre los campos esenciales en las historias de usuario se encuentran los siguientes: [3]

- **ID o Número:** Es el identificador de la historia de usuario, este debe ser único.
- **Usuario:** Hace referencia al interesado o la persona que solicita dicho requerimiento.
- **Nombre historia:** Es el nombre de la historia de usuario, debe ser conciso y claro.
- **Prioridad en negocio:** Se refiere al valor que representa dicha historia para el negocio.
- **Riesgo en desarrollo:** Es el riesgo técnico o funcional que puede estar asociado a la implementación de la historia de usuario.
- **Descripción:** Es un texto breve que describe la historia de usuario.
- **Validación:** Son las pruebas que se le realizan al requerimiento solicitado, puede ser realizada con el interesado.

[3] SCRUM, Manager, Historias de usuario, (2022), [Consultado el 17, septiembre, 2022]. Disponible en internet: <https://scrummanager.net/files/scrum_manager_historias_usuario.pdf>

A continuación, se mencionan las listas de usuario relacionadas con el desarrollo del proyecto:

Historia de Usuario	
Número:	1
Usuario:	Administrador
Nombre historia	Consulta de estudiante
Prioridad en negocio:	Alta
Riesgo en desarrollo:	Bajo
Estimación:	8 horas
Descripción: El usuario administrador al ingresar al aplicativo debe contar con la posibilidad de consultar si el estudiante está registrado o no en la base de datos de manera manual, en caso de que el lector de código de barras no se encuentre en funcionamiento.	
Validación: ● Debe existir un cuadro de texto en el cual el administrador ingrese el código del estudiante. ● Se debe generar una alerta que muestre si el usuario está registrado en el sistema o no. ● Si el estudiante existe debe mostrar la información asociada. ● Si el estudiante no existe debe permitir realizar el registro de este.	

Tabla 10. Historia de usuario numero 1

Historia de Usuario	
Número:	2
Usuario:	Administrador
Nombre historia	Consulta de estudiante por código de barras

Prioridad en negocio:	Alta
Riesgo en desarrollo:	Bajo
Estimación:	16 horas
Descripción: El usuario administrador al ingresar al aplicativo debe contar con la posibilidad de consultar si el estudiante está registrado o no en la base de datos con la ayuda de un lector de código de barras.	
Validación: ● El lector de código de barras deberá leer el código de barras alojado en el carné estudiantil del estudiante. ● Se debe generar una alerta que muestre si el usuario está registrado en el sistema o no. ● Si el estudiante existe debe mostrar la información asociada. ● Si el estudiante no existe debe permitir realizar el registro de este.	

Tabla 11. Historia de usuario numero 2

	Historia de Usuario
Número:	3
Usuario:	Administrador
Nombre historia	Registro de estudiante
Prioridad en negocio:	Alta
Riesgo en desarrollo:	Bajo
Estimación:	10 horas
Descripción: El usuario administrador al ingresar al aplicativo debe contar con la posibilidad de registrar los estudiantes de manera manual, ya que el lector de código de barras solo captura el código estudiantil del	

estudiante.
Validación: ● Debe existir un formulario que cuente con los datos básicos del estudiante, nombre completo, identificación y código estudiantil ● Una vez diligenciados todos los datos del estudiante se deberá realizar la respectiva verificación, con el fin de no ingresar registros duplicados en el sistema.

Tabla 12. Historia de usuario numero 3

	Historia de Usuario
Número:	4
Usuario:	Administrador
Nombre historia	Edición de estudiante
Prioridad en negocio:	Normal
Riesgo en desarrollo:	Bajo
Estimación:	4 horas
Descripción: El usuario administrador podrá editar la información del estudiante en caso de encontrar algún dato erróneo.	
Validación: ● Debe existir una opción que permite editar la información del estudiante. ● El formulario de edición debe traer todos los datos asociados al estudiante.	

Tabla 13. Historia de usuario numero 4

Historia de Usuario

Número:	5
Usuario:	Administrador
Nombre historia	Eliminación de estudiantes
Prioridad en negocio:	Normal
Riesgo en desarrollo:	Bajo
Estimación:	4 horas
Descripción: El usuario administrador podrá eliminar estudiantes en caso de haber realizado un mal registro.	
Validación: ● Debe existir una opción que permite eliminar un estudiante registrado. ● Debe generar una alerta en caso de oprimir eliminar usuario por equivocación o tan solo para confirmar la eliminación.	

Tabla 14. Historia de usuario numero 5

Historia de Usuario	
Número:	6
Usuario:	Administrador
Nombre historia	Consulta de motocicletas
Prioridad en negocio:	Alta
Riesgo en desarrollo:	Bajo
Estimación:	8 horas
Descripción: El usuario administrador podrá consultar todas las motocicletas que	

el estudiante tenga a su nombre.
Validación: ● El sistema debe permitir realizar la consulta de las motocicletas asociadas a un estudiante por medio de su código estudiantil. ● Si el estudiante no cuenta con ninguna motocicleta registrada debe mostrar una alerta y permitir registrar los datos. ● Si el estudiante cuenta con una o más motocicletas registradas deberá mostrar el listado.

Tabla 15. Historia de usuario numero 6

Historia de Usuario	
Número:	7
Usuario:	Administrador
Nombre historia	Registro de motocicletas
Prioridad en negocio:	Alta
Riesgo en desarrollo:	Bajo
Estimación:	10 horas
Descripción: El usuario administrador podrá registrar las motocicletas asociadas a un determinado estudiante, este proceso se deberá realizar de manera manual.	
Validación: ● El sistema debe contar con un formulario de registro de motocicletas, el cual genere una relación entre el estudiante y la motocicleta. ● El sistema debe permitir al administrador realizar ilimitados registros de motocicletas asociadas a un estudiante en específico.	

Tabla 16. Historia de usuario numero 7

Historia de Usuario	
Número:	8
Usuario:	Administrador
Nombre historia	Edición de motocicletas
Prioridad en negocio:	Normal
Riesgo en desarrollo:	Bajo
Estimación:	4 horas
Descripción: El usuario administrador podrá editar las motocicletas asociadas a un determinado estudiante, este proceso se deberá realizar de manera manual.	
Validación: ● El sistema debe contar con un formulario de edición de motocicletas, el cual genere una relación entre el estudiante y la motocicleta. ● Se debe generar una alerta en la cual se acepte o rechace la edición realizada.	

Tabla 17. Historia de usuario numero 8

Historia de Usuario	
Número:	9
Usuario:	Administrador
Nombre historia	Eliminación de motocicletas
Prioridad en negocio:	Normal
Riesgo en desarrollo:	Bajo

Estimación:	4 horas
Descripción: El usuario administrador podrá eliminar las motocicletas asociadas a un determinado estudiante, en caso de existir un registro erróneo.	
Validación: ● Debe existir una opción que permite eliminar una motocicleta registrada. ● Debe generar una alerta en caso de oprimir eliminar motocicleta por equivocación o tan solo para confirmar la eliminación.	

Tabla 18. Historia de usuario numero 9

Historia de Usuario	
Número:	10
Usuario:	Administrador
Nombre historia	Consulta de listado actual y listado histórico
Prioridad en negocio:	Bajo
Riesgo en desarrollo:	Bajo
Estimación:	8 horas
Descripción: El usuario administrador podrá consultar el listado actual y el listado histórico de las motocicletas que han interactuado con la universidad.	
Validación: ● Se debe visualizar los listados generados por el administrador	

Tabla 19. Historia de usuario numero 10

2.1.1.2. PRIORIZACIÓN Y ESTIMACIÓN DEL PRODUCT BACKLOG

Utilizando como base la prioridad descrita en cada una de las historias de usuario descritas en el inciso 2.1.1, se realiza la priorización de las actividades descritas.

ITEM	ACTIVIDAD
3	Registro de estudiante
2	Consulta de estudiante por código de barras
1	Consulta de estudiante
4	Edición de estudiante
5	Eliminación de estudiantes
7	Registro de motocicletas
6	Consulta de motocicletas
8	Edición de motocicletas
9	Eliminación de motocicletas
10	Consulta de listado actual y listado histórico

Tabla 20. Priorización y estimación del Product Backlog

Para la estimación del Product Backlog se utiliza la estrategia “Planning Poker”, descrita en el inciso 1.8.1, ya que ayudará a organizar en base a la dificultad de realización de cada una de las historias de usuarios. A continuación, se muestra la lista de actividades con sus respectivas estimaciones en horas.

ITEM	ACTIVIDAD	Estimación (horas)
1	Consulta de estudiante	8
2	Consulta de estudiante por código de barras	16
3	Registro de estudiante	10
4	Edición de estudiante	4
5	Eliminación de estudiantes	4
6	Consulta de motocicletas	8

7	Registro de motocicletas	10
8	Edición de motocicletas	4
9	Eliminación de motocicletas	4
10	Consulta de listado actual y listado histórico	8
Total en horas		**76**

Tabla 21. Priorización y estimación del Product Backlog según Planning Poker

Para poder visualizar de mejor manera la estimación realizada en horas, se ordena de mayor a menor.

ITEM	ACTIVIDAD	Estimación (horas)
2	Consulta de estudiante por código de barras	16
3	Registro de estudiante	10
7	Registro de motocicletas	10
1	Consulta de estudiante	8
6	Consulta de motocicletas	8
10	Consulta de listado actual y listado histórico	8
4	Edición de estudiante	4
5	Eliminación de estudiantes	4
8	Edición de motocicletas	4
9	Eliminación de motocicletas	4
Total en horas		**76**

Tabla 22. Priorización y estimación del Product Backlog según Planning Poker ordenado por estimación

2.1.2. DEFINICIÓN DE USUARIOS

En el aplicativo para el control de flujo de las motocicletas en la Facultad Tecnológica de la Universidad Distrital, se cuenta con solo un tipo de usuario el cual se describe a continuación:

USUARIO	FUNCIÓN
Administrador	Este tipo de usuario será el encargado de gestionar el aplicativo, donde será el responsable de la creación, modificación y eliminación de usuario y motocicletas si se requiere. Del mismo modo será el responsable y única persona que podrá ver los listados actuales e históricos en el aplicativo.

Tabla 23. Definición de usuarios

CAPÍTULO 3

3. DISEÑO

3.1. DEFINICIÓN DE SPRINTS

Según la terminología Scrum en el inciso 1.8.1 los Sprint en la metodología Scrum, hacen referencia a aquellos ciclos o iteraciones expresadas en tiempo, que requiere el equipo de desarrollo Scrum para completar las tareas definidas en el Product Backlog y en los requerimientos plasmados en las historias de usuarios.

El desarrollo de un Sprint suele estar entre 4 semanas como máximo, en donde se siguen los siguientes pasos: Planeación del Sprint, Refinamiento del Sprint, Revisión del Sprint y Retrospectiva del Sprint. [4]

3.1.1. PLANEACIÓN DEL SPRINT

En la planeación de un Sprint se realizan reuniones de trabajo con todo el equipo de trabajo, no suele tener un tiempo definido, se busca dejar cualquier duda resuelta, antes de iniciar con su implementación. En esta fase se tiene en cuenta las tareas a realizar y cómo realizarlas, tiempos, distribución en el equipo entre otros, cabe resaltar que si algo no queda claro se tienen los Scrum Diarios, que son reuniones cortas en donde se podrán retomar los temas que no quedaron claros.

3.1.2. REFINAMIENTO DEL BACKLOG

En esta fase se definen las actividades a realizar, estas deben estar directamente relacionadas con los requerimientos de los interesados y las historias de usuario.

3.1.3. REVISIÓN DEL SPRINT

En esta fase se realizará la entrega de resultados acordes a lo realizado con el fin de recibir la retroalimentación acorde.

3.1.4. RETROSPECTIVA DEL SPRINT

Es la fase final del Sprint, en donde se realiza la evaluación de los resultados entregados en la revisión del Sprint, se menciona el desempeño del equipo del trabajo y finalmente se dan a conocer las correcciones si es necesario.

[4] REHKOPF, De Max, Atlassian, Sprints de scrum, (2022). [Consultado el 17, septiembre, 2022]. Disponible en internet: <https://www.atlassian.com/es/agile/scrum/sprints>

3.2. SPRINT 1

3.2.1. PLANEACIÓN DEL SPRINT

Según lo descrito en el Product Backlog en el inciso 2.1.1, se seleccionan las siguientes actividades a desarrollar:

ITEM	NOMBRE	ACTOR
3	Registro de estudiante	Administrador

Tabla 24. Planeación Sprint 1

3.2.2. REFINAMIENTO DEL BACKLOG

Con el fin de actualizar el listado Backlog se definen las siguientes actividades que ayudarán a la finalización de la historia de usuario número 3. Registro de estudiantes definida en la planeación del Sprint:

ITEM	NOMBRE	HISTORIA DE USUARIO	TIEMPO (horas)
1	Diseño del logotipo del aplicativo	1	3 hora
2	Diseño del formulario de registro de estudiantes	1	6 horas
3	Implementación de la vista para el registro de estudiantes	1	4 horas
4	Validación de campos del formulario de registro de estudiantes	1	2 horas

Tabla 25. Refinamiento del Backlog Sprint 1

3.2.3. REVISIÓN DEL SPRINT

Se realiza los entregables asociados a las actividades propuestas en el refinamiento del backlog, entre los cuales se encuentran:

- Se diseña el logotipo del aplicativo en formato .JPG (Ilustración 4) correspondiente al punto de refinamiento 1 del Backlog (Tabla 25).
- Se crea el diseño del formulario de registro de estudiantes (Ilustración 5) correspondiente al punto de refinamiento 2 del Backlog (Tabla 25).

- Se implementa del formulario de registro de estudiantes basado en el diseño realizado, (Ilustración 6) correspondiente al punto de refinamiento 3 del Backlog (Tabla 25).
- Se realiza la verificación de campos asociados al formulario de registro de estudiantes, es decir, en los campos de nombres y apellidos solo se deberán aceptar caracteres, en los campos de cédula y código estudiantil solo se deben aceptar números, esto se logra apreciar en los campos de texto, en donde los primeros no tienen ninguna marca y los segundos cuentan con scroll de aumento y disminución de numero (Ilustración 7), además en el código generado se especifica el tipo de dato a recibir y el sistema obliga al usuario a ingresar correctamente (Ilustración 8) correspondiente al punto de refinamiento 5 del Backlog (Tabla 25).

Ilustración 4. Logotipo del aplicativo

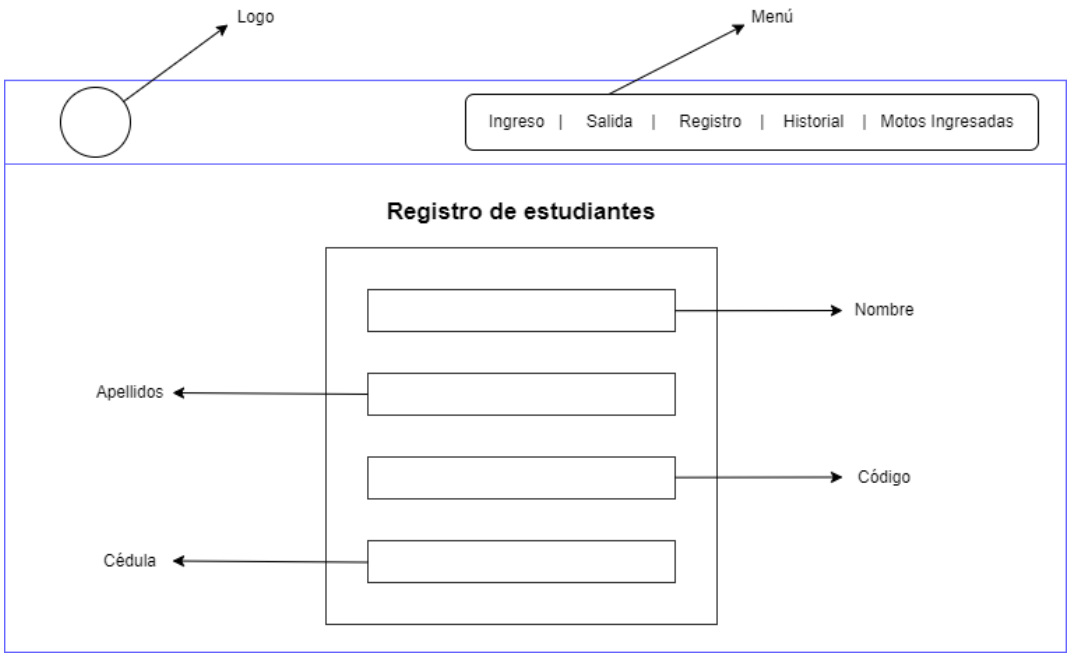


Ilustración 5. Diseño del formulario de registro de estudiantes

Ilustración 6. Implementación de la vista para el registro de estudiantes

Ilustración 7. Vista de validación de campos

```
return (
  <div className={styles.container}>
    <h2>Registro Estudiante</h2>
    <form onSubmit={sendData} className={`d-flex flex-column align-items-center ${styles.form}`}>
      <div className={styles.formContainer}>
        <h3>Nombre:</h3>
        <input required value={name} onChange={handleName} className={styles.formContainer_input}
        type="text" placeholder='Digite nombre del estudiante...' />
        <h3>Apellido:</h3>
        <input required value={lastName} onChange={handleLastName} className={styles.formContainer_input}
        type="text" placeholder='Digite apellido del estudiante...' />
        <h3>Código:</h3>
        <input required value={code} onChange={handleCode} className={styles.formContainer_input}
        type="number" placeholder='Digite código del estudiante...' />
        <h3>Cédula:</h3>
        <input required value={id} onChange={handleId} className={styles.formContainer_input}
        type="number" placeholder='Digite cédula del estudiante...' />
      </div>
      <button type='submit' className={styles.formContainer_button}>Continuar</button>
    </form>
  </div>
);
```

Ilustración 8.Validación de campos del formulario de registro de estudiantes

3.2.4. RETROSPECTIVA DEL SPRINT

Se concluye de manera exitosa el Sprint, todas las actividades asociadas al refinamiento del Backlog fueron entregadas y se encuentran funcionando de manera correcta.

