

UNIVERSIDAD AUTONOMA TOMAS FRIAS
CARRERA DE INGENERIA DE SISTEMAS



TRABAJO DIRIGIDO

**SISTEMA DE INFORMACIÓN WEB DE CONTROL VEHICULAR DEL
TRANSPORTE PÚBLICO PARA LA FUNDACIÓN SOFTWARE LIBRE
DE LA CARRERA DE INGENIERIA DE SISTEMAS DE LA
UNIVERSIDAD AUTÓNOMA TOMAS FRIAS**

Para optar el título de

Licenciado en

Ingeniería de Sistemas

POR: SANTOS MACHACA LOPEZ

TUTOR: M. SC. LIC. ANNY MERCADO ALGARAÑAZ

Potosí – Bolivia

2024

DEDICATORIA

A mi amada familia y amigos, por su apoyo incondicional y comprensión durante largas jornadas de trabajo y desarrollo del presente proyecto. Su aliento y amor son el motor que me impulsa a seguir adelante y ser un profesional exitoso.

AGRADECIMIENTOS

En primer lugar, mi más profundo agradecimiento a Dios, por ser mi guía y fortaleza, iluminado cada paso de mi camino y otorgándome la resiliencia para superar los restos de la vida.

A mi madrecita Reina López M. que fue un pilar para cumplir mis metas, a mi papito Angel Machaca C. que me ilumina desde el cielo, y mis hermanos y hermanas que me brindaron su apoyo durante el constante de recorrido del aprendizaje.

A mi querida tía Eleuteria López M. que fue como mi segunda madre, a mi padrino Sandro Cayo y mi madrina Beatriz Moya, y mis primos y primas que fueron como mis hermanos y me abrieron las puertas de su hogar, no tengo palabras para expresar la gratitud por el amor, el cariño y el apoyo que me brindaron. Gracias por cada consejo, cada momento compartido, y hacerme sentir parte de su maravillosa familia, su apoyo emocional ha sido una fuente de motivación y parte esencial de mi viaje y siempre ocuparan un lugar en mi corazón.

Quiero expresar mi más sincero agradecimiento a la Universidad Autónoma Tomas Frías en especial a la carrera de Ingeniería de Sistemas por haberme brindado una experiencia académica enriquecedora y transformadora. A lo largo de estos años, he tenido la oportunidad de formalizarme de los docentes altamente capacitados y apasionados, quienes han compartido su conocimiento y experiencia con generosidad y dedicación. Su orientación y apoyo han sido fundamentales para mi crecimiento académico y personal.

Un agradecimiento especial a mi tutora M. Sc. Lic. Anny Mercado Algarañaz y mi encargado, Ing. José David Mamani Figueroa, gracias por brindarme su sabiduría, paciencia, comprensión y orientación, que fueron vitales para el desarrollo del proyecto, gracias a las observaciones y recomendaciones he concluido de manera satisfactoria el presente proyecto.

Quiero expresar mi más profundo agradecimiento a los amigos, quienes compartieron conmigo momentos de tristeza, miedo y estrés. Su apoyo incondicional y mutuo fue fundamental para concluir con éxito este maravilloso meta.

RESUMEN

El presente proyecto se presenta en cumplimiento de los requisitos establecidos por la Universidad Autónoma Tomas Frías como parte del proceso de titulación para la carrera de Ingeniería de Sistemas, bajo la supervisión de la Fundación Software Libre de la carrera de Ingeniería de Sistemas.

El desarrollo del presente sistema representa un avance significativo en la modernización del transporte público de los microbuses, este proyecto busca mejorar la eficiencia operativa, y aumentar la seguridad, elevando la calidad del servicio ofrecido a los usuarios. Mediante el monitoreo en tiempo real.

El sistema se desarrolló utilizando la metodología de Scrum para la representación de la estructura del proyecto, se utilizaron las tecnologías framework next.js para la parte frontend, y nest.js para la parte backend, gestor de base de datos mongo DB para almacenamiento de datos y los lenguajes de programación TypeScript para el desarrollo del sistema de control y monitoreo de los microbuses, y java para el desarrollo de la aplicación GPS, junto a OpenStreetMap de software libre.

Para comprobar el correcto funcionamiento del sistema desarrollado se emplearon las pruebas de funcionamiento y pruebas de aceptación, que no solo aseguraron la funcionalidad técnica del sistema, sino que también validaron su utilidad práctica en entorno real.

CONTENIDO	pág.
INTRODUCCIÓN.....	1
CAPITULO I: MARCO REFERENCIAL	8
1.1. INTRODUCCION	8
1.2. FUNDAMENTOS SOBRE CONTROL VEHICULAR DEL TRANSPORTE PÚBLICO.....	8
1.2.1. MEDIOS DE TRANSPORTE	8
1.2.2. AUTOBÚS URBANO.....	9
1.2.3. TRANSPORTE PÚBLICO	9
1.2.4. AUTOBUSES O COLECTIVOS URBANOS.....	10
1.2.5. CONTROL VEHICULAR URBANO.....	10
1.3. FUNDAMENTOS SOBRE DESARROLLO SISTEMA DE INFORMACIÓN WEB.....	11
1.3.1. SISTEMAS.....	11
1.3.2. SISTEMAS DE INFORMACIÓN GEOGRÁFICA.....	11
1.3.3. SISTEMAS EN LA WEB	12
1.4. FUNDAMENTOS SOBRE METODOLOGÍA DE DESARROLLO	12
1.4.1. LA METODOLOGÍA.....	12
1.4.1.1. METODOLOGÍA SCRUM	13
1.4.1.1.1. SCRUM MASTER	13
1.4.1.1.2. PRODUCT OWNER	14
1.4.1.1.3. DEVELOPER.....	14
1.4.1.1.4. TEST	15
1.4.1.1.5. BACKLOG DE PRODUCTO	15
1.4.1.1.6. BACKLOG DE SPRINT.....	16
1.4.1.1.6.1. HISTORIA DE USUARIO.....	16
1.5. FUNDAMENTOS SOBRE IMPLEMENTACIÓN	17
1.5.1. LENGUAJES DE PROGRAMACIÓN	17
1.5.3. TYPESCRIPT.....	17
1.5.4. JAVA	18
1.5.4. SERVIDOR	19
1.5.5. DOCKER	19
1.5.6. BACKEND	20
1.5.6.1. FRAMEWORK NESTJS.....	20

1.5.7. MONGODB	20
1.5.8. SWAGGER	21
1.5.9. FRONTEND.....	21
1.5.9.1. FRAMEWORK NEXTJS	21
1.5.9.1.1. MATERIAL UI PARA NEXTJS.....	22
1.5.10. HTML 5	22
1.5.11. CSS 3.....	23
1.5.12. OPENSTREETMAP	23
1.5.14. ANDROID STUDIO	24
1.5.15. DISPOSITIVOS GPS	24
1.5.16. DISTANCIA DE RECORRIDO	25
1.5.16.1. OBTENCIÓN DE LA DISTANCIA A TRAVÉS DE LAS COORDENADAS DE LATITUD Y LONGITUD	25
1.6. FUNDAMENTOS SOBRE PRUEBAS	26
1.6.1. PRUEBAS EN INFORMÁTICA	26
1.6.2. TIPOS DE PRUEBAS	26
1.6.2.1. CAJA NEGRA	26
1.6.2.2. CAJA BLANCA.....	27
1.6.2.3. PRUEBA DE INTEGRIDAD	27
1.6.2.4. PRUEBA DE FUNCIONAMIENTO.....	28
1.6.2.5. PRUEBA DE ACEPTACIÓN.....	28
CAPÍTULO II: PLANIFICACIÓN Y DISEÑO.....	31
2.1. INTRODUCCIÓN	31
2.2. DIAGNÓSTICO DEL CONTEXTO	31
2.2.1. ANTECEDENTES	32
2.3. ANÁLISIS DE INSTRUMENTOS.....	32
2.4. PLANIFICACIÓN	33
2.4.1. PRE-JUEGO	33
2.4.1.1. ROLES	33
2.4.2. PLANIFICACIÓN DE LA ITERACIÓN (PRODUCT BACKLOG)	39
2.4.3. PLANIFICACIÓN DE SPRINT	40
2.4.3.1. Planificación de Sprint 1	40
2.4.3.2. Planificación de Sprint 2	41

2.4.3.3. Planificación de Sprint 3	42
2.4.3.4. Planificación de Sprint 4	47
2.5. ESTUDIO DE FACTIBILIDAD	50
2.5.1. Factibilidad operacional	50
2.5.2. Factibilidad técnica.....	51
2.5.2.1. Recursos de hardware	51
2.5.2.1. RECURSOS DE SOFTWARE	51
2.5.3. Factibilidad económica.....	52
2.6. Diseño	53
2.6.1. Sprint 1	53
2.6.2. Sprint 2	54
2.6.3. Sprint 3	55
2.6.4. Sprint 4.....	63
2.7. Arquitectura de datos	64
2.7.1. Diagrama de entidad relación	64
CAPITULO III: IMPLEMENTACIÓN Y PRUEBAS	66
3.1. Introducción	66
3.2. Implementación de sprint 1	67
3.2.1. Control de acceso al sistema	67
3.2.2. Roles y permisos.....	68
3.3. Implementación de sprint 2	71
3.3.1. Aplicación GPS.....	71
3.4. Implementación de sprint 3	72
3.4.1. Listar Usuarios	72
3.4.2. Registrar Usuarios	73
3.4.3. Eliminar Usuarios.....	74
3.4.4. Listar Buses.....	75
3.4.5. Registrar buses	76
3.4.6. Asignar chofer	77
3.4.7. Eliminar bus.....	78
3.4.8. Listar línea	79
3.4.9. Asignar horarios	80
3.4.10. Asignar Tarifas.....	82

3.4.11. Asignar buses.....	84
3.4.12. Registrar línea.....	88
3.4.13. Eliminar líneas	89
3.4.14. Registrar Rutas y paradas	90
3.4.15. Listar Rutas	91
3.4.16. Eliminar rutas	92
3.4.17. Registro de rutas	93
3.4.18. Listar tarifas.....	94
3.4.19. Listar tipos de tarifa.....	95
3.4.20. Registrar Tarifas.....	96
3.4.21. Eliminar Tarifas.....	97
3.4.22. Listar horarios.....	98
3.4.23. Listado de días.....	99
3.4.24. Registro de horario.....	100
3.4.25. Registro de licencia de conducir	101
3.5. Implementación de sprint 4	102
3.5.1. Monitoreo, listar rutas y paradas, ver velocidad de microbuses y ver reporte de las solicitudes	102
3.5.2. Listar líneas.....	103
3.5.3. Listar tarifas.....	104
3.5.4. Listar Horarios	107
3.5.5. Estados de conexión de usuarios.....	108
3.6. Seguridad	109
3.7. Pruebas	109
3.7.1. Pruebas Funcionales.....	110
3.8. Pruebas de aceptación	120
CONCLUSIONES	129
RECOMENDACIONES	130
REFERENCIAS BIBLIOGRÁFICAS	131
BIBLIOGRAFÍA	137

ÍNDICE DE TABLAS

2.4.1. HISTORIA DE USUARIOS.....	34
Tabla 1. Historia de usuario 1	34
Tabla 2. Historia de usuario 2	34
Tabla 3. Historia de usuario 3	34
Tabla 4. Historia de usuario 4	35
Tabla 5. Historia de usuario 5	35
Tabla 6. Historia de usuario 6	35
Tabla 7. Historia de usuario 7	36
Tabla 8. Historia de usuario 8	36
Tabla 9. Historia de usuario 9	37
Tabla 10. Historia de usuario 10	37
Tabla 11. Historia de usuario 11	37
Tabla 12. Historia de usuario 12	38
Tabla 13. Historia de usuario 13	38
Tabla 14. Historia de usuario 14	38
Tabla 15. Historia de usuario 15	39
Tabla 16. Historia de usuario 16	39
Tabla 17: Producto Backlog	39
Tabla 18. Sprint 1 backlog – Control de acceso al sistema.....	40
Tabla 19. Sprint 1 backlog – Roles y permisos	41
Tabla 20. Sprint 2 backlog – Aplicación GPS.....	41
Tabla 21. Sprint 3 backlog – Registrar usuarios	42
Tabla 22. Sprint 3 backlog – Registrar microbuses	43

Tabla 23. Sprint 3 backlog – Registrar líneas.....	44
Tabla 24. Sprint 3 backlog – Registrar rutas y paradas.....	44
Tabla 25. Sprint 3 backlog – Registrar tarifas de los microbuses.....	45
Tabla 26. Sprint 3 backlog – Registrar horarios de salida	45
Tabla 27. Sprint 3 backlog – Registrar licencia de conducir.....	46
Tabla 28. Sprint 4 backlog – Monitoreo.....	47
Tabla 29. Sprint 4 backlog – Ver rutas y paradas.....	48
Tabla 30. Sprint 4 backlog – Ver Tarifas.....	48
Tabla 31. Sprint 4 backlog – Ver estados de conexión.....	48
Tabla 32. Sprint 4 backlog – Ver velocidad de microbuses.....	49
Tabla 33. Sprint 4 backlog – Ver reporte de las solicitudes de las líneas.....	49
Tabla 34: recursos de hardware.....	51
Tabla 35: recursos de software.....	52
Tabla 36: tabla de costos.....	52
Tabla 37: especificación de sprints	53
Tabla 38: especificación de sprints	54
Tabla 39: especificación de sprints	55
Tabla 40: especificación de sprints	63
Tabla 41. Pruebas Funcionales control de acceso al sistema.....	110
Tabla 42. Pruebas Funcionales roles y permisos	110
Tabla 43. prueba funcionales Aplicación GPS.....	111
Tabla 44. Pruebas funcionales Registrar usuarios	112
Tabla 45. Pruebas funcionales Registrar microbuses	113
Tabla 46. Pruebas funcionales líneas	114
Tabla 47. Pruebas funcionales Registrar rutas y pardas.....	115

Tabla 48. Pruebas funcionales Registrar tarifas de los microbuses	116
Tabla 49. Pruebas funcionales registrar horarios de salida	116
Tabla 50. Pruebas funcionales registrar licencia de conducir.....	117
Tabla 51. Pruebas funcionales Monitoreo.....	118
Tabla 52. Pruebas funcionales Ver rutas y paradas.....	119
Tabla 53. Pruebas funcionales Ver tarifas	119
Tabla 54. Pruebas funcionales ver estado de conexión	119
Tabla 55. Pruebas funcionales ver velocidad de microbuses.....	120
Tabla 56. Pruebas funcionales ver reporte de las solicitudes de las líneas.....	120
Tabla 57. Pruebas de aceptación control de acceso al sistema	121
Tabla 58. Pruebas de aceptación roles y permisos	121
Tabla 59. Pruebas de aceptación aplicación GPS.....	122
Tabla 60. Pruebas de aceptación registrar usuarios	122
Tabla 61. Pruebas de aceptación registrar usuarios	123
Tabla 62. Pruebas de aceptación registrar microbús.....	123
Tabla 63. Pruebas de aceptación registrar líneas.....	124
Tabla 64. Pruebas de aceptación registrar rutas y paradas	125
Tabla 65. Pruebas de aceptación registrar tarifas de los microbuses.....	125
Tabla 66. Pruebas de aceptación registrar horarios de salida	126
Tabla 67. Pruebas de aceptación Monitoreo	126
Tabla 68. Pruebas de aceptación ver rutas y paradas.....	127
Tabla 69. Pruebas de aceptación ver rutas y tarifas	127
Tabla 70. Pruebas de aceptación ver estados de conexión.....	127
Tabla 71. Pruebas de aceptación ver velocidad de microbuses.....	127
Tabla 72. Pruebas de aceptación ver reporte de solicitudes de líneas	128

ÍNDICE DE FIGURAS

Figura 1: fórmula para la calcular distancia	25
Figura 2: Arquitectura de interfaz de usuario autenticación.....	53
Figura 3: Arquitectura de interfaz de usuario Roles y permisos	54
Figura 4: Arquitectura de interfaz de usuario Aplicación GPS	55
Figura 5: Arquitectura de interfaz de usuario registrar usuarios	56
Figura 6: Arquitectura de interfaz de usuario lista de usuarios	57
Figura 7: Arquitectura de interfaz de usuario lista de usuarios	57
Figura 8: Arquitectura de interfaz de usuario registro de microbuses.....	58
Figura 9: Arquitectura de interfaz de usuario lista de microbuses	58
Figura 10: Arquitectura de interfaz de usuario registro de línea	59
Figura 11: Arquitectura de interfaz de usuario lista de línea.....	59
Figura 12: Arquitectura de interfaz de usuario registro de ruta.....	60
Figura 13: Arquitectura de interfaz de usuario lista de rutas.....	60
Figura 14: Arquitectura de interfaz de usuario registro de tarifa	61
Figura 15: Arquitectura de interfaz de usuario lista de tarifas	61
Figura 16: Arquitectura de interfaz de usuario registro de horario	62
Figura 17: Arquitectura de interfaz de usuario lista de horario	62
Figura 18: Arquitectura de interfaz de usuario monitoreo	63
Figura 19: Diagrama de clases de entidad	64
Figura 20: Implementación de interfaz Login	67
Figura 21: Código fuente de autenticación.....	67
Figura 22: Lista de roles y permisos.....	68
Figura 23: Código fuente de roles y permisos.....	68
Figura 24: Agregar roles.....	69
Figura 25: Código fuente de Agregar roles	69
Figura 26: Editar roles	70

Figura 27: Código fuente de editar roles	70
Figura 28: Aplicación GPS.....	71
Figura 29. Código fuente de la aplicación GPS	72
Figura 30. Listar usuarios	72
Figura 31. Código fuente de listar usuarios.....	73
Figura 32: Registrar Usuarios.....	73
Figura 33. Código fuente de Registrar Usuarios	74
Figura 34. Eliminar Usuarios	74
Figura 35. Código fuente de eliminar usuarios	75
Figura 36. Listar Buses.....	75
Figura 37. Código fuente de Listar Buses	76
Figura 38. Registrar buses	76
Figura 39. Código fuente de Registrar buses.....	77
Figura 40. Asignar chofer	77
Figura 41. Código fuente de Asignar chofer	78
Figura 42. Eliminar bus	78
Figura 43. Código fuente de Eliminar bus.....	79
Figura 44. Listar línea.....	79
Figura 45. Código fuente de Listar línea	80
Figura 46. Asignar horarios	80
Figura 47. Código fuente de Asignar horarios	81
Figura 48. Lista de horario	81
Figura 49. Código fuente de lista de horario	82
Figura 50. Asignar Tarifas.....	82
Figura 51. Código fuente de Asignar Tarifas	83
Figura 52. Vista previa ver tarifas	83
Figura 53. Código fuente vista previa ver tarifas	84
Figura 54. Asignar buses	84

Figura 55. Código fuente de Asignar buses.....	85
Figura 56. Asignar rutas	85
Figura 57: código fuente de asignar rutas.....	86
Figura 58: Vista previa rutas	86
Figura 59: código fuente de vista previa ver rutas.....	87
Figura 60: Asignar o desasignar rutas a bus de la línea	87
Figura 61: Código fuente de Asignar o desasignar rutas a bus de la línea	88
Figura 62. Registrar línea	88
Figura 63. Código fuente de Registrar línea.....	89
Figura 64. Eliminar líneas	89
Figura 65. Código fuente de Eliminar líneas.....	90
Figura 66. Registrar Rutas y paradas.....	90
Figura 67. Código fuente de Registrar Rutas y paradas	91
Figura 68. Ver rutas	91
Figura 69. Código fuente de Listar Rutas.....	92
Figura 70. Eliminar rutas	92
Figura 71. Código fuente de Eliminar rutas	93
Figura 72. Registro de rutas.....	93
Figura 73. Código fuente de Registro de rutas	94
Figura 74. Listar tarifas	94
Figura 75. Código fuente de Listar tarifas	95
Figura 76. Listar tipos de tarifa	95
Figura 77. Código fuente de Listar tipos de tarifa	96
Figura 78. Registrar Tarifas.....	96
Figura 79. Código fuente de Registrar Tarifas	97
Figura 80. Eliminar Tarifas	97
Figura 81. Código fuente de Eliminar Tarifas.....	98
Figura 82. Listar horarios	98

Figura 83. Código fuente de Listar horarios.....	99
Figura 84. Listado de días	99
Figura 85. Código fuente de Listado de días	100
Figura 86. Registro de horario.....	100
Figura 87. Código fuente de Registro de horario	101
Figura 88. Registro de licencia de conducir	101
Figura 89. Código fuente de Registro de licencia de conducir.....	102
Figura 90. Monitoreo	102
Figura 91. Código fuente de Monitoreo	103
Figura 92. Listar líneas	103
Figura 93. Código fuente de Listar líneas	104
Figura 94. Listar tarifas	104
Figura 95. Código fuente de Listar tarifas.....	105
Figura 96: Listar rutas de la línea	105
Figura 97: Código fuente de Listar rutas de la línea.....	106
Figura 98: Ver buses cercanos.....	106
Figura 99: Código fuente de Ver buses cercanos	107
Figura 100. Listar Horarios	107
Figura 101. Código fuente de Listar Horarios.....	108
Figura 102. Estados de conexión de usuarios.....	108
Figura 103. Código fuente de Estados de conexión de usuarios	109

INTRODUCCIÓN

En el mundo entero existen muchos transportes públicos que son fundamental para el desplazamiento y mejora de calidad de vida de los ciudadanos. Los medios de transporte fueron inventados para reducir el tiempo de movimientos y trasportar las mercancías de los individuos. En el contexto de desarrollo de un sistema de control para los medios de trasportes vehicular es vital debidamente que no solo busca la modernización de la gestión vehicular de trasportes públicos, sino también aumenta la seguridad y calidad de vida de los ciudadanos mejorando la experiencia de los usuarios y contribuye el desarrollo sostenible de la ciudad.

En el ámbito laboral de los choferes de trasportes públicos como microbuses afrontan un desafíos y oportunidades que exigen el control y monitoreo constante de los mismo, debido que muchos de ellos aplican velocidades muy altas, sin importar las vidas de los ciudadanos ocasionando accidentes no deseados. Por otra parte, el control de los cobros y pagos de los pasajes tiene poca relevancia que todos estos desafíos afectan no solo a la organización o ciudadanos, sino que también afecta al patrimonio cultural de la ciudad.

En este sentido, se destaca la importancia de desarrollar un sistema de control de microbuses como un medio para abordar los desafíos actuales mencionados, que proporcionará a las autoridades de transporte público de microbuses y los usuarios herramientas para la toma de decisiones más informada, reduciendo los tiempos de esperas mejorando la puntualidad, además mencionará las respectivas rutas de circulación de los microbuses.

Habiendo mencionado las debilidades del transporte público de los microbuses, muchas ciudades enfrentan estos desafíos relacionados con la gestión vehicular del transporte público de los microbuses, incluyendo los problemas de seguimiento ineficiente de rutas y falta de herramientas para la toma de decisiones informadas. La implementación de un sistema de control vehicular bajo la plataforma web surge como una respuesta innovadora y tecnológica para abordar estos desafíos.

Mediante el presente proyecto se pretende implementar, un sistema para el control vehicular para los microbuses, basándose en la administración de los microbuses de la ciudad de Potosí con el fin de mejorar la calidad de vida de los ciudadanos y el patrimonio cultural de la ciudad,

siendo una herramienta de apoyo para la organización y los usuarios de la misma identidad. Para ello se plantea el problema de la siguiente manera.

La fundación de Software Libre de la carrera de Ingeniería de Sistemas de la Universidad Autónoma Tomas Frías, siendo una institución competente enfrenta desafíos, en la actualidad la identidad cuenta con muchas softwares gratuitos adaptables para cualquiera institución, sin embargo no cuenta con un software como para el control y monitoreo de los transportes de microbuses de las ciudades que es muy novedoso la implementación de este software, y además es muy útil a la hora de toma de decisiones.

Por otra parte, en muchas ciudades el transporte público de microbuses tiene poca seguridad debido a que muchos de ellos aplican velocidades muy altas y no hay un personal que los monitoree debido a estas acciones se provocan accidentes no deseados y malestares de los ciudadanos. De igual manera en cuanto los pasajes y obtención de las rutas de los microbuses es poco eficiencia, los choferes en algunas ocasiones cobran pasajes fuera de reglamentos y en algunas ocasiones los ciudadanos no conocen las rutas de las líneas, esto de igual manera provoca una eficiencia en cuanto al turismo.

Por lo tanto, se hace necesario llevar a cabo la implementación de un sistema de información para el control y monitoreo de transporte público de los microbuses, para abordar estos desafíos y mejorar la calidad de vida de los ciudadanos. Esta iniciativa no solo busca optimizar los procesos y la formar de controlar los microbuses, sino también mejorar el turismo en las ciudades que utilicen este software. (**VER ANEXO 1**).

Tomando en cuenta la problemática descrita se establece el siguiente **problema de investigación:**

¿Cómo coadyuvar a la Fundación de Software Libre en el control vehicular de microbuses del transporte público?

El **objeto de estudio** del presente trabajo se centra en los sistemas de información de control con aplicación de los sistemas de información geográfica bajo plataforma web.

El **campo de acción** se centra en la automatización del control vehicular de microbuses del transporte público para la Fundación Software Libre de la carrera de Ingeniería de Sistemas de la Universidad Autónomas Tomas Frías.

Se tiene como **objetivo**: Desarrollar un sistema de información bajo plataforma web para el control vehicular de microbuses del transporte público de la ciudad de Potosí para la Fundación Software Libre de la carrera de Ingeniería de Sistemas de la Universidad Autónoma Tomas Frías.

Para este proyecto se plantean las siguientes **preguntas científicas**:

- ¿Cuál es el fundamento teórico de respaldo al desarrollo sistema de información bajo plataforma web para el control vehicular de microbuses del transporte público?
- ¿Cuál es la situación actual del control vehicular de transporte público en la ciudad de Potosí?
- ¿Cómo obtener el sistema de información bajo plataforma web para el control vehicular de microbuses del transporte público ?
- ¿Cómo demostrar la funcionalidad y validez sistema de información bajo plataforma web para el control vehicular de microbuses del transporte público?

Las tareas de investigación abordan aspectos clave para el desarrollo del sistema que coadyuban fundamentar y sustentar el software desarrollado, las cuales son:

- Elaboración de un marco teórico que permita fundamentar adecuadamente el desarrollo sistema de información bajo plataforma web para el control vehicular de microbuses del transporte público a través de la lectura del material bibliográfico.
- Realización de un diagnóstico que permita determinar la situación actual del proceso de administración del trasporte públicos de microbuses, utilizando los métodos y técnicas de recolección de información como la observación y la entrevista.
- Obtención sistema de información bajo plataforma web para el control vehicular de microbuses del transporte público utilizando la metodología de SCRUM.
- Implementación del sistema de información bajo plataforma web para el control vehicular de microbuses del transporte público utilizando OpenStreetMap para el manejo

de la información geográfica, las tecnologías de framework Nextjs para el diseño de frontend, Nestjs para la parte backend, Android Studio para el desarrollo de la aplicación gps, y los lenguajes de programación TypeScript, Java y mongo DB como base de datos.

- Realización de pruebas de funcionalidad y validación que permita garantizar el correcto funcionamiento del sistema y la verificación del logro de los requisitos establecidos para el desarrollo del sistema de información bajo plataforma web para el control vehicular de microbuses del transporte público.