3.3. SPRINT 2

3.3.1. PLANEACIÓN DEL SPRINT

Según lo descrito en el Product Backlog en el inciso 2.1.1, se seleccionan las siguientes actividades a desarrollar:

ITEM	NOMBRE	ACTOR
1	Consulta de estudiante	Administrador
2	Consulta de estudiante con lector de código de barras	Administrador

Tabla 26. Planeación del Sprint 2

3.3.2. REFINAMIENTO DEL BACKLOG

Con el fin de actualizar el listado Backlog se definen las siguientes actividades que ayudarán a la finalización de la historia de usuario número 1. Consulta de estudiante y 2. Consulta de estudiante con lector de código de barras, definidas en la planeación del Sprint:

ITEM	NOMBRE	HISTORIA DE USUARIO	TIEMPO (horas)
1	Diseño del formulario de consulta de estudiantes	1	4 hora
2	Implementación del formulario de consulta de estudiantes	1	4 horas
3	Implementación del lector de código de barras	1	10 horas
4	Diseño de la vista con los datos del estudiante	1	4 hora
5	Implementación de la vista con los datos del estudiante	1	2 horas
6	Validación de campos del	1	2 horas

	formulario de consulta del estudiante		

Tabla 27. Refinamiento del Backlog Sprint 2

3.3.3. REVISIÓN DEL SPRINT

Se realiza los entregables asociados a las actividades propuestas en el refinamiento del backlog, entre los cuales se encuentran:

- Se crea el diseño del formulario de consulta de estudiantes, (Ilustración 9) correspondiente al punto de refinamiento 1 del Backlog (Tabla 27).
- Se implementa del formulario de consulta de estudiantes basado en el diseño realizado, este realiza la búsqueda comparando el código del estudiante guardado en base de datos y el ingresado por el usuario, (Ilustración 10) correspondiente al punto de refinamiento 2 del Backlog (Tabla 27).
- Se implementa el lector de código de barras al sistema, esta captura el código estudiantil del estudiante alojado en el carné estudiantil y lo compara con el guardado en la base de datos correspondiente al punto de refinamiento 3 del Backlog (Tabla 27).
- Una vez el estudiante sea encontrado en la información alojada en la base de datos se procede a mostrar toda su información y además la de las motos asociadas al mismo (Ilustración 11) correspondiente al punto de refinamiento 4 del Backlog (Tabla 27).
- Se implementa la vista con los datos del estudiante basado en el diseño realizado, en este se visualizan la información del estudiante (Nombre, apellido, cedula y código estudiantil) junto los datos de las motos asociadas que pueden ser n (Placa e Id de tarjeta de propiedad) la búsqueda comparando el código del estudiante guardado en base de datos y el ingresado por el usuario, (Ilustración 12) correspondiente al punto de refinamiento 5 del Backlog (Tabla 27).
- Se realiza la verificación de campos asociados al formulario de consulta del estudiante, es decir, si el código estudiantil perteneciente al estudiante no se encuentra en la base de datos, se mostrará el mensaje “El estudiante no está registrado” (Ilustración 13) correspondiente al punto de refinamiento 6 del Backlog (Tabla 27).

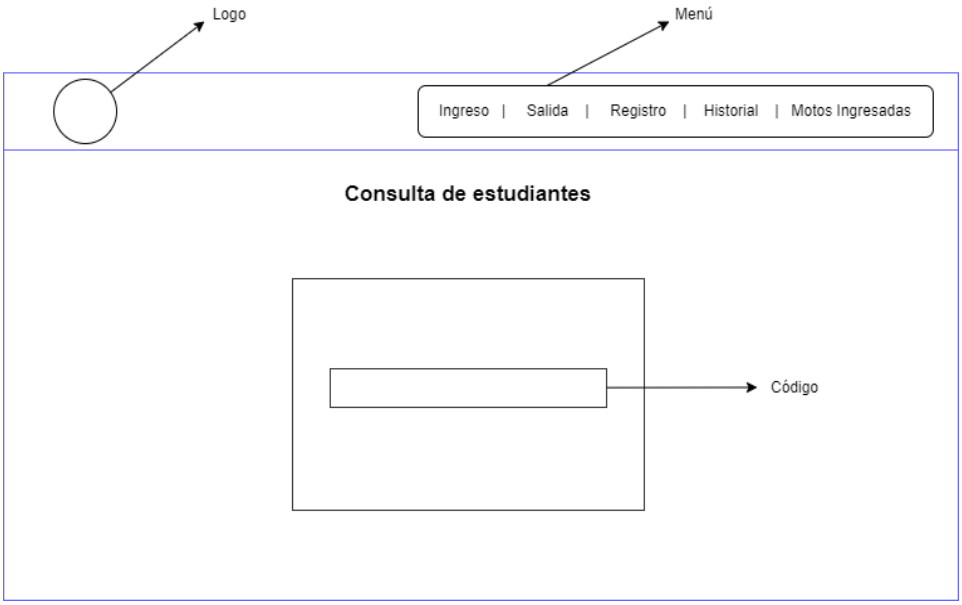


Ilustración 9. Diseño del formulario de consulta de estudiante

Ilustración 10. Implementación del formulario de consulta de estudiante

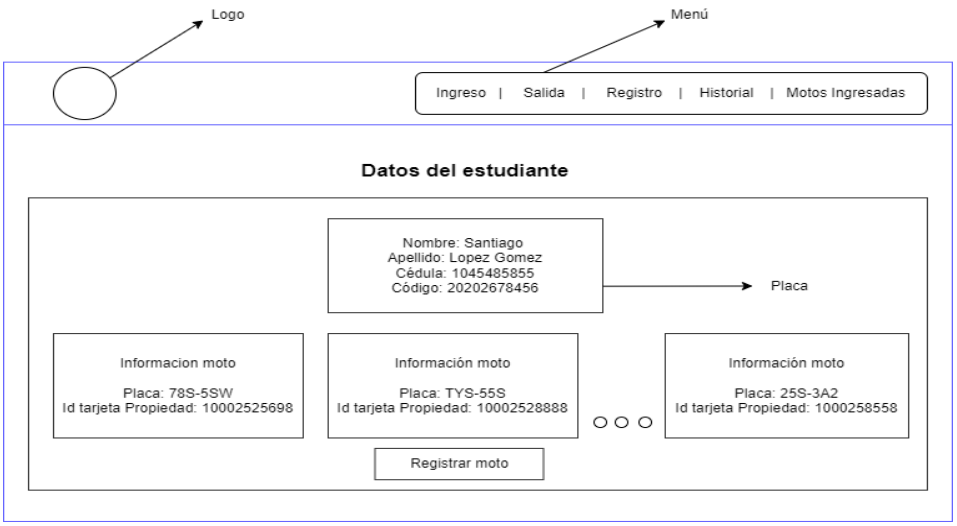


Ilustración 11. Diseño de la vista con los datos del estudiante

Ilustración 12. Implementación de la vista con los datos del estudiante

Ilustración 13. Validación de campos del formulario de consulta del estudiante

3.3.4. RETROSPECTIVA DEL SPRINT

Se concluye de manera exitosa el Sprint, todas las actividades asociadas al refinamiento del Backlog fueron entregadas y se encuentran funcionando de manera correcta.

3.4. SPRINT 3

3.4.1. PLANEACIÓN DEL SPRINT

Según lo descrito en el Product Backlog en el inciso 2.1.1, se seleccionan las siguientes actividades a desarrollar:

ITEM	NOMBRE	ACTOR
4	Edición de estudiante	Administrador

5	Eliminación de estudiante	Administrador

Tabla 28. Planeación de Sprint 3

3.4.2. REFINAMIENTO DEL BACKLOG

Con el fin de actualizar el listado Backlog se definen las siguientes actividades que ayudarán a la finalización de la historia de usuario número 4. Edición de estudiante y 5. Eliminación de estudiante, definidas en la planeación del Sprint:

ITEM	NOMBRE	HISTORIA DE USUARIO	TIEMPO (horas)
1	Diseño del formulario de edición de estudiantes	1	4 hora
2	Implementación del formulario de edición de estudiantes	1	4 horas
3	Creación de alertas para el formulario de edición de estudiantes	1	2 horas
4	Validación de campos del formulario de edición	1	2 horas

Tabla 29. Refinamiento del Backlog Sprint 3

3.4.3. REVISIÓN DEL SPRINT

Se realiza los entregables asociados a las actividades propuestas en el refinamiento del backlog, entre los cuales se encuentran:

- Se crea el diseño del formulario de edición de estudiantes, (Ilustración 14) correspondiente al punto de refinamiento 1 del Backlog (Tabla 29).
- Se implementa del formulario de edición de estudiantes basado en el diseño realizado, este realiza la edición del usuario primero buscando el código del estudiante guardado en base de datos y el ingresado por el usuario, segundo permite realizar la actualización de los datos asociados al estudiante, (Ilustración 15) correspondiente al punto de refinamiento 2 del Backlog (Tabla 29).
- Se crean las alertas correspondientes al éxito de edición del usuario (Ilustración 16) correspondiente al punto de refinamiento 3 del Backlog (Tabla 29).

- Se realiza la verificación de campos asociados al formulario de edición de estudiantes, es decir, en los campos de nombres y apellidos solo se deberán aceptar caracteres, en los campos de cédula y código estudiantil solo se deben aceptar números correspondiente al punto de refinamiento 4 del Backlog (Tabla 29).
- El estudiante no será eliminado, con el fin de mantener el histórico intacto.