El **diseño metodológico** que muestra el conjunto de métodos teóricos y empíricos que permitieron llevar a cabo la investigación, además de ser fundamental para la recopilación de información se muestra en los siguientes párrafos.

Los **métodos teóricos** que fueron utilizados son:

- **Análisis y Síntesis:** Este método permitió a estudiar el proceso de la administración del control y monitoreo vehicular de los trasportes públicos de los microbuses, permitiendo desarrollar los componentes de este medio y formar un sistema funcional.
- **Inducción y Deducción:** coadyubó realizar un estudio exhaustivo de la funcionalidad de proceso de control y monitoreo actual del transporte público de microbuses, basándose en la administración de control de los microbuses de la ciudad de potosí, permitiendo solucionar las fallas identificadas.

Entre los **métodos empíricos** se utilizaron:

- **Entrevista:** Este método ayudó a recopilar la información para el desarrollo del sistema a la necesidad de la institución, se empleó una entrevista al encargado de la institución de Software Libre de la carrera de Ingeniería de Sistemas de la Universidad Autónoma Tomas Frías (**Ver anexo 2, 3**).
- **Encuesta:** Este método fue utilizado para recopilar la información de los trasportes públicos de microbuses en las distintas ciudades del mundo se empleó una encuesta a la población de las distintas ciudades con el fin de obtener la información del manejo de microbuses (**Ver anexo 4**).

El presente trabajo se encuentra plenamente **justificado** tanto social, como económica y tecnológicamente.

La **justificación social** del presente proyecto es de gran importancia, puesto que beneficia a la institución brindando una solución efectiva para el control del transporte público vehicular, por otra parte, beneficia a la población dándole información sobre las rutas del transporte público, permitiendo ver en tiempo real la ubicación de los microbuses.

Asimismo, se justifica desde el punto de vista **económico**, el software es de gran importancia porque monitorea en tiempo real, el recorrido y sus velocidades de los microbuses, por el cual puede ser importante para evitar accidente y ocasionar consto en los daños del bus y seguridad de cliente, por otra parte, muestra el tiempo de llegadas de microbuses a partir de la ubicación de usuario, que es beneficioso y eficiente para el usuario, evitando gastos en otros servicios similares. En algunos países, la implementación de tecnologías que mejoran la eficiencia energética y la seguridad vial puede calificar para incentivos fiscales, reduciendo el costo total de la inversión.

Al presentar estos argumentos, puedes construir un caso sólido para justificar la inversión en la implementación de un sistema de control y monitoreo vehicular de microbuses, destacando cómo los beneficios a corto y largo plazo superan los costos iniciales. El costo de software es gratuito, utiliza herramientas y servicios gratuitos y es adaptable a cualquiera empresa de microbuses o sindicatos, sin embargo, tiene costos para la implementación de mismo debido a que se deben adquirir equipos computacionales para su funcionamiento.

En cuanto a la justificación **técnica**, el sistema que se desarrolla utiliza herramientas actuales las cuales son:

- **TypeScript: (versión 5.2)**, Es una extensión del lenguaje de programación JavaScript que se caracteriza por ampliar su sintaxis en el ámbito de tipeado, que gracias a ellos permite controlar los tipos de datos y mejora el control de datos de manera eficiente.
- **Nestjs**: framework de desarrollo web basado en Node. Js que utiliza TypeScript para proporcionar una estructura de programación sólida y altamente escalable, este framework se utilizará para el desarrollo de software la parte backend del lado servidor.

- **Nextjs: (versión 14.0.0),** Es un framework de react que facilita la creación de aplicaciones web modernas y eficientes, tiene la capacidad de realizar la renderización en el lado del servidor como en el lado cliente. En el presente proyecto se utilizará para el desarrollo del software la parte frontend.
- **MongoDB: (versión 6.0.7),** Es un gestor de base de datos no relacional, su forma de trabajar en documentos, esta base de datos se utilizará para dar seguridad a la información almacenada y permitirá el almacenamiento de datos de manera masiva.
- **Java y Android Studio:** Android Studio es un entorno de desarrollo integrado (IDE) oficial para el desarrollo de aplicaciones Android, basado en el software JetBrains IntelliJ IDEA. Para soportar el desarrollo de aplicaciones sobre el sistema operativo Android, Android Studio utiliza un sistema de construcción basado en Gradle, emuladores, plantillas de código e integración con Github. Por otra parte, java es un lenguaje de programación orientada a objetos o polimorfismo.
- **OpenStreetMap y leaflet:** OpenStreetMap es un proveedor de mapas gratuito donde se puede crear mapas editables, se distribuye bajo la licencia abierta. En cambio, leaflet es una biblioteca de JavaScript que permite el manejo del mapa de manera eficiente.

La **estructura** del trabajo se encuentra dividida en tres capítulos:

- Capítulo I: Este capítulo se centra en el marco referencial que abarca los fundamentos teóricos para el desarrollo del trabajo.
- Capítulo II: Este capítulo se enfoca a la planificación y diseño del sistema en donde se realiza un diagnóstico al control de los microbuses públicos urbano para determinar los requerimientos del sistema mediante el uso de la metodología Scrum y sus fases.
- Capítulo III: Está orientado a la implementación y las pruebas que demostraron el correcto funcionamiento del sistema desarrollado.

Finalmente, las actividades y el tiempo requerido para el desarrollo del trabajo fueron establecidas mediante la elaboración de un **cronograma** (ver anexo 5).

CAPÍTULO I

MARCO

REFERENCIAL

CAPITULO I: MARCO REFERENCIAL

1.1. INTRODUCCION

El marco referencial es una sección fundamental en cualquier trabajo de investigación, ya que proporciona el contexto necesario para entender el estudio. En esta parte, se establecen los conceptos y teorías relevantes, se revisa la literatura existente y se justifican las bases teóricas y metodológicas que sustentan el estudio. A través del marco referencial, se clarifican los antecedentes del problema investigado y se identifican las principales variables y sus relaciones. Esto no solo ayuda a situar el estudio dentro del campo de conocimiento, sino que también facilita la comprensión del problema de investigación y orienta el diseño metodológico del estudio.

1.2. FUNDAMENTOS SOBRE CONTROL VEHICULAR DEL TRANSPORTE PÚBLICO

1.1.1. MEDIOS DE TRANSPORTE

Los medios de transporte son las distintas formas que utiliza y ha utilizado el ser humano para llevar de un punto a otro una carga o un pasajero. Estos han sido de vital importancia para la historia de la humanidad, pues nos han permitido llevar productos de un sitio a otro, nos han permitido viajar por las diferentes regiones del planeta y por fuera de este, nos han facilitado la movilidad dentro de las urbes y entre el campo y las ciudades (encyclopedia online, 2024).

Los medios de transporte son vehículos que se utilizan para el traslado de personas o mercancías. Esto, permitiría una primera clasificación. No obstante, en muchos casos, estos medios de transporte pueden transportar a personas y mercancías al mismo tiempo. Estos vehículos han sido construidos para desplazarse en diversos ambientes (Páez, 2020).

En tanto de tomar los conceptos se determina que los medios de transporte son una forma de desplazarse de un punto a otro, este mecanismo es utilizado por los seres vivientes humanos para el traslado de las mercancías u otros.

Existen varios tipos de medios de transporte, la finalidad de cada uno es el desplazamiento, sin embargo, en el presente proyecto se enfoca en medios de transporte de los microbuses urbanos públicos.

1.2.2. AUTOBÚS URBANO

“Un autobús urbano, autobús de ciudad, autobús público o autobús de tránsito es un autobús utilizado para el transporte público de distancias cortas. Las funciones y especificaciones de los autobuses de tránsito no son claras y varían en función del operador y de la región.” (wikipedia, 2023)

En el transporte urbano de personas se puede distinguir el público y el privado. Dentro de los transportes públicos existen los de libre conducción, que en general actúan por el rodamiento neumático, como son el taxi, autobuses, etc. Y el de Conducción forzada, que actúan por la rodadura sobre guías (una o más), como son el tranvía, ferrocarriles, etc. Según su definición autobús u ómnibus es un vehículo automóvil de transporte público dotado de muchas plazas. (casiopea, 2015)

De tomar estos conceptos se determina que el autobús urbano se destaca como un medio de transporte público de corta distancia, con funciones y especificaciones que pueden variar según el operador y la región. Este vehículo, diseñado para el transporte masivo, se diferencia de otros transportes públicos, como los taxis, por su capacidad para albergar a un gran número de pasajeros.

1.2.3. TRANSPORTE PÚBLICO

El transporte público es la red de servicio colectivo para garantizar que la sociedad tenga a disposición diversos medios de traslado que se ajusten a sus necesidades, preferencias y posibilidades económicas. A diario, millones de personas utilizan líneas de autobús o colectivo, trenes, taxis, bicicletas públicas y subterráneos o metro (Gudiña, 2023).

El transporte público es el término aplicado al transporte colectivo de pasajeros. A diferencia del transporte privado, los viajeros del transporte público tienen que adaptarse a los horarios y a las rutas que ofrezca el operador y dependen en mayor o menor medida de la intervención regulatoria del Gobierno (wikipedia, 2023).

De haber tomado estos conceptos se define que el transporte público es un medio que permite moverse a los humanos de un punto a otro, este medio está a disposición del público de bajo costo en cuanto a pasajes, sin embargo, tienen una sola ruta por donde transitar.

1.2.4. AUTOBUSES O COLECTIVOS URBANOS

Un autobús urbano, autobús de ciudad, autobús público o autobús de tránsito es un autobús utilizado para el transporte público de distancias cortas. Las funciones y especificaciones de los autobuses de tránsito no son claras y varían en función del operador y de la región (wikipedia).

Se define como autobús a un vehículo que cuenta con la capacidad de transportar entre 10 y 120 pasajeros. Por lo general, son usados en las líneas urbanas e interurbanas que funcionan en la ciudad y cuentan con un trayecto fijo, es decir, cumplen con una ruta establecida. Estos se encuentran elaborados en distintos tamaños, modelos y diseños que se ajustan a las necesidades de las carreteras (Mediotrasporte, 2023).

Tomando estos conceptos de determinan que los medios de transporte tienen la función de trasladar a los pasajeros de un punto a otro de en corta distancia.

1.2.5. CONTROL VEHICULAR URBANO

Los sistemas de Control de Tránsito Urbano (UTC) requieren señales de tránsito, controladores de señal, medición en rampas y carteles de mensajes variables para controlar el tránsito. También se requiere un sistema de comunicaciones para la transferencia de los datos del sensor de tránsito hasta los controladores del equipamiento y de señales. comunicaciones de datos entre los diferentes controladores algoritmos inteligentes que usan la información acerca de las condiciones actuales del tránsito para predecir las futuras cargas vehiculares y soportar las decisiones sobre las medidas de control de la red y de la optimización del tránsito de diversas maneras, para minimizar las demoras, mejorar el rendimiento vehicular y reducir la cantidad de arranques y paradas, las emisiones tóxicas y el consumo de combustible (Piarc, 2024).

Tomando estos conceptos se define que el control de vehicular varía dependiendo a cada ciudad, sin embargo, el control vehicular es una forma de mejorar el rendimiento y seguridad del transporte público autobuses urbanos.

1.3. FUNDAMENTOS SOBRE DESARROLLO SISTEMA DE INFORMACIÓN WEB

1.3.1. SISTEMAS

Se entiende por un sistema a un conjunto ordenado de componentes relacionados entre sí, ya se trate de elementos materiales o conceptuales, dotado de una estructura, una composición y un entorno particulares. Se trata de un término que aplica a diversas áreas del saber, como la física, la biología y la informática o computación. (concepto, 2023)

Del latín *systema*, un sistema es modulo ordenado de elementos que se encuentran interrelacionados y que interactúan entre sí. El concepto se utiliza tanto para definir a un conjunto de conceptos como a objetos reales dotados de organización (Quinteros, 2020)

Entonces se puede decir que un sistema es un conjunto de elementos que están relacionados entre sí, un sistema puede tener otro sistema dentro de ella o también puede formar de otro sistema, a cuál se la llaman también como ecosistema.

1.3.2. SISTEMAS DE INFORMACIÓN GEOGRÁFICA

Un sistema de información geográfica es un conjunto de herramientas que integra y relaciona diversos componentes que permiten la organización, almacenamiento, manipulación, análisis y modelización de grandes cantidades de datos procedentes del mundo real que están vinculados a una referencia espacial, facilitando la incorporación de aspectos sociales-culturales, económicos y ambientales que conducen a la toma de decisiones de una manera más eficaz. (wikipedia, 2023)

Un sistema de información geográfica (SIG) es un sistema empleado para describir y categorizar la Tierra y otras geografías con el objetivo de mostrar y analizar la información a la que se hace referencia espacialmente. Este trabajo se realiza fundamentalmente con los mapas. (esri, 2023)

Entonces se puede concluir que un Sistema de Información Geográfica (SIG) es una herramienta integral que aborda la organización, almacenamiento, manipulación, análisis y modelización de grandes cantidades de datos del mundo real vinculados a una referencia espacial. Estos sistemas permiten la incorporación de aspectos sociales, culturales, económicos y ambientales, lo que contribuye a una toma de decisiones más eficaz.

1.3.3. SISTEMAS EN LA WEB

El sistema web o también denominado aplicaciones web se define como aplicaciones de software que se puede usar en un servicio web por medio de internet o de una intranet desde un navegador. Actualmente, el sistema web es muy utilizado por la razón de que es muy rápida y práctica en el navegador web. De hecho, las aplicaciones web evita gastos lo que significa a que no será necesario en aprender a manejar nuevos programas que puedan ser costosos. (Crea System, 2023)

Los sistemas web son aplicaciones o plataformas informáticas que funcionan a través de una red, generalmente internet, y son accesibles desde navegadores web en dispositivos como computadoras, tabletas o teléfonos móviles. Estos sistemas permiten a los usuarios interactuar, compartir información, realizar transacciones y de interactuar en línea. (Borovsky, 2023)

De haber tomas estos conceptos se define que los sistemas en la web son llamados también aplicaciones web o plataformas en la web que no son instalables si no que están alojados en un servidor, esto hace que sea accesible por dispositivos móviles o computadoras entre otros dispositivos que cuenten con un navegador, para poder acceder a estas plataformas solo necesitan la conexión de datos.

1.4. FUNDAMENTOS SOBRE METODOLOGÍA DE DESARROLLO

1.4.1. LA METODOLOGÍA

La metodología es una serie de métodos y técnicas de rigor científico que se aplican sistemáticamente durante un proceso de investigación para alcanzar un resultado teóricamente válido. (Coelho, 2023)

Hace referencia al conjunto de procedimientos racionales utilizados para alcanzar el objetivo o la gama de objetivos que rige una investigación científica, una exposición doctrinal o tareas que requieran habilidades, conocimientos o cuidados específicos. Con frecuencia puede definirse la metodología como el estudio o elección de un método pertinente o adecuadamente aplicable a determinado objeto. (wikipwdia, 2023)

De acuerdo de estas definiciones recuperados de las distintas fuentes se determina que la metodología son conjunto de procesos a seguir para concluir y demostrar de manera eficiente el desarrollo de una actividad propuesta.

1.4.1.1. METODOLOGÍA SCRUM

Scrum es un marco de trabajo ágil a través del cual las personas pueden abordar problemas complejos adaptativos a la vez que se entregan productos de forma eficiente y creativa con el máximo valor. Así, Scrum es una metodología que ayuda a los equipos a colaborar y realizar un trabajo de alto impacto. (Martines, 2023)

En SCRUM se trabaja con Sprints, es decir, el proyecto se divide en pequeñas partes para poder abordarlas de forma más rápida y eficiente. Un proyecto puede estar compuesto por varios sprints que cuando se concluyen dan el resultado esperado. (ilimit, 2022)

Entonces de haber tomado los conceptos se lo define la metodología scrum como el trabajo ágil que trabaja por sprints, aunque no es recomendable el uso de esta metodología para un equipo de un solo integrante, pero se puede aplicar, no existe una regla de la metodología que indique al respecto.

1.4.1.1.1. SCRUM MASTER

“El scrum master (sm) o facilitador de proyectos, es la figura que lidera los equipos en la gestión ágil de proyectos. su misión es que los equipos de trabajo alcancen sus objetivos hasta llegar a la fase de sprint final, eliminando cualquier dificultad que puedan encontrar en el camino”. (Paula, 2022)

Un Scrum Master es el líder del equipo de Scrum. Está a cargo de establecer la metodología Scrum y mantener a los miembros del equipo enfocados en los principios y las prácticas de Scrum. Los Scrum Masters suelen tener habilidades interpersonales y disfrutan de ayudar a los miembros del equipo a crecer y mejorar. (Raeburn, asana, 2023)

Habiendo obtenido las fuentes de información de scrum mastes se determina scrum master es el líder del equipo de desarrollo que ayudan a crecer y mejorar las dificultades de equipo, como scrum master tiene varias responsabilidades a diferencia de un gerente tradicional cuyo objetivo es mantener al equipo y al proyecto en el camino correcto.

1.4.1.1.2. PRODUCT OWNER

El Product Owner o propietario del producto es un rol estándar en los equipos de scrum que se enfoca en entregar el mejor producto posible a los usuarios finales. Para hacerlo, los propietarios desarrollan una visión de cómo debería funcionar el producto, definen características específicas y transforman esas características en tareas del trabajo pendiente(backlog) del producto, para que el equipo de Scrum trabaje en ellas. El Product Owner es responsable del producto final y actúa como nexo entre las partes interesadas del negocio, los miembros del equipo de Scrum y los usuarios finales. (MacNeil, 2023)

Un Product Owner es el profesional encargado de optimizar el valor del producto desarrollado por una empresa. Traducido al castellano como propietario de producto y mencionado a menudo con las siglas PO, el Product Owner debe asegurarse de que el equipo con el que trabaja aporta auténtico valor al proyecto, encargándose de gestionar las tareas del backlog y decidiendo sobre las características del producto. (Molina, 2021)

Por lo tanto, producto owner es el encargado de entrega de productos, su responsabilidad es optimizar el valor del producto, gestionar las tareas del backlog. El producto Owner se encarga de preocuparse por la mejora constante del producto, atendiendo tanto su funcionalidad como su coherencia con la misión de la empresa.

1.4.1.1.3. DEVELOPER

Developer o desarrollador es la persona experta en escribir código, es decir, idear el conjunto de secuencias de órdenes que llevan a un sistema informático a realizar una acción concreta. La herramienta principal de lo que es un developer son los lenguajes de programación. Estos son sistemas de escritura con sus normas sintácticas y gramaticales que permiten transmitir información e instrucciones a un software o computador. (Keepcoding, 2023)

Un software developer, o desarrollador de software, es un profesional encargado de crear, diseñar y mantener programas y aplicaciones informáticas. Su labor fundamental consiste en escribir líneas de código que permitan a las computadoras ejecutar tareas específicas. Estos profesionales utilizan lenguajes de programación y herramientas especializadas para dar vida a ideas y convertirlas en una herramienta funcional. (kodigo, 2023)

Tomando estos conceptos se determinan que developer es la persona que está a lado de líneas de código, es el que desarrolla el producto informático, las lógicas de los programas depende de él. Su papel fundamental es el dominio de los lenguajes de programación, a estos desarrolladores se les denomina también como backend a los que trabajan en la parte del servidor, frontend a los que trabajan en la parte de diseño del producto o la cliente, a los que trabajan en ambos se las llama como FullStack.

1.4.1.4. TEST

El tester es aquella persona encargada de realizar el control de calidad de productos de software (Quality Control), es quien debe buscar en programas y aplicaciones diseñados bajo ciertas características, fallas o errores antes de que salga al mercado digital. Existe también el analista de aplicaciones, que a diferencia de los tester, éstos utilizan aplicaciones ya existentes para crear y modelar otras nuevas. (Euroinnova, 2023)

Un tester de software es un profesional especializado en el área de pruebas de software que se ocupa principalmente de evaluar la calidad y el funcionamiento de un desarrollo antes de su lanzamiento o implementación. Así, recae en él la tarea de identificar y revisar la corrección de errores, además de garantizar que el software cumpla con los requisitos y expectativas del cliente. (ICARIA, 2023)

De acuerdo a las definiciones se concluye que el test es un profesional especializado que realiza las pruebas de software, su objetivo principal es enfocar en la calidad de producto antes del lanzamiento, el tester tiene que identificar los errores del software, él tiene que identificar las vulnerabilidades y falencias del software.

1.4.1.5. BACKLOG DE PRODUCTO

El backlog de un producto es una lista de trabajo ordenado por prioridades para el equipo de desarrollo que se obtiene de la hoja de ruta y sus requisitos. Los elementos más importantes se muestran al principio del backlog del producto para que el equipo sepa qué hay que entregar primero. El equipo de desarrollo no trabaja con el backlog al ritmo que dicta el propietario de producto, y este no presiona al equipo de desarrollo para que saque el trabajo adelante. En su lugar, el equipo de desarrollo saca trabajo del backlog del producto en la medida de sus capacidades, ya sea de forma continua (kaban) o por iteraciones (scrum). (Radigan, 2023)

product backlog a una lista de funciones y elementos, ordenados según las prioridades, que son necesarios para cumplir con los objetivos y las expectativas del proyecto. La regla general es desarrollar uno de estos conjuntos de trabajo pendiente del producto para cada producto y asignarle a un equipo ese trabajo pendiente en particular. (Raeburn,2022)

de tomar estas definiciones de concluye que el product backlog es una lista de trabajo como una herramienta colaborativa, esta herramienta proporciona al usuario la prioridad que se tienen que desarrollar o debe ser desarrollado por el equipo, es una lista de identificación de las actividades o de historias de usuario que se tienen que realizar.

1.4.1.1.6. BACKLOG DE SPRINT

“El sprint backlog, es una imagen en tiempo real del trabajo que se planea realizar durante el Sprint, es un plan que hace visible todo el trabajo que el equipo de desarrollo identifica como necesario para cumplir con el objetivo del Sprint” (Friend, 2023)

El backlog del Sprint es una lista de tareas que define el trabajo a realizar por los miembros de equipo scrum durante un sprint. La lista surge durante la planificación del sprint. Las tareas del backlog del sprint son las que el equipo definió como necesarias para convertir a ítems del backlog de producto en funcionalidad del sistema. Cada tarea identifica quién es responsable de hacer el trabajo y un estimado de la cantidad de trabajo faltante para la tarea durante cualquier día del Sprint. (IDEAS, 2020)

De acuerdo a las fuentes obtenidas se determina que el backlog de sprint proporciona una guía clara para el equipo durante el sprint, detallando las tareas específicas que deben completarse para lograr los objetivos establecidos al inicio del sprint. Durante las reuniones diarias de Scrum, el equipo discute el progreso hacia la finalización de las tareas del backlog de sprint y ajusta si es necesario para garantizar el éxito del sprint.

1.4.1.1.6.1. HISTORIA DE USUARIO

“Una historia de usuario es una oración concisa e informal escrita desde la perspectiva de un usuario final o cliente, utilizada para informar las decisiones de diseño. Son utilizadas a menudo en el diseño de la experiencia del usuario (UX) y el desarrollo de productos”. (free, 2023)

Las Historias de Usuario son representaciones de requisitos, elaboradas en una o dos frases y recogidas en un lenguaje común y entendible por el usuario. También conocidas como User Stories o simplemente US se han convertido en un estándar a la hora de definir requisitos. Aunque su popularidad viene de la mano de Scrum. (Álvarez, 2020)

De esta manera de obtener las definiciones se determina que las historias de usuario son los requerimientos de los usuarios, en scrum se realiza una tabla donde se escribe las necesidades de usuario, con la identificación de un código, es muy útil esta herramienta que ayuda a desarrollar el software de acuerdo, a las necesidades del usuario.

1.5. FUNDAMENTOS SOBRE IMPLEMENTACIÓN

1.5.1. LENGUAJES DE PROGRAMACIÓN

Un lenguaje de programación es un conjunto de reglas gramaticales (tanto sintácticas como semánticas) que instruyen a que un ordenador o dispositivo se comporte de una cierta manera. Cada lenguaje de programación tiene un vocabulario, un conjunto único de palabras clave que sigue a una sintaxis especial para formar y organizar instrucciones del ordenador. (chakray, 2023)

En informática, se conoce como lenguaje de programación a un programa destinado a la construcción de otros programas informáticos. Su nombre se debe a que comprende un lenguaje formal que está diseñado para organizar algoritmos y procesos lógicos que serán luego llevados a cabo por un ordenador o sistema informático, permitiendo controlar así su comportamiento físico, lógico y su comunicación con el usuario humano. (concepto, 20203)

Por lo tanto, de tomar las definiciones anteriores se determina que los lenguajes de programación es la comunicación de humanos con las máquinas, se podría decir que también un conjunto de reglas o claves que se indican mediante un editor de código, esto se interpretará a lenguaje máquina y se procederá los procesos lógicos indicadas.

1.5.3. TYPESCRIPT

TypeScript es una extensión al lenguaje de programación JavaScript que se caracteriza por ampliar su sintaxis en el ámbito de los tipos. En este sentido, es un lenguaje de programación propio que se basa en JavaScript para darnos herramientas de desarrollo en cualquier escala de

proyectos. Además de agregar elementos a la sintaxis de JavaScript, TypeScript se conecta de manera más profunda con los editores de código, revisando errores de manera más oportuna. (KeepCoding, 2022)

Actualmente, TypeScript está tomando mucha relevancia, tanto en el desarrollo a nivel backend utilizando Node.js, con la aparición de Deno, por ejemplo, como a nivel de desarrollo frontend a la hora de trabajar con librerías como React o como Vue, que desde su versión 3 ya es TypeScript friendly, porque ha desarrollado su core utilizando TypeScript. (Barrios, 2023)

De haber tomado los conceptos de typescript se define que es una extensión de lenguaje de programación de JavaScript, fue inventado para mejorar al lenguaje JavaScript, debido que maneja los datos de manera tipeado, esto hace que tenga relevancia al momento de manejar los elementos de sintaxis.

1.5.4. JAVA

Java es un lenguaje de programación ampliamente utilizado para codificar aplicaciones web. Ha sido una opción popular entre los desarrolladores durante más de dos décadas, con millones de aplicaciones Java en uso en la actualidad. Java es un lenguaje multiplataforma, orientado a objetos y centrado en la red que se puede utilizar como una plataforma en sí mismo. Es un lenguaje de programación rápido, seguro y confiable para codificarlo todo, desde aplicaciones móviles y software empresarial hasta aplicaciones de macrodatos y tecnologías del servidor (aws, 2023).

Java es un lenguaje de programación orientado a objetos y una plataforma de software ampliamente utilizado que se ejecuta en miles de millones de dispositivos, que incluyen computadoras portátiles, dispositivos móviles, consolas de juegos, dispositivos médicos y muchos otros. Las reglas y la sintaxis de Java se basan en los lenguajes C y C++ (IBM, 2023).

Tomando estos conceptos se define que java es un lenguaje de programación orientada a objetos y es de multiplataforma, basado en las sintaxis de c y c++. Una de las principales ventajas de desarrollar software con Java es su portabilidad. Una vez que haya escrito el código para un programa Java en una computadora portátil, es muy fácil mover el código a un dispositivo móvil.

1.5.4. SERVIDOR

Un servidor es un sistema que proporciona recursos, datos, servicios o programas a otros ordenadores, conocidos como clientes, a través de una red. En teoría, se consideran servidores aquellos ordenadores que comparten recursos con máquinas cliente. Existen muchos tipos de servidores, como los servidores web, los servidores de correo y los servidores virtuales. (Paessler, 2023)

La puesta a disposición de los servicios del servidor a través de una red informática se basa en el modelo cliente-servidor, concepto que hace posible distribuir las tareas entre los diferentes ordenadores y hacerlas accesibles para más de un usuario final de manera independiente. (Ionos, 2023)

Entones se concluye que los servidores son computadoras más potentes que la de uso personal, están orientados para una sola función, su objetivo principal es procesar las tareas de manera rápida y en volúmenes muy grandes.

1.5.5. DOCKER

Docker es una plataforma de código abierto que permite a los desarrolladores crear, implementar, ejecutar, actualizar y gestionar contenedores, que son componentes estandarizados y ejecutables que combinan el código fuente de la aplicación con las bibliotecas y dependencias del sistema operativo (SO) necesarias para ejecutar ese código en cualquier entorno. (IBM, 2023)

A diferencia de una máquina virtual que proporciona virtualización de hardware, un contenedor proporciona virtualización ligera a nivel de sistema operativo mediante la abstracción del "espacio del usuario". Los contenedores comparten el núcleo del sistema host con otros contenedores. Un contenedor, que se ejecuta en el sistema operativo host, es una unidad de software estándar que empaqueta código y todas sus dependencias, para que las aplicaciones se puedan ejecutar de forma rápida y fiable de un entorno a otro. Los contenedores no son persistentes y se activan desde imágenes. (Oracle, 2023)

De acuerdo de tomar las definiciones de determina que Docker es una plataforma de virtualización, esta herramienta esta enfocado para los sistemas operativos Linux que proporciona el trabajo ligero y mas limpio a la hora de instalar aplicaciones en el sistema

operativo Linux. Docker las aplicaciones maneja en contenedores, que se activan mediante las imágenes.

1.5.6. BACKEND

1.5.6.1. FRAMEWORK NESTJS

Nest.js es un framework de desarrollo web basado en Node.js que utiliza TypeScript para proporcionar una estructura de programación sólida y altamente escalable. Se inspira en otros frameworks de desarrollo populares, como Angular y Spring. Y se centra en la creación de aplicaciones de Backend altamente escalables y modulares, aunque también puedes desarrollar aplicaciones web. (Itequia, 2023)

Nestjs proporciona a los desarrolladores una arquitectura escalable, débilmente acopladas y fácil de mantener. Nestjs en su core utiliza patrones de diseños base para el apoyo de una mejor solución, como lo son el patrón repositorio e inyección de dependencias. El aporte de Nestjs va desde las diferentes estrategias modulares hasta plugins elaborados por la comunidad, todo esto inspirado en el framework Angular. (Diego, 2022)

Entonces se define que el framework nestjs es una herramienta para el desarrollo del software de la parte de backend o servidor, proporciona a los desarrolladores una arquitectura escalable, esta herramienta utiliza el patrón de diseño modelo vista y controlador, esta framework está desarrollado en el lenguaje de programación de typescript.

1.5.7. MONGODB

MongoDB (del inglés humongous, "enorme") es un sistema de base de datos NoSQL orientado a documentos de código abierto y escrito en C++, que en lugar de guardar los datos en tablas lo hace en estructuras de datos BSON (similar a JSON) con un esquema dinámico. Al ser un proyecto de código abierto, sus binarios están disponibles para los sistemas operativos Windows, GNU/Linux, OS X y Solaris y es usado en múltiples proyectos o implementaciones en empresas como MTV Network, Craigslist, BCI o Foursquare. (Robledano, 2019)

MongoDB es un sistema para la gestión de datos NoSQL o no relacional. Se trata de un modelo orientado a documentos que se almacenan en BSON, una representación binaria de JSON, y que no usa tablas como los sistemas SQL ni necesita seguir un esquema. (Inesdi, 2022)

Por lo tanto, se concluye que nextjs es un framework de react que esta a lado de servidor y cliente, que ya tiene funciones definidas que permite simplificar el trabajo de los desarrolladores y optimiza las líneas de código, haciendo que los sistemas desarrollados con sus librerías de este framework sean escalables poque sigue un patrón de diseño MVC (Modelo Vista Controlador).

1.5.8. SWAGGER

Debido al amplio abanico de tecnologías y lenguajes de programación, describir las API era una tarea muy compleja en el pasado. Un primer punto importante: para el desarrollo de las API, actualmente se utiliza el paradigma de programación REST. Sitios web como Google, Amazon y Twitter utilizan API de RESTFul. (ionos, 2021)

Swagger es un conjunto de herramientas de software de código abierto para diseñar, construir, documentar, y utilizar servicios web Restful. Fue desarrollado por SmartBear software e incluye documentación automatizada, generación de código, y generación de casos de prueba. (wikipedia, 2023)

Por lo Tanto, Swagger es una herramienta para realizar peticiones a un API, además esta herramienta cuenta con librerías que permiten documentar de la manera eficiente las peticiones que se van a realizar, además permite validar los tokens de sesión. Es una ayuda fundamental para los desarrolladores que se dedican a desarrollar la parte backend.

1.5.9. FRONTEND

1.5.9.1. FRAMEWORK NEXTJS

Next.js es un framework de React que permite construir sitios web y aplicaciones web estáticas supercargadas, amigables con el SEO y extremadamente orientadas al usuario, utilizando el framework de React. Next.js es conocido por la mejor experiencia de los desarrolladores a la hora de construir aplicaciones listas para usar en producción con todas las características que necesitas. (kinsta, 2022)

NextJS es un framework JavaScript ligero y de código abierto creado sobre React, que permite desarrollar aplicaciones y sitios web muy rápidos y fáciles de usar. Aunque tiene una curva de aprendizaje, incluso los desarrolladores nuevos de front-end pueden aprenderlo rápidamente. Se basa en Babel y Node.js, integrándose con React para desarrollar aplicaciones. (aplyca, 2022)

Por lo tanto, se concluye que nextjs es un framework de react que esta a lado de servidor y cliente, que ya tiene funciones definidas que permite simplificar el trabajo de los desarrolladores y optimiza las líneas de código, mejorando las buenas prácticas de programación porque se basa en los componentes y utiliza los hooks de react.

1.5.9.1.1. MATERIAL UI PARA NEXTJS

Material UI es una biblioteca de componentes de interfaz de usuario para React, diseñada para ayudar a los desarrolladores a construir aplicaciones web modernas, siguiendo los principios de Material Design, el sistema de diseño creado por Google. Proporciona una amplia variedad de componentes predefinidos, como botones, formularios, iconos, cuadros de diálogo, tablas, etc. (Gonzales, 2023)

Sigue los diseños de Material Design de Google. Proporciona componentes estilizados y listos para usar para acelerar el desarrollo de aplicaciones web y móviles. Material UI permite a los desarrolladores crear interfaces de usuario atractivas y funcionales con facilidad, mientras mantiene una consistencia en la apariencia y la experiencia de usuario. (Skeleton , 2023)

De acuerdo a las definiciones obtenidas se puede decir que material ui es una biblioteca para el diseño de los componentes de un sitio web, esta librería es para el uso de react que, que tiene muchas funcionalidades y amigable a la hora de desarrollar un software, fue creado pensando en las buenas prácticas de codificación del sistema como tal.

1.5.10. HTML 5

HTML 5 es un Lenguaje de Marcado de Hipertexto (HyperText Markup Language) que funciona con las bases del HTML de siempre, pero que cuenta con etiquetas adicionales que permiten crear sitios webs más modernos y más compatibles con las tecnologías digitales actuales. (Workana, 2021)

HTML, cuyas siglas corresponden a HyperText Markup Language o lenguaje de marcado de hipertexto en español. Lo primero sería aclarar que, aunque se considere así, HTML no es un lenguaje de programación. No posee características propias de este, como las funciones, los bucles y las condiciones. (pablo, 2022)

Se concluye que HTML 5 es la quinta versión de HTML lenguaje marcado de Hipertexto que da la funcionalidad a los sitios web. Para aplicar esto en un editor de código se debe indicar en un apóstrofe menor que y mayor que cerrando de la misma forma agregando el apóstrofe de la división como se puede ver en el siguiente ejemplo:<html>tus etiquetas HTML y textos aquí</html>.

1.5.11. CSS 3

Abreviado en sus siglas en inglés, Cascading Style Sheets, que significa hojas de estilo en cascada, el CSS es una función que se agrega a HTML que proporciona tanto a los desarrolladores de sitios Web, así como a los usuarios, más control sobre cómo se muestran las páginas. Con CSS, los diseñadores y los usuarios pueden crear hojas de estilo que definen cómo aparecen los diferentes elementos, como los encabezados y los enlaces. Estas hojas de estilo se pueden aplicar a cualquier página Web, y nos permite optimizarla para mejorar su posicionamiento. (arimetrics, 2022)

CSS, al igual que HTML, es uno de los lenguajes centrales de Internet. Mientras que para añadir texto a un sitio web se utiliza HTML y se estructura semánticamente, para definir el diseño del contenido se utiliza CSS. Aunque HTML y CSS se utilizan en combinación, las instrucciones de diseño de CSS y los elementos de HTML existen por separado. Esto significa que una máquina puede leer un documento electrónico incluso sin CSS. Con la ayuda de CSS, el contenido del navegador se prepara visualmente y se presenta de forma atractiva. (ionos, 2021)

Por tanto, se define que CSS son hojas de estilo que da forma a la página web, cuando se diseñan sitios web los estilos CSS es imprescindible, porque dará la mejor interactividad al usuario y una bonita vista, con HTML los sitios web son como esqueletos a la comparación del ser humano que funcionan, pero es difícil de manejar.

1.5.12. OPENSTREETMAP

OpenStreetMap (también conocido como OSM) es un proyecto colaborativo para crear mapas editables y libres. En lugar del mapa en sí, los datos generados por el proyecto se consideran su salida principal (wikipedia, 2024).

OpenStreetMap es un mapa editable y libre del mundo entero que está siendo en gran medida elaborado desde cero por voluntarios y publicado con una licencia de contenido abierto.

La licencia de OpenStreetMap permite el acceso libre (o casi libre) a nuestras imágenes de mapas y a todos los datos cartográficos subyacentes. El proyecto tiene como objetivo promover usos nuevos e interesantes de estos datos (wiki, 2023).

Por lo tanto, se define que OpenStreetMap es un proveedor de mapas gratuito que está a disposición de los usuarios que permite realizar ediciones de los mapas y de código abierto.

1.5.14. ANDROID STUDIO

es un entorno de desarrollo integrado (IDE) oficial para el desarrollo de aplicaciones Android, basado en el software JetBrains IntelliJ IDEA. Para soportar el desarrollo de aplicaciones sobre el sistema operativo Android, Android Studio utiliza un sistema de construcción basado en Gradle, emuladores, plantillas de código e integración con Github. Android Studio es el entorno de desarrollo integrado (IDE) oficial de Google para el sistema operativo Android, construido sobre el software JetBrains IntelliJ IDEA y diseñado específicamente para el desarrollo de Android (Navarro, 2024).

Es una plataforma de desarrollo de software que proporciona un conjunto de herramientas y recursos para crear aplicaciones móviles para Android. Es desarrollado y mantenido por Google, y se ha convertido en la opción preferida de muchos desarrolladores debido a su amplia gama de características y su estrecha integración con el ecosistema de Android (franko, 2023).

Tomando estos conceptos de determina que Android Studio es una herramienta para el desarrollo de aplicaciones móviles basado en Android, de igual forma se puede utilizar como un emulador de Android, trabaja con los lenguajes de programación java y kotlin.

1.5.15. DISPOSITIVOS GPS

El Sistema de Posicionamiento Global (GPS; en inglés: Global Positioning System), originalmente Navstar GPS, es un sistema que permite a un dispositivo receptor localizar su propia posición sobre la Tierra con una precisión de hasta centímetros (si se utiliza GPS diferencial), aunque lo común son unos pocos metros (wikipedia, 2024).

El Sistema de Posicionamiento Global (GPS) es un sistema de navegación que utiliza satélites, un receptor y algoritmos para sincronizar datos de ubicación, velocidad y tiempo para viajes aéreos, marítimos y terrestres (Martín González Soto, 2023).

Tomado estos conceptos se determina que un dispositivo GPS tiene la función de mostrar el posicionamiento del propio dispositivo sobre la tierra, por el cual utiliza algoritmos para realizar cálculos a través de los satélites.

1.5.16. DISTANCIA DE RECORRIDO

La distancia recorrida en física es la longitud total del camino que un objeto ha recorrido durante su movimiento. Esta medida es esencial para comprender y cuantificar los desplazamientos y trayectorias de los objetos en el espacio (significados, 2023).

La distancia recorrida es una cantidad escalar representada por un valor numérico seguido de una unidad de medida que responde al recorrido total entre dos puntos, es decir, es la suma de todos los desplazamientos que realiza un objeto en movimiento hasta llegar a su destino. Este valor siempre es positivo, ya que para este cálculo es indiferente la dirección en la que se mueve el objeto, solo se considera la longitud del camino (Marín, 2024).

Por lo tanto, la distancia es una magnitud de física que indica la longitud de recorrido del camino o también se puede decir el recorrido de un punto a otro y el valor siempre es positivo.

1.5.16.1. OBTENCIÓN DE LA DISTANCIA A TRAVÉS DE LAS COORDENADAS DE LATITUD Y LONGITUD

El cálculo de la distancia entre dos puntos se hace a partir de la longitud y la latitud de cada punto. Estas coordenadas se pueden obtener a través de dispositivos GPS o bien, a través de mapas y herramientas en línea (Educacion Activa, 2024).

Para calcular la distancia, se puede utilizar la fórmula del Semiverseno, como te había sugerido en el comentario. Consiste en tomar dos puntos de una esfera y conociendo su radio, puedes calcular la distancia que hay entre cada extremo. Cada punto (P) está compuesto por una tupla (*latitud, longitud*) y ajustaremos $R = 6371$ (radio de la esfera/tierra)

Figura 1: fórmula para la calcular distancia

$$d = 2r \arcsin \left(\sqrt{\sin^2\left(\frac{\phi_2 - \phi_1}{2}\right) + \cos(\phi_1) \cos(\phi_2) \sin^2\left(\frac{\lambda_2 - \lambda_1}{2}\right)} \right)$$

Fuente: Educación activa

1.6. FUNDAMENTOS SOBRE PRUEBAS

1.6.1. PRUEBAS EN INFORMÁTICA

Las pruebas de software son el proceso de evaluar y verificar que un producto o aplicación de software hace lo que se supone que debe. Los beneficios de unas buenas pruebas incluyen la prevención de errores y la mejora del rendimiento (IBM, 2023).

Las pruebas de software (en inglés software testing) son las investigaciones empíricas y técnicas cuyo objetivo es proporcionar una devolución sobre el funcionamiento de un programa de software. Es una actividad más en el proceso de desarrollo de software, usualmente parte del control de calidad (wikipedia).

Por lo tanto, se define que las pruebas son un conjunto de procesos que permite verificar la calidad y el cumplimiento de la función de un producto o aplicación. En las pruebas de software existen varios tipos de pruebas, en este capítulo las cuales los más importantes son:

1.6.2. TIPOS DE PRUEBAS

1.6.2.1. CAJA NEGRA

La prueba de caja negra, test funcional o prueba comportamental es un tipo de prueba de software directa, cuya finalidad es analizar la compatibilidad entre las interfaces de cada uno de los componentes del software. No tiene en consideración la lógica interna del sistema. Permite la revisión final de las especificaciones y codificación de un programa. La prueba es considerada aceptable cuando su ejecución conlleva una probabilidad elevada de encontrar un error y es satisfactoria cuando lo detecta. (wikipedia, 2023)

Las pruebas de caja negra se centran en los requisitos funcionales y en las especificaciones del sistema. Están diseñadas para buscar errores, por medio de la interacción con la interfaz del sistema o software a probar, aquí se tienen en cuenta las entradas y salidas del software sin preocuparnos por tener un conocimiento previo de la estructura interna del software y de cómo el código fue construido. (SQA, 2022)

Tomando las dos definiciones se determina que las pruebas de caja negra son pruebas que se emplean sin importar los errores de las líneas de código, las pruebas de caja negra están

enfocados a las pruebas de la interface del software, la finalidad de estas pruebas es detectar los errores a nivel de diseño del software.

1.6.2.2. CAJA BLANCA

Las pruebas de blanca (también conocidas como pruebas de caja de cristal o pruebas estructurales) se centran en los detalles procedimentales del software, por lo que su diseño está fuertemente ligado al código fuente. El ingeniero de pruebas escoge distintos valores de entrada para examinar cada uno de los posibles flujos de ejecución del programa y cerciorarse de que se devuelven los valores de salida adecuados. (wikipedia, 2022)

La caja blanca es una categoría de las pruebas de software que se refiere a los métodos de comprobación del funcionamiento de la estructura interna y el diseño del software. Contrastá con las pruebas de caja negra, que no se ocupan de las operaciones internas del software, sino que sólo comprueban sus resultados externos. (Chernyak, 2023)

Entonces se concluye que las pruebas de caja blanca son fundamentales para evaluar la lógica interna y el diseño del software, asegurando una cobertura exhaustiva de los diferentes caminos de ejecución. Complementan las pruebas de caja negra al proporcionar una visión detallada de cómo funciona internamente el software, lo que contribuye a mejorar la calidad y la confiabilidad del producto final.

1.6.2.3. PRUEBA DE INTEGRIDAD

La integridad de los datos en una base de datos se preserva mediante una serie de procedimientos, reglas y principios de verificación y validación de errores que se ejecutan durante la fase de diseño del flujo de integración. Estos controles y procedimientos de corrección se basan en reglas comerciales predefinidas. Por ejemplo, las reglas dictan que se filtren los datos con un valor de fecha u hora incorrecto. (Astera, 2023)

Esta analogía es aplicable en los sistemas de software, en los que deben ejecutarse distintos tipos de pruebas para garantizar su correcto funcionamiento. Cada línea de código es equiparable a un ingrediente, que al mezclarse con otros fragmentos puede llegar a sufrir alteraciones en sus funciones originales. (testing, 2023)

Tomando las definiciones anteriores se concluye que las pruebas de integridad son las pruebas que se realizan para determinar la integridad de los datos, esto quiere decir que puede ver alteraciones de los datos de lo original. Las alteraciones de datos generalmente se emplean cuando se realiza un ataque de fishing a la base de datos.

1.6.2.4. PRUEBA DE FUNCIONAMIENTO

Las pruebas funcionales en las pruebas de software son una forma de determinar si el software o una aplicación funcionan como se espera. Las pruebas funcionales no se ocupan de cómo se produce el procesamiento, sino de si éste ofrece los resultados correctos o tiene algún fallo. (Carlos, 2022)

Una prueba funcional es una prueba de tipo caja negra basada en la ejecución, revisión y retroalimentación de las funcionalidades previamente diseñados para el software. Las pruebas funcionales se hacen mediante el diseño de modelos de prueba que buscan evaluar cada una de las opciones con las que cuenta el paquete informático. Dicho de otro modo, son pruebas específicas, concretas y exhaustivas para probar y validar que el software hace lo que debe y, sobre todo, lo que se ha especificado. (wikipedia, 2023)

En conclusiones las pruebas funcionales en el ámbito del software se enfocan en determinar si una aplicación o software opera según lo esperado, sin profundizar en los detalles internos del procesamiento. Estas pruebas se centran en los resultados y la funcionalidad, adoptando un enfoque de caja negra que evalúa la ejecución, revisión y retroalimentación de las funcionalidades previamente diseñadas para el software.

1.6.2.5. PRUEBA DE ACEPTACIÓN

Las pruebas de aceptación son las últimas pruebas realizadas donde el cliente prueba el software y verifica que cumpla con sus expectativas. Estas pruebas generalmente son funcionales y se basan en los requisitos definidos por el cliente y deben hacerse antes de la salida a producción. (linkedin, 2023)

Las pruebas de aceptación de usuario o User Acceptance Testing (UAT) resultan un paso crucial en el proceso de desarrollo. Es un punto sensible en el último paso antes del lanzamiento al mercado, gracias al cual este lanzamiento contará con las garantías de calidad y funcionalidad necesarias. (tester, 2023)

Tomando las definiciones se determina que las pruebas de aceptación de usuario (UAT), representan la fase final y crítica en el proceso de desarrollo de software. Estas pruebas implican que el cliente verifique directamente el software para asegurarse de que cumpla con sus expectativas y requisitos previamente definidos. Al ser pruebas funcionales centradas en los criterios del cliente, las pruebas de aceptación desempeñan un papel fundamental para garantizar la calidad y funcionalidad del producto antes de su lanzamiento al mercado.

CAPÍTULO II

PLANIFICACIÓN Y DISEÑO

CAPÍTULO II: PLANIFICACIÓN Y DISEÑO

2.1. INTRODUCCIÓN

La introducción de un análisis y diseño del sistema de control vehicular es fundamental para comprender la importancia y el alcance de este tipo de sistema en la gestión eficiente y segura del tráfico automotor. Este sistema se ha vuelto cada vez más relevante en un mundo donde la movilidad urbana y la seguridad vial son prioridades constantes.

El análisis y diseño del sistema de control vehicular implica la evaluación exhaustiva de diversos aspectos, como la infraestructura vial, la tecnología disponible, las necesidades de los usuarios y los objetivos de seguridad y eficiencia. Además, requiere una comprensión profunda de los procesos involucrados en la gestión del tráfico, desde la detección de vehículos hasta la implementación de medidas de control y regulación.

En última instancia, este análisis y diseño del sistema de control vehicular busca ofrecer una visión integral y estratégica para mejorar la movilidad urbana, reducir los congestionamientos, prevenir accidentes de tráfico y promover un entorno vial más seguro y sostenible para todos los usuarios de la vía.

2.2. DIAGNÓSTICO DEL CONTEXTO

El transporte público es un medio esencial en la movilidad urbana que cumple una función vital en la conectividad y accesibilidad de las personas en las ciudades. Se define como un sistema de transporte que está abierto al público en general y que proporciona servicios de desplazamiento en masa, ya sea a través de autobuses, trenes, tranvías, metro u otros medios similares, en el proyecto que se está desarrollando se enfoca en el transporte público de microbuses.

Al mismo tiempo, el transporte público contribuye a la integración social al ofrecer una opción de movilidad accesible para todos los estratos socioeconómicos de la población. Facilita el acceso a empleo, educación, servicios de salud, actividades recreativas y culturales, promoviendo la inclusión y la cohesión social.

Sin embargo, el transporte público también enfrenta desafíos importantes, como la necesidad de mejorar la calidad del servicio, la seguridad de los usuarios, la infraestructura y la sostenibilidad

ambiental. Estos desafíos requieren una planificación cuidadosa, inversión en infraestructura moderna y eficiente, así como políticas públicas que fomenten su desarrollo y mejora continua.

2.2.1. ANTECEDENTES

Como antecedentes se tiene a la fundación software libre, es la institución para el cual se está desarrollado el sistema de control y monitoreo vehicular, por tal motivo se enfoca en los antecedentes de la institución mencionado.

La institución fundación software libre fue creado el 26 de marzo del año 2023, es una de las empresas más joven que opera dentro la Carrera de Ingeniería de Sistemas, su función el desarrollo de software libre gratuito.

2.3. ANÁLISIS DE INSTRUMENTOS

El análisis de instrumentos es de gran ayuda para la toma de decisiones, la cual permite evaluar los aspectos que se deben tomar en cuenta para la implementación del sistema de control y monitoreo vehicular.

2.3.1. ENTREVISTA

Es una técnica de adquisición de datos en forma directa establecida entre un investigador y su objeto de estudio a través de individuos o grupos con el fin de obtener testimonios orales.

Se empleo esta técnica para recabar la información del manejo del sistema vehicular, esta técnica se realizó a los conductores como encargado y jefes de sindicatos, que permitió conocer de manera minuciosa y detalla el proceso del manejo de los microbuses, en cual se utilizó un cuestionario de guía (**Ver anexo 6, 7**)

2.3.2. OBSERVACIÓN

Esta técnica fue empleada para observar atentamente el fenómeno del comportamiento del movimiento vehicular de los microbuses, así tomar la información para su uso, y posterior análisis, esto ayudo a determinar los requerimientos del sistema de control y monitoreo vehicular.

2.3.3. ENCUESTA

Esta técnica fue de gran ayuda para recolectar información valiosa sobre las opiniones y percepciones de la población respecto al control y monitoreo vehicular de los microbuses. Mediante la aplicación de encuestas, pudimos obtener una visión detallada de las actitudes y expectativas de los usuarios sobre la implementación de estos sistemas. La encuesta permitió identificar tanto el nivel de conocimiento existente sobre los sistemas de control y monitoreo como el grado de interés y aceptación de su posible implementación (Ver anexo 8, 9).

2.4. PLANIFICACIÓN

La planificación detallada del desarrollo del sistema de control y monitoreo de microbuses se llevará a cabo utilizando la metodología Scrum. En donde se define los roles y responsabilidades de los equipos, así como los requisitos específicos del producto en base a la retroalimentación de los usuarios y las necesidades identificadas. La estructura de sprints. Reuniones regulares y la flexibilidad para ajustar las prioridades en función de la evolución del proyecto que son elementos claves para la planificación.

2.4.1. PRE-JUEGO

2.4.1.1. ROLES

El equipo Scrum se está formando por los siguientes roles

SCRUM MASTER(SM)
MSC. ANNY MERCADO ALGARAÑAZ
PRODUCT OWNER(PO)
ING. DAVID MAMANI FIGUEROA
DEVELOPER(DP)
UNIV. SANTOS MACHACA LOPEZ

2.4.1. HISTORIA DE USUARIOS

Tabla 1. Historia de usuario 1

Historia de usuario		
Identificador (ID) de la historia	HUC-0001	
Título de la historia	Control de acceso al sistema	
Enunciado de la historia		
Rol	Característica/Funcionalidad	Razón/Resultado
Usuario del sistema.	Se requiere que el sistema tenga un control de acceso al sistema por un login.	Control del sistema.
Criterios de aceptación		
<ul style="list-style-type: none"> • Validación de usuario y contraseña • Validación de tamaños de contraseñas 		

Fuente: Elaboración propia

Tabla 2. Historia de usuario 2

Historia de usuario		
Identificador (ID) de la historia	HUC-0002	
Título de la historia	Roles y permisos	
Enunciado de la historia		
Rol	Característica/Funcionalidad	Razón/Resultado
Usuario del sistema	Se necesita asignar los roles y permisos a los usuarios.	Permisos del acceso del sistema.
Criterios de aceptación		
<ul style="list-style-type: none"> • Contar con el subsistema de usuarios • Validación de campos vacíos y textos. 		