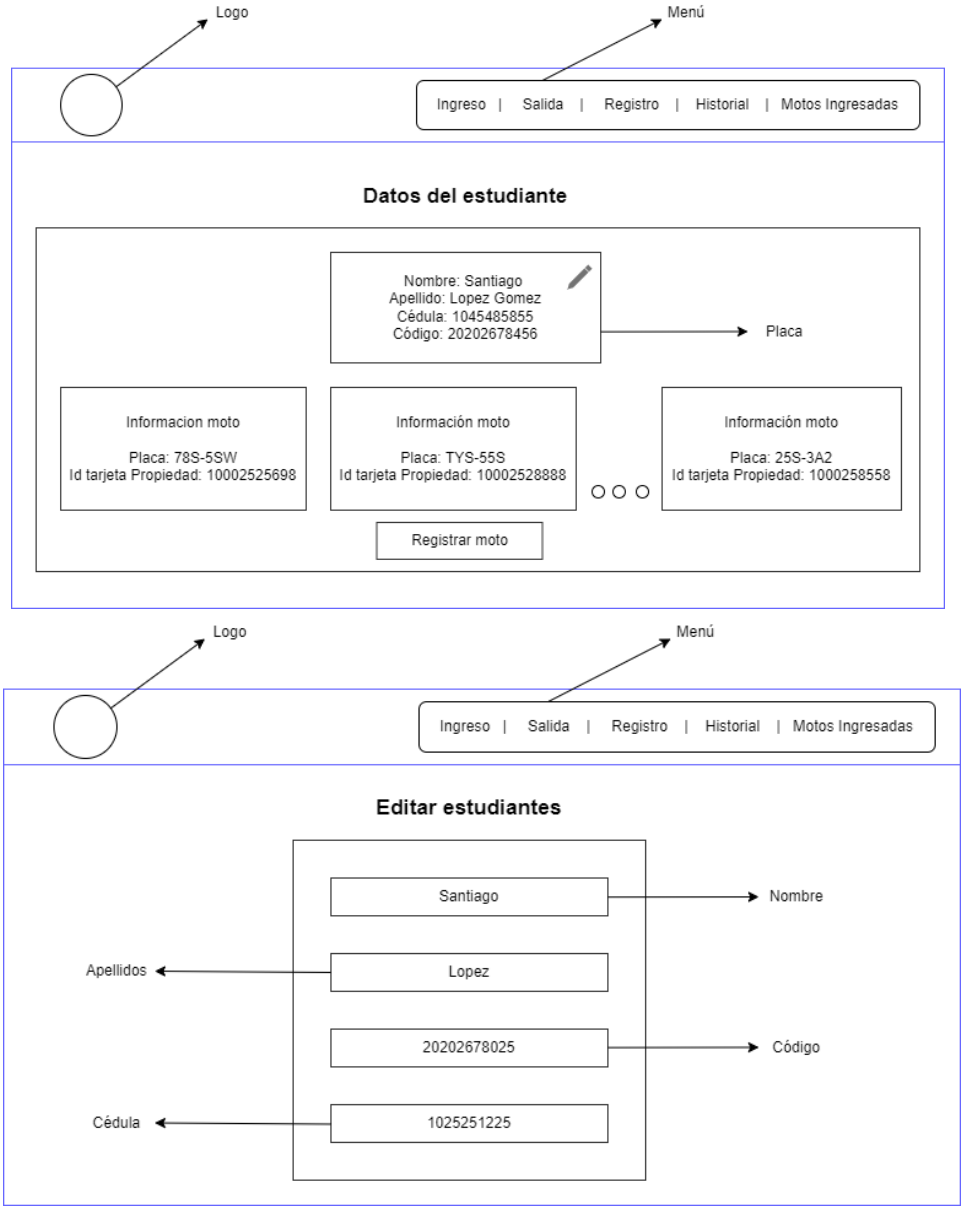


Ilustración 14.Diseño del formulario de edición de estudiantes

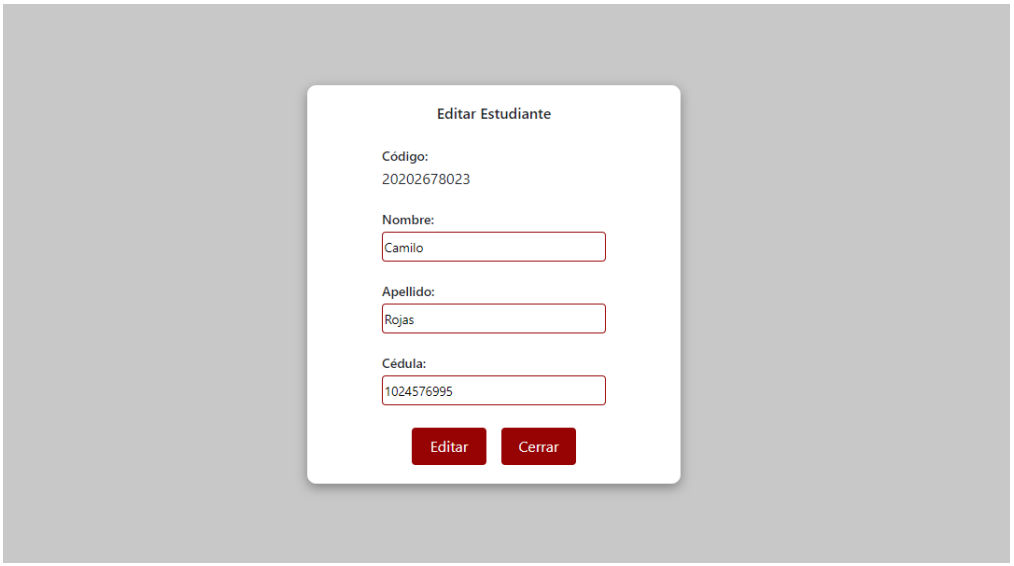


Ilustración 15. Implementación del formulario de edición de estudiantes

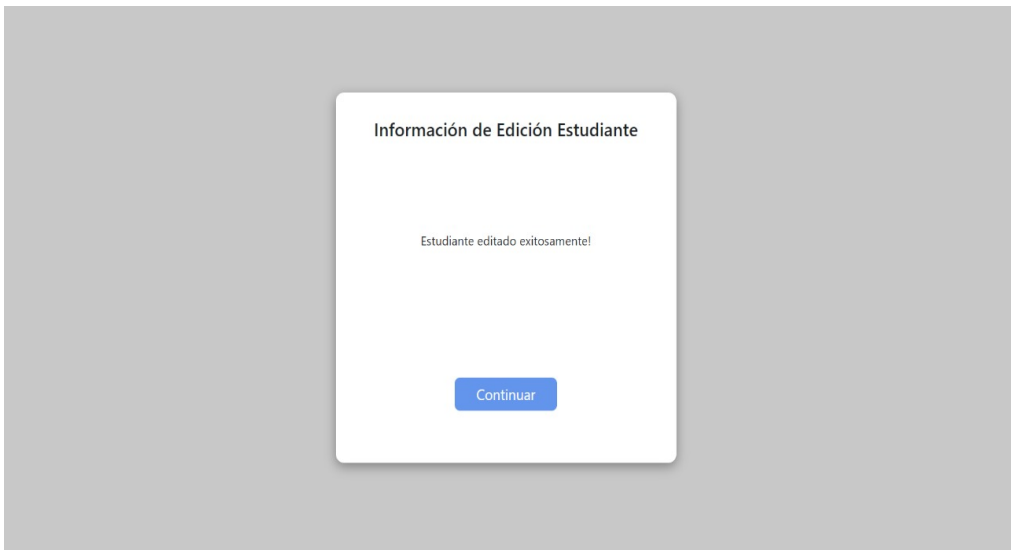


Ilustración 16. Creación de alertas para el formulario de edición de estudiantes

3.4.4. RETROSPECTIVA DEL SPRINT

Se concluye de manera exitosa el Sprint, todas las actividades asociadas al refinamiento del Backlog fueron entregadas y se encuentran funcionando de manera correcta.

3.5. SPRINT 4

3.5.1. PLANEACIÓN DEL SPRINT

Según lo descrito en el Product Backlog en el inciso 2.1.1, se seleccionan las siguientes actividades a desarrollar:

ITEM	NOMBRE	ACTOR
7	Registro de motocicletas	Administrador

Tabla 30. Planeación de Sprint 4

3.5.2. REFINAMIENTO DEL BACKLOG

Con el fin de actualizar el listado Backlog se definen las siguientes actividades que ayudarán a la finalización de la historia de usuario número 7. Registro de motocicletas definida en la planeación del Sprint:

ITEM	NOMBRE	HISTORIA DE USUARIO	TIEMPO (horas)
1	Diseño del formulario de registro de motocicletas	1	6 horas
2	Implementación de la vista para el registro de motocicletas	1	4 horas
3	Creación de alertas en el formulario de registro de motocicletas	1	2 horas
4	Validación de campos del formulario de registro de motocicletas	1	2 horas

Tabla 31. Refinamiento del Backlog Sprint 4

3.5.3. REVISIÓN DEL SPRINT

Se realiza los entregables asociados a las actividades propuestas en el refinamiento del backlog, entre los cuales se encuentran:

- Se crea el diseño del formulario de registro de motocicletas (Ilustración 17) correspondiente al punto de refinamiento 1 del Backlog (Tabla 31).
- Se implementa del formulario de registro de motocicletas basado en el diseño realizado, (Ilustración 18) correspondiente al punto de refinamiento 2 del Backlog (Tabla 31).
- Se crean las alertas correspondientes al éxito de creación de la motocicleta (Ilustración 19) correspondiente al punto de refinamiento 3 del Backlog (Tabla 31).
- Se realiza la verificación de campos asociados al formulario de registro de motocicletas, es decir, en los campos de paca se aceptan caracteres y números, en los campos de identificación de tarjeta de propiedad solo se deben aceptar números (Ilustración 20) correspondiente al punto de refinamiento 4 del Backlog (Tabla 31).

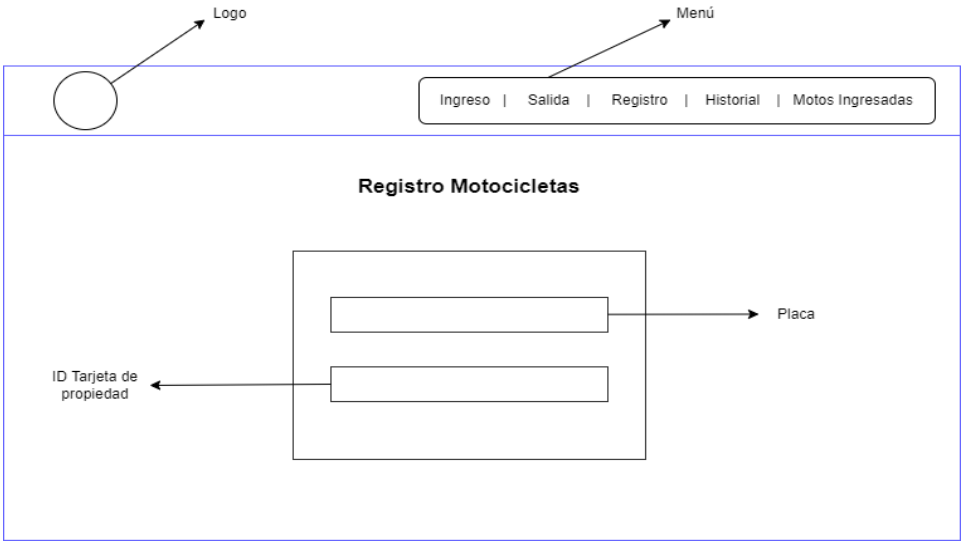


Ilustración 17. Diseño del formulario de registro de motocicletas

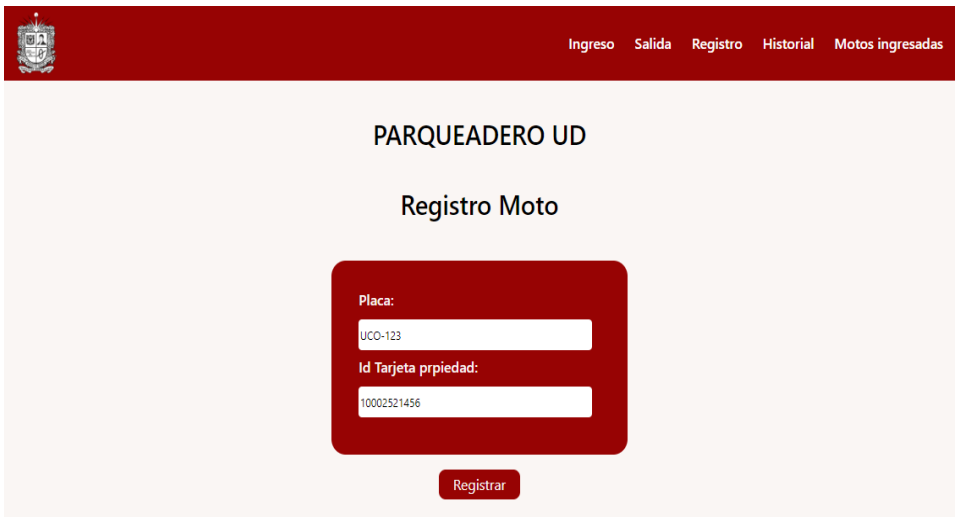


Ilustración 18. Implementación de la vista para el registro de motocicletas

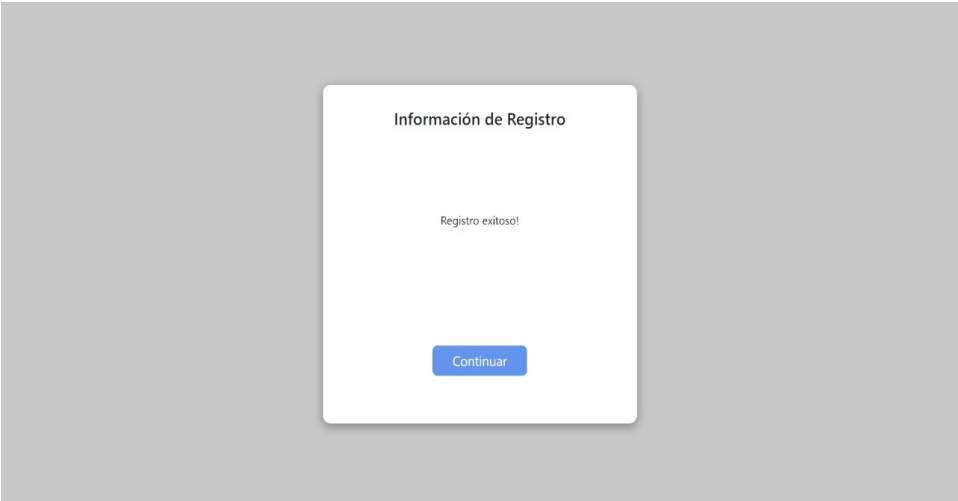


Ilustración 19. Alertas en el formulario de registro de motocicletas

```
return (
    <div className={styles.container}>
        <h2>Registro Moto</h2>
        <div className={styles.prueba}></div>
        <form onSubmit={sendDataMoto} className={`d-flex flex-column align-items-center ${styles.form}`}>
            <div className={styles.formContainer}>
                <h3>Placa:</h3>
                <input required value={placa} onChange={handlePlaca} className={styles.formContainer_input}
                type="text" placeholder='Digite placa moto...' />
                <h3>Id Tarjeta prpiedad:</h3>
                <input required value={idTarjeta} onChange={handleIdTarjeta} className={styles.formContainer_input}
                type="number" placeholder='Digite Id tarjeta de propiedad...' />
            </div>
            <button type='submit' className={styles.formContainer_button}>Registrar</button>

        </form>
```

Ilustración 20. Validación de campos del formulario de registro de motocicletas

3.5.4. RETROSPECTIVA DEL SPRINT

Se concluye de manera exitosa el Sprint, todas las actividades asociadas al refinamiento del Backlog fueron entregadas y se encuentran funcionando de manera correcta.

3.6. SPRINT 5

3.6.1. PLANEACIÓN DEL SPRINT

Según lo descrito en el Product Backlog en el inciso 2.1.1, se seleccionan las siguientes actividades a desarrollar:

ITEM	NOMBRE	ACTOR
8	Edición de motocicleta	Administrador
9	Eliminación de motocicleta	Administrador

Tabla 32. Planeación del Sprint 5

3.6.2. REFINAMIENTO DEL BACKLOG

Con el fin de actualizar el listado Backlog se definen las siguientes actividades que ayudarán a la finalización de la historia de usuario número 8. Edición de motocicleta y 9. Eliminación de motocicleta, definidas en la planeación del Sprint:

ITEM	NOMBRE	HISTORIA DE USUARIO	TIEMPO (horas)
1	Diseño del formulario de edición de motocicletas	1	4 hora
2	Diseño de la funcionalidad de eliminación de motocicletas	1	2 hora
3	Implementación del formulario de edición de motocicletas	1	4 horas
4	Implementación de la funcionalidad de eliminación de motocicletas	1	2 horas
5	Creación de alertas para el formulario de edición de motocicletas	1	1 horas
6	Creación de alertas para la funcionalidad eliminación de motocicletas	1	2 horas
7	Validación de campos del formulario de edición de motocicletas	1	2 horas

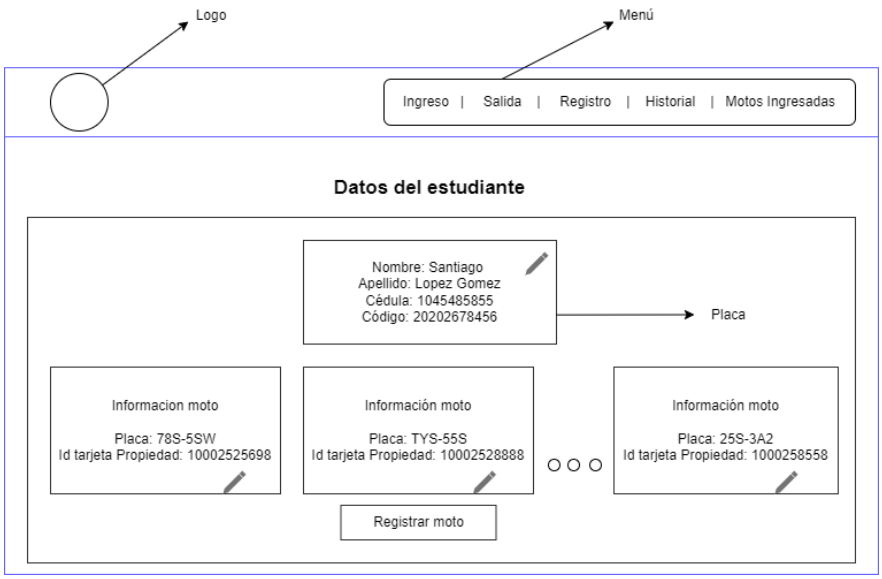
Tabla 33. Refinamiento del Backlog Sprint 5

3.6.3. REVISIÓN DEL SPRINT

Se realiza los entregables asociados a las actividades propuestas en el refinamiento del backlog, entre los cuales se encuentran:

- Se crea el diseño del formulario de edición de motocicletas, (Ilustración 21) correspondiente al punto de refinamiento 1 del Backlog (Tabla 33).
- Se crea el diseño de la funcionalidad de eliminación de motocicletas, (Ilustración 22) correspondiente al punto de refinamiento 2 del Backlog (Tabla 33).

- Se implementa del formulario de edición de motocicletas basado en el diseño realizado, este realiza la edición del usuario primero buscando la información de la motocicleta, segundo permite realizar la actualización de los datos asociados a la motocicleta, (Ilustración 23) correspondiente al punto de refinamiento 3 del Backlog (Tabla 33).
- Se implementa la funcionalidad de eliminar una motocicleta, este buscará la motocicleta según el código del estudiante ingresado por el usuario, seguido de esto permitirá eliminar si encuentra alguna coincidencia (Ilustración 24) correspondiente al punto de refinamiento 4 del Backlog (Tabla 33).
- Se crean las alertas correspondientes al éxito de edición de la motocicleta (Ilustración 25) correspondiente al punto de refinamiento 4 del Backlog (Tabla 33).
- Se crean las alertas correspondientes al éxito de eliminación de la motocicleta (Ilustración 26) correspondiente al punto de refinamiento 4 del Backlog (Tabla 33).
- Se realiza la verificación de campos asociados al formulario de edición de motocicletas, es decir, en los campos de placa y id tarjeta de propiedad solo se deberán aceptar caracteres, correspondientes al punto de refinamiento 5 del Backlog (Tabla 33).