Fuente: Elaboración propia

Tabla 3. Historia de usuario 3

Historia de usuario		
Identificador (ID) de la historia	HUC-0003	
Título de la historia	Aplicación gps	
Enunciado de la historia		
Rol	Característica/Funcionalidad	Razón/Resultado
Usuario del sistema	Se necesita una aplicación que envíe la ubicación del dispositivo.	Monitoreo de los dispositivos.
Criterios de aceptación		

- Contar con el sistema de autenticación
- Validación de campos vacíos
- Contar con la implementación de servidor para la conexión del dispositivo

Fuente: Elaboración propia

Tabla 4. Historia de usuario 4

Historia de usuario		
Identificador (ID) de la historia		HUC-0004
Título de la historia		Registrar usuarios
Enunciado de la historia		
Rol	Característica/Funcionalidad	Razón/Resultado
Usuario del sistema	Se requiere un sistema para registrar los usuarios.	Control de usuarios.
Criterios de aceptación		
<ul style="list-style-type: none"> • validación de campos vacíos, texto y numéricos • tener una lista de los países 		

Fuente: Elaboración propia

Tabla 5. Historia de usuario 5

Historia de usuario		
Identificador (ID) de la historia		HUC-0005
Título de la historia		Registrar microbuses
Enunciado de la historia		
Rol	Característica/Funcionalidad	Razón/Resultado
Usuario del sistema	Se requiere un sistema para registrar los microbuses.	Control de micros
Criterios de aceptación		
<ul style="list-style-type: none"> • validación de campos vacíos y texto • validación de campos numéricos • contar con una lista de marcas de microbuses 		

Fuente: Elaboración propia

Tabla 6. Historia de usuario 6

Historia de usuario		
Identificador (ID) de la historia		HUC-0006
Título de la historia		Registrar líneas
Enunciado de la historia		
Rol	Característica/Funcionalidad	Razón/Resultado
Usuario del sistema	Se requiere un sistema para registrar las líneas.	Control de las líneas.

Criterios de aceptación		
<ul style="list-style-type: none"> • Contar con el subsistema de registro de microbuses • Contar con el subsistema de registro de horarios • Contar con el subsistema de registro de rutas y paradas • Contar con el subsistema de registro de tarifas 		

Fuente: Elaboración propia

Tabla 7. Historia de usuario 7

Historia de usuario		
Identificador (ID) de la historia		HUC-0007
Título de la historia		Registrar rutas y paradas
Enunciado de la historia		
Rol	Característica/Funcionalidad	Razón/Resultado
Usuario del sistema	Se requiere registrar las rutas y paradas de los microbuses mediante los mapas geográficos.	Control de rutas y paradas
Criterios de aceptación		
<ul style="list-style-type: none"> • validación campos vacíos y numéricos • contar con Apis Geo mapas gratuitos 		

Fuente: Elaboración propia

Tabla 8. Historia de usuario 8

Historia de usuario		
Identificador (ID) de la historia		HUC-0008
Título de la historia		Registrar tarifas de los microbuses
Enunciado de la historia		
Rol	Característica/Funcionalidad	Razón/Resultado
Usuario del sistema	Se necesita un sistema para registrar las tarifas de los microbuses.	Control de tarifas
Criterios de aceptación		
<ul style="list-style-type: none"> • validación de campos vacíos y numéricos 		

Fuente: Elaboración propia

Tabla 9. Historia de usuario 9

Historia de usuario		
Identificador (ID) de la historia	HUC-0009	
Título de la historia	Registrar horarios de salidas	
Enunciado de la historia		
Rol	Característica/Funcionalidad	Razón/Resultado
Usuario del sistema	Se requiere un sistema para registrar los horarios de salida de los microbuses.	Control de horarios de salida
Criterios de aceptación		
<ul style="list-style-type: none"> • Validación de campos vacíos • Validación de tipos de datos y numéricos 		

Fuente: Elaboración propia

Tabla 10. Historia de usuario 10

Historia de usuario		
Identificador (ID) de la historia	HUC-0010	
Título de la historia	Registrar licencia de conducir	
Enunciado de la historia		
Rol	Característica/Funcionalidad	Razón/Resultado
Usuario del sistema	Se requiere un sistema para registrar las licencias de los conductores de los microbuses.	Registro de las licencias
Criterios de aceptación		
<ul style="list-style-type: none"> • Contar con el subsistema de registro de usuarios • Validación de campos vacíos y numéricos • Validación de las imágenes 		

Fuente: Elaboración propia

Tabla 11. Historia de usuario 11

Historia de usuario		
Identificador (ID) de la historia	HUC-0011	
Título de la historia	Monitoreo	
Enunciado de la historia		
Rol	Característica/Funcionalidad	Razón/Resultado
Usuario del sistema	Se requiere un sistema para monitorear el desplazamiento de los microbuses en tiempo real.	Monitoreo de los microbuses
Criterios de aceptación		
<ul style="list-style-type: none"> • Contar con la aplicación GPS 		

- Contar con el subsistema de registro de rutas y paradas
- Contar con el subsistema de registro de usuarios
- Contar con el subsistema de registro de microbuses
- Contar con el subsistema de registro de líneas
- Contar con el subsistema de registro de horarios

Fuente: Elaboración propia

Tabla 12. Historia de usuario 12

Historia de usuario		
Identificador (ID) de la historia	HUC-0012	
Título de la historia	Ver rutas y paradas	
Enunciado de la historia		
Rol	Característica/Funcionalidad	Razón/Resultado
Usuario del sistema	Se requiere un sistema para ver las rutas y paradas de las líneas.	Mostrar rutas y paradas
Criterios de aceptación		
<ul style="list-style-type: none"> • Contar con el sistema de registro de rutas y paradas. 		

Fuente: Elaboración propia

Tabla 13. Historia de usuario 13

Historia de usuario		
Identificador (ID) de la historia	HUC-0013	
Título de la historia	Ver tarifas	
Enunciado de la historia		
Rol	Característica/Funcionalidad	Razón/Resultado
Usuario del sistema	Se requiere un sistema para ver las tarifas de las líneas.	Mostrar tarifas
Criterios de aceptación		
<ul style="list-style-type: none"> • Contar con el sistema de registro de tarifas 		

Fuente: Elaboración propia

Tabla 14. Historia de usuario 14

Historia de usuario		
Identificador (ID) de la historia	HUC-0014	
Título de la historia	Estados de conexión	
Enunciado de la historia		
Rol	Característica/Funcionalidad	Razón/Resultado
Usuario del sistema	Se necesita un sistema para ver los estados de conexión de los choferes.	Monitoreo de los choferes

Criterios de aceptación
• Contar con la aplicación de GPS

Fuente: Elaboración propia

Tabla 15. Historia de usuario 15

Historia de usuario		
Identificador (ID) de la historia		HUC-0015
Título de la historia		Ver velocidad de los microbuses
Enunciado de la historia		
Rol	Característica/Funcionalidad	Razón/Resultado
Usuario del sistema	Se necesita un sistema para ver las velocidades de los microbuses.	Monitoreo de los microbuses
Criterios de aceptación		
<ul style="list-style-type: none"> • Contar con registro de rutas • Contar con registro de paradas • Contar con la aplicación de GPS 		

Fuente: Elaboración propia

Tabla 16. Historia de usuario 16

Historia de usuario		
Identificador (ID) de la historia		HUC-0016
Título de la historia		Ver reporte de las líneas solicitadas
Enunciado de la historia		
Rol	Característica/Funcionalidad	Razón/Resultado
Usuario del sistema	Se necesita ver los reportes de solicitudes de las líneas.	Reporte de las solicitudes de las líneas
Criterios de aceptación		
<ul style="list-style-type: none"> • Contar con el subsistema de registro de líneas • Contar con el subsistema monitoreo 		

Fuente: Elaboración propia

2.4.2. PLANIFICACIÓN DE LA ITERACIÓN (PRODUCT BACKLOG)

Tabla 17: Producto Backlog

Nro.	ID de la historia	Título de la historia	Estado	Estimación (Semanas)	Sprint	% de finalización
1	HUC-0001	Control de acceso al sistema	Terminado	1	1	100%

2	HUC-0002	Roles y permisos	Terminado	2	1	100%
3	HUC-0003	Aplicación GPS	Terminado	2	2	100%
4	HUC-0004	Registrar usuarios	Terminado	1	3	100%
5	HUC-0005	Registrar microbuses	Terminado	1	3	100%
6	HUC-0006	Registrar líneas	Terminado	1	3	100%
7	HUC-0007	Registrar rutas y paradas	Terminado	3	3	100%
8	HUC-0008	Registrar tarifas de los microbuses	Terminado	1	3	100%
9	HUC-0009	Registrar horarios de salida	Terminado	1	3	100%
10	HUC-0010	Registrar licencia de conducir	Terminado	1	3	100%
11	HUC-0011	Monitoreo	Terminado	4	4	100%
12	HUC-0012	Ver rutas y paradas	Terminado	2	4	100%
13	HUC-0013	Ver Tarifas	Terminado	1	4	100%
14	HUC-0014	Ver estados de conexión	Terminado	1	4	100%
15	HUC-0015	Ver velocidad de microbuses	Terminado	1	4	100%
16	HUC-0016	Ver reporte de las solicitudes de las líneas	Terminado	2	4	100%

Fuente: Elaboración propia

2.4.3. PLANIFICACIÓN DE SPRINT

2.4.3.1. Planificación de Sprint 1

Tabla 18. Sprint 1 backlog – Control de acceso al sistema

Requisito	backlog	Tarea	Responsable	Tipo	Estado
Ninguno	Control de acceso al sistema	Diseño y planificación	Santos Machaca L.	Análisis y diseño	Completado
		Codificación de la base de datos	Santos Machaca L.	Codificación	Completado

		Diseño de la interfaz	Santos Machaca L.	Diseño y Codificación	Completado
		Conexión con la base de datos	Santos Machaca L.	Codificación	Completado
		Validación de datos	Santos Machaca L.	Codificación	Completado
		Validación de usuario y contraseña	Santos Machaca L.	Codificación	Completado
		Ejecución de pruebas	Santos Machaca L.	pruebas	Completado

Fuente: Elaboración propia

Tabla 19. Sprint 1 backlog – Roles y permisos

Requisito	backlog	Tarea	Responsable	Tipo	Estado
Control de acceso al sistema - Subsistema registro de usuarios	Roles y permisos	Diseño y planificación	Santos Machaca L.	análisis y diseño	Completado
		Codificación de la base de datos	Santos Machaca L.	Codificación	Completado
		Diseño de la interfaz	Santos Machaca L.	Diseño y Codificación	Completado
		Conexión con la base de datos	Santos Machaca L.	Codificación	Completado
		Validación de datos	Santos Machaca L.	Codificación	Completado
		Visualización de los usuarios	Santos Machaca L.	Diseño y Codificación	Completado
		Asignación de los roles	Santos Machaca L.	Diseño y Codificación	Completado
		Ejecución de pruebas	Santos Machaca L.	pruebas	Completado

Fuente: Elaboración propia

2.4.3.2. Planificación de Sprint 2

Tabla 20. Sprint 2 backlog – Aplicación GPS

Requisito	backlog	Tarea	Responsable	Tipo	Estado
		Diseño y planificación	Santos Machaca L.	análisis y diseño	Completado

Control de acceso al sistema	Aplicación GPS	Codificación de la base de datos	Santos Machaca L.	codificación	Completado
		Diseño de la interfaz	Santos Machaca L.	Diseño y codificación	Completado
		Conexión con la base de datos	Santos Machaca L.	codificación	Completado
		Validación de datos	Santos Machaca L.	codificación	Completado
		Extracción de la ubicación de dispositivo	Santos Machaca L.	codificación	Completado
		Conexión de socket	Santos Machaca L.	codificación	Completado
		Cálculo de la distancia de las latitudes y longitudes obtenidas	Santos Machaca L.	codificación	Completado
		Cálculo de la velocidad	Santos Machaca L.	codificación	Completado
		verificación de los estados de conexión de dispositivo	Santos Machaca L.	codificación	Completado
		Validación de usuario y contraseña	Santos Machaca L.	Codificación	Completado
		Ejecución de la aplicación en segundo plano	Santos Machaca L.	Codificación	Completado
		Ejecución de pruebas	Santos Machaca L.	pruebas	Completado

Fuente: Elaboración propia

2.4.3.3. Planificación de Sprint 3

Tabla 21. Sprint 3 backlog – Registrar usuarios

Requisito	backlog	Tarea	Responsable	Tipo	Estado
Control de acceso al sistema	Registrar usuarios	Diseño y planificación	Santos Machaca L.	Análisis y diseño	Completado
		Codificación de la base de datos	Santos Machaca L.	Codificación	Completado

		Diseño de la interfaz de usuario	Santos Machaca L.	Diseño y codificación	Completado
		Conexión con la base de datos	Santos Machaca L.	codificación	Completado
		Validación de datos	Santos Machaca L.	codificación	Completado
		Lista de los usuarios	Santos Machaca L.	codificación	Completado
		Editar datos de los usuarios	Santos Machaca L.	Codificación	Completado
		Validar tipo de archivo	Santos Machaca L.	Codificación	Completado
		Ejecución de pruebas	Santos Machaca L.	Pruebas	Completado

Fuente: Elaboración propia

Tabla 22. Sprint 3 backlog – Registrar microbuses

Requisito	backlog	Tarea	Responsable	Tipo	Estado
Control de acceso al sistema	Registrar microbuses	Diseño y planificación	Santos Machaca L.	análisis y diseño	Completado
		codificación de la base de datos	Santos Machaca L.	codificación	Completado
		Diseño de la interfaz de usuario	Santos Machaca L.	Diseño y codificación	Completado
		Conexión con la base de datos	Santos Machaca L.	codificación	Completado
		validación de datos	Santos Machaca L.	codificación	Completado
		Lista de los microbuses	Santos Machaca L.	Diseño y codificación	Completado
		Editar datos de microbuses	Santos Machaca L.	Diseño y codificación	Completado
		validar tipo de archivo	Santos Machaca L.	Codificación	Completado
		Asignación de choferes	Santos Machaca L.	Diseño y codificación	Completado
		Ejecución de pruebas	Santos Machaca L.		Completado

Fuente: Elaboración propia

Tabla 23. Sprint 3 backlog – Registrar líneas

Requisito	Backlog	Tarea	Responsable	Tipo	Estado
Control de acceso al sistema - Registro de rutas y paradas - Registro de microbuses - Registro de líneas - Registro de tarifas	Registrar líneas	Diseño y planificación	Santos Machaca L.	análisis y diseño	Completado
		codificación de la base de datos	Santos Machaca L.	codificación	Completado
		Diseño de la interfaz del usuario	Santos Machaca L.	Diseño y codificación	Completado
		Conexión con la base de datos	Santos Machaca L.	codificación	Completado
		validación de datos	Santos Machaca L.	codificación	Completado
		Lista de las líneas	Santos Machaca L.	Diseño y codificación	Completado
		Editar registro de la línea	Santos Machaca L.	Diseño y codificación	Completado
		asignar microbuses	Santos Machaca L.	Diseño y codificación	Completado
		asignar rutas y paradas	Santos Machaca L.	Diseño y codificación	Completado
		Asignar horarios	Santos Machaca L.	Diseño y codificación	Completado
		Asignar tarifas	Santos Machaca L.	Diseño y codificación	Completado
		Ejecución de pruebas	Santos Machaca L.	pruebas	Completado

Fuente: Elaboración propia

Tabla 24. Sprint 3 backlog – Registrar rutas y paradas

Requisito	backlog	Tarea	Responsable	Tipo	Estado
Control de acceso al sistema	Registrar rutas y paradas	Diseño y planificación	Santos Machaca L.	análisis y diseño	Completado
		codificación de la base de datos	Santos Machaca L.	codificación	Completado
		Diseño de la interfaz de usuario	Santos Machaca L.	Diseño y codificación	Completado
		Conectar con los proveedores de mapas	Santos Machaca L.	codificación	Completado
		Conexión con la base de datos	Santos Machaca L.	codificación	Completado

		validación de datos	Santos Machaca L.	codificación	Completado
		Lista de los rutas y paradas	Santos Machaca L.	codificación	Completado
		Editar rutas y paradas	Santos Machaca L.	Codificación	Completado
		Diseñar un lápiz para dibujar las rutas	Santos Machaca L.	Codificación	Completado
		Generar un geojson para guardar los datos en la base de datos	Santos Machaca L.	Codificación	Completado
		Ejecución de pruebas	Santos Machaca L.	pruebas	Completado

Fuente: Elaboración propia

Tabla 25. Sprint 3 backlog – Registrar tarifas de los microbuses

Requisito	backlog	Tarea	Responsable	Tipo	Estado
Control de acceso al sistema	Registrar tarifas de los microbuses	Diseño y planificación	Santos Machaca L.	análisis y diseño	Completado
		codificación de la base de datos	Santos Machaca L.	codificación	Completado
		Diseño de la interfaz de usuario	Santos Machaca L.	Diseño y codificación	Completado
		Conexión con la base de datos	Santos Machaca L.	codificación	Completado
		validación de datos	Santos Machaca L.	codificación	Completado
		Lista de las tarifas	Santos Machaca L.	Diseño y codificación	Completado
		Editar datos de tarifas	Santos Machaca L.	Diseño y codificación	Completado
		Ejecución de pruebas	Santos Machaca L.	pruebas	Completado

Fuente: Elaboración propia

Tabla 26. Sprint 3 backlog – Registrar horarios de salida

Requisito	backlog	Tarea	Responsable	Tipo	Estado
		Diseño y planificación	Santos Machaca L.	análisis y diseño	Completado

Control de acceso al sistema	Registrar horarios de salida	codificación de la base de datos	Santos Machaca L.	codificación	Completado
		Diseño de la interfaz de usuario	Santos Machaca L.	Diseño y codificación	Completado
		Conexión con la base de datos	Santos Machaca L.	codificación	Completado
		validación de datos	Santos Machaca L.	codificación	Completado
		Lista de los horarios	Santos Machaca L.	Diseño y codificación	Completado
		Editar datos de los horarios	Santos Machaca L.	Diseño y codificación	Completado
		validar tipo de datos	Santos Machaca L.	Codificación	Completado
		Ejecución de pruebas	Santos Machaca L.	pruebas	Completado

Fuente: Elaboración propia

Tabla 27. Sprint 3 backlog – Registrar licencia de conducir

Requisito	backlog	Tarea	Responsable	Tipo	Estado
Control de acceso al sistema	Registrar licencia de conducir	Diseño y planificación	Santos Machaca L.	Análisis y diseño	Completado
		Codificación de la base de datos	Santos Machaca L.	Codificación	Completado
		Diseño de la interfaz de usuario	Santos Machaca L.	Diseño y codificación	Completado
		Conexión con la base de datos	Santos Machaca L.	codificación	Completado
		Validación de datos	Santos Machaca L.	codificación	Completado
		Lista de usuarios y las licencias de conducir	Santos Machaca L.	Diseño y codificación	Completado
		Editar datos de los usuarios y las licencias	Santos Machaca L.	Diseño y codificación	Completado
		Validar tipo de archivo	Santos Machaca L.	Codificación	Completado
		Ejecución de pruebas	Santos Machaca L.	pruebas	Completado

Fuente: Elaboración propia

2.4.3.4. Planificación de Sprint 4

Tabla 28. Sprint 4 backlog – Monitoreo

Requisito	backlog	Tarea	Responsable	Tipo	Estado
Control de acceso al sistema	Monitoreo	Diseño y planificación	Santos Machaca L.	análisis y diseño	Completado
- registro de usuarios		codificación de la base de datos	Santos Machaca L.	codificación	Completado
- registro de líneas		Diseño de la interfaz del usuario	Santos Machaca L.	Diseño y codificación	Completado
- Registro de paradas		Conexión de socket	Santos Machaca L.	codificación	Completado
- Registro de rutas		Conexión con la base de datos	Santos Machaca L.	codificación	Completado
- Registro de horarios		Lista de las líneas	Santos Machaca L.	Diseño y codificación	Completado
		Lista de los estados de conexión	Santos Machaca L.	Diseño y codificación	Completado
		Reporte de las líneas solicitadas	Santos Machaca L.	Diseño y codificación	Completado
		Lista de rutas y paradas	Santos Machaca L.	Diseño y codificación	Completado
		Lista de Tarifas	Santos Machaca L.	Diseño y codificación	Completado
		Lista de Horarios	Santos Machaca L.	Diseño y codificación	Completado
		Mostrar los desplazamientos de los microbuses en tiempo real	Santos Machaca L.	Diseño y codificación	
		Mostrar velocidades del microbús	Santos Machaca L.	Diseño y codificación	Completado
		Ejecución de pruebas y pruebas	Santos Machaca L.	pruebas	Completado

Fuente: Elaboración propia

Tabla 29. Sprint 4 backlog – Ver rutas y paradas

Requisito	backlog	Tarea	Responsable	Tipo	Estado
Control de acceso al sistema - registro de paradas y rutas	Ver rutas y paradas	Diseño y planificación	Santos Machaca L.	análisis y diseño	Completado
		codificación de la base de datos	Santos Machaca L.	codificación	Completado
		Diseño de la interfaz de usuario	Santos Machaca L.	Diseño y codificación	Completado
		Conexión con la base de datos	Santos Machaca L.	codificación	Completado
		Listar rutas y paradas	Santos Machaca L.	Diseño y codificación	Completado
		Ejecución de pruebas y pruebas	Santos Machaca L.	pruebas	Completado

Fuente: Elaboración propia

Tabla 30. Sprint 4 backlog – Ver Tarifas

Requisito	backlog	Tarea	Responsable	Tipo	Estado
Control de acceso al sistema - registro de tarifas	Ver Tarifas	Diseño y planificación	Santos Machaca L.	análisis y diseño	Completado
		codificación de la base de datos	Santos Machaca L.	codificación	Completado
		Diseño de la interfaz de usuario	Santos Machaca L.	Diseño y codificación	Completado
		Conexión con la base de datos	Santos Machaca L.	codificación	Completado
		Listar tarifas	Santos Machaca L.	Diseño y codificación	Completado
		Ejecución de pruebas y pruebas	Santos Machaca L.	pruebas	Completado

Fuente: Elaboración propia

Tabla 31. Sprint 4 backlog – Ver estados de conexión

Requisito	backlog	Tarea	Responsable	Tipo	Estado
	Ver estados de conexión	Diseño y planificación	Santos Machaca L.	análisis y diseño	Completado

Control de acceso al sistema - registro usuarios		codificación de la base de datos	Santos Machaca L.	codificación	Completado
		Diseño de la interfaz de usuario	Santos Machaca L.	Diseño y codificación	Completado
		Conexión con la base de datos	Santos Machaca L.	codificación	Completado
		Conexión de socket	Santos Machaca L.	codificación	Completado
		Listar usuarios	Santos Machaca L.	Diseño y codificación	Completado
		Ejecución de pruebas	Santos Machaca L.	pruebas	Completado

Fuente: Elaboración propia

Tabla 32. Sprint 4 backlog – Ver velocidad de microbuses

Requisito	backlog	Tarea	Responsable	Tipo	Estado
Control de acceso al sistema - registro usuarios - calculo de velocidades	Ver velocidad de microbuses	Diseño y planificación	Santos Machaca L.	análisis y diseño	Completado
		codificación de la base de datos	Santos Machaca L.	codificación	Completado
		Diseño de la interfaz de usuario	Santos Machaca L.	Diseño y codificación	Completado
		Conexión con la base de datos	Santos Machaca L.	codificación	Completado
		Conexión de socket	Santos Machaca L.	codificación	Completado
		Listar líneas	Santos Machaca L.	Diseño y codificación	Completado
		Listar usuarios	Santos Machaca L.	Diseño y codificación	Completado
		Listar velocidades	Santos Machaca L.	Diseño y codificación	Completado
		Ejecución de pruebas	Santos Machaca L.	pruebas	Completado

Fuente: Elaboración propia

Tabla 33. Sprint 4 backlog – Ver reporte de las solicitudes de las líneas

Requisito	backlog	Tarea	Responsable	Tipo	Estado
		Diseño y planificación	Santos Machaca L.	análisis y diseño	Completado

Control de acceso al sistema	Ver reporte de las solicitudes de las líneas	codificación de la base de datos	Santos Machaca L.	codificación	Completado
		Diseño de la interfaz de usuario	Santos Machaca L.	Diseño y codificación	Completado
		Conexión con la base de datos	Santos Machaca L.	codificación	Completado
		Listar reporte de solicitudes de líneas	Santos Machaca L.	Diseño y codificación	Completado
		Grafiar las solicitudes de las líneas	Santos Machaca L.	Diseño y codificación	Completado
		Ejecución de pruebas	Santos Machaca L.	pruebas	Completado

Fuente: Elaboración propia

2.5. ESTUDIO DE FACTIBILIDAD

2.5.1. Factibilidad operacional

La Factibilidad operacional hace referencia a todos aquellos recursos donde interviene algún tipo de actividad que vaya a participar durante la creación del sistema.

Durante esta etapa se identifican todas aquellas actividades que son necesarios para lograr el objetivo y se evalúa y determina todo lo necesario para llevar a cabo el sistema. En este sentido la factibilidad operacional es viable para el presente proyecto.

Para la justificación de la factibilidad operacional se ha corroborado mediante encuestas a la población y entrevistas a los choferes (**Ver Anexo 6,8,9**), en el cual la mayor parte necesitan un sistema de control y monitoreo vehicular de los microbuses, donde indican que sería una herramienta tecnológica para el patrimonio cultural y además esto ayudará a los ciudadanos dar la seguridad.

Por otra parte, se realizó entrevistas al personal de los sindicatos donde indican que este sistema será de gran apoyo y útil para la administración de los microbuses, ya que mediante este sistema se podrá ver en tiempo real de las ubicaciones de los microbuses del sindicato (**Ver Anexo 7**).

2.5.2. Factibilidad técnica

La factibilidad técnica es la evaluación de los recursos técnicos que influyen al desarrollo del sistema, de tal forma contemplan todas las restricciones del hardware, y el software necesario para la implementación de presente proyecto.

En este proyecto se ha establecido un conjunto de herramientas que coadyuva el desarrollo del software para el control y monitoreo vehicular de los microbuses de las distintas ciudades, siendo así útil para determinar los recursos técnicos de la empresa desde un punto de vista la culminación del software en el tiempo establecido.

2.5.2.1. Recursos de hardware

En este punto se identifican los recursos de hardware que será necesario para el desarrollo del dicho sistema para el control y monitoreo vehicular.

Tabla 34: recursos de hardware

	REQUERIMIENTO MÍNIMO	REQUERIMIENTO ÓPTIMO
CARACTERÍSTICAS	Computadora integrada	Computadora integrada
Arquitectura	64 bits	
Procesador	Core i5 o Ryzen 5	
Disco duro	500GB	
Memoria Ram	8GB	

Fuente: Elaboración propia

2.5.2.1. RECURSOS DE SOFTWARE

En este punto se identifican los recursos de software que será necesario para el desarrollo del sistema.

Tabla 35: recursos de software

Software	Requerimientos Mínimos	Softwares utilizados
Software de editor código	Ninguna	Visual code Studio
Lenguaje de programación	Ninguna	TypeScript, JavaScript Es6
Framework de desarrollo web	Ninguna	Nextjs (frontend) Nestjs (backend)
S. O.	Ninguna	Windows, Linux
Software proveedores de mapas	openStreetMap	openStreetMap
Software geolocalización	openStreetMap	openStreetMap
Software geográficas	openStreetMap	openStreetMap

Fuente: Elaboración propia

2.5.3. Factibilidad económica

El software desarrollado no tiene costos debido que se utilizan herramientas y software de terceros gratuitas, sin embargo, se debe considerar los costos que serán de inversión de la empresa para puesto en marcha del sistema concluida, que sería el esfuerzo, los costos de la adquisición de equipos y los costos de proveedores de red.