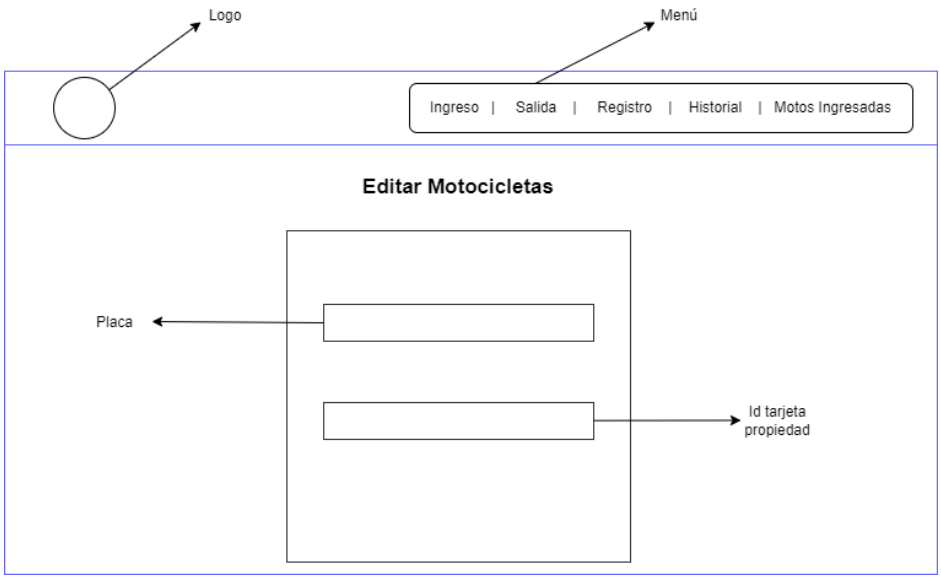


Ilustración 21. Diseño del formulario de edición de motocicletas

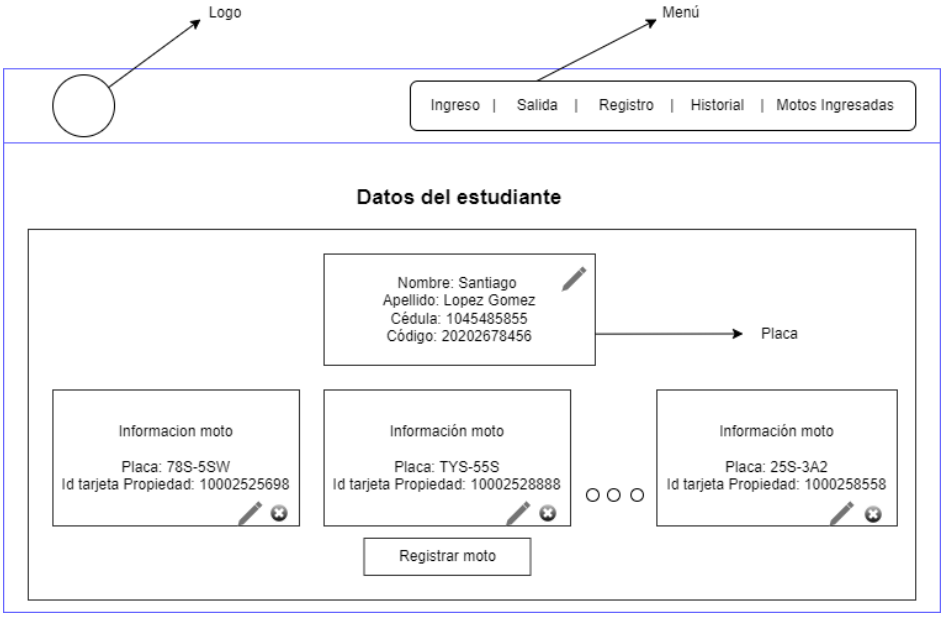


Ilustración 22. Diseño de la funcionalidad de eliminación de motocicletas

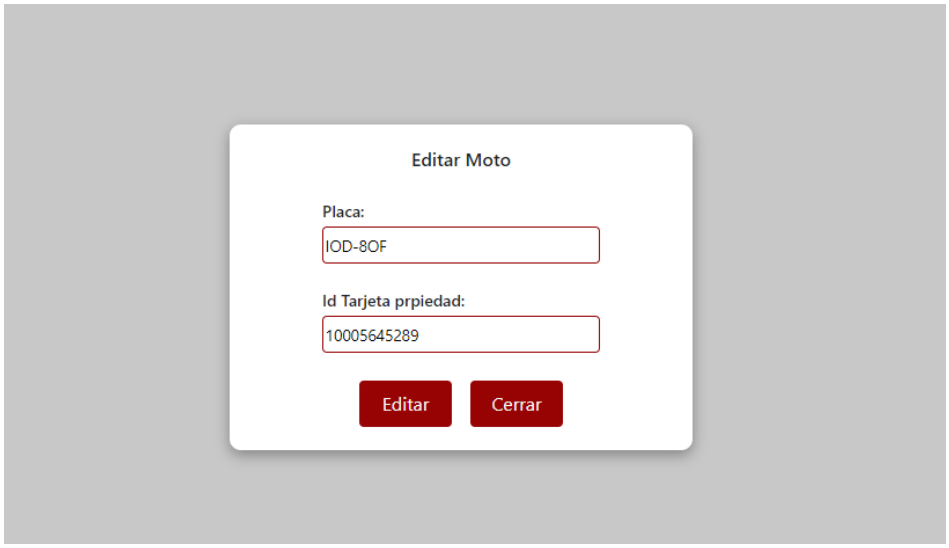


Ilustración 23. Implementación del formulario de edición de motocicletas

Ilustración 24. Implementación de la funcionalidad de eliminación de motocicletas

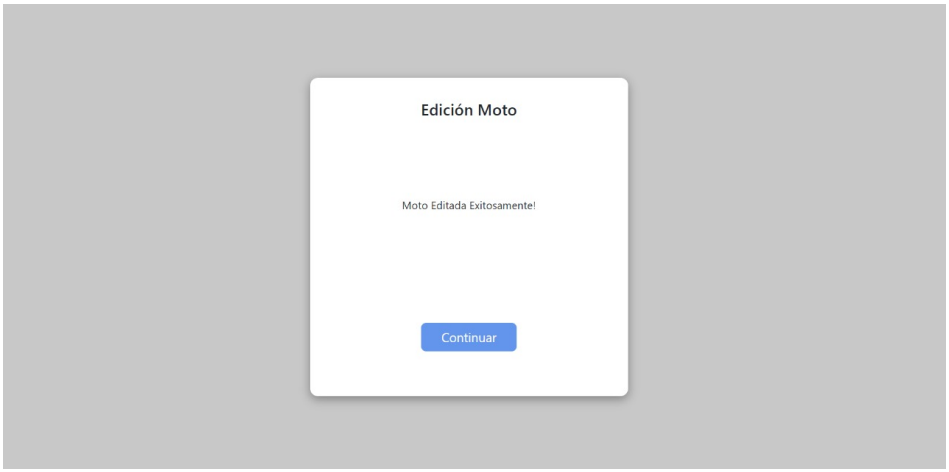


Ilustración 25. Creación de alertas para el formulario de edición de motocicletas

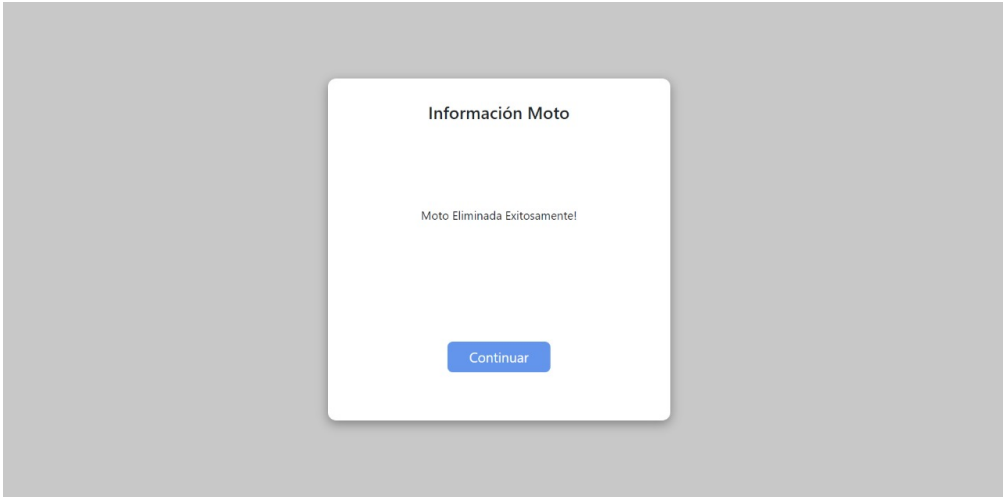


Ilustración 26. Creación de alertas para la funcionalidad eliminación de motocicletas

3.6.4. RETROSPECTIVA DEL SPRINT

Se concluye de manera exitosa el Sprint, todas las actividades asociadas al refinamiento del Backlog fueron entregadas y se encuentran funcionando de manera correcta.

3.7. SPRINT 6

3.7.1. PLANEACIÓN DEL SPRINT

Según lo descrito en el Product Backlog en el inciso 2.1.1, se seleccionan las siguientes actividades a desarrollar:

ITEM	NOMBRE	ACTOR
10	Consulta de listado actual y listado histórico	Administrador

Tabla 34. Planeación Sprint 6

3.7.2. REFINAMIENTO DEL BACKLOG

Con el fin de actualizar el listado Backlog se definen las siguientes actividades que ayudarán a la finalización de la historia de usuario número 10. Consulta de listado actual y listado histórico, definidas en la planeación del Sprint:

ITEM	NOMBRE	HISTORIA DE USUARIO	TIEMPO (horas)
1	Diseño del formulario de consulta del listado actual	1	6 hora

2	Diseño del formulario de consulta del listado histórico	1	6 hora
3	Implementación del formulario de consulta del listado actual	1	4 horas
4	Implementación del formulario de consulta del listado histórico	1	4 horas
5	Configuración del buscador de los formularios de búsqueda del listado actual y listado histórico	1	2 horas

Tabla 35. Refinamiento Backlog Sprint 6

3.7.3. REVISIÓN DEL SPRINT

Se realiza los entregables asociados a las actividades propuestas en el refinamiento del backlog, entre los cuales se encuentran:

- Se crea el diseño del formulario de consulta del listado actual, (Ilustración 27) correspondiente al punto de refinamiento 1 del Backlog (Tabla 35).
- Se crea el diseño del formulario de consulta del listado histórico, (Ilustración 28), correspondiente al punto de refinamiento 2 del Backlog (Tabla 35).
- Se implementa del formulario de consulta del listado actual basado en el diseño realizado, este realiza la búsqueda del día actual, (Ilustración 29), correspondiente al punto de refinamiento 3 del Backlog (Tabla 35).
- Se implementa del formulario de consulta del listado actual y listado histórico basado en el diseño realizado, este realiza la búsqueda del histórico de ingresos y salidas de las motocicletas, (Ilustración 30). correspondiente al punto de refinamiento 4 del Backlog (Tabla 35).
- Se implementa la configuración del buscador de los formularios de búsqueda del listado actual y listado histórico (Ilustración 31), correspondiente al punto de refinamiento 5 del Backlog (Tabla 35)

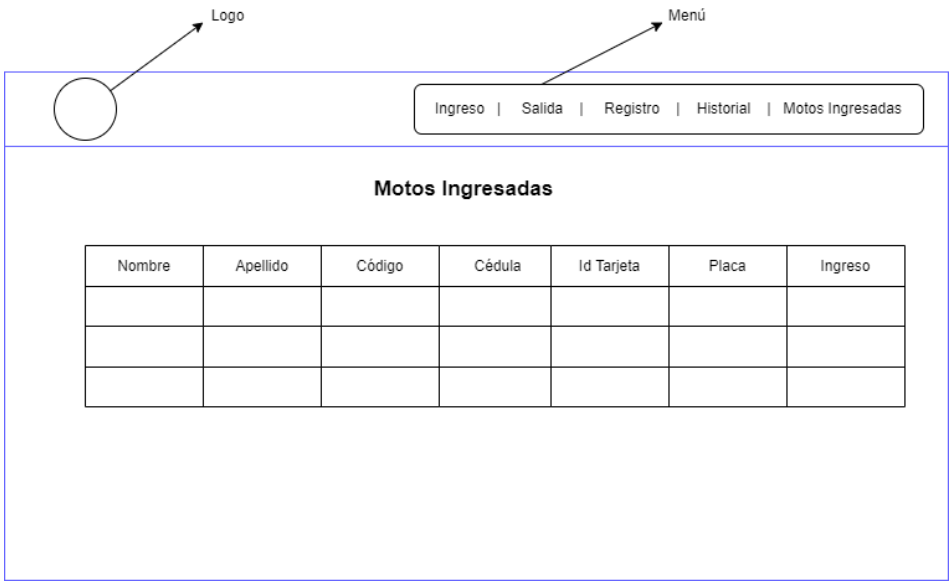


Ilustración 27. Diseño del formulario de consulta del listado actual

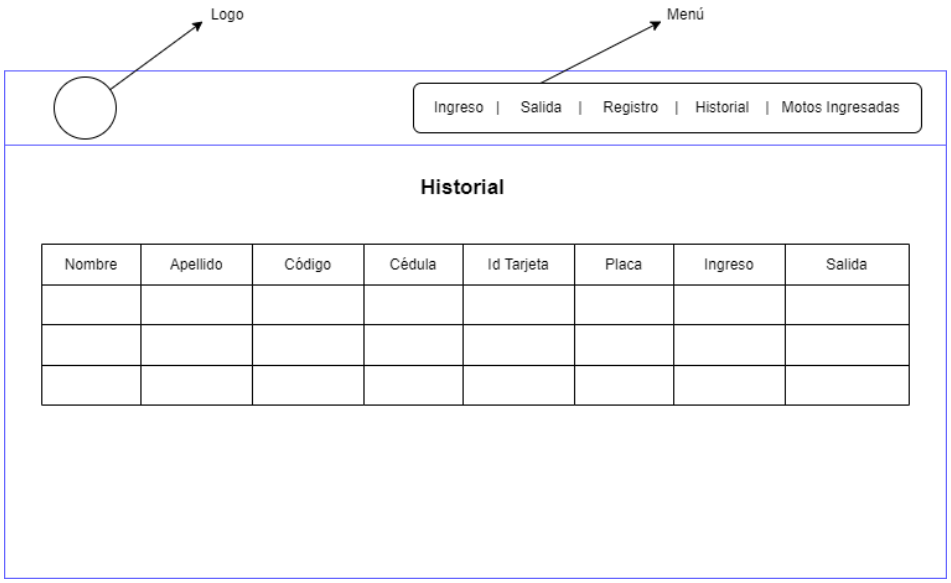


Ilustración 28. Diseño del formulario de consulta del listado histórico

Ilustración 29. Implementación del formulario de consulta del listado actual

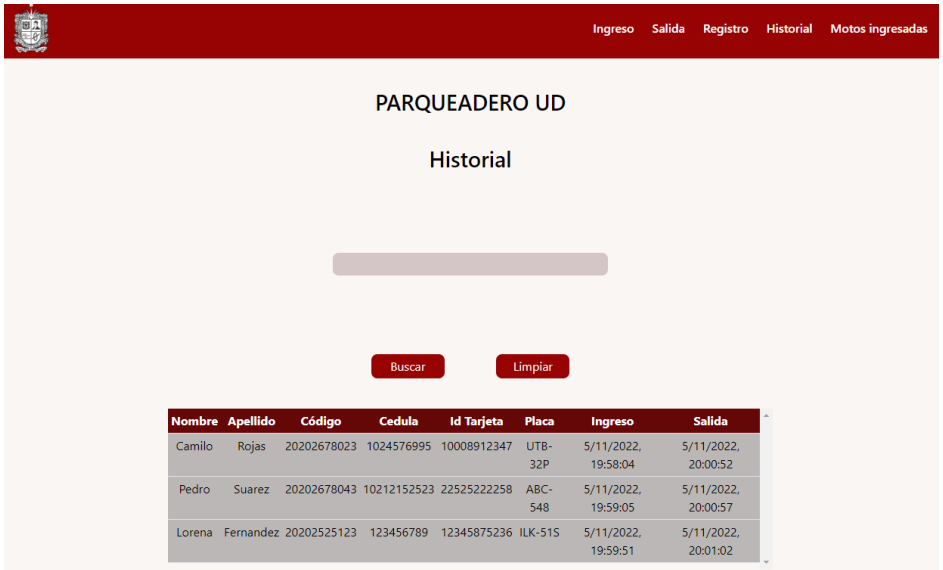


Ilustración 30. Implementación del formulario de consulta del listado histórico

Ilustración 31. Configuración del buscador de los formularios de búsqueda del listado actual y listado histórico

3.7.4. RETROSPECTIVA DEL SPRINT

Se concluye de manera exitosa el Sprint, todas las actividades asociadas al refinamiento del Backlog fueron entregadas y se encuentran funcionando de manera correcta.