Para tal efecto se demuestra la información de los diferentes tipos de costos para la implantación y puesto en marcha del sistema, que permite la determinación de los requisitos de factibilidad económica tanto del hardware como software y entre otros.

Tabla 36: tabla de costos

Tabla resumen	
Costo SIA:	107047 Bs.
Costo Software:	100 Bs.
Costo Hardware:	23127 Bs.
Otros gastos:	4351 Bs.
TOTAL	134625 Bs.

Fuente: Elaboración propia

Después de realizar los cálculos del software que se está desarrollando se puede determinar el costo de aproximación de **134625 Bs. (CIENTO TREINTA Y CUATRO MIL SEISCIENTOS VEINTICINCO**

BOLIVIANOS), dicho costo esta expresado en dólares. El costo más elevado son los esfuerzos y el tiempo que se va a tardar en desarrollar el software, en este caso el presente proyecto está determinado la culminación del proyecto en un plazo de 9 meses, donde el proyecto ya estaría en funcionamiento.

2.6. Diseño

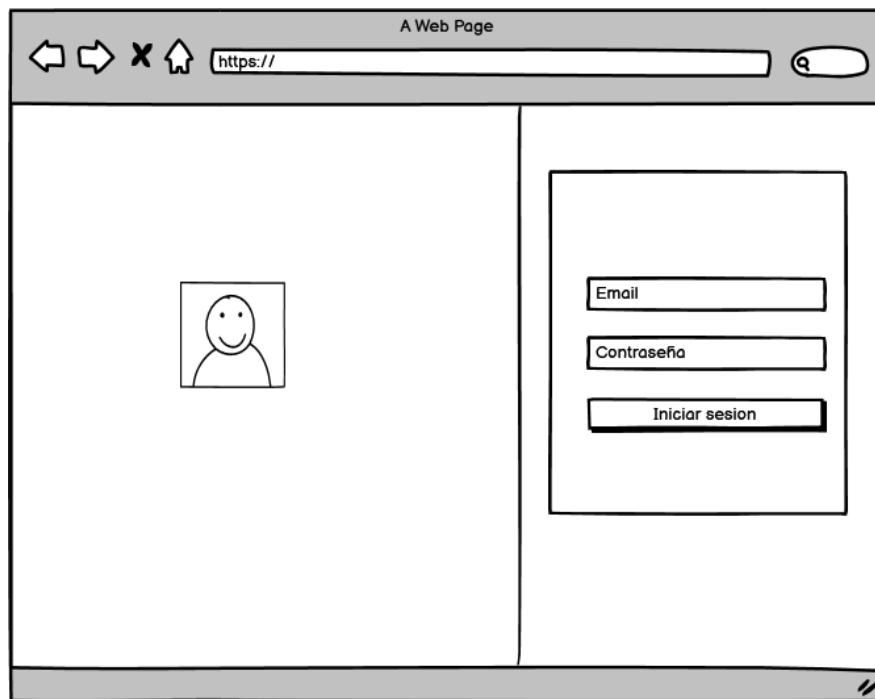
2.6.1. Sprint 1

Tabla 37: especificación de sprints

Sprint 1				
Fecha de Inicio			Tareas Pendientes	0
Fecha de culminación			días pendientes	0
Prioridad	descripción de la tarea	Responsable	duración/semanas	Estado
Alta	Control de acceso al sistema	Santos Machaca L.	1	Terminado
Alta	Roles y permisos	Santos Machaca L.	2	Terminado

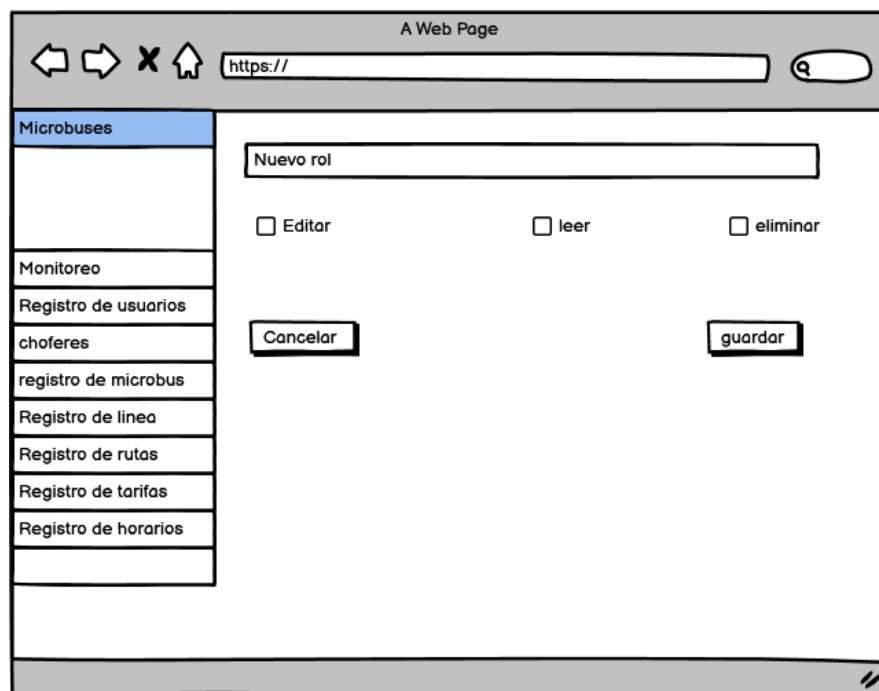
Fuente: Elaboración propia

Figura 2: Arquitectura de interfaz de usuario autenticación



Fuente: Elaboración propia

Figura 3: Arquitectura de interfaz de usuario Roles y permisos



Fuente: Elaboración propia

2.6.2. Sprint 2

Tabla 38: especificación de sprints

Sprint 2				
Fecha de Inicio			Tareas Pendientes	0
Fecha de culminación			días pendientes	0
Prioridad	descripción de la tarea	Responsable	duración/semanas	Estado
Alta	Aplicación GPS	Santos Machaca L.	2	Terminado

Fuente: Elaboración propia

Figura 4: Arquitectura de interfaz de usuario Aplicación GPS



Fuente: Elaboración propia

2.6.3. Sprint 3

Tabla 39: especificación de sprints

Sprint 3				
Fecha de Inicio		Tareas Pendientes		0
Fecha de culminación		días pendientes		0
Prioridad	descripción de la tarea	Responsable	duración/semanas	Estado
Alta	Registrar usuarios	Santos Machaca L.	1	Terminado
Media	Registrar microbuses	Santos Machaca L.	1	Terminado
Media	Registrar líneas	Santos Machaca L.	1	Terminado

Media	Registrar rutas y paradas	Santos Machaca L.	3	Terminado
Media	Registrar tarifas de los microbuses	Santos Machaca L.	1	Terminado
Media	Registrar horarios de salida	Santos Machaca L.	1	Terminado
Alta	Registrar licencia de conducir	Santos Machaca L.	1	Terminado

Fuente: Elaboración propia

Figura 5: Arquitectura de interfaz de usuario registrar usuarios

Fuente: Elaboración propia

Figura 6: Arquitectura de interfaz de usuario lista de usuarios

A Web Page

https://

Apellidos y nombre	Ci	dirección	celular	genero	nacionalidad	rol	acciones
Giacomo Guilizzo Founder & CEO	40	Peldi					
Marco Botton Tuttofare	38						
Mariah MacLachl Better Half	41	Patata	723748				
Valerie Liberty Head Chef	75	Val					

Fuente: Elaboración propia

Figura 7: Arquitectura de interfaz de usuario lista de usuarios

A Web Page

https://

Nombres y Apellido	ci	fecha de emi	fecha de expira	catego	opcion
Miguel Alejandr	105317	20/03/2024	20/03/2024	A	:
Miguel Alejandr	105317	20/03/2024	20/03/2024	A	:
Miguel Alejandr	105317	20/03/2024	20/03/2024	A	:
Miguel Alejandr	105317	20/03/2024	20/03/2024	A	:

Fuente: Elaboración propia

Figura 8: Arquitectura de interfaz de usuario registro de microbuses

The wireframe shows a web page titled 'A Web Page' with a URL bar containing 'https://'. On the left is a vertical sidebar menu with items: Microbuses (highlighted in blue), Monitoreo, Registro de usuarios, choferes, registro de microbus (highlighted in blue), Registro de linea, Registro de rutas, Registro de tarifas, and Registro de horarios.

The main content area has a search bar with a magnifying glass icon and a 'buscar' button. To the right is a table with columns: marca, modelo, and tipo. The table contains four rows of data: nissan, 2006, bus; nissan, 2006, bus; nissan, 2006, bus; and nissan, 2006, bus. To the right of the table is a form with fields: 'seleccionar imgan' (with a file icon), 'Marca' (text input), 'Modelo' (text input), 'Tipo' (text input), 'placa' (text input), and 'cantidad de asientos' (text input). At the bottom are 'cancelar' and 'guardar' buttons.

Fuente: Elaboración propia

Figura 9: Arquitectura de interfaz de usuario lista de microbuses

The wireframe shows a web page titled 'A Web Page' with a URL bar containing 'https://'. On the left is a vertical sidebar menu with items: Microbuses (highlighted in blue), Monitoreo, Registro de usuarios, choferes, registro de microbus, Registro de linea, Registro de rutas, Registro de tarifas, and Registro de horarios.

The main content area has a search bar with a magnifying glass icon and a 'buscar' button. To the right is a table with columns: Marc, Model, tip, placa, cantidad, chofer, estado, and accione. The table contains four rows of data: Nissa, 2006, bu, xyz - 78, 32, santos machac, activo, :; Nissa, 2006, bu, xyz - 78, 32, santos machac, Mantenimie, :; Nissa, 2006, bu, xyz - 78, 32, santos machac, inactivo, :; and Nissa, 2006, bu, xyz - 78, 32, santos machac, activo, :. There is also a 'Nuevo bus' button above the table.

Fuente: Elaboración propia

Figura 10: Arquitectura de interfaz de usuario registro de línea

A Web Page
https://

Microbuses

Monitoreo
Registro de usuarios
choferes
registro de microbus
Registro de linea
Registro de rutas
Registro de tarifas
Registro de horarios

search buscar

linea	rutas	horario
012	rutas	horarios
012	rutas	horarios
012	rutas	horarios

Línea
asignar rutas
asignar horario
asignar tarifa
asignar minibus

cancelar guardar

Fuente: Elaboración propia

Figura 11: Arquitectura de interfaz de usuario lista de línea

A Web Page
https://

Microbuses

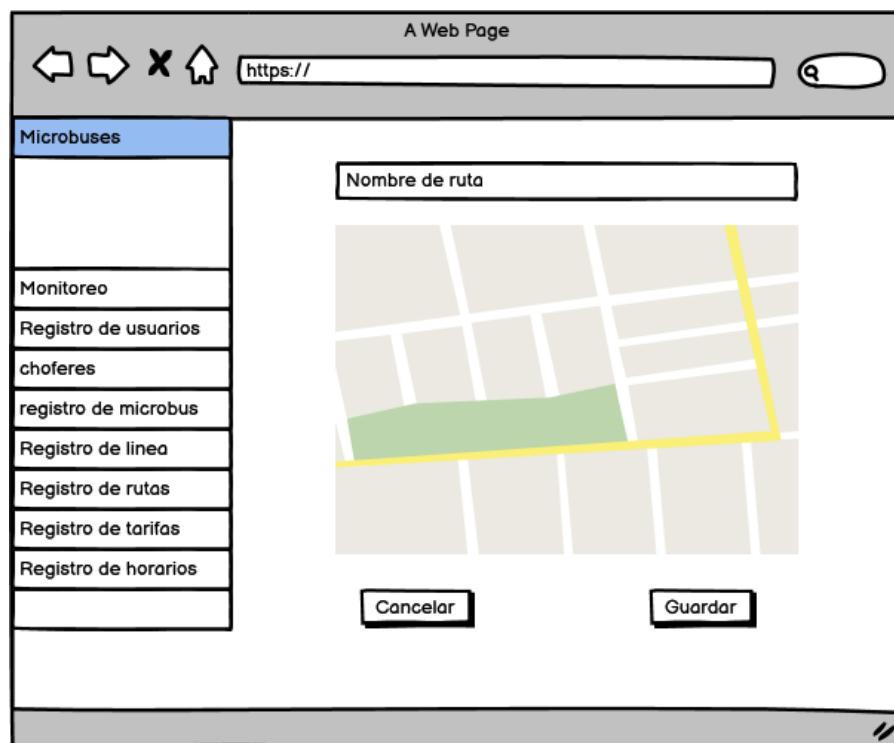
Monitoreo
Registro de usuarios
choferes
registro de microbus
Registro de linea
Registro de rutas
Registro de tarifas
Registro de horarios

search buscar Nueva linea

Línea	rutas y paradas	Horarios	tarifa	buses	acciones
012	ruta 012	horarios 012	tarifa 012	ver	:
012	ruta 012	horarios 012	tarifa 012	ver	:
012	ruta 012	horarios 012	tarifa 012	ver	:
012	ruta 012	horarios 012	tarifa 012	ver	:

Fuente: Elaboración propia

Figura 12: Arquitectura de interfaz de usuario registro de ruta



Fuente: Elaboración propia

Figura 13: Arquitectura de interfaz de usuario lista de rutas

Diagrama de la arquitectura de interfaz de usuario para la lista de rutas. La página web muestra el mismo menú lateral que en la Figura 12. En la parte superior derecha hay un campo de búsqueda ('search') y un botón 'buscar'. A la derecha de estos botones hay un botón 'Nueva ruta'. La sección principal contiene una tabla que muestra una lista de rutas registradas:

nombre de la ruta	fecha de creacion	rutas	acciones
ruta 012	30/02/2024	ver ruta	:
ruta 012	30/02/2024	ver ruta	:
ruta 012	30/02/2024	ver ruta	:
ruta 012	30/02/2024	ver ruta	:

Fuente: Elaboración propia

Figura 14: Arquitectura de interfaz de usuario registro de tarifa

A Web Page
https://

Microbuses

Monitoreo
Registro de usuarios
choferes
registro de microbus
Registro de linea
Registro de rutas
Registro de tarifas
Registro de horarios

search buscar

tarif	fecha de creacio	detalie
012	20/03/2024	ver detail
012	20/03/2024	ver detail
012	20/03/2024	ver detail

adulto Bs. 1
niños Bs. 0.50
escolar Bs. 1

+

Nombre de la tarifa

cancelar guardar

Fuente: Elaboración propia

Figura 15: Arquitectura de interfaz de usuario lista de tarifas

A Web Page
https://

Microbuses

Monitoreo
Registro de usuarios
choferes
registro de microbus
Registro de linea
Registro de rutas
Registro de tarifas
Registro de horarios

search buscar Nueva tarifa

nombre de la tarifa	fecha de creacion	tarifas	acciones
tarifa 012	30/02/2024	ver tarifa	:
tarifa 012	30/02/2024	ver tarifa	:
tarifa 012	30/02/2024	ver tarifa	:
tarifa 012	30/02/2024	ver tarifa	:

Fuente: Elaboración propia

Figura 16: Arquitectura de interfaz de usuario registro de horario

A Web Page
https://

Microbuses

Buscar bus

Tarif	Fecha de creacion	detalle
012	20/03/2024	<input type="button" value="ver detalle"/>
012	20/03/2024	<input type="button" value="ver detalle"/>
012	20/03/2024	<input type="button" value="ver detalle"/>
012	20/03/2024	<input type="button" value="ver detalle"/>

Lunes Martes Miércoles
 Jueves Viernes Sábado
 Domingo

Fuente: Elaboración propia

Figura 17: Arquitectura de interfaz de usuario lista de horario

A Web Page
https://

Microbuses

search Nueva horario

nombre del horario	horario	detalles	acciones
horario 012	30/02/2024	08:30 - 20:30	:
horario 012	30/02/2024	08:30 - 20:30	:
horario 012	30/02/2024	08:30 - 20:30	:
horario 012	30/02/2024	08:30 - 20:30	:

Fuente: Elaboración propia

2.6.4. Sprint 4

Tabla 40: especificación de sprints

Sprint 4				
Fecha de Inicio			Tareas Pendientes	0
Fecha de culminación			Dias pendientes	0
Prioridad	descripción de la tarea	Responsable	Duracion/semanas	Estado
Baja	Monitoreo	Santos Machaca L.	4	Terminado
Baja	Ver rutas y paradas	Santos Machaca L.	2	Terminado
Baja	Ver Tarifas	Santos Machaca L.	1	Terminado
Media	Ver estado de conexión	Santos Machaca L.	1	Terminado
Media	Ver velocidad de microbuses	Santos Machaca L.	1	Terminado
Media	Ver reporte de las solicitudes de las líneas	Santos Machaca L.	2	Terminado

Fuente: Elaboración propia

Figura 18: Arquitectura de interfaz de usuario monitoreo



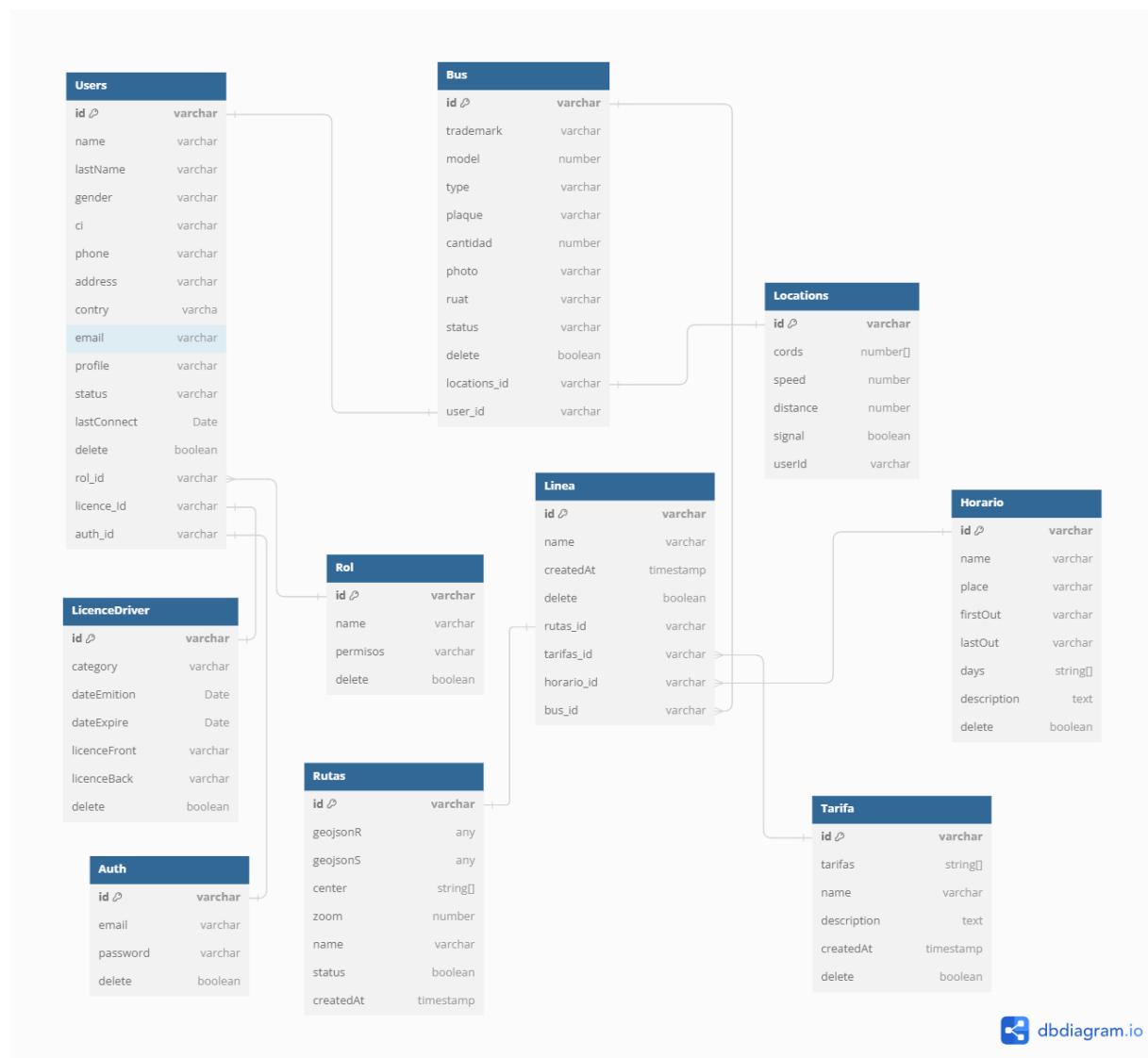
Fuente: Elaboración propia

2.7. Arquitectura de datos

Para la compresión de arquitectura de datos se propuso un diccionario de datos (**Ver Anexo 11**) el cual muestra a detalle la estructura de la base de datos, junto con el diagrama de clases que se muestra a continuación.

2.7.1. Diagrama de entidad relación

Figura 19: Diagrama de clases de entidad



Fuente: Elaboración propia

CAPÍTULO III

IMPLEMENTACIÓN Y PRUEBAS

CAPITULO III: IMPLEMENTACIÓN Y PRUEBAS

3.1. Introducción

En este capítulo se aborda la fase crucial de la implementación y pruebas del proyecto, donde las ideas y diseños conceptuales se transforman en una solución funcional. La implementación consiste en convertir las especificaciones del diseño en código ejecutable y ensamblar los diferentes componentes del sistema para crear un producto cohesivo. Este proceso incluye la programación, integración de módulos y configuración del entorno, asegurando que cada parte del sistema funcione según lo esperado.

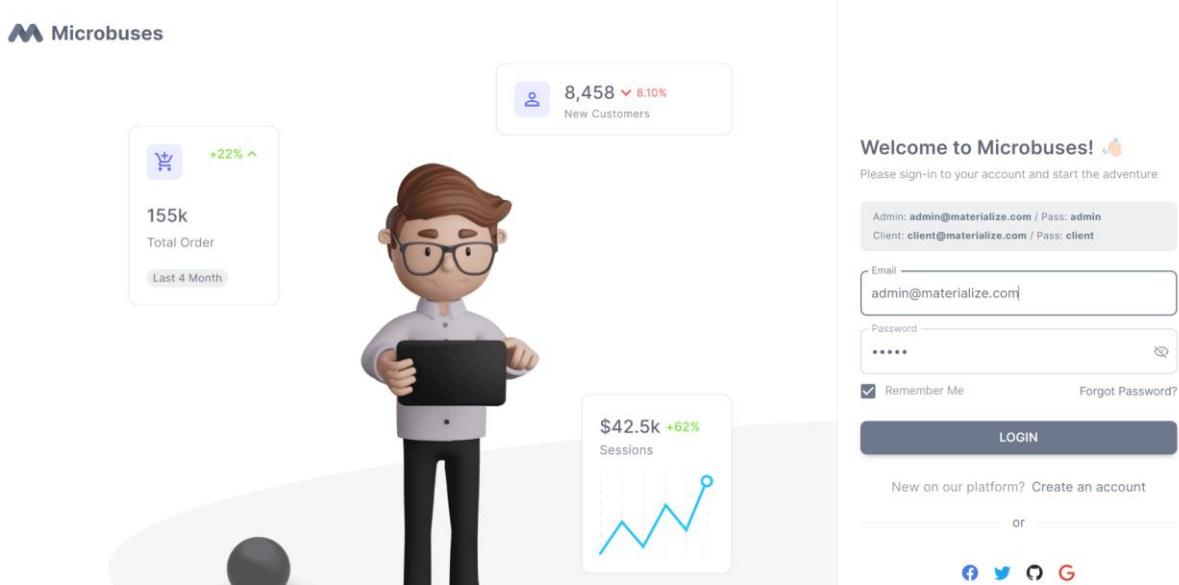
La fase de pruebas es igualmente esencial, ya que garantiza la calidad y fiabilidad del software desarrollado. A través de pruebas exhaustivas, se identifican y corrigen errores, se verifica el cumplimiento de los requisitos y se asegura que el sistema cumpla con los estándares de rendimiento y seguridad. Las pruebas se dividen en varios niveles, incluyendo pruebas unitarias, de integración, de sistema y de aceptación, cada una dirigida a validar diferentes aspectos del software.

Este capítulo describe detalladamente los pasos seguidos para implementar el sistema, las herramientas utilizadas, las metodologías de desarrollo adoptadas y los resultados obtenidos durante las pruebas. También se discuten los desafíos enfrentados y las soluciones implementadas para superarlos, proporcionando una visión completa del proceso de desarrollo y aseguramiento de la calidad del proyecto.

3.2. Implementación de sprint 1

3.2.1. Control de acceso al sistema

Figura 20: Implementación de interfaz Login



Fuente: Elaboración propia

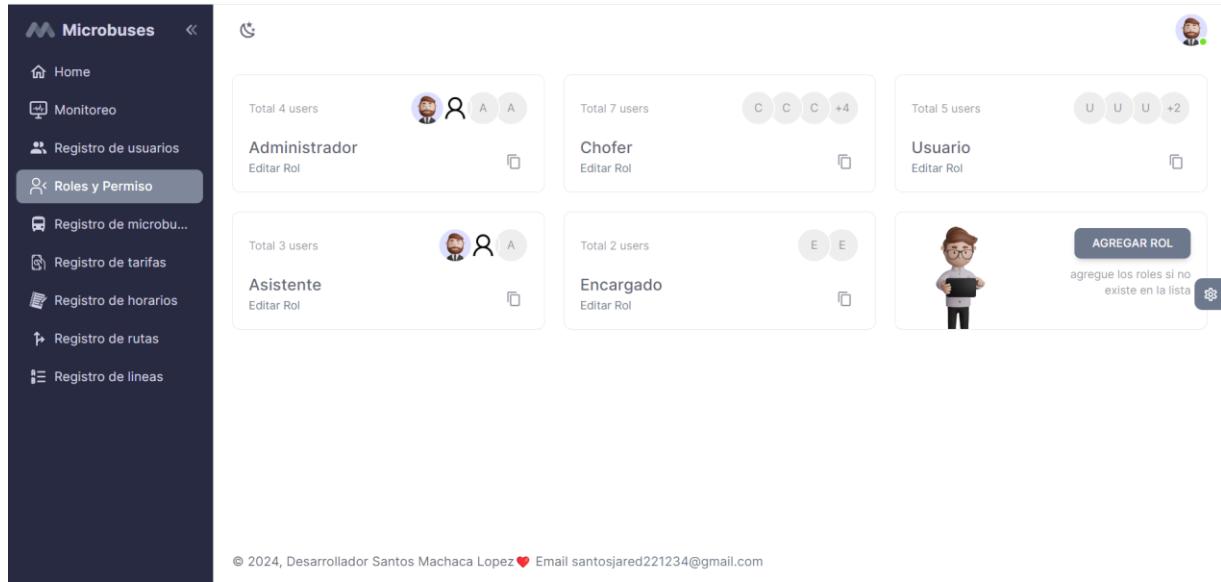
Figura 21: Código fuente de autenticación

A screenshot of a code editor showing the "auth.service.ts" file. The code is written in TypeScript and defines an "AuthService" class with various methods for managing user authentication. The code includes imports for HttpStatus, Injectable, and several DTO classes. It uses the Mongoose framework for database interactions and JWT for token generation. The complexity of the code is noted as 11.

Fuente: Elaboración propia

3.2.2. Roles y permisos

Figura 22: Lista de roles y permisos



Fuente: Elaboración propia

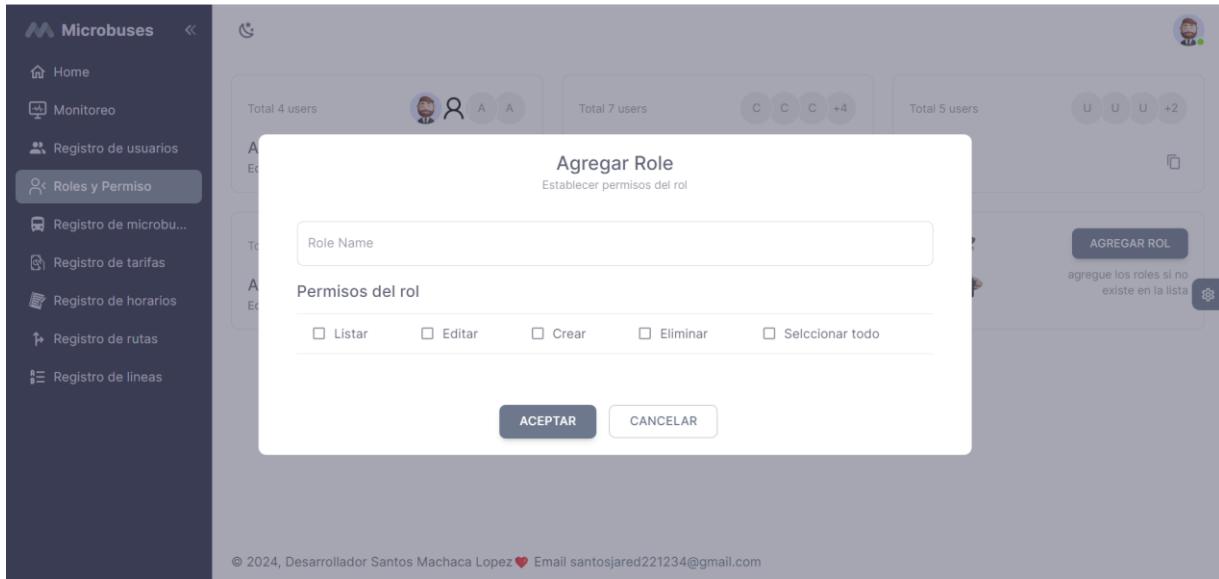
Figura 23: Código fuente de roles y permisos

```
src > roles > roles.service.ts M
1 import { Injectable } from '@nestjs/mongoose';
2 import { Role, RolDocument } from './schema/roles.schema';
3 import { FilterQuery, Model } from 'mongoose';
4 import { isEmptyDB } from 'src/utils/isEmptyDB';
5 import { RolSeeder } from 'src/seeders/rol.seeder';
6 import { Role } from './entities/role.entity';
7
8 You, 6 days ago | 1 author (You) | Complexity is 3 Everything is cool!
9
10 @Injectable()
11 export class RoleService {
12   constructor(@InjectModel(Role.name)private readonly rolModel:Model<RolDocument>){}
13   async create(createRoleDto: CreateRoleDto) {
14     return await this.rolModel.create(createRoleDto);
15   }
16
17   async findAll(filters:string):Promise<Role[]> {
18     const regexPattern = new RegExp(filters, 'i');
19     return this.rolModel.find({name:$regex:regexPattern}).exec();
20   }
21
22   async findOne(id: string) {
23     return await this.rolModel.findOne({_id, delete:false}).exec();
24   }
25
26   async update(id: string, updateRoleDto: UpdateRoleDto) {
27     return await this.rolModel.findOneAndUpdate({_id},updateRoleDto);
28   }
29
30 }
```

[Nest] 5584 - 25/06/2024, 19:05:02 LOG [RouterExplorer] Mapped {/componentes/:id, DELETE} route +0ms
[Nest] 5584 - 25/06/2024, 19:05:02 LOG [NestApplication] Nest application successfully started +18ms
cliente conectado b1gPk w5Y0ucUdpqAAB
cliente conectado bQgE90x4HfRr_0IHAAE
se desconecta cliente b1gPk w5Y0ucUdpqAAB
cliente conectado ZK9iUsgfB23lwz0JAAAG

Fuente: Elaboración propia

Figura 24: Agregar roles



Fuente: Elaboración propia

Figura 25: Código fuente de Agregar roles

La captura de pantalla muestra el entorno de desarrollo en Visual Studio Code. El explorador de archivos muestra la estructura del proyecto, incluyendo carpetas como "src", "dist", "node_modules", "auth", "bus", "componentes", "entities", "schema" y "config". La vista de código muestra el archivo "roles.controller.ts" que contiene el siguiente código:

```

import { Controller, Get, Post, Delete, Patch, Param, Body, Query } from '@nestjs/common';
import { RolesService } from './roles.service';
import { CreateRoleDto, UpdateRoleDto } from './schemas/roles.schema';

@Controller('roles')
export class RolesController {
    constructor(private readonly rolesService: RolesService) {}

    @Post()
    create(@Body() createRoleDto: CreateRoleDto) {
        return this.rolesService.create(createRoleDto);
    }

    @Get()
    findAll(@Query() query: FiltersRoleDto) {
        const { filter } = query;
        const items = await this.rolesService.findAll(filter);
        return items;
    }

    @Get(':id')
    findOne(@Param('id') id: string) {
        return await this.rolesService.findOne(id);
    }

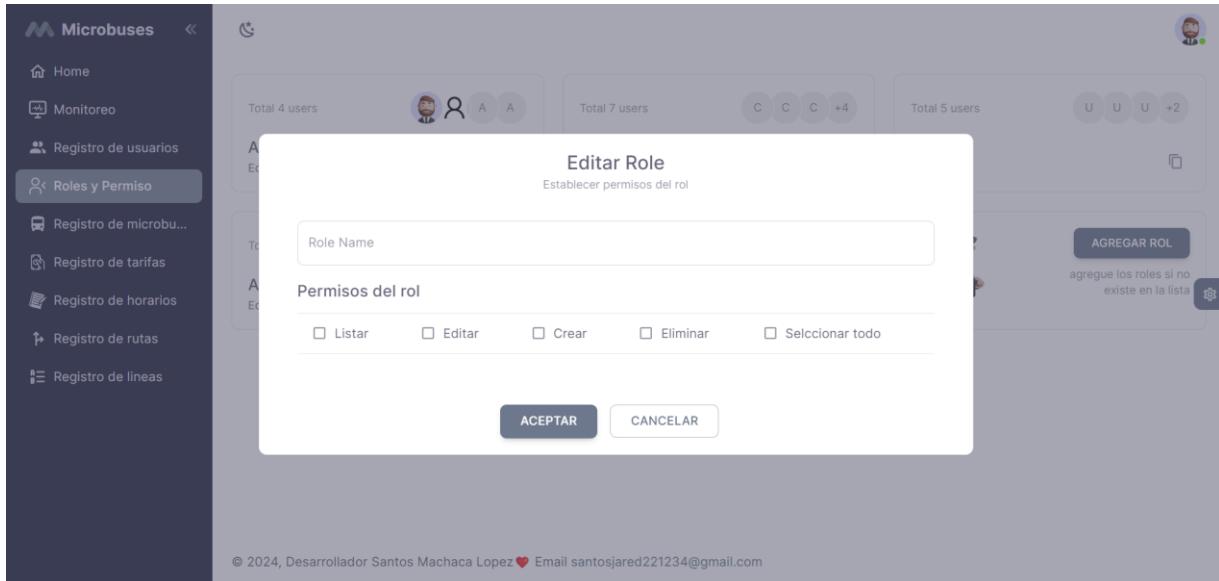
    @Patch(':id')
    update(@Param('id') id: string, @Body() updateRoleDto: UpdateRoleDto) {
        return this.rolesService.update(id, updateRoleDto);
    }

    @Delete(':id')
    remove(@Param('id') id: string) {
        return this.rolesService.remove(id);
    }
}
  
```

El terminal muestra logs de ejecución, indicando que el servicio se inició exitosamente y que se han conectado clientes.

Fuente: Elaboración propia

Figura 26: Editar roles



Fuente: Elaboración propia

Figura 27: Código fuente de editar roles

```

src > componentes > componentes.controller.ts > ...
1 import { Controller, Get, Post, Body, Patch, Param, Delete } from '@nestjs/common';
2 import { ComponentesService } from './componentes.service';
3 import { CreateComponenteDto } from './dto/create-componente.dto';
4 import { UpdateComponenteDto } from './dto/update-componente.dto';
5
6 @controller('componentes')
7 export class ComponentesController {
8   constructor(private readonly componentesService: ComponentesService) {}
9
10  @Post()
11  create(@Body() createComponenteDto: CreateComponenteDto) {
12    return this.componentesService.create(createComponenteDto);
13  }
14
15  @Get()
16  findAll() {
17    return this.componentesService.findAll();
18  }
19
20  @Get(':id')
21  findOne(@Param('id') id: string) {
22    return this.componentesService.findOne(+id);
23  }
24
25  @Patch(':id')
26  update(@Param('id') id: string, @Body() updateComponenteDto: UpdateComponenteDto) {
27    return this.componentesService.update(+id, updateComponenteDto);
28  }
29}

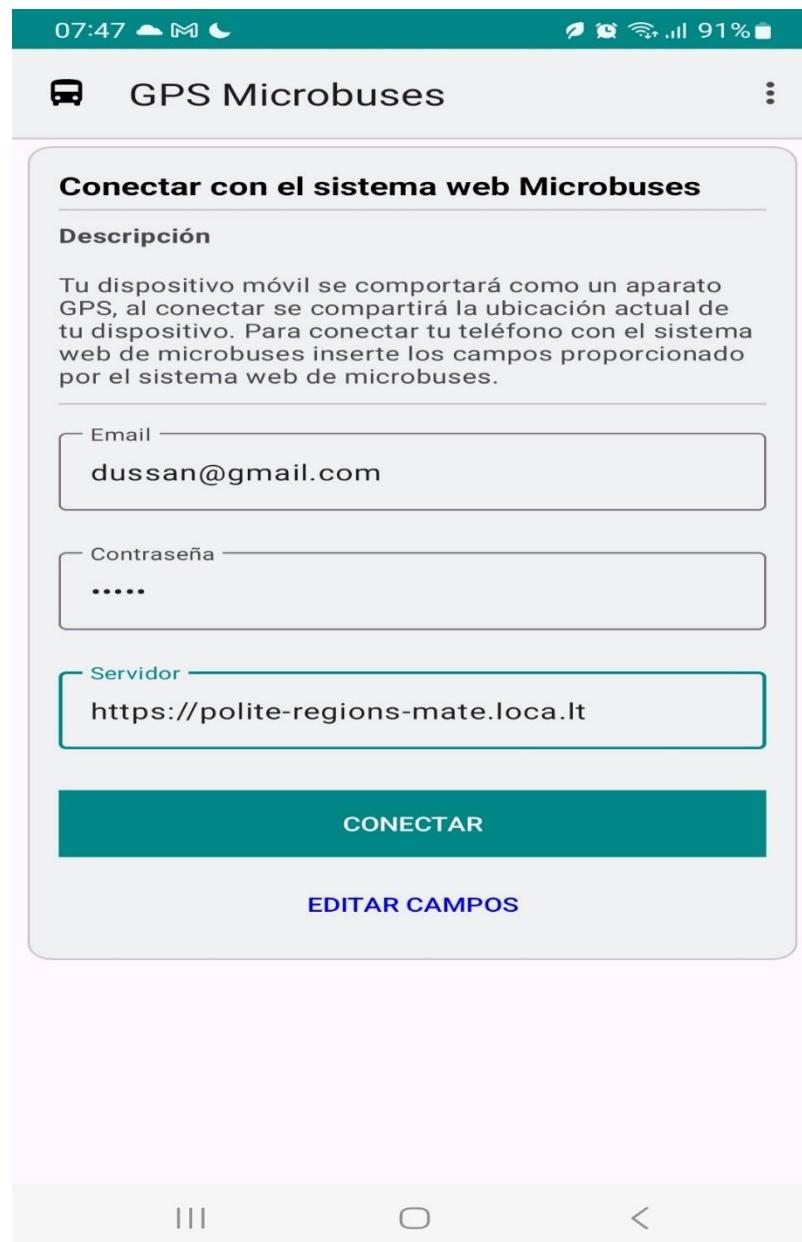
```

Fuente: Elaboración propia

3.3. Implementación de sprint 2

3.3.1. Aplicación GPS

Figura 28: Aplicación GPS



Fuente: Elaboración propia

Figura 29. Código fuente de la aplicación GPS

```

1 Usage santosjared *
private void connectToSocket(String server) {
    sendSocketConnectionStatus( isLoading: true, isSuccess: false, isError: false, isConnected: false, credencial: false);
    if (!server.startsWith("http://") && !server.startsWith("https://")) {
        server = "http://" + server;
    }
    try {
        socket = IO.socket(server); // Reemplaza "your_server_url" con la URL de tu servidor Socket.io
        socket.connect();
        socket.on(Socket.EVENT_CONNECT, new Emitter.Listener() {
            santosjared *
            @Override
            public void call(Object... args) {
                initChat();
                statusConnect=true;
            }
        }).on(Socket.EVENT_DISCONNECT, new Emitter.Listener() {
            santosjared *
            @Override
            public void call(Object... args) {
                sendSocketConnectionStatus( isLoading: false, isSuccess: false, isError: false, isConnected: true, credencial: false);
                statusConnect=false;
            }
        }).on(Socket.EVENT_CONNECT_ERROR, new Emitter.Listener(){
            santosjared *
            @Override
            public void call(Object... args){
                statusConnect=false;
            }
        });
    } catch (Exception e) {
        e.printStackTrace();
    }
}

```

Fuente: Elaboración propia

3.4. Implementación de sprint 3

3.4.1. Listar Usuarios

Figura 30. Listar usuarios

NOMBRES Y APELLIDOS	CI	DIRECCIÓN	CELULAR	GÉNERO	NACIONALIDAD	ROL	ACCIONES
juan carlos rodriguez gasdfhja@gmail.com	3435354	sdfsfdsf	3453535	Masculino	Bolivia	Niguno	⋮
manuel torrez asdas asfdsdf@hmail.com	34535	xcsxsd	3453534	Masculino	Bolivia	Niguno	⋮
ghggvhgh,m,m,m hjhgfgh@gmail.com	9889879	lkhkhjg	6676767	Masculino	Afganistán	Niguno	⋮
dussan canaviri dussan@gmail.com	10531753	Av. las bande...	72381722	Masculino	Bolivia	Niguno	⋮
Jhony Contreras Vas jhony@gmail.com	1053175	Av. los pinos	72381722	Masculino	Bolivia	Niguno	⋮
Jheny Benitez Relos Jheny@gmail.com	1385678	Av. Las bande...	72381722	Femenina	Bolivia	Niguno	⋮

Fuente: Elaboración propia

Figura 31. Código fuente de listar usuarios

```

src > users > users.service.ts > UsersService > findUser
13   export class UsersService {
14
15     Complexity is 6 It's time to do something...
16
17     async findAll(filters: FiltersDTO) {
18       const { filter, skip, limit } = filters
19       const regexPattern = new RegExp(filter, 'i');
20       const columnstoSearch = ['name', 'lastName', 'gender', 'ci', 'phone', 'address', 'contry', 'email', 'profile'];
21
22       const orConditions = columnstoSearch.map(column => ({
23         [column]: { $regex: regexPattern }
24       }));
25
26       if (skip && limit) {
27         const result = await this.userModel.find({$or:orConditions, delete:false}).populate('rol').populate('licenceId').skip(skip).limit(limit)
28         const total = await this.userModel.countDocuments({$or:orConditions, delete:false})
29         return { result, total }
30       }
31
32       const result = await this.userModel.find({$or:orConditions, delete:false}).populate('rol').populate('licenceId').exec();
33       const total = await this.userModel.countDocuments({$or:orConditions, delete:false})
34       return { result, total }
35     }
36
37     async findOne(id: string) {
38       return await this.userModel.findOne({ id, delete: false });
39     }
40
41     Complexity is 6 It's time to do something...
42
43     async findUser(filters) {
44       const { filter, skip, limit } = filters
45       const regexPattern = new RegExp(filter, 'i');
46       const columnstoSearch = ['name', 'lastName', 'gender', 'ci', 'phone', 'address', 'contry', 'email', 'profile'];
47
48       const orConditions = columnstoSearch.map(column => ({
49         [column]: { $regex: regexPattern }
50       }));
51
52       if (skip && limit) {
53
54       }
55     }
56
57   }
58
59   const columnstoSearch = ['name', 'lastName', 'gender', 'ci', 'phone', 'address', 'contry', 'email', 'profile'];
60
61   const orConditions = columnstoSearch.map(column => ({
62     [column]: { $regex: regexPattern }
63   }));
64
65   if (skip && limit) {
66     const result = await this.userModel.find({$or:orConditions, delete:false}).populate('rol').populate('licenceId').skip(skip).limit(limit)
67     const total = await this.userModel.countDocuments({$or:orConditions, delete:false})
68     return { result, total }
69   }
70
71   const result = await this.userModel.find({$or:orConditions, delete:false}).populate('rol').populate('licenceId').exec();
72   const total = await this.userModel.countDocuments({$or:orConditions, delete:false})
73   return { result, total }
74 }
75
76
77
78
79
80
81
82
83
84
85
86
87

```

Fuente: Elaboración propia

3.4.2. Registrar Usuarios

Figura 32: Registrar Usuarios

NOMBRES Y APELLIDOS	CI	DIRECCIÓN	CELULAR
juan carlos rodriguez gasdthja@gmail.com	3435354	sdfsfdsf	3453535
manuel torrez asdasd asfdsof@hotmail.com	34535	xcsxsd	3453534
ghggvvhgh,m,m,m hjhhgf@gmail.com	9889879	lkhjhjg	6676767
dussan canaviri dussan@gmail.com	10531753	Av. las bander...	72381722
Jhony Contreras Vas jhony@gmail.com	1053175	Av. los pinos	72381722
Jhenny Benitez Relos Jhenny@gmail.com	1385678	Av. Las bande...	72381722

Nombre: Apellido:
 Género: País:
 Celular: Dirección:
 Correo electrónico:

Fuente: Elaboración propia

Figura 33. Código fuente de Registrar Usuarios

```

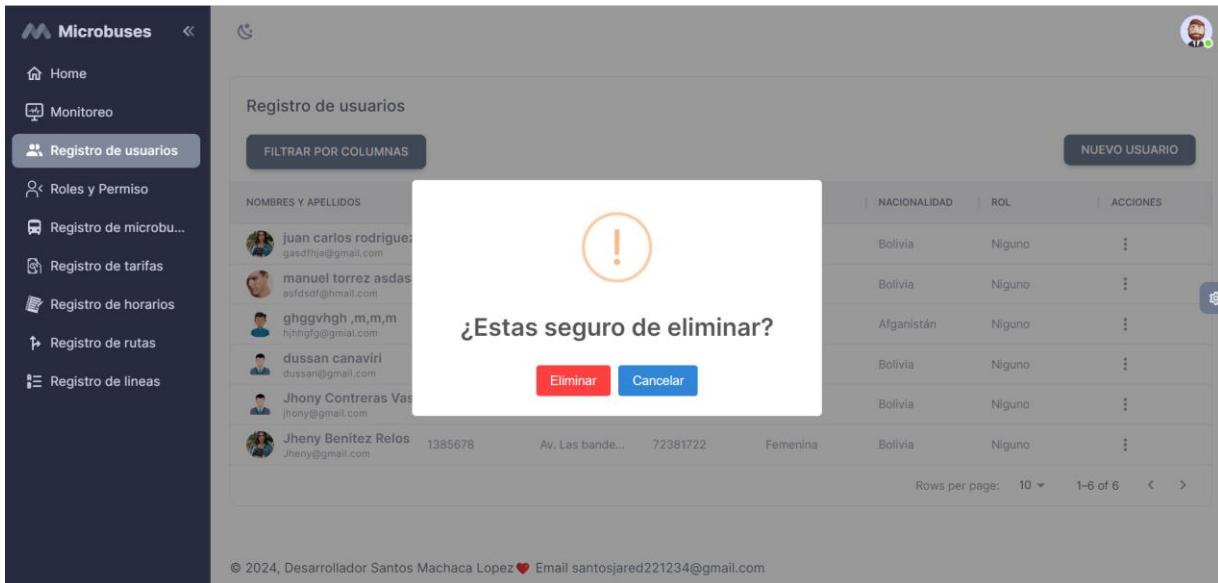
You, 2 weeks ago | author (You) | Complexity is 14 You must be kidding
@injectable()
export class UserService {
  constructor(@injectModel('Users.name') private readonly userModel: Model<UsersDocument>,
  private authService: AuthService,
  @injectModel('Bus.name') private readonly busModel: Model<BusDocument>
) {}

Complexity is 14 You must be kidding
async create(createUserDto: CreateUserDto, file: Express.Multer.File) {
  const { email, password, ci } = createUserDto
  const dbEmail = await this.userModel.findOne({ email })
  if (dbEmail) {
    if (dbEmail.delete) {
      createUserDto.delete=false
      this.authService.update(email,{email,delete:false})
      return this.update(dbEmail.id, createUserDto, file)
    }
    UsersMessageError.email = 'el Correo Electrónico ya se encuentra registrado'
    throw new HttpException(UsersMessageError, HttpStatus.BAD_REQUEST);
  }
  const dbCi = await this.userModel.findOne({ ci })
  if (dbCi) {
    if (dbCi.delete) {
      createUserDto.delete=false
      this.authService.update(email,{email,delete:false})
      return this.update(dbEmail.id, createUserDto, file)
    }
    UsersMessageError.ci = 'El ci ya se encuentra registrado'
    throw new HttpException(UsersMessageError, HttpStatus.BAD_REQUEST);
  }
  if (file) {
    const fs = require('fs');
    fs.renameSync(file.path, `./uploads/${file.originalname}`)
    createUserDto.profile = `./uploads/${file.originalname}`;
  }
}
  
```

Fuente: Elaboración propia

3.4.3. Eliminar Usuarios

Figura 34. Eliminar Usuarios



Fuente: Elaboración propia

Figura 35. Código fuente de eliminar usuarios

The screenshot shows a developer's workspace with multiple tabs open in a code editor. The left sidebar displays a project structure for 'TRANSPORT-BACKEND' with several modules like auth, bus, config, etc. The main editor area shows a file named 'users.service.ts'. A search bar at the top right contains the text 'Complexity is 4 Everything is cool'. The code itself is a class definition for 'UserService' with various methods like 'findUser', 'update', 'updateStatus', 'remove', and 'Users'. Below the code, a note says 'Complexity is 10 It's time to do something...'. The bottom status bar indicates the code was last updated 'You, 2 weeks ago'.

```
src > users > users.service.ts > UserService > findUser
  export class UserService {
    async findUser(filters) {
      const result = await this.userModel.find({$or:[{conditions}, {delete:false}, {busId:null}]}).exec();
      const total = await this.userModel.countDocuments({$or:[{conditions}, {delete:false}, {busId:null}]});
      return { result, total }
    }
  }

Complexity is 4 Everything is cool
async update(id: string, updateUserDto: UpdateUserDto, file: Express.Multer.File) {
  const { email } = updateUserDto
  if (file) {
    const fs = require('fs');
    fs.renameSync(file.path, `./uploads/${file.originalname}`);
    updateUserDto.profile = `/uploads/${file.originalname}`;
    this.authService.update(email, { email });
    return await this.userModel.findOneAndUpdate({id}, updateUserDto);
  }
  this.authService.update(email, { email })
  return await this.userModel.findOneAndUpdate({id}, updateUserDto);
}

async updateStatus(id: string, status: string) {
  return await this.userModel.findOneAndUpdate({ id }, { status, lastConnect: new Date(Date.now()) });
}

async remove(id: string) {
  const remove = await this.userModel.findOneAndUpdate({ id }, { delete: true }, { new: true });
  this.authService.remove(remove.email)
  return remove
}

}

Complexity is 10 It's time to do something...
async Users(filters) {
  const usersAssignedToBus = await this.busModel.find().distinct('userId').exec();
  if (usersAssignedToBus === null) {
    return [];
  }
  const userIdsAssignedToBus = usersAssignedToBus
    .filter(userId => userId !== null)
```

Fuente: Elaboración propia

3.4.4. Listar Buses

Figura 36. Listar Buses

Lista de Microbuses									
CERRAR FILTRADO NUEVO MICROBUS									
Marca	Modelo	Tipo	Placa	Cantidad de Asientos	Chofer	Ruat	Estado		
FILTRAR RESTABLECER									
MARCA	MODELO	TIPO	PLACA	ASIENTOS	CHOFER	RUAT	ESTADO		ACCIONES
 Mercedes Benz	2006	Bus	HGV 778	28	 juan carlos rodriguez 3435354	Abrir	Activo	...	
 Tata	1990	Microbus	SAX 897	32	 dussan canaviri 1053175	Abrir	Activo	...	
Nissan	1990	Microbus	ZSD 3232	16	 Jhony Contreras Vasc 1053175	Sin Docu...	En Manteni...	...	
Nissan	1990	Microbus	ZSD 789	34	 manuel torrez asdasa 34535	Sin Docu...	Inactivo	...	
Nissan	1990	Microbus	xdfg 678	16	No asignado	Sin Docu...	Proceso	...	
 Nissan	1990	Microbus	GHIY 890	32	 Jheny Benitez Relos 1385678	Abrir	Activo	...	