3.9. DIAGRAMA DE DESPLIEGUE

El siguiente diagrama describe la arquitectura de ejecución del sistema realizado, en donde se puede evidenciar el Stack Mern, el cual utiliza las tecnologías asociadas al lenguaje de programación JavaScript, como lo son React JS el cual junto HTML/CSS, Bootstrap y JavaScript componen el Frontend de la aplicación (parte que interactúa con los usuarios), por otra parte se tiene Express JS y Node JS que componen el Backend de la aplicación (parte que interactúa con el servidor), finalmente se cuenta con MongoDB que es la base de datos no relacional del sistema.

La unión entre Stack Mern y el lector de código de barras, se realiza por medio de React JS, de esta manera se logra realizar la recolección de datos en la aplicación.

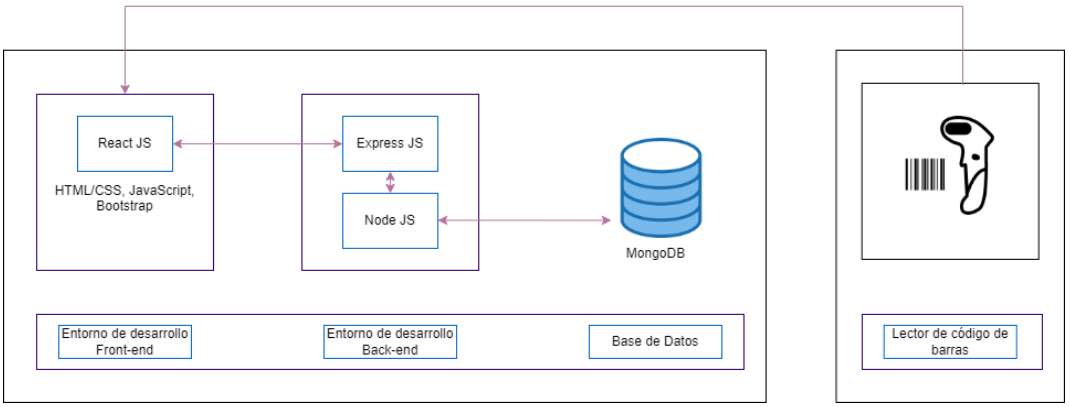


Ilustración 32. Diagrama de despliegue

3.10. DIAGRAMA DE PROCESOS

3.10.1. DIAGRAMA DE INGRESO DE MOTOCICLETAS

En el diagrama de procesos se describen las acciones que se realizan en el ingreso de una motocicleta asociada a un estudiante a la Facultad Tecnológica de la Universidad.

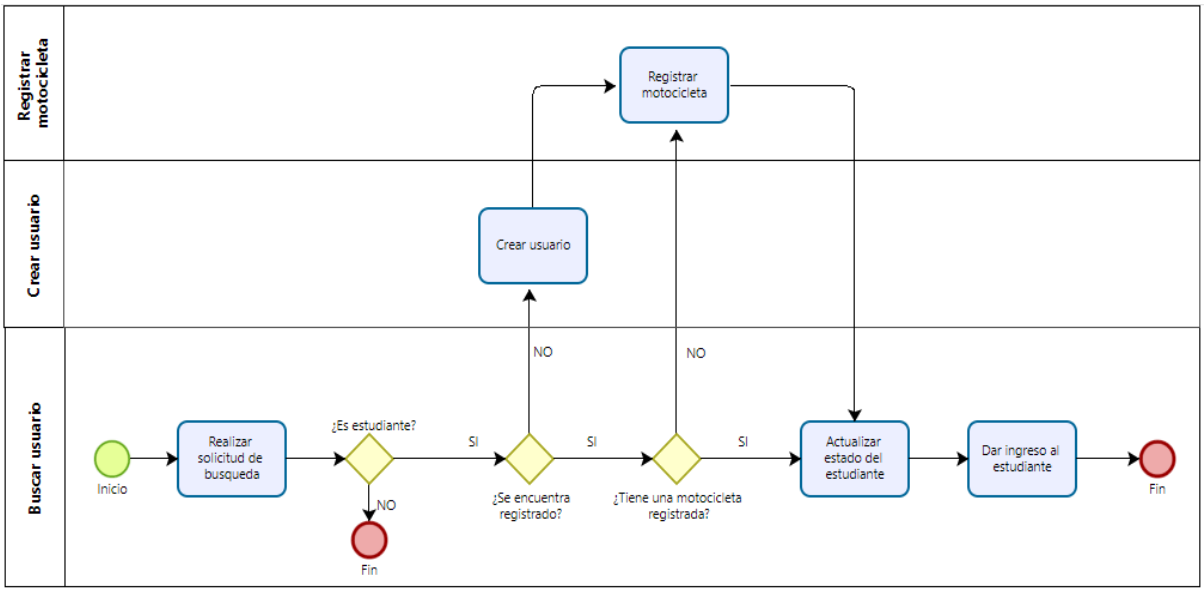


Ilustración 33. Diagrama de entrada de motocicletas

3.10.1. DIAGRAMA DE SALIDA DE MOTICICLETAS

En el diagrama de procesos se describen las acciones que se realizan en la salida de una motocicleta asociada a un estudiante de la Facultad Tecnológica de la Universidad.

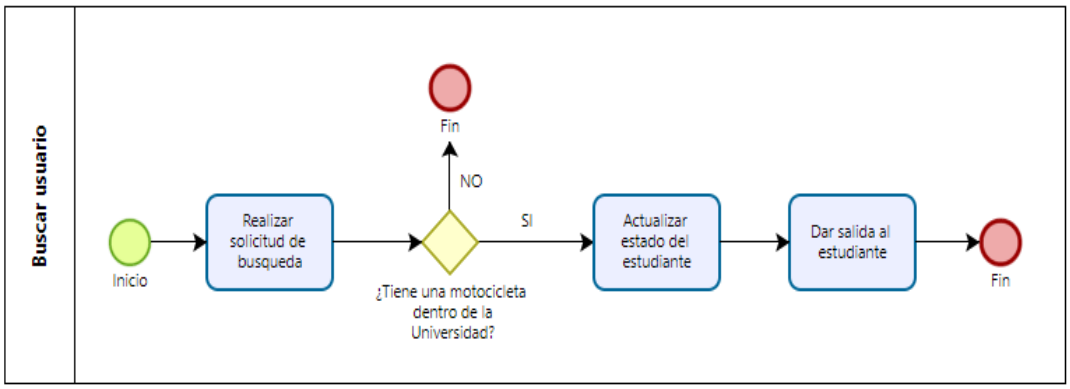


Ilustración 34. Diagrama de salida de motocicletas

CAPÍTULO 4

4. IMPLEMENTACIÓN

4.1. MODELO DE BASE DE DATOS NO RELACIONAL

Actualmente las bases de datos no relacionales carecen de un modelo definido, estas suelen estar definidas por colecciones de datos no estructurados, por lo que resulta difícil la generación de una diagramación estática. Sin embargo, se plantea un diagrama entidad relación para entender el funcionamiento y conexión de los datos dentro del aplicativo.

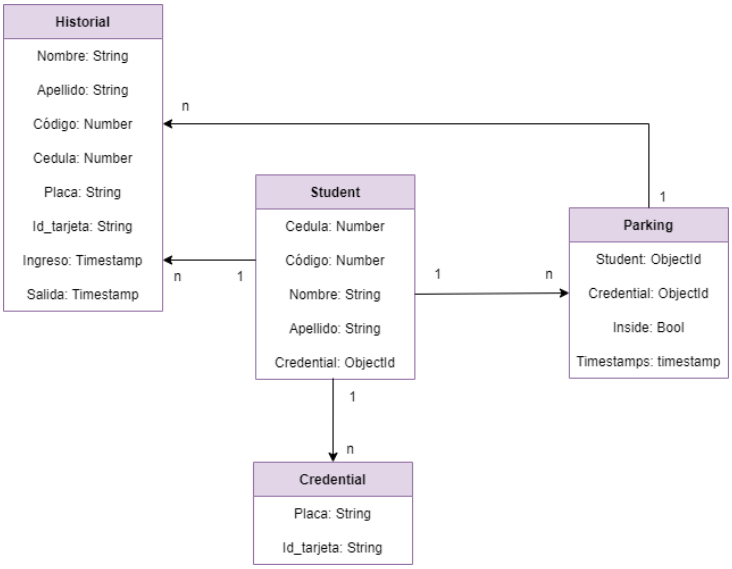


Ilustración 35. Modelo de base de datos

4.2. DICCIONARIO DE DATOS

Actualmente se utilizan tres documentos de tipo JSON dentro de la base de datos no relacional, manejada en MongoDB, entre las cuales se encuentran: Student, Parking y Credential. Se debe tener en cuenta que los tipos de datos en MongoDB son completamente compatibles en los demás lenguajes utilizados en el Stack Mern, al estar todos basados en el lenguaje de programación JavaScript.

4.2.1 DOCUMENTO MONGODB STUDENT

Este documento recolecta todos los datos asociados al estudiante actualmente activo en la Universidad y con una motocicleta a ingresar o retirar de la misma. Los datos que componen esta colección de datos son los siguientes:

ITEM	NOMBRE	DESCRIPCIÓN
1	Cedula	Este dato contiene el número de identidad del estudiante. Dentro del aplicativo se le asignó el tipo de dato “Number”, es decir que solo podrá recibir números.
2	Código	Es el número identificador del estudiante dentro de la Universidad Distrital FJC. Dentro del aplicativo se le asignó el tipo de dato “Number”, es decir que solo podrá recibir números.
3	Nombre	Hace referencia a los nombres del estudiante. Dentro del aplicativo se le asignó el tipo de dato “String”, es decir que acepta cualquier tipo de carácter.
4	Apellido	Hace referencia a los apellidos del estudiante. Dentro del aplicativo se le asignó el tipo de dato “String”, es decir que acepta cualquier tipo de carácter.
5	Credential	Este campo contiene la información de las motocicletas asociadas al estudiante. Dentro del aplicativo se le asignó el tipo de dato “ObjectId”, es decir que puede hacer referencia a otra clase y relacionar su información.

Tabla 36. Documento MongoDB Student

4.2.2 DOCUMENTO MONGODB CREDENTIAL

Este documento recolecta todos los datos asociados a las motocicletas asociadas a un estudiante. Los datos que componen esta colección de datos son los siguientes:

ITEM	NOMBRE	DESCRIPCIÓN
1	Placa	Este dato contiene la información de la placa de la motocicleta a entrar o retirar de la Universidad.

		Dentro del aplicativo se le asignó el tipo de dato “String”, es decir que acepta cualquier tipo de carácter.
2	Id_tarjeta	Este dato contiene el número de identificación de la tarjeta de propiedad de la motocicleta. Dentro del aplicativo se le asignó el tipo de dato “String”, es decir que acepta cualquier tipo de carácter.

Tabla 37. Documento MongoDB Credential

4.2.3 DOCUMENTO MONGODB PARKING

Este documento recolecta todos los datos asociados al registro o retiro de una moto asociada a un estudiante dentro de la Universidad Distrital. Los datos que componen esta colección de datos son los siguientes:

ITEM	NOMBRE	DESCRIPCIÓN
1	Student	Este dato contiene la información del estudiante. Dentro del aplicativo se le asignó el tipo de dato “ObjectId”, es decir que puede hacer referencia a otra clase y relacionar su información.
2	Credential	Este dato contiene la información de las motocicletas asociadas al estudiante. Dentro del aplicativo se le asignó el tipo de dato “ObjectId”, es decir que puede hacer referencia a otra clase y relacionar su información.
3	Inside	Hace referencia al estado actual del estudiante, es decir informa si la moto de un estudiante se encuentra dentro de la Universidad o no. Dentro del aplicativo se le asignó el tipo de dato “Boolean”, es decir, solo maneja el estado true o false.
4	Timestamps	Hace referencia a la fecha en tiempo real en la que el estudiante ingresa una

		motocicleta a la Universidad. Dentro del aplicativo se le asignó el tipo de dato “timestamps”, es decir acepta fechas.

Tabla 38. Documento MongoDB Parking

4.2.4 DOCUMENTO MONGODB HISTORIAL

Este documento recolecta todos los datos asociados al registro o retiro de una moto asociada a un estudiante dentro de la Universidad Distrital para así crear el histórico de datos. Los datos que componen esta colección de datos son los siguientes:

ITEM	NOMBRE	DESCRIPCIÓN
1	Nombre	Este dato contiene el nombre del estudiante. Dentro del aplicativo se le asignó el tipo de dato “String”, es decir que acepta cualquier tipo de carácter.
2	Apellido	Este dato contiene el apellido del estudiante. Dentro del aplicativo se le asignó el tipo de dato “String”, es decir que acepta cualquier tipo de carácter.
3	Código	Hace referencia al estado actual del estudiante, es decir informa si la moto de un estudiante se encuentra dentro de la Universidad o no. Dentro del aplicativo se le asignó el tipo de dato “Boolean”, es decir, solo maneja el estado true o false.
4	Cedula	Este dato contiene el número de identidad del estudiante. Dentro del aplicativo se le asignó el tipo de dato “Number”, es decir que solo podrá recibir números.
5	Id_tarjeta	Este dato contiene el número de identificación de la tarjeta de propiedad de la motocicleta.

		Dentro del aplicativo se le asignó el tipo de dato “String”, es decir que acepta cualquier tipo de carácter.
6	Placa	Este dato contiene la información de la placa de la motocicleta a entrar o retirar de la Universidad. Dentro del aplicativo se le asignó el tipo de dato “String”, es decir que acepta cualquier tipo de carácter.
7	Ingreso	Hace referencia a la fecha en que el estudiante ingresa a la Universidad. Dentro del aplicativo se le asignó el tipo de dato “timestamps”, es decir acepta fechas.
8	Salida	Hace referencia a la fecha en que el estudiante sale de la Universidad. Dentro del aplicativo se le asignó el tipo de dato “timestamps”, es decir acepta fechas.

Tabla 39. Documento MongoDB Historial

4.3. ESTRUCTURA DE ARCHIVOS

El presente proyecto es pasado en API RESTful, en donde se realiza una petición y un servicio da respuesta a él, es decir las API’s generadas en el Backend se consumen en el Frontend. El proyecto en si se organiza de varias formas, las cuales se explicarán a continuación:

4.3.1 DESARROLLO BACKEND Y FRONTEND

El proyecto se encuentra separado en dos grandes componentes, el Backend y Frontend, donde el primero mencionado hace referencia a la lógica del aplicativo, lo cual engloba el servidor, las bases de datos y todo el desarrollo realizado para el tratamiento de información, en el Stack Mern esto lo representa Express JS, Node JS y MongoDB.

Por otra parte, el Frontend es aquella parte del aplicativo que tiene contacto con el usuario, es decir engloba todo el diseño del aplicativo, en el Stack Mern lo representa React JS, junto HTML, CSS y JavaScript.

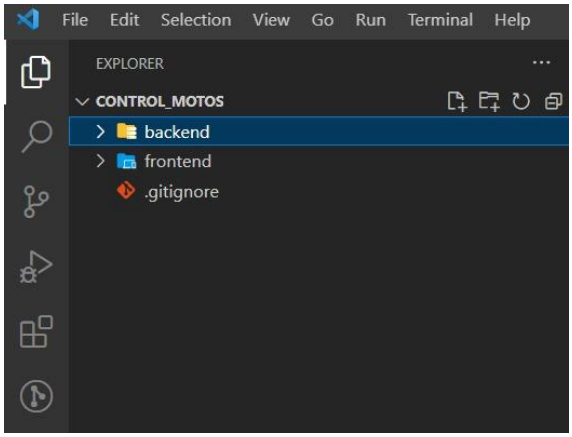


Ilustración 36. Distribución Backend y Frontend

4.3.2 ORGANIZACIÓN DEL FRONTEND

La organización del Frontend es realizada según la metodología de desarrollo Atomic Desing, la cual sigue la arquitectura basada en componentes totalmente compatible con React JS.

Gracias a este principio cada servicio en el proyecto utiliza un entorno aislado, es decir, cada estilo, acción a realizar o prueba son independientes entre sí. Lo anterior mencionado ayuda a que el código sea mucho más legible y modular, facilitando la ubicación de funcionalidades, actualizaciones y detección de errores.

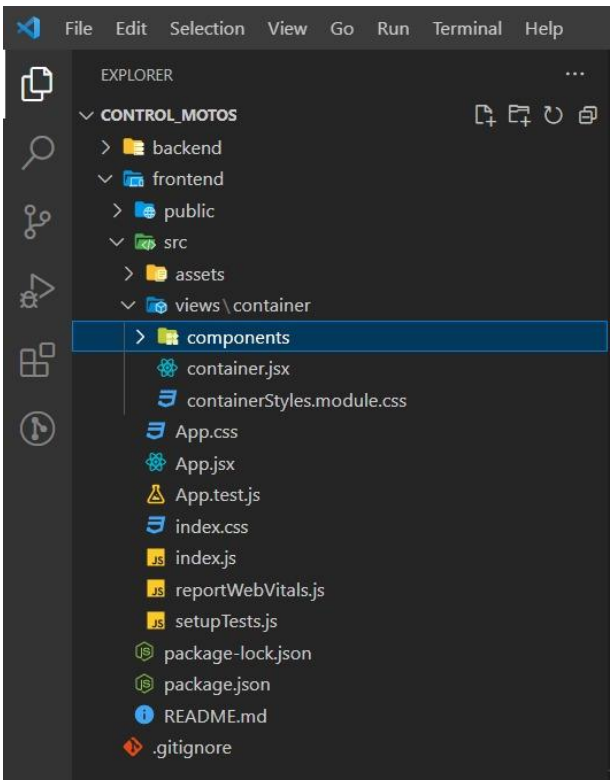


Ilustración 37. Distribución Frontend

A continuación, se explican las funcionalidades de los principales archivos y carpetas alojadas en el Frontend del aplicativo:

- Carpeta src: En esta carpeta se encuentra el código React, utilizado en el proyecto, es decir es donde se realiza todo el desarrollo. Esta se compone de:
  - **Carpeta Assets:** Esta contiene todos los recursos a utilizar en el proyecto, puede alojar, imágenes, fuentes, sonidos y cualquier archivo multimedia.

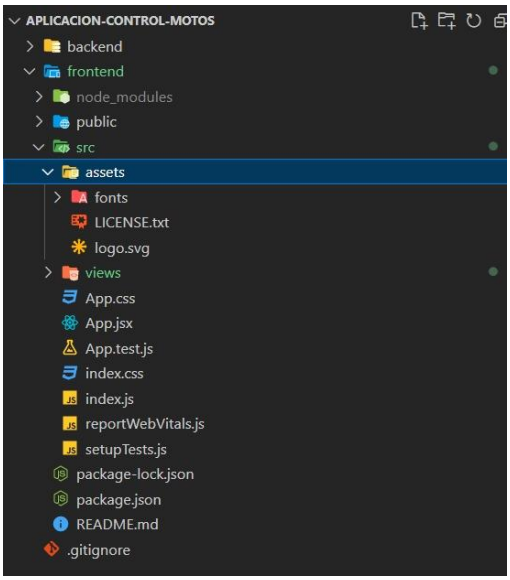


Ilustración 38. Carpeta assets - Frontend

  - **Carpeta Views:** Esta carpeta contiene la construcción del proyecto a realizar, ya que es donde se realiza la maquetación de las vistas que vera el usuario.

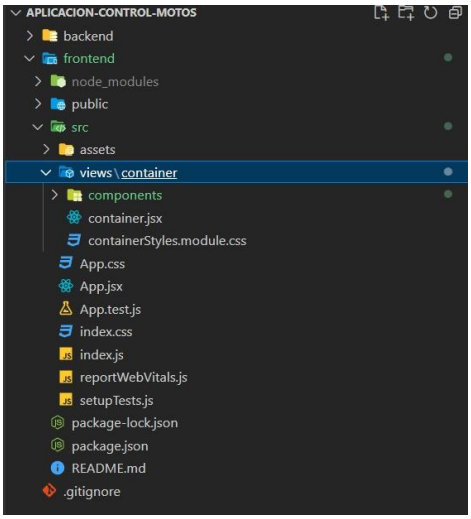


Ilustración 39. Carpeta views - Frontend

En esta carpeta cada acción realizada esta agrupada en un componente, basados en los principios de Atomic Desing mencionados anteriormente.

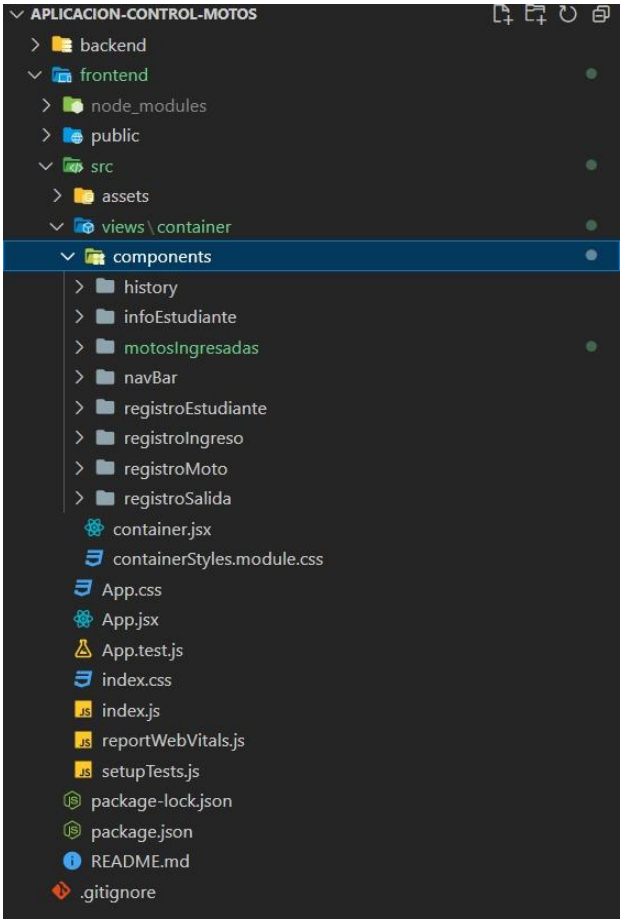


Ilustración 40. Componentes en el Frontend

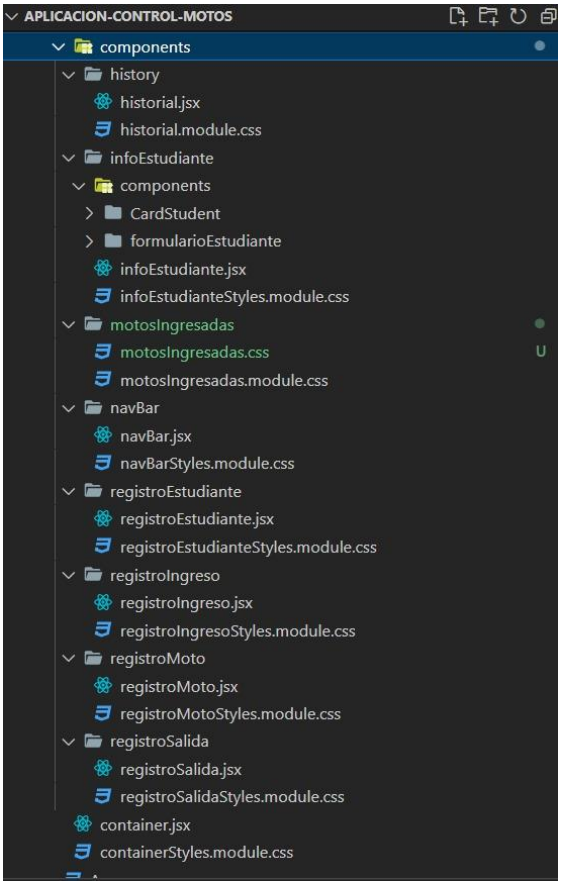


Ilustración 41. Distribución de los componentes en el Frontend

4.3.2 ORGANIZACIÓN DEL BACKEND

En la distribución del Backend se utiliza el principio de programación por capas, en donde el controllers es el puente de conexión entre el Backend y el Frontend, el models se encarga de toda la esturctura de datos a utilizar y finalmente el routes controla el enrutamiento en el aplicativo.

A continuación, se explican las funcionalidades de los principales archivos y carpetas alojadas en el Backend del aplicativo:

- **Carpeta src:** Contiene todo el conjunto de archivos y carpetas que componen el proyecto.
- **Carpeta Controllers:** En esta carpeta se encuentran todos los archivos que sirven de puente entre el Backend y Frontend.

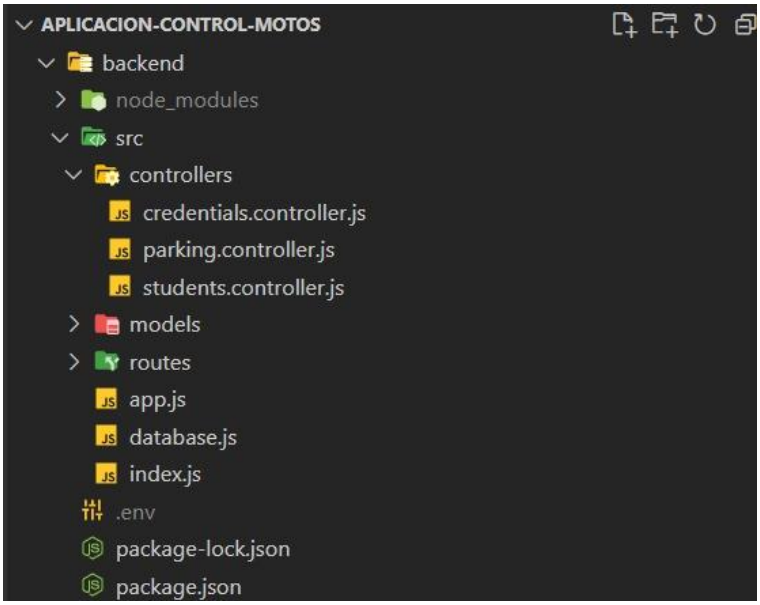


Ilustración 42. Carpeta controllers - Backend

- **Carpeta Models:** En esta carpeta se encuentran todos los archivos en donde se estructuran lo datos a utilizar en todo el aplicativo. Es decir, los datos de los estudiantes, las motos y los registros de entrada y salida de estos.

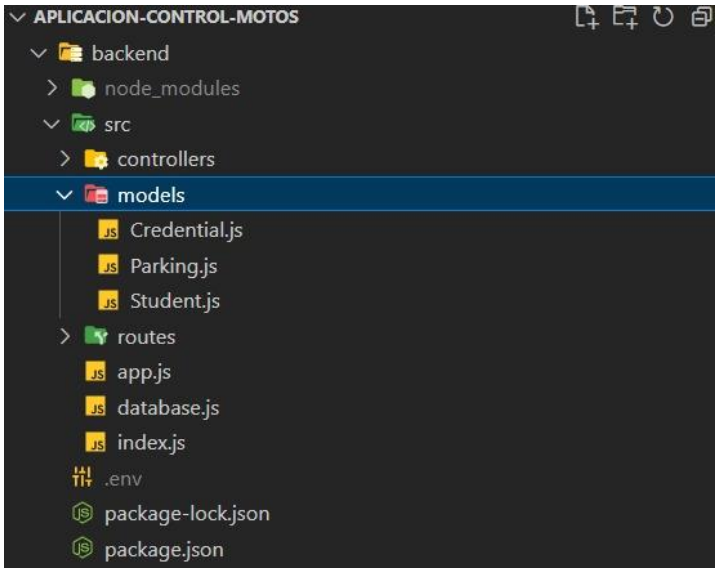


Ilustración 43. Carpeta models - Backend

- **Carpeta Routes:** En esta carpeta se realiza la configuración de las rutas a utilizar.

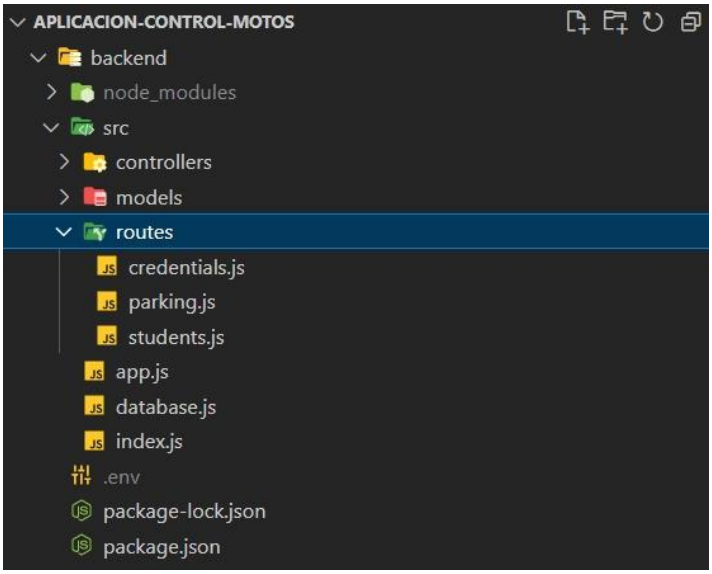


Ilustración 44. Carpeta routes - Backend

- **Archivo app.js:** Este archivo es el servidor del aplicativo.
- **Archivo database.js:** En este archivo se realiza la conexión con la base de datos a utilizar en este caso MongoDB.
- **Archivo index.js:** Este archivo se encarga de llamar el servidor (app.js) cada vez que se realice una petición.

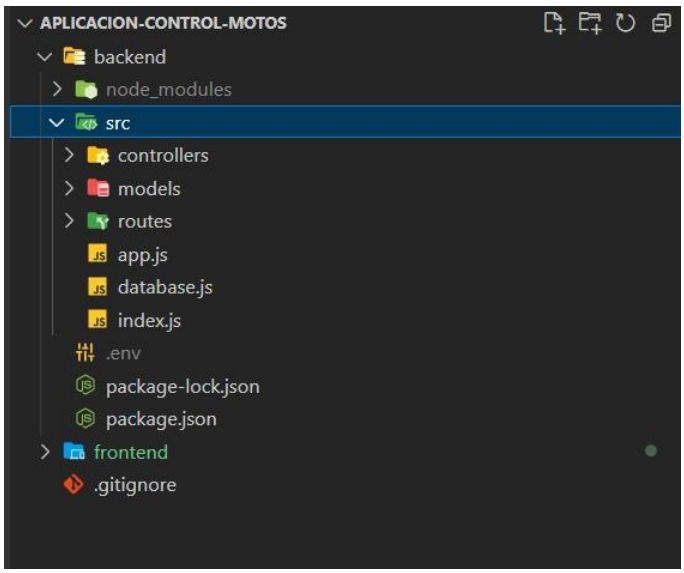


Ilustración 45. Distribución Backend

4.3.3 CONTROL DE VERSIONAMIENTO POR GIT

En el presente proyecto se utilizó el control de versiones gestionado por Git, lo cual ayuda al equipo de desarrollo a realizar seguimiento a todas las modificaciones realizadas en el código.
Además, ayuda a realizar un respaldo de la aplicación en caso de algún inconveniente y no perder información.

En el desarrollo del aplicativo, se manejaron ramas de Git, lo que quiere decir que cada una de ellas contiene una versión del proyecto pero que no afecta con ninguna otra a la hora de unificar el código.

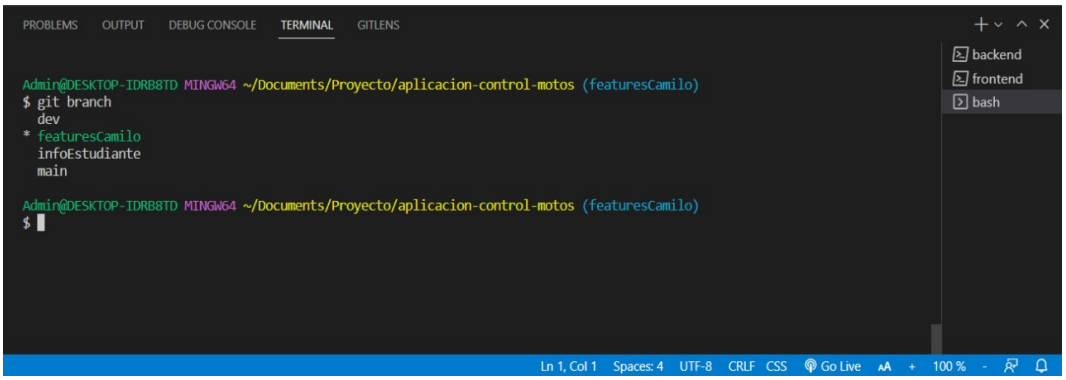


Ilustración 46. Ramas utilizadas en Git

CAPÍTULO 5

5. PRUEBAS

En el siguiente apartado se realizan las pruebas correspondientes a cada uno de los Sprint desarrollados, descritos en el inciso 3.1.