Fuente: Elaboración propia

Figura 37. Código fuente de Listar Buses

```

import { Injectable } from '@nestjs/common';
import { InjectModel } from '@nestjs/mongoose';
import { Model } from 'mongoose';
import { Bus, BusDocument } from './bus.schema';
import { LineaService } from 'src/linea/linea.service';
import { AsignacionBus } from 'src/linea/entities/asignacion-bus';
import { LineaController } from 'src/linea/linea.controller';
import { SocketGateway } from 'src/socket.gateway';

@Injectable()
export class BusService {
    constructor(
        @InjectModel(Bus.name) private busModel: Model,
        private lineaService: LineaService,
        private lineaController: LineaController,
        private socketGateway: SocketGateway
    ) {}

    Complexity is 12 You must be kidding
    async findAll(filters: FiltersDto): Promise {
        const { filter, skip, limit } = filters;
        const columnstoSearch = ['trademark', 'type', 'plaque'];
        const filterNumber = parseFloat(filter);
        const isFilterNumber = !isNaN(filterNumber);

        if (isFilterNumber) {
            columnstoSearch.push('model');
            columnstoSearch.push('cantidad');
        }

        Complexity is 6 It's time to do something...
        const orConditions = columnstoSearch.map(column => {
            if (isFilterNumber && (column === 'cantidad' || column === 'model')) {
                return { [column]: filterNumber };
            } else {
                return { [column]: new RegExp(filter, 'i') };
            }
        });
        if (skip && limit) {
            const result = await this.busModel.find({$or:orConditions,delete:false})
                .populate({path:'userId',model:'User',populate:{path:'licenceId',model:'LicenceDriver'}})
                .populate('locationId',skip,limit).exec();
            const total = await this.busModel.countDocuments({$or:orConditions,delete:false});
            return { result, total };
        }
        const result = await this.busModel.find({$or:orConditions,delete:false})
            .populate({path:'userId',model:'User',populate:{path:'licenceId',model:'LicenceDriver'}})
            .populate('locationId').exec();
        const total = await this.busModel.countDocuments({$or:orConditions,delete:false});
        return { result, total };
    }

    async findOne(id: string): Promise {
        You, 3 months ago dev backend
        return await this.busModel.findOne({id, delete:false}).populate('userId').populate('locationId');
    }
}

```

Fuente: Elaboración propia

3.4.5. Registrar buses

Figura 38. Registrar buses

MARCA	MODELO	TIPO	PLACA	ASIENTOS
Mercedes Benz	2006	Bus	HGV 778	28
Tata	1990	Microbus	SAX 897	32
Nissan	1990	Microbus	ZSD 3232	16
Nissan	1990	Microbus	ZSD 789	34
Nissan	1990	Microbus	xdfg 678	16
Nissan	1990	Microbus	GHIY 890	32

Fuente: Elaboración propia

Figura 39. Código fuente de Registrar buses

```

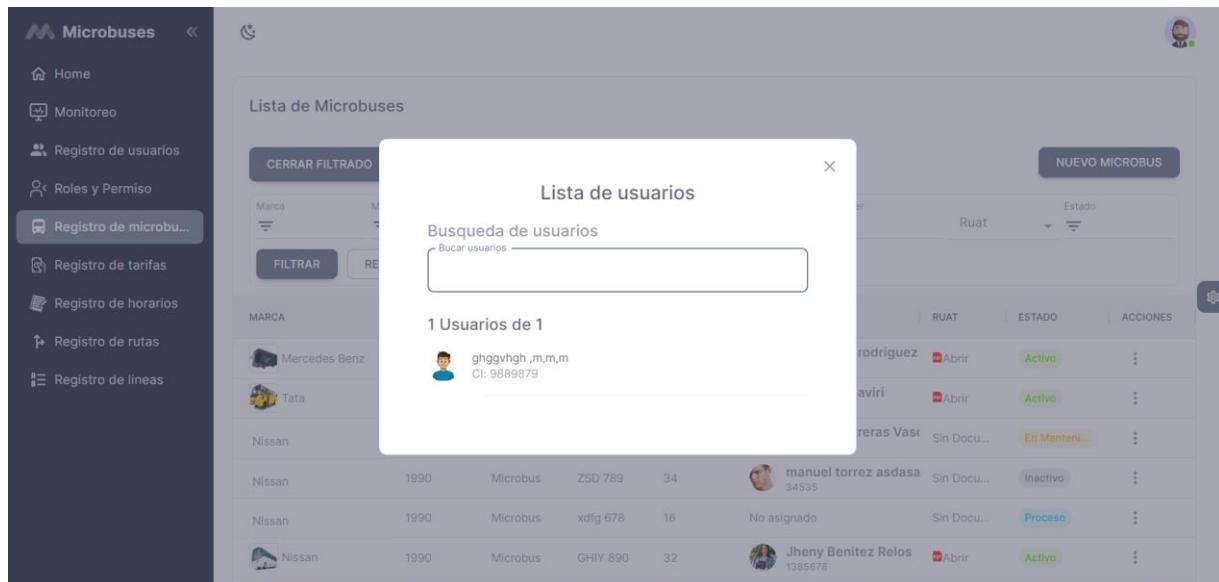
src > bus > bus.service.ts > BusService > isValidData
You, 2 weeks ago | 1 author (You)
1 import { HttpException, HttpStatus, Injectable, NotFoundException } from '@nestjs/common';
2 import { CreateBusDto } from './dto/create-bus.dto';
3 import { UpdateBusDto } from './dto/update-bus.dto';
4 import { InjectModel } from '@nestjs/mongoose';
5 import { Bus } from './entities/bus.entity';
6 import { Model } from 'mongoose';
7 import { BusDocument } from './schema/bus.schema';
8 import { AsigneDriverDto } from './dto/asigne-driver.dto';
9 import { FiltersDto } from 'src/utils/filters.dto';
10 import * as fs from 'fs';
11 import { Users, UsersDocument } from 'src/users/schema/users.schema';
12
You, 2 weeks ago | 1 author (You) | Complexity is 24 You must be kidding
13 @Injectable()
14 export class BusService {
15   constructor(@InjectModel(Bus.name) private readonly busModel:Model<BusDocument>,
16             @InjectModel(Users.name) private readonly userModel:Model<UsersDocument>)
17 }
Complexity is 12 You must be kidding
18 async create(createBusDto: CreateBusDto, files: (photo?: Express.Multer.File[], ruat?: Express.Multer.File[])) {
19   const errorMessages= await this.isValidData(createBusDto)
20   if(errorMessages){
21     if(files){
22       const { photo, ruat } = files;
23       if(photo && ruat){
24         if (photo && ruat.length > 0) {
25           const busPhoto = photo[0];
26           const frontImagePath = `./uploads/${busPhoto.originalname}`;
27           await fs.promises.rename(busPhoto.path, frontImagePath);
28           createBusDto.photo = `/uploads/${busPhoto.originalname}`;
29         }
30       }else{
31         if(photo){
32           const busPhoto = photo[0];
33           const frontImagePath = `./uploads/${busPhoto.originalname}`;
34           await fs.promises.rename(busPhoto.path, frontImagePath);
35         }
36       }
37     }
38   }
39 }

```

Fuente: Elaboración propia

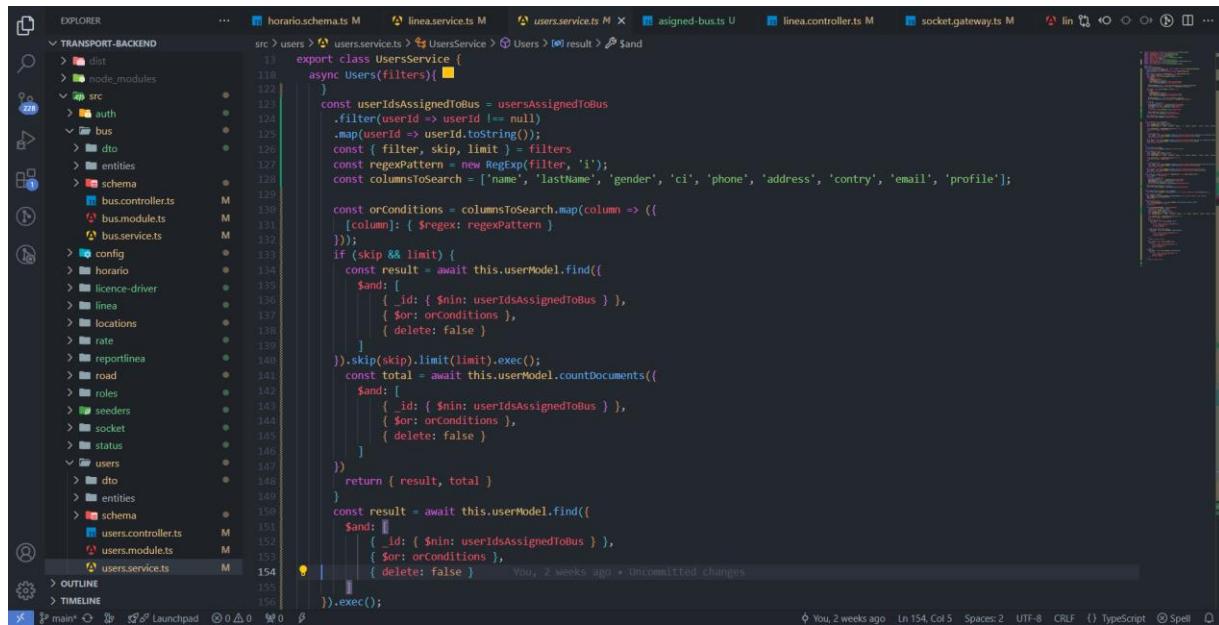
3.4.6. Asignar chofer

Figura 40. Asignar chofer



Fuente: Elaboración propia

Figura 41. Código fuente de Asignar chofer



```

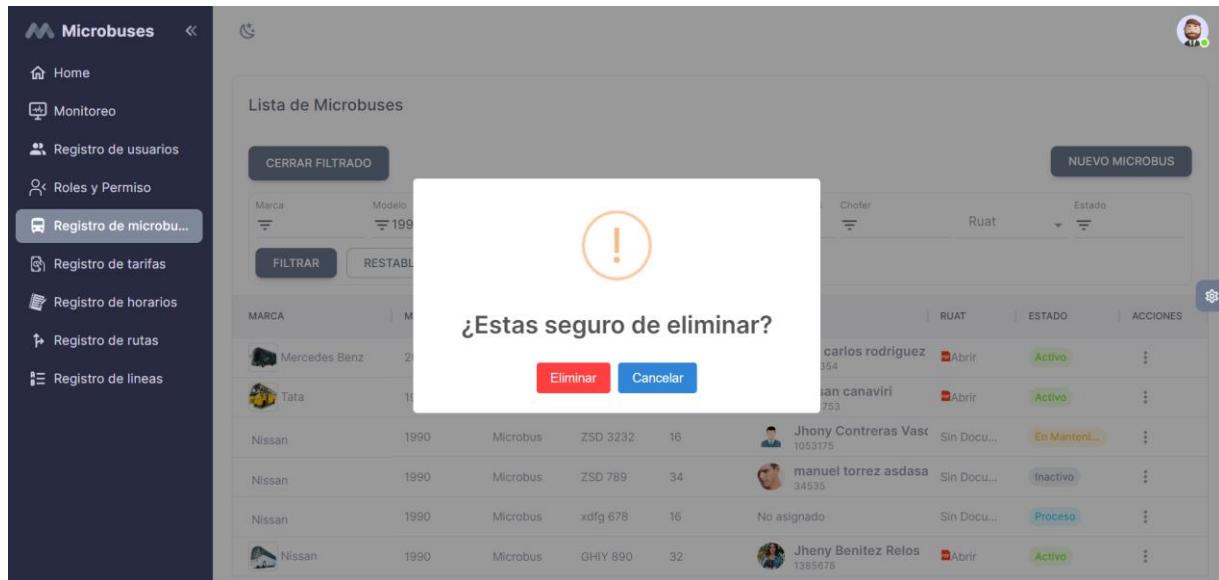
src > users > users.service.ts > UsersService > Users > result > $and
119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154
    export class UsersService {
        async usersAssignedToBus(filters) {
            const userIdsAssignedToBus = usersAssignedToBus
                .filter(userId => userId !== null)
                .map(userId => userId.toString());
            const { filter, skip, limit } = filters;
            const regexPattern = new RegExp(filter, 'i');
            const columnstoSearch = ['name', 'lastName', 'gender', 'ci', 'phone', 'address', 'contrary', 'email', 'profile'];
            const orConditions = columnstoSearch.map(column => ({
                [column]: { $regex: regexPattern }
            }));
            if (skip && limit) {
                const result = await this.usermodel.find({
                    $and: [
                        { _id: { $in: userIdsAssignedToBus } },
                        { $or: orConditions },
                        { delete: false }
                    ]
                }).skip(skip).limit(limit).exec();
                const total = await this.usermodel.countDocuments({
                    $and: [
                        { _id: { $in: userIdsAssignedToBus } },
                        { $or: orConditions },
                        { delete: false }
                    ]
                });
                return { result, total }
            }
            const result = await this.usermodel.find({
                $and: [
                    { _id: { $in: userIdsAssignedToBus } },
                    { $or: orConditions },
                    { delete: false }
                ]
            });
            return result;
        }
    }
}

```

Fuente: Elaboración propia

3.4.7. Eliminar bus

Figura 42. Eliminar bus



Fuente: Elaboración propia

Figura 43. Código fuente de Eliminar bus

```

src > bus > buservice.ts > BusService > desassigned > updatedBus
14
15 export class BusService {
16     async update(id: string, updateBusDto: UpdateBusDto, files: { photo?: Express.Multer.File[], route?: Express.Multer.File[] }) {
17         throw new HttpException(errorMessages, HttpStatus.BAD_REQUEST);
18     }
19
20     async remove(id: string) {
21         return this.busModel.findOneAndUpdate({id}, {delete:true}, {new:true});
22     }
23
24     async getUsers() {
25         return this.busModel.aggregate([
26             { $lookup: {
27                 from: 'buses',
28                 localField: '_id',
29                 foreignField: 'users',
30                 as: 'assignedBuses'
31             } },
32             { $match: {
33                 assignedBuses: { $size: 0 }
34             } }
35         ]).exec();
36     }
37
38 Complexity is 4 Everything is cool!
39     async asignedriver(id:string, asignedriverDto:AsigneDriverDto){
40         const(userid)->asignedriver
41         const response = await this.busModel.findOneAndUpdate((id),{userId:(new:true))
42         if(response){
43             return response
44         }
45         throw new HttpException('', HttpStatus.BAD_REQUEST);
46     }
47
48 Complexity is 4 Everything is cool!
49     async desassigned(id: string) {
50         const updatedBus = await this.busModel.findOneAndUpdate(
51             { id },
52             { userId: null },
53             { new: true }
54         )
55     }
56
57 Complexity is 4 Everything is cool!
58 
```

You, 2 weeks ago · In 146, Col 1 · Spaces: 2 · JST-8 · CR LF · {} · TypeScript · Spell · ⚙

Fuente: Elaboración propia

3.4.8. Listar línea

Figura 44. Listar línea

NOMBRE DE LA LINEA	RUTAS Y PARADAS	HORARIOS	TARIFAS	BUSES	RUTAS BUSES	ACCIONES
123	<input type="checkbox"/> ver rutas	<input type="checkbox"/> ver horario	<input type="checkbox"/> ver tarifas	<input type="checkbox"/> ver buses	<input type="checkbox"/> asignar o desas...	⋮
234	<input type="checkbox"/> ver rutas	<input type="checkbox"/> ver horario	<input type="checkbox"/> ver tarifas	<input type="checkbox"/> ver buses	<input type="checkbox"/> asignar o desas...	⋮
G	<input type="checkbox"/> ver rutas	<input type="checkbox"/> ver horario	<input type="checkbox"/> ver tarifas	<input type="checkbox"/> ver buses	<input type="checkbox"/> asignar o desas...	⋮
h	<input type="checkbox"/> ver rutas	<input type="checkbox"/> ver horario	<input type="checkbox"/> ver tarifas	<input type="checkbox"/> ver buses	<input type="checkbox"/> asignar o desas...	⋮
110	<input type="checkbox"/> ver rutas	<input type="checkbox"/> ver horario	<input type="checkbox"/> ver tarifas	<input type="checkbox"/> ver buses	<input type="checkbox"/> asignar o desas...	⋮
M	<input type="checkbox"/> ver rutas	<input type="checkbox"/> ver horario	<input type="checkbox"/> ver tarifas	<input type="checkbox"/> ver buses	<input type="checkbox"/> asignar o desas...	⋮

Rows per page: 10 ▾ 1–6 of 6 < >

© 2024, Desarrollador Santos Machaca Lopez ❤ Email santosjared22123@gmail.com

Fuente: Elaboración propia

Figura 45. Código fuente de Listar línea

```

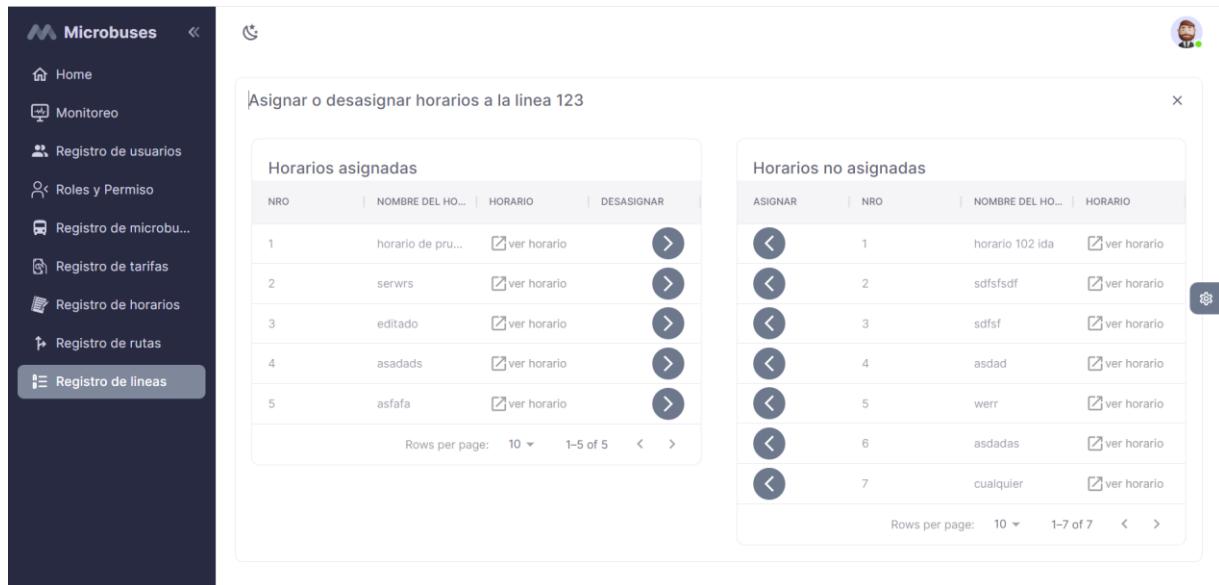
src > linea > linea.service.ts > LineaService > findAll > [0] orConditions
17  export class LineaService {
32    async findAll(filters: FiltersDto) {
33      const orConditions = columnsToSearch.map((column) => ((
34        | [column]: { $regex: regexPattern }
35        | You, 5 days ago • Uncommitted changes
36      )));
37
38      const query = { delete: false };
39
40      const populateOptions = [
41        { path: 'road', match: { delete: false } },
42        { path: 'horario', match: { delete: false } },
43        { path: 'rate', match: { delete: false } },
44        { path: 'buses', match: { delete: false } },
45        { path: 'userId', model: 'Users', match: { delete: false } },
46        { path: 'locationId', model: 'locations', match: { delete: false } },
47        { path: 'licencieDriver', model: 'licenceDriver', match: { delete: false } }
48      ];
49
50      const result = await this.lineaModel
51        .find(query)
52        .populate(populateOptions)
53        .exec();
54
55      return result;
56    }
57
58    const seeders = [
59      { name: 'seeder1', args: {} },
60      { name: 'seeder2', args: {} },
61      { name: 'seeder3', args: {} },
62      { name: 'seeder4', args: {} },
63      { name: 'seeder5', args: {} },
64    ];
65
66    const result = await this.lineaModel
67      .find()
68      .exec();
69
70    return result;
71  }
72}
73
74  let result;
75
76  You, 5 days ago • In 39, Col 8 • Spaces: 2 • UTF-8 • CRLF • {} TypeScript • Spell

```

Fuente: Elaboración propia

3.4.9. Asignar horarios

Figura 46. Asignar horarios



Fuente: Elaboración propia

Figura 47. Código fuente de Asignar horarios

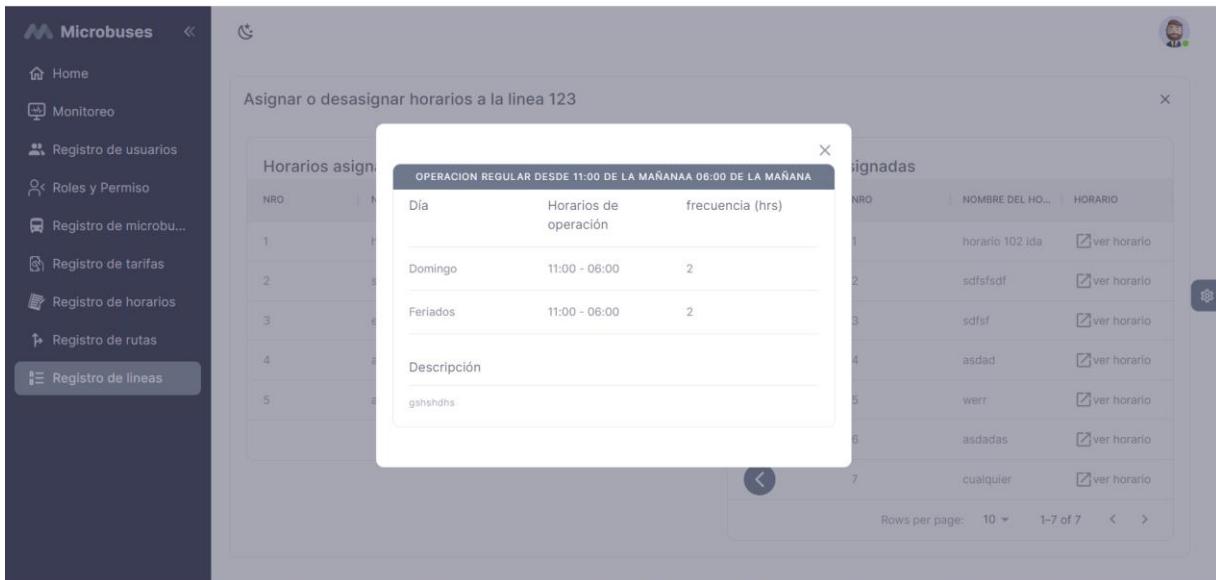
```

src > linea > linea.service.ts > LineaService > desasignadoHorario > update
    export class LineaService {
        ...
        async assignedRoad (id:string,assignedRoadto:AssignedRoadto){
            ...
        }
        Complexity is 4 Everything is cool
        ...
        const update = await this.lineModel.findOneAndUpdate({id},{$pull:{horario:$in:desasignadoHorario.horario}}),(new:true))
        if(update){
            return update.populate([ path: 'horario', match: { delete: false } ])
        }
        throw new NotFoundException('linea no encontrado');
    }
    Complexity is 4 Everything is cool
    ...
    const update = await this.lineModel.findOneAndUpdate({id},{$addToSet:{horario:$each:asignedHorario.horario}}),(new:true))
    if(update){
        return update.populate([ path: 'horario', match: { delete: false } ])
    }
    throw new NotFoundException('linea no encontrado');
}
Complexity is 8 It's time to do something...
async allHorarioNotAssigned (id:string){
    const linea = await this.lineModel.findOne({id:id,delete:false})
    if(linea){
        if(linea.horario.length>0){
            const horarioids = linea.horario.map((id)=>{
                return id.toString()
            })
            return await this.horarioModel.find({_id:$in:horarioids}, {delete:false})
        }
        return await this.horarioModel.find({delete:false})
    }
    throw new NotFoundException('linea no encontrado');
}
Complexity is 8 It's time to do something...
async alltarifaNotAssigned (id:string){
    const linea = await this.lineModel.findOne({id:id, delete:false})
    if(linea){
        ...
    }
}

```

Fuente: Elaboración propia

Figura 48. Lista de horario



Fuente: Elaboración propia

Figura 49. Código fuente de lista de horario

```

export class HorarioService {
  async findAll(filters: any) {
    if (filter.place) {
      searchFilters['place'] = { $regex: new RegExp(filter.place, 'i') };
    }
    if (filter.firstout) {
      searchFilters['firstout'] = { $regex: new RegExp(filter.firstout, 'i') };
    }
    if (filter.lastout) {
      searchFilters['lastout'] = { $regex: new RegExp(filter.lastout, 'i') };
    }
    if (filter.days) {
      searchFilters['days'] = { $regex: new RegExp(filter.days, 'i') };
    }
  }
  if (skip !== undefined && limit !== undefined) {
    const result = await this.horarioModel.find(searchFilters).skip(skip).limit(limit).exec();
    const total = await this.horarioModel.countDocuments(searchFilters);
    return { result, total };
  }

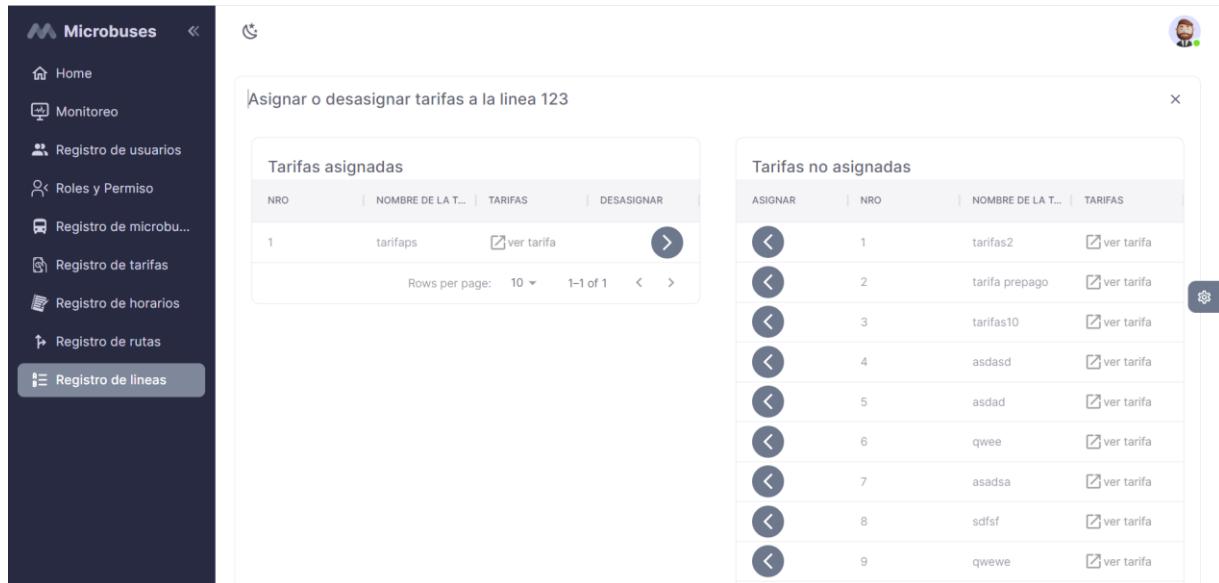
  const result = await this.horarioModel.find(searchFilters);
  const total = await this.horarioModel.countDocuments({ delete: false });
  return { result, total };
}

const result = await this.horarioModel.find({ delete: false });
const total = await this.horarioModel.countDocuments({ delete: false });
return { result, total };
}
  