5.1. PRUEBAS SPRINT 1

5.1.1 PRUEBA 1 DEL SPRINT 1

ID Prueba	01
Usuario	Administrador
Descripción de la prueba	Comprobar que el formulario de registro de estudiantes permita ser diligenciado de manera adecuada.
Acciones	• No ingresar ningún dato en el formulario. • Ingresar algunos campos e intentar registrar al estudiante.
Resultado	• Si no se ingresa ningún dato, el sistema lanza una alerta que dice “Completa este campo”. (Ilustración 47). • Si los datos no son ingresados en su totalidad, el sistema lanza una alerta que dice “Completa este campo”, dejara de aparecer hasta que los campos sean diligenciados totalmente. (Ilustración 48).

Tabla 40. Prueba 1 del Sprint 1

Ilustración 47. Prueba 1 del Sprint 1 – Diligenciamiento de información en registro de estudiante

Ilustración 48. Prueba 1 del Sprint 1 - Validación de campos formulario registro de estudiante

5.1.2 PRUEBA 2 DEL SPRINT 1

ID Prueba	02
Usuario	Administrador
Descripción de la prueba	Comprobar que los campos del formulario registrar estudiante acepten los tipos de datos correctos
Acciones	• Intentar ingresar letras o caracteres especiales en los campos cedula y código.
Resultado	• En los campos cedula y código, no se logra ingresar letras o caracteres especiales, se puede visualizar que los campos contienen las flechas de incremento y disminución, esto significa que solo aceptan números.

Tabla 41. Prueba 2 del Sprint 1

Ilustración 49. Prueba 2 del Sprint 1 - Validación de tipos de datos en el formulario registro de estudiante

5.1.3 PRUEBA 3 DEL SPRINT 1

ID Prueba	03
Usuario	Administrador
Descripción de la prueba	Comprobar que la información registrada en el formulario registro de estudiantes se guarde correctamente en las bases de datos.
Acciones	• Ingresar todos los datos y dar clic en continuar. • Revisar el servicio por medio de la aplicación de escritorio Postman. • Revisar en las bases de datos la información guardada.
Resultado	• El formulario deja guardar sin ningún problema (Ilustración 50). • Se revisa Postman y la información se guarda correctamente (Ilustración 51). • Se revisa la base de datos en MongoDB y la información se guarda correctamente (Ilustración 52).

Tabla 42. Prueba 3 del Sprint 1

Ilustración 50. Prueba 3 del Sprint 1 - Validación de guardado de información

```
[
    {
        "_id": "632cbc5ced99678b9ed9a1ca",
        "cedula": 1024576995,
        "codigo": 20202678023,
        "nombre": "Camilo",
        "apellido": "Rojas",
        "credential": [
            "632d05271de46553779a0f48",
            "632cbcc1ed99678b9ed9a1cd",
            "632cbe3ced99678b9ed9a1d4"
        ],
        "createdAt": "2022-09-22T19:49:48.036Z",
        "updatedAt": "2022-09-23T01:05:39.243Z",
        "__v": 0
    },
```


Ilustración 51. Prueba 3 del Sprint 1 - Validación de guardado de información del estudiante en Postman

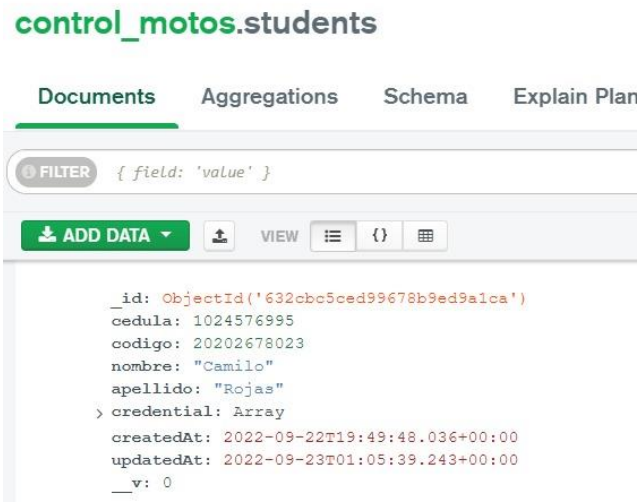


Ilustración 52. Prueba 3 del Sprint 1 - Validación de guardado de información del estudiante en MongoDB

5.1.4 PRUEBA 4 DEL SPRINT 1

ID Prueba	04
Usuario	Administrador
Descripción de la prueba	Comprobar que no acepte duplicidad de estudiantes
Acciones	• Ingresar todos los datos y dar clic en continuar.
Resultado	• El sistema si encuentra duplicidad de información lanza la alerta correspondiente (Ilustración 53).

Tabla 43. Prueba 4 del Sprint 1

Ilustración 53. Prueba 4 del Sprint 1 - Validación de duplicidad de estudiantes

5.2. PRUEBAS SPRINT 2

5.2.2 PRUEBA 1 DEL SPRINT 2

ID Prueba	05
Usuario	Administrador
Descripción de la prueba	Comprobar que la búsqueda de un estudiante sea exitosa
Acciones	• Ingresar el código estudiantil y dar clic en Buscar
Resultado	• El sistema encuentra el estudiante, adicionalmente muestra la información del estudiante y las motos a su nombre. (Ilustración 54)

Tabla 44. Prueba 1 del Sprint 2

Ilustración 54. Prueba 1 Sprint 2 - Validación de la búsqueda de un estudiante

5.1.2 PRUEBA 2 DEL SPRINT 2

ID Prueba	06
Usuario	Administrador
Descripción de la prueba	Comprobar que sucede si un estudiante no está registrado en la plataforma.
Acciones	• Ingresar el código estudiantil no existente en la base de datos y dar clic en Buscar
Resultado	• El sistema arroja el respectivo mensaje. (Ilustración 55)

Tabla 45. Prueba 2 del Sprint 2

Ilustración 55. Prueba 2 Sprint 2 - Validación de la búsqueda de un estudiante

5.3. PRUEBAS SPRINT 3

5.3.1 PRUEBA 1 DEL SPRINT 3

ID Prueba	07
Usuario	Administrador
Descripción de la prueba	Comprobar la funcionalidad de editar estudiante en el sistema
Acciones	• Dar clic en editar (Icono), ingresar los datos y guardar
Resultado	• El sistema arroja la respectiva alerta de edición exitosa. (Ilustración 57)

Tabla 46. Prueba 1 Sprint 3

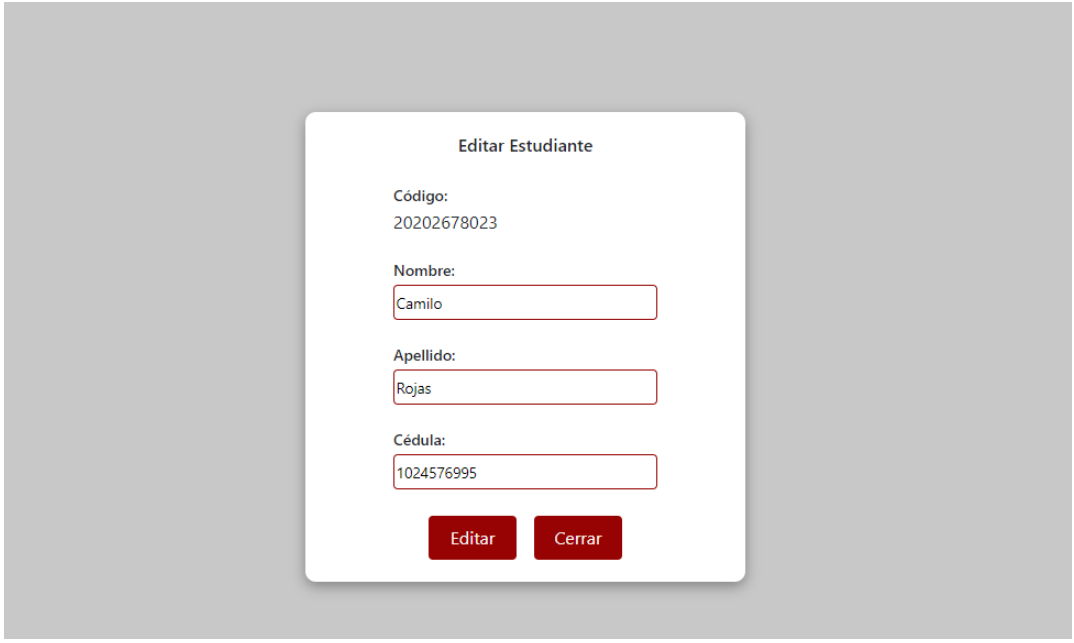


Ilustración 56. Información del estudiante a editar

Ilustración 57. Prueba 1 Sprint 3 - Validación de alerta en la edición del estudiante

5.3.2 PRUEBA 2 DEL SPRINT 3

ID Prueba	08
Usuario	Administrador
Descripción de la prueba	Comprobar si la información del estudiante fue actualizada por Postman y MongoDB.
Acciones	• Buscar la información la información del estudiante en Postman y compararla con la ingresada (Ilustración 60). • Buscar la información la información del estudiante en MongoDB y compararla

	con la ingresada (Ilustración 60).
Resultado	• El sistema actualiza correctamente la información consultada en Postman. (Ilustración 58). • El sistema actualiza correctamente la información consultada en MongoDB. (Ilustración 59).

Tabla 47. Prueba 2 Sprint 3

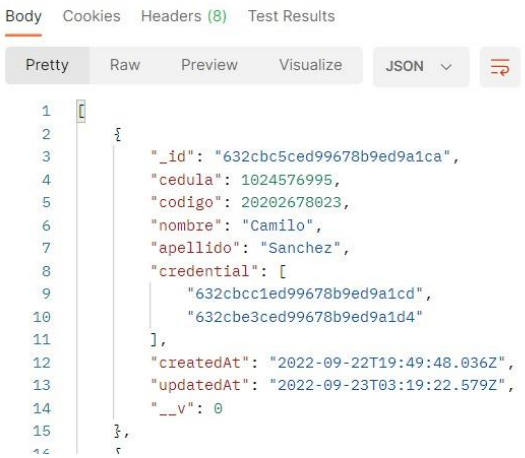


Ilustración 58. Prueba 2 Sprint 3 - Validación de la información editada del estudiante en Postman

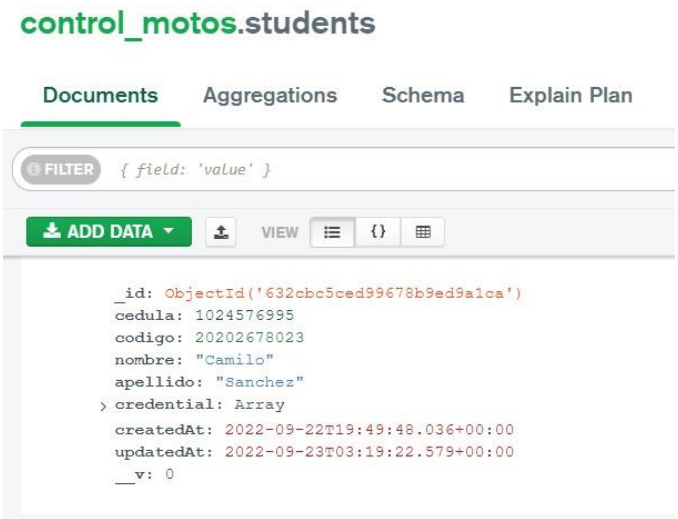


Ilustración 59. Prueba 2 Sprint 3 - Validación de la información editada del estudiante en MongoDB

5.4. PRUEBAS SPRINT 4

5.4.1 PRUEBA 1 DEL SPRINT 4

ID Prueba	09
Usuario	Administrador
Descripción de la prueba	Comprobar que el formulario de registro de motocicletas permita ser diligenciado de manera adecuada.
Acciones	• Ingresar algunos campos e intentar registrar la motocicleta.
Resultado	• Si no se ingresa ningún dato o incompletos, el sistema lanza una alerta que dice “Completa este campo”. (Ilustración 60).

Tabla 48. Prueba 1 Sprint 4

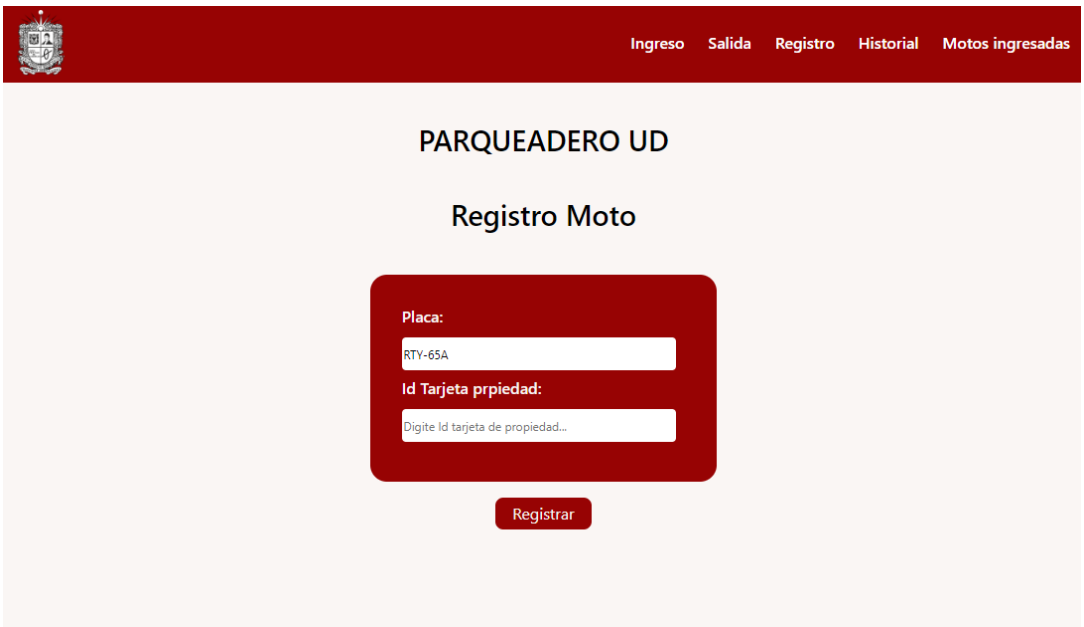


Ilustración 60. Prueba 1 del Sprint 4 - Validación de campos formulario registro de motocicletas

5.5. PRUEBAS SPRINT 5

5.5.1 PRUEBA 1 DEL SPRINT 5

ID Prueba	10
Usuario	Administrador
Descripción de la prueba	Comprobar que el formulario de edición de motocicletas permita ser diligenciado de manera adecuada.

Acciones	• Actualizar la información de la motocicleta.
Resultado	• El sistema genera la actualización sin problemas, muestra la alerta de edición exitosa. (Ilustración 62)

Tabla 49. Prueba 1 Sprint 5

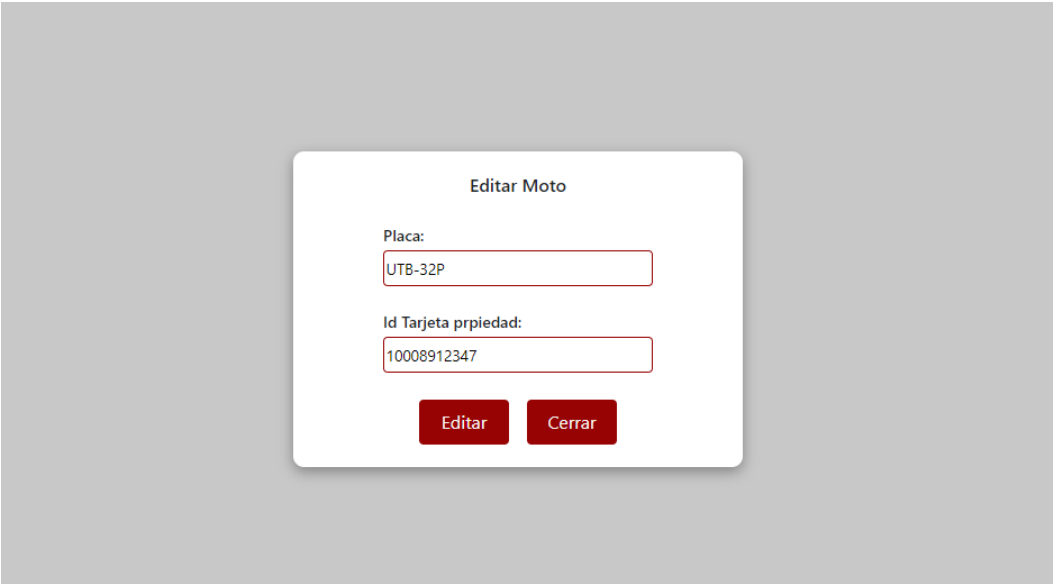


Ilustración 61. Prueba 1 del Sprint 5 - Formulario edición de motocicletas

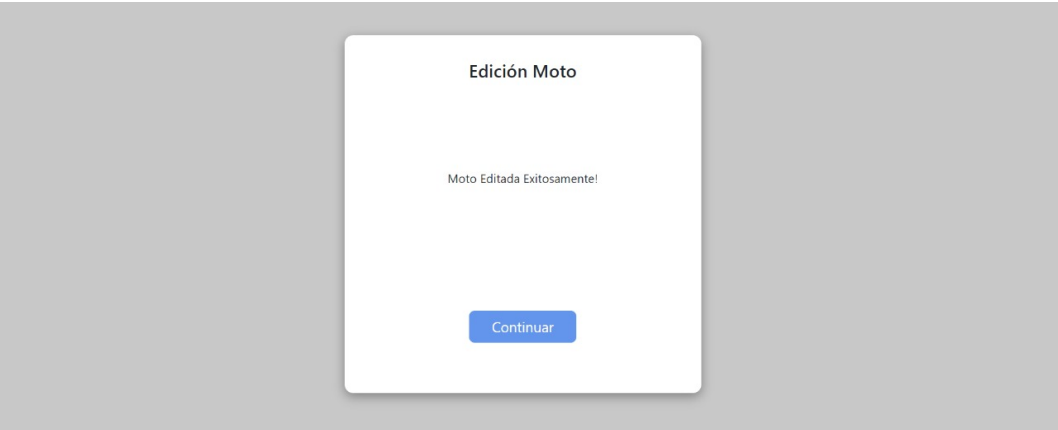


Ilustración 62. Prueba 1 del Sprint 5 - Alerta de edición exitosa

5.5.2 PRUEBA 2 DEL SPRINT 5

ID Prueba	11
Usuario	Administrador
Descripción de la prueba	Comprobar que la información haya sido actualizada correctamente.
Acciones	• Verificar en la pestaña de motocicletas del aplicativo.

	• Verificar la información actualizada en Postman. • Verificar la información actualizada en Postman
Resultado	• La información mostrada en el aplicativo es correcta (Ilustración 63). • La información mostrada en Postman es correcta (Ilustración 64). • La información mostrada en la base de datos en MongoDB es correcta (Ilustración 65).

Tabla 50. Prueba 2 del Sprint 5 - Verificación de información actualizada sobre motocicletas

Ilustración 63. Prueba 2 del Sprint 5 - Verificación de la actualización de motocicleta en el aplicativo

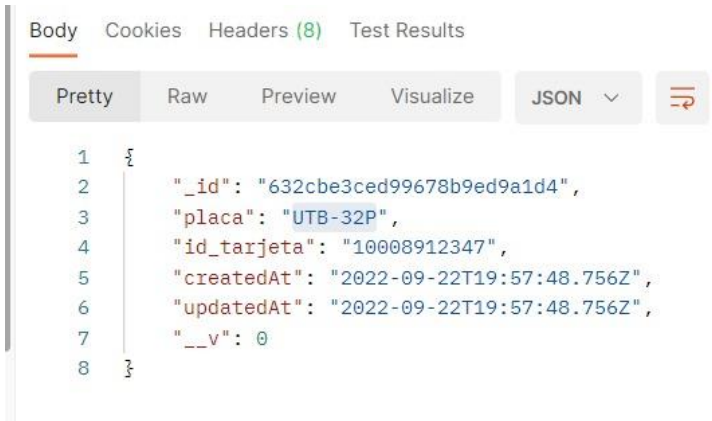


Ilustración 64. Prueba 2 del Sprint 5 - Verificación de la actualización de motocicleta en Postman

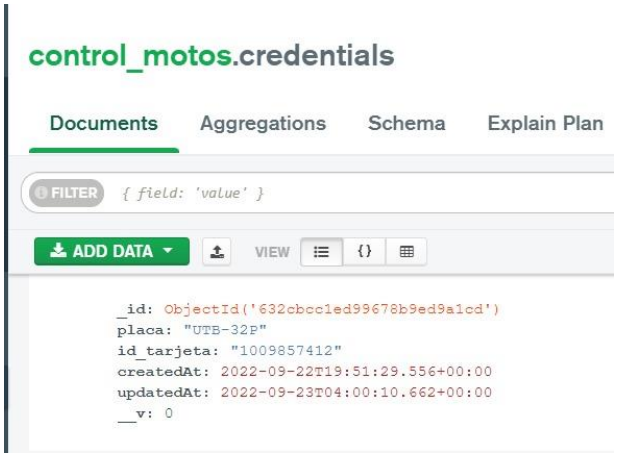


Ilustración 65. Prueba 2 del Sprint 5 - Verificación de la actualización de motocicleta en MongoDB

5.6. PRUEBAS SPRINT 6

5.6.1 PRUEBA 1 DEL SPRINT 6

ID Prueba	11
Usuario	Administrador
Descripción de la prueba	Comprobar el buscador de las tablas de listado actual e histórico funcionen correctamente
Acciones	• Ingresar el código de un estudiante y buscar.
Resultado	• Se realiza la búsqueda del código ingresado. (Ilustración 67).

Tabla 51. Prueba 1 del Sprint 6

Ilustración 66. Prueba 1 del Sprint 6 - Listado Histórico de entradas y salidas de motocicletas.

Ilustración 67. Prueba 1 del Sprint 6 - Búsqueda exitosa del código ingresado

CONCLUSIONES

- El uso de Stack Mern facilitó el desarrollo de la aplicación desarrollada, ya que, al contar con diversas tecnologías basadas en el mismo lenguaje de programación, el aprendizaje y adaptación al conjunto es creciente. Adicionalmente al ser un Framework pensado a futuro, la aplicación no quedará obsoleta en corto plazo.
- El tener en cuenta una arquitectura basada en capas (Backend, Frontend y Base de datos), asegura que el sistema sea altamente escalable, dando facilidad de actualizaciones, migraciones, potencializaciones, entre otros.
- La implementación de un lector de código de barras es de gran ayuda para el sistema, ya que ayuda a realizar la verificación de los estudiantes en un menor tiempo, mejorando el flujo de entradas y salidas de la facultad.
- La metodología Scrum fue de gran importancia para la culminación del proyecto, ya que simplifica el desarrollo realizado, separándolo en pequeñas porciones de las cuales se deben realizar entregas parciales. Es una metodología con la que se pueden establecer objetivos y alcances desde el inicio del proyecto, sigue un plan de trabajo organizado y está sujeto a mejoras constantes que aporten al proyecto.
- El desarrollo de cada uno de los subsistemas da como resultado un proyecto pensado en satisfacer las necesidades que tiene la Universidad Distrital respecto al flujo de entrada y salida de motocicletas pertenecientes a los estudiantes de esta, su implementación ayudará no solo al estudiante y si no al personal de seguridad de la facultad, reduciendo tiempos y minimizando riesgos.

REFERENCIAS BIBLIOGRÁFICAS

1. LEAN, Management. Metodología Rup: ¿Qué es, ¿cuál es su objetivo y cómo se utiliza? Lean Management Blog [página web]. [Consultado el 8, julio, 2022]. Disponible en Internet: <https://lean-management.site/rup/#Las_4_fases_del_RUP>.

2. MIXPANEL, Stack tecnológico: qué es y cómo construir uno, Mixpanel [página web]. [Consultado el 9, julio, 2022]. Disponible en Internet: <https://mixpanel.com/es/topics/what-is-a-technology-stack/>.

3. ETECE, Editorial. HTML - Concepto, historia, ¿Cómo funciona y etiquetas, Concepto? [página web]. (2021). [Consultado el 9, julio, 2022]. Disponible en Internet: <https://concepto.de/html/>.

4. PEIRÓ, Rosario. Lenguaje CSS - Definición, qué es y concepto. Economipedia [página web]. (2017). [Consultado el 9, julio, 2022]. Disponible en Internet: <https://economipedia.com/definiciones/lenguaje-css.html>.

5. PARADA, Miguel. MERN Stack: Qué es y qué ventajas ofrece. OpenWebinars.net [página web]. (21, octubre, 2020). [Consultado el 9, julio, 2022]. Disponible en Internet: <https://openwebinars.net/blog/mern-stack-que-es-y-que-ventajas-ofrece/>.

6. RUBENFA. MongoDB: qué es, cómo funciona y cuándo podemos usarlo (o no). Genbeta - Software, descargas, aplicaciones web y móvil, desarrollo [página web]. (3, febrero, 2014). [Consultado el 9, julio, 2022]. Disponible en Internet: <https://www.genbeta.com/desarrollo/mongodb-que-es-como-funciona-y-cuando-podemos-usarlo-o-no>.

7. REACT. Empezando – React. React – Resumen de la documentación de React y recursos relacionados. [página web]. [Consultado el 9, julio, 2022]. Disponible en Internet: <https://es.reactjs.org/docs/getting-started.html>.

8. GUEST. Bootstrap: ¿Qué es, para qué sirve y cómo instalarlo? Rock Content - ES [página web]. (12, abril, 2022). [Consultado el 9, julio, 2022]. Disponible en Internet: <https://rockcontent.com/es/blog/bootstrap/>.

9. SIMÕES, Chiyana. ¿Qué es Node.js, y para qué sirve? Blog ITDO - Agencia de desarrollo Web, Apps y Marketing en Barcelona [página web]. (27, julio, 2021). [Consultado el 9, julio, 2022]. Disponible en Internet: <https://www.itdo.com/blog/que-es-node-js-y-para-que-sirve/>.

10. JSON. Introducción a JSON, JSON [página web]. [Consultado el 9, julio, 2022]. Disponible en Internet: <https://www.json.org/json-es.html>.

11. EL COMERCIO. ¿Qué es HTTP? Y otros datos sobre protocolos en Internet. El Comercio Perú [página web]. [Consultado el 9, julio, 2022]. Disponible en

Internet: <https://elcomercio.pe/respuestas/que/http-otros-datos-protocolos-internet-noticia-451298-noticia/>.

12. AWS. ¿Qué es el middleware? | Middleware explicado | AWS. Amazon Web Services, Inc. [página web]. [Consultado el 9, julio, 2022]. Disponible en Internet: <https://aws.amazon.com/es/what-is/middleware/#:~:text=El%20middleware%20es%20un%20software,se%20pueda%20innovar%20más%20rápido.>.

13. MURADAS, Yanina. Qué es Postman y primeros pasos. OpenWebinars.net [página web]. (3, junio, 2019). [Consultado el 9, julio, 2022]. Disponible en Internet: <https://openwebinars.net/blog/que-es-postman/>.

14. MICROSOFT. Documentation for Visual Studio Code. Visual Studio Code - Code Editing. Redefined [página web]. (3, noviembre, 2021). [Consultado el 9, julio, 2022]. Disponible en Internet: <https://code.visualstudio.com/docs>.

15. DISTRITAL, Universidad Distrital Francisco José de Caldas. Obtenido de Consejo superior universitario. [página web]. (11, octubre, 2012). [Consultado el 9, julio, 2022] Disponible en Internet: <https://sgral.udistrital.edu.co/xdata/csu/acu_2012-004.pdf>

16. GOV. Ley 1581 de 2012 - Gestor Normativo. El servicio público es de todos - Función Pública [página web]. [Consultado el 9, julio, 2022]. Disponible en Internet: <https://www.funcionpublica.gov.co/eva/gestornormativo/norma.php?i=49981>.

17. GOV. Decreto 2851 de 2013 - Gestor Normativo. El servicio público es de todos - Función Pública [página web]. [Consultado el 9, julio, 2022]. Disponible en Internet: <https://www.funcionpublica.gov.co/eva/gestornormativo/norma.php?i=55853>.

18. GOV. Ley 23 de 1982 - Gestor Normativo. El servicio público es de todos - Función Pública [página web]. [Consultado el 9, julio, 2022]. Disponible en Internet: <https://www.funcionpublica.gov.co/eva/gestornormativo/norma.php?i=3431#:~:text=Esta%20Ley%20protege%20exclusivamente%20la,obras%20literarias,%20científicas%20y%20artísticas.>.

19. ARANGO GRISALES, Claudia Viviana y APARICIO MARTINEZ, Jeison Camilo. Software de control de acceso y registro de vehículos para los parqueaderos de la Facultad Tecnológica en la Universidad Distrital Francisco José de Caldas, con lector de huella digital. Repositorio Institucional Universidad Distrital - RIUD [página web]. (2019). [Consultado el 9, julio, 2022]. Disponible en Internet:

<https://repository.udistrital.edu.co/bitstream/handle/11349/22453/Aparicio%20Martinez%20Jeison%20Camilo%202019.pdf?sequence=1&isAllowed=y>.

20. CAMARGO RUIZ, Diego Fernando y TORRES MENDOZA, Jorge David. Sistema telemático para la gestión del parqueadero de motocicletas de la facultad tecnológica - RIUD [página web]. (2022). [Consultado el 9, julio, 2022]. Disponible en Internet: <https://repository.udistrital.edu.co/bitstream/handle/11349/28126/TorresMendozaJorgeDavid2021.pdf?sequence=1&isAllowed=y>.

21. AZURE. Estilo de arquitectura de n niveles - Azure Architecture Center. Developer tools, technical documentation, and coding examples | Microsoft Docs [página web]. [Consultado el 10, julio, 2022]. Disponible en Internet: <https://docs.microsoft.com/es-es/azure/architecture/guide/architecture-styles/n-tier>.

22. EDIX. Framework: Qué es, para qué sirve y algunos ejemplos. Edix España [página web]. (2022). [Consultado el 10, julio, 2022]. Disponible en Internet: <https://www.edix.com/es/instituto/framework/>.

23. DEEMER, Pete, BENEFIELD, Gabrielle, LARMAR, Craig y VODDE, Bas. Una introducción básica a la teoría y práctica de Scrum. Versión 2.0. [Libro online]. (2022) [Consultado el 13, septiembre, 2022]. Disponible en Internet: <https://scrumprimer.org/primers/es_scrumprimer20.pdf>.

24. ATLASSIAN. Scrum, aprende a utilizar scrum con lo mejor de él. [página web]. (2022) [Consultado el 13, septiembre, 2022]. Disponible en internet: <https://www.atlassian.com/es/agile/scrum>.

25. RED HAT, ¿Que es una Api de REST? [página web]. (2020). [Consultado el 13, septiembre, 2022]. Disponible en internet: <https://www.redhat.com/es/topics/api/what-is-a-rest-api>.

26. BRADY, ¿Qué es y cómo funciona un lector de código de barras? [página web] (2022). [Consultado el 13, septiembre, 2022]. Disponible en internet: <https://www.etiquetas-laboratorio.com/blog/que-es-y-como-funciona-un-lector-de-codigo-de-barras/>.

Bogotá D. C., 22 de septiembre de 2022

Señores:
CONSEJO CURRICULAR
TECNOLOGÍA EN SISTEMATIZACIÓN DE DATOS E INGENIERÍA EN TELÉMATICA
Universidad Distrital F.J.C
Ciudad

Ref. Cumplimiento de Objetivos Modalidad de Grado Culminada

Respetados Señores:

En cumplimiento del reglamento de trabajo de grado, atentamente me permito presentar el trabajo de grado titulado "SISTEMA DE REGISTRO Y CONTROL DE ACCESO Y SALIDA DE MOTOCICLETAS EN LA FACULTAD TECNOLÓGICA DE LA UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS BASADO EN API RESTFUL" en la modalidad de monografía desarrollado por los estudiantes Jeisson Camilo Rojas Sánchez, 20202678023, 1.024.576.995 y Santiago Lopez Gomez, 20202678041, 1.073.720.013) para optar al título de Ingeniero(s) en Telemática , el cual cumple con los objetivos de la propuesta inicial.

Cordialmente,

LUIS FELIPE WANUMEN SILVA
Docente Universidad Distrital
Facultad Tecnológica