```

Fuente: Elaboración propia

3.4.10. Asignar Tarifas

Figura 50. Asignar Tarifas



Fuente: Elaboración propia

Figura 51. Código fuente de Asignar Tarifas

```

import { LineaService } from './linea.service';
import { TarifaModel } from '../models/tarifa.model';
import { LineaModel } from '../models/linea.model';

export class LineaService {
    // ...
}

async allTarifaNotAssigned (id:string) {
    const linea = await this.lineaModel.findOne({id:id, delete:false})
    if(linea){
        if(linea.rate.length>0){
            const TarifaIds = linea.rate.map((id)=>{
                return id.toString()
            })
            return await this.tarifaModel.find({_id:$in:TarifaIds}, {delete:false})
        }
        return await this.tarifaModel.find({delete:false})
    }
    throw new NotFoundException('linea no encontrado');
}

Complexity is 4 Everything is cool
async desasignatedTarifa (id:string,desasignedTarifa:AsignedRateDto) {
    const update = await this.lineaModel.findOneAndUpdate({id},{$pull:{rate:$in:desasignedTarifa.rate}}, {new:true})
    if(update){
        return update.populate([ { path: 'rate', match: { delete: false } }])
    }
    throw new NotFoundException('linea no encontrado');
}

Complexity is 4 Everything is cool
async assignedTarifa (id:string,assignedTarifa:AsignedRateDto) {
    const update = await this.lineaModel.findOneAndUpdate({id},{$addToSet:{rate:$each:assignedTarifa.rate}}, {new:true})
    if(update){
        return update.populate([ { path: 'rate', match: { delete: false } }])
    }
    throw new NotFoundException('linea no encontrado');
}

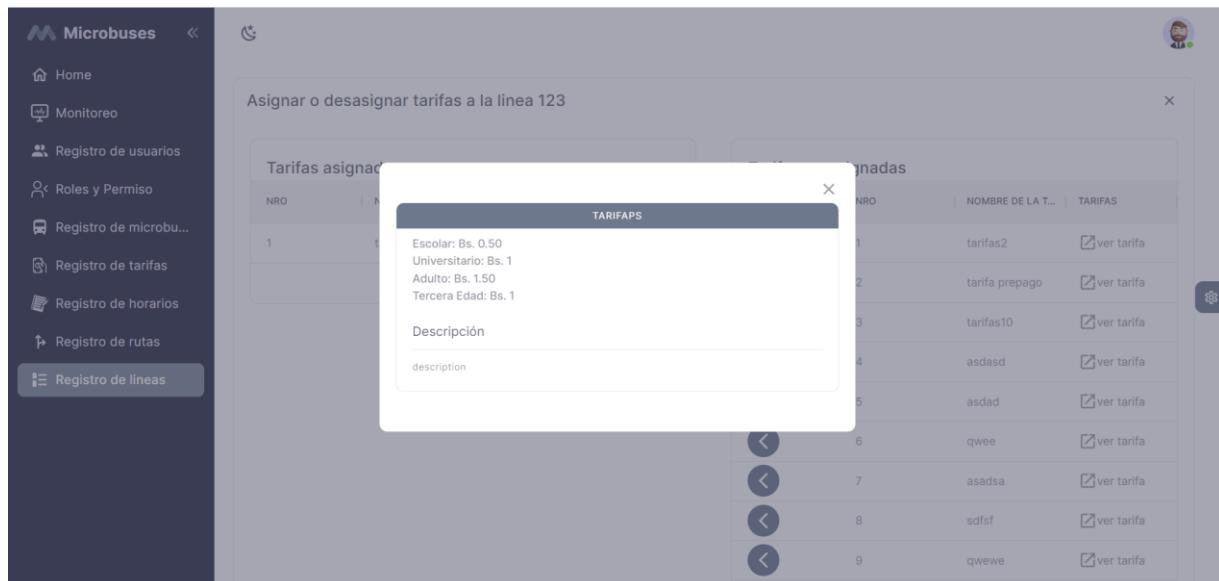
Complexity is 10 It's time to do something...
async allBusNotAssigned(filters) {
    const busAssignedToLinea = await this.lineaModel.find().distinct('buses').exec();
    if (busAssignedToLinea === null) {
        return [];
    }
}

```

You, 5 days ago · Ln 216, Col 6 · Spaces: 2 · UTF-8 · CRLF · { TypeScrip... · ⚡ Spell · ⚡

Fuente: Elaboración propia

Figura 52. Vista previa ver tarifas



Fuente: Elaboración propia

Figura 53. Código fuente vista previa ver tarifas

```

src > rate > rate.service.ts > RateService > findAll
10  export class RateService {
11    async findAll(filters: any) {
12      if (!filters) {
13        const { filter, skip, limit } = filters;
14        const searchFilters = { [Symbol.iterator]: false };
15
16        if (filter) {
17          if (filter.name) {
18            searchFilters['name'] = { $regex: new RegExp(filter.name, 'i') };
19          }
20
21          if (filter.createdAt) {
22            const date = filter.createdAt;
23            const startDate = new Date(`${date}T00:00:00.000Z`);
24            const endDate = new Date(`${date}T23:59:59.999Z`);
25            searchFilters['createdAt'] = {
26              $gte: startDate,
27              $lte: endDate
28            };
29
30        }
31
32        if (skip !== undefined && limit !== undefined) {
33          const result = await this.tarifaModel.find(searchFilters).skip(skip).limit(limit).exec();
34          return result;
35        }
36      }
37
38      return [];
39    }
40  }
41
42  if (skip === undefined && limit === undefined) {
43    const result = await this.tarifaModel.find(searchFilters).skip(skip).limit(limit).exec();
44    return result;
45  }

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS

[Nest] 5584 - 25/06/2024, 19:05:02 LOG [RouterExplorer] Mapped /{componentes:id}, DELETE route +0ms
[Nest] 5584 - 25/06/2024, 19:05:02 LOG [NestApplication] Nest application successfully started +18ms
cliente conectado btpK_w5Y0ucUdgRAAB
cliente conectado btpK_w5Y0ucUdgRAAE
se desconecta cliente btpK_w5Y0ucUdgRAAB
cliente conectado ZKe9jUsgeB2J3MzvJAAAG

Fuente: Elaboración propia

3.4.11. Asignar buses

Figura 54. Asignar buses

MARCA	MODELO	TIPO	PLACA	ASIENTOS	CHOFER	ESTADO	DESASIGNAR
Toyota	1990	Micro	ADS 456	16	Manuel Julian Contreras 1987645	Activo	>

Buses no asignadas							
ASIGNAR	MARCA	MODELO	TIPO	PLACA	Cantidad de Asientos	Chofer	Estado
<	Mitsubishi	1990	Microbus	ASD34	16	No asignado	Activo
<	Nissan	1990	Microbus	ADS 345	16	cafdsafa ASASDA 2342	Activo
<	Nissan	1990	Microbus	ASD 456	16	No asignado	Activo
<	Otra marca	1990	Minibus	Asdf45	16	No asignado	Activo

Fuente: Elaboración propia

Figura 55. Código fuente de Asignar buses

Fuente: Elaboración propia

Figura 56. Asignar rutas

**Microbuses** < 

Home

Monitoreo

Registro de usuarios

Roles y Permisos

Registro de microbuses

Registro de tarifas

Registro de horarios

Registro de rutas

Registro de líneas

Asignar o desasignar rutas a la linea 123

Rutas asignadas		Rutas no asignadas	
NRO	NOMBRE DE LA R...	PREVIA	ASIGNAR
1	ruta de la line...	<input checked="" type="checkbox"/> previa	>
2	ruta ida 012	<input checked="" type="checkbox"/> previa	>
3	lufa	<input checked="" type="checkbox"/> previa	>

Rows per page: 10 ▾ 1–3 of 3 < >

Rutas no asignadas		Rutas asignadas	
ASIGNAR	NRO	NOMBRE DE LA R...	PREVIA
<	1	yo	<input type="checkbox"/> previa
<	2	rutas las band...	<input type="checkbox"/> previa
<	3	x	<input type="checkbox"/> previa
<	4	rutas 32	<input type="checkbox"/> previa

Rows per page: 10 ▾ 1–4 of 4 < >

© 2024, Desarrollador Santos Machaca Lopez ❤ Email santosjared221234@gmail.com

Fuente: Elaboración propia

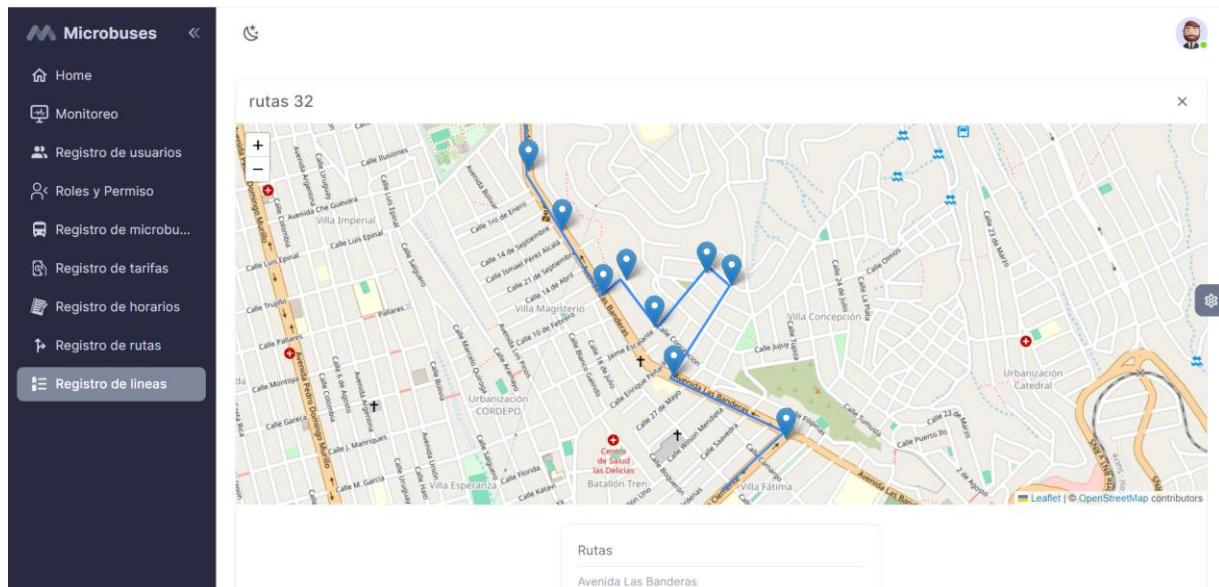
Figura 57: código fuente de asignar rutas

The screenshot shows a developer's workspace with the following details:

- File Explorer:** Shows the project structure under the root folder "linea". The "TRANSPORT-BACKEND" folder is expanded, showing sub-folders like "src", "dto", "entities", "schema", and "locations".
- Code Editor:** Displays the file "linea.service.ts" with several code snippets highlighted in yellow, indicating they are currently selected or being navigated.
- Terminal:** Shows logs from Nest.js and Node.js, indicating successful application startup and connection events.
- Bottom Bar:** Includes tabs for "PROBLEMS", "OUTPUT", "DEBUG CONSOLE", "TERMINAL", "PORTS", and "GITLENS".

Fuente: Elaboración propia

Figura 58: Vista previa rutas



Fuente: Elaboración propia

Figura 59: código fuente de vista previa ver rutas

```

src > linea > linea.service.ts > LineaService > allBusNotAssigned > queryConditions
export class LineaService {
    ...
    async allBusNotAssigned(id: string, assignedBus: AssignedBusDto) {
        ...
    }
}

Complexity is 8 it's time to do something...
async allRoadNotAssigned(id: string) {
    const linea = await this.lineaModel.findOne({ id: id, delete: false })
    if (linea) {
        if (linea.road.length > 0) {
            const roadIds = linea.road.map((id) => {
                return id.toString()
            })
            return await this.roadModel.find({ _id: { $in: roadIds }, delete: false })
        }
        return await this.roadModel.find({ delete: false })
    }
    return new NotFoundException('linea no encontrado')
}

Complexity is 7 it's time to do something...
async assigneBusRoute(id, data: AssignedBusRoadDto) {
    const linea = await this.lineaModel.findOne({ id }).populate('buses')
    if (linea) {
        const filterBus = linea.buses.filter(bus => bus.id === data.busId)
        if (filterBus.length !== 0) {
            const updateBus = await this.busModel.findOneAndUpdate({ id: data.busId }, { road: data.road })
            return await this.lineaModel.findOne({ id }).populate('buses')
        }
        throw new NotFoundException('bus no asignado a la linea')
    }
}

```

Fuente: Elaboración propia

Figura 60: Asignar o desasignar rutas a bus de la línea



Fuente: Elaboración propia

Figura 61: Código fuente de Asignar o desasignar rutas a bus de la línea

```

    export class LineaService {
      async desasignarBus(id: string, desasignedBus: AssignedBusDto) {
        return update.populate([{ path: 'buses', match: { delete: false } }]);
      }
      throw new NotFoundException('linea no encontrado');
    }

    Complexity is 4 Everything is cool!
    async asignarBus(id: string, assignedBus: AssignedBusDto) {
      const update = await this.lineaModel.findOneAndUpdate({ id }, { $addToSet: { buses: { $each: assignedBus.buses } } }, { new: true });
      if (update) {
        return update.populate([{ path: 'buses', match: { delete: false } }]);
      }
      throw new NotFoundException('linea no encontrado');
    }

    Complexity is 8 It's time to do something...
    async allRoadNotAssigned(id: string) {
      const linea = await this.lineaModel.findOne({ id, delete: false });
      if (linea) {
        if (linea.road.length > 0) {
          const roadIds = linea.road.map((id) => {
            return id.toString();
          });
          return await this.roadModel.find({ _id: { $nin: roadIds }, delete: false });
        }
        return await this.roadModel.find({ delete: false });
      }
      throw new NotFoundException('linea no encontrado');
    }
  }

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS

[Nest] 5584 - 25/06/2024, 19:05:02 LOG [RouterExplorer] Mapped {/componentes/:id, DELETE} route +0ms
[Nest] 5584 - 25/06/2024, 19:05:02 LOG [NestApplication] Nest application successfully started +18ms
cliente conectado btpk_w5Y0urUdgRAAB
cliente conectado btpk_w5Y0ucUdrRAAB
se desconecto cliente btpk_w5Y0ucUdrRAAB
cliente conectado ZKe9iUsgE823kZwJAAG

Fuente: Elaboración propia

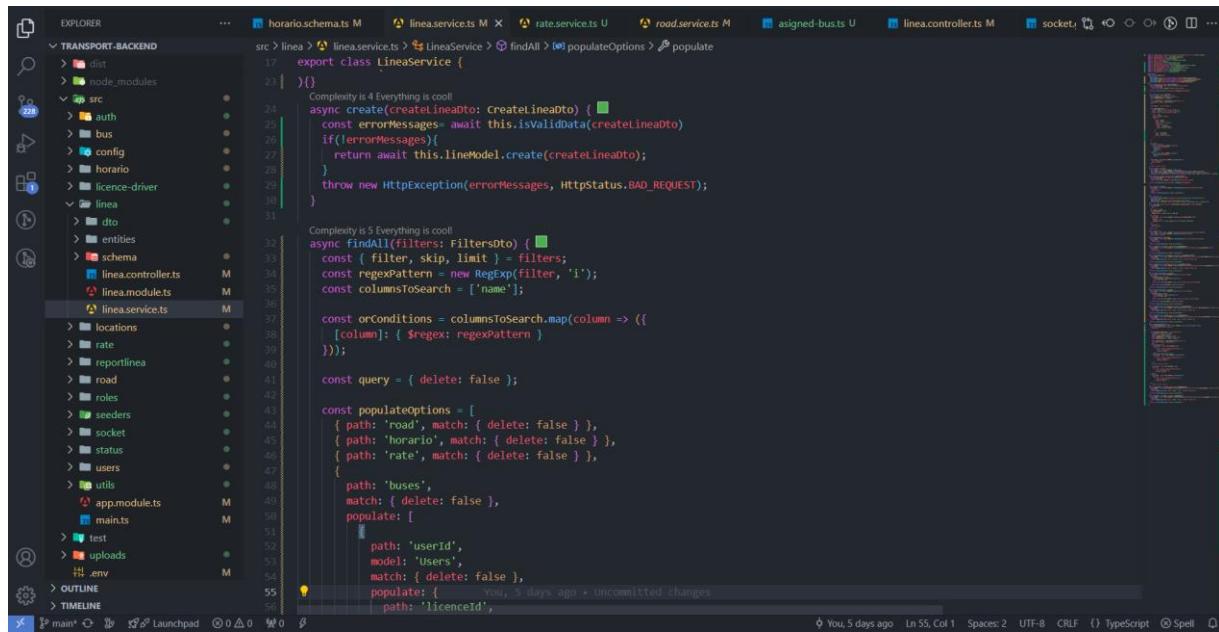
3.4.12. Registrar línea

Figura 62. Registrar línea

NOMBRE DE LA LINEA	RUTAS Y PARADAS	HORARIOS	TARIFAS
11	sasd	ver horario	
012	sasd	ver horario	
F	ruta 09	ver horario	
09	ruta 09	ver horario	
Y	ruta 09	ver horario	
08	sasd	ver horario	
110	ruta 09	ver horario	
J	sasd	ver horario	
H	ruta 09	ver horario	

Fuente: Elaboración propia

Figura 63. Código fuente de Registrar línea



```

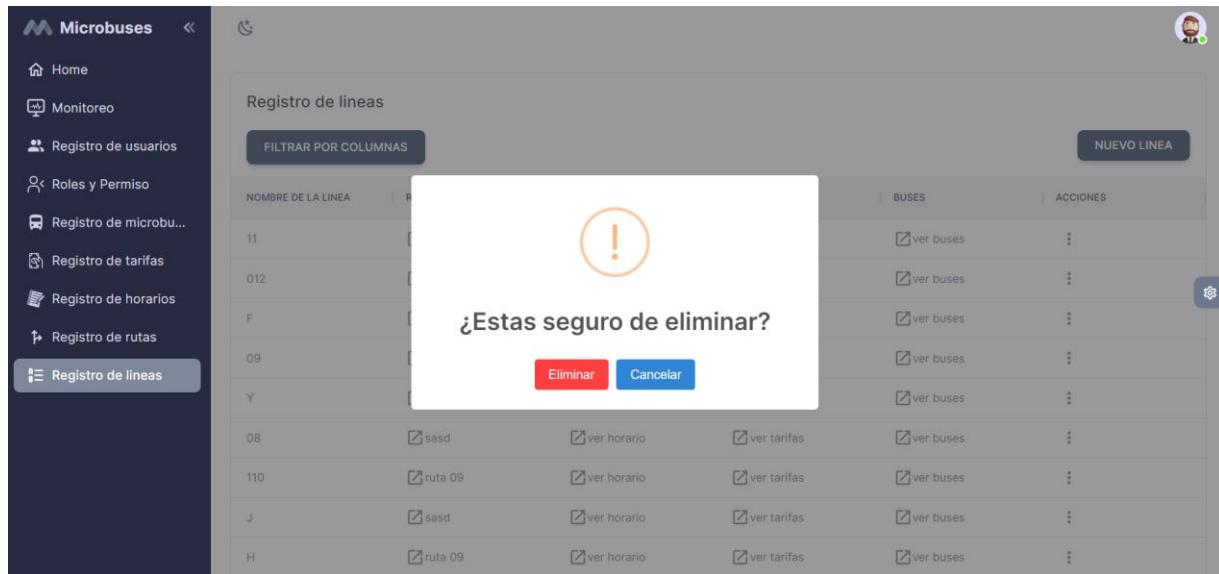
17  export class LineaService {
18
19    constructor() {}
20
21    Complexity is 4 Everything is cool
22    async create(createLineaDto: CreateLineaDto) {
23      const errorMessages = await this.isValidData(createLineaDto)
24      if(errorMessages){
25        return await this.lineaModel.create(createLineaDto);
26      }
27      throw new HttpException(errorMessages, HttpStatus.BAD_REQUEST);
28    }
29
30    Complexity is 5 Everything is cool
31    Complexity is 6 Everything is cool
32    async findAll(filters: FiltersDto) {
33      const { filter, skip, limit } = filters;
34      const regexPattern = new RegExp(filter, 'i');
35      const columnstoSearch = ['name'];
36
37      const orConditions = columnstoSearch.map(column => ({
38        [column]: { $regex: regexPattern }
39      }));
40
41      const query = { delete: false };
42
43      const populateOptions = [
44        { path: 'road', match: { delete: false } },
45        { path: 'horario', match: { delete: false } },
46        { path: 'rate', match: { delete: false } },
47        {
48          path: 'buses',
49          match: { delete: false },
50          populate: [
51            { path: 'userId',
52              model: 'Users',
53              match: { delete: false },
54              populate: {
55                path: 'licenseId'
56              }
57            }
58          ]
59        }
60      ];
61
62      const result = await this.lineaModel.find(query).skip(skip).limit(limit).exec();
63
64      return result;
65    }
66  }
67
68  Complexity is 7 Everything is cool
69  Complexity is 8 Everything is cool
70
71  module.exports = LineaService;

```

Fuente: Elaboración propia

3.4.13. Eliminar líneas

Figura 64. Eliminar líneas



Fuente: Elaboración propia

Figura 65. Código fuente de Eliminar líneas

```

import { LineaService } from './linea.service';
import { UpdateLineaDto } from './dto/update-linea.dto';

export class LineaService {
    update(id: string, updateLineaDto: UpdateLineaDto): Promise<void> {
        const update = await this.lineaModel.findOneAndUpdate({id}, updateLineaDto);
        if(errorMessages){
            return await this.lineaModel.findOneAndDelete({id}, updateLineaDto);
        }
        throw new HttpException(errorMessages, HttpStatus.BAD_REQUEST);
    }

    remove(id: string): Promise<void> {
        const delete = await this.lineaModel.findOneAndDelete({id}, {delete:true}, {new:true});
        if(delete){
            return delete;
        }else{
            throw new NotFoundException('lnea no encontrado');
        }
    }

    async findlinea () {
        return await this.lineaModel.find({delete:false}).populate('road').populate('horario')
        .populate('rate')
        .populate([{path:'buses',model:'Users', populate:[{path:'userId',model:'Users', populate:[{path:'licenceld', model:'licenceDriver'}], path:'locationid',model:'Locations'}]}])
    }
}

Complexity is 4 Everything is cool!
async remove(id: string): Promise<void> {
    const delete = await this.lineaModel.findOneAndDelete({id}, {delete:true}, {new:true});
    if(delete){
        return delete;
    }else{
        throw new NotFoundException('lnea no encontrado');
    }
}

Complexity is 7 Its time to do something...
async isValidData(createLineaDto: CreateLineaDto | UpdateLineaDto, update?: boolean): Promise<boolean> {
    const message = {
        name: ''
    }
    let iserror = false
    if(!createLineaDto.name){
        iserror=true
        message.name='el campo Lnea es requerido'
    }
    if(update){
        const data = await this.lineaModel.findOne({name:createLineaDto.name});
        if(data){
            iserror= true
        }
    }
    return !iserror
}

```

Fuente: Elaboración propia

3.4.14. Registrar Rutas y paradas

Figura 66. Registrar Rutas y paradas

NOMBRES DE RUTAS	FECHA DE CREACIÓN	RUTAS	ESTADO	ACCIONES
sasd	29/05/2024	[ver rutas]	Activo	...
ruta 09	02/06/2024	[ver rutas]	Activo	...
ruta leina E	13/06/2024	[ver rutas]	Activo	...
ruta linea 012	13/06/2024	[ver rutas]	Activo	...
ruta 08	13/06/2024	[ver rutas]	Activo	...

Fuente: Elaboración propia

Figura 67. Código fuente de Registrar Rutas y paradas

```

import { RoadModel } from '../models/Road';
import { Filter } from 'mongoose';

export class RoadService {
    Complexity is 6 It's time to do something...
    async findAll(filters: Filter) {
        const { filter, skip, limit } = filters;
        const regexPattern = new RegExp(filter, 'i');
        const columnstoSearch = ['name', 'rates'];

        const orConditions = columnstoSearch.map(column => (
            [column]: { $regex: regexPattern }
        ));

        if (skip && limit) {
            const result = await this.roadModel.find({$or: orConditions}, {delete: false}).skip(skip).limit(limit).exec();
            const total = await this.roadModel.countDocuments({$or: orConditions}, {delete: false});
            return { result, total };
        }

        const result = await this.roadModel.find({$or: orConditions}, {delete: false}).exec();
        const total = await this.roadModel.countDocuments({$or: orConditions}, {delete: false});
        return { result, total };
    }

    async findOne(id: string) {
        return await this.roadModel.findOne({id:id});
    }

    Complexity is 4 Everything is cool!
    async update(id: string, updateRoadDto: UpdateRoadDto) {
        const errorMessage = await this.isValidData(updateRoadDto);
        if(errorMessage){
            return await this.roadModel.findOneAndUpdate({id},updateRoadDto);
        }
        throw new HttpException(errorMessage, HttpStatus.BAD_REQUEST);
    }

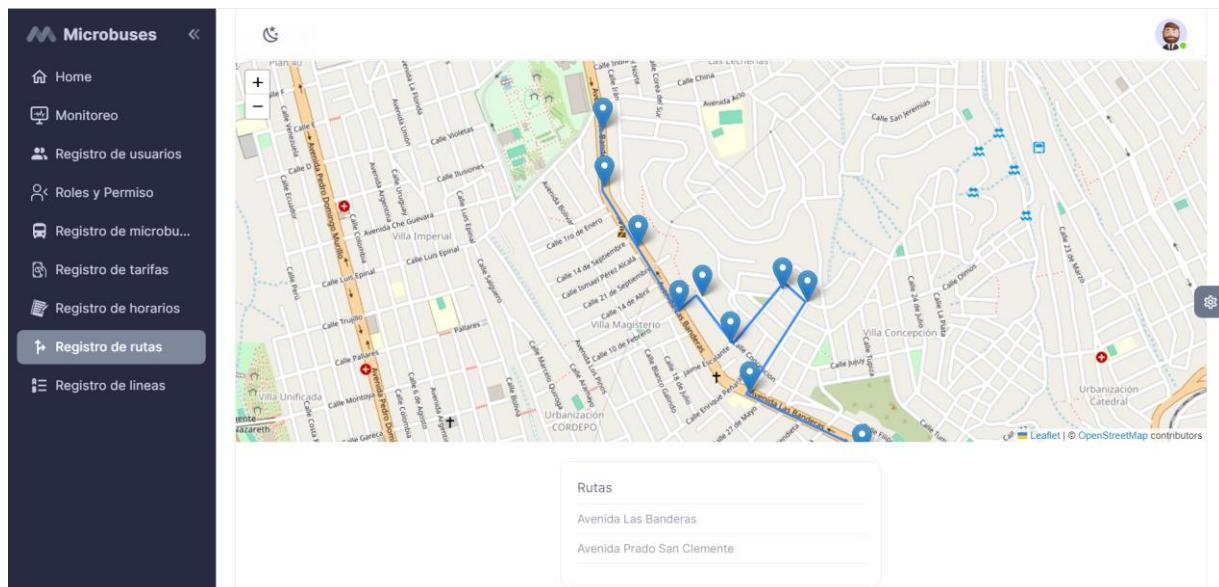
    async remove(id: string) {
        const delete = await this.roadModel.findOneAndUpdate({id},{delete:true},{new:true});
        You, 3 months ago + add road
    }
}

```

Fuente: Elaboración propia

3.4.15. Listar Rutas

Figura 68. Ver rutas



Fuente: Elaboración propia

Figura 69. Código fuente de Listar Rutas

```

src > road > road.service.ts > RoadService > remove
  export class RoadService {
    async findAll(filters: FiltersDto) {
      const orConditions = columnsToSearch.map(column => ({
        column: ( $regex: regexPattern )
      }));
      if (skip && limit) {
        const result = await this.roadModel.find({$or:orConditions,delete:false}).skip(skip).limit(limit).exec();
        const total = await this.roadModel.countDocuments({$or:orConditions,delete:false});
        return { result, total };
      }
      const result = await this.roadModel.find({$or:orConditions,delete:false}).exec();
      const total = await this.roadModel.countDocuments({$or:orConditions,delete:false});
      return { result, total };
    }

    async findOne(id: string) {
      return await this.roadModel.findOne({id:id});
    }

    Complexity is 4 Everything is cool
    async update(id: string, updateRoadDto: UpdateRoadDto) {
      const errorMessage = await this.isValidData(updateRoadDto);
      if(errorMessage){
        return await this.roadModel.findOneAndUpdate({id},updateRoadDto);
      }
      throw new HttpException(errorMessage, HttpStatus.BAD_REQUEST);
    }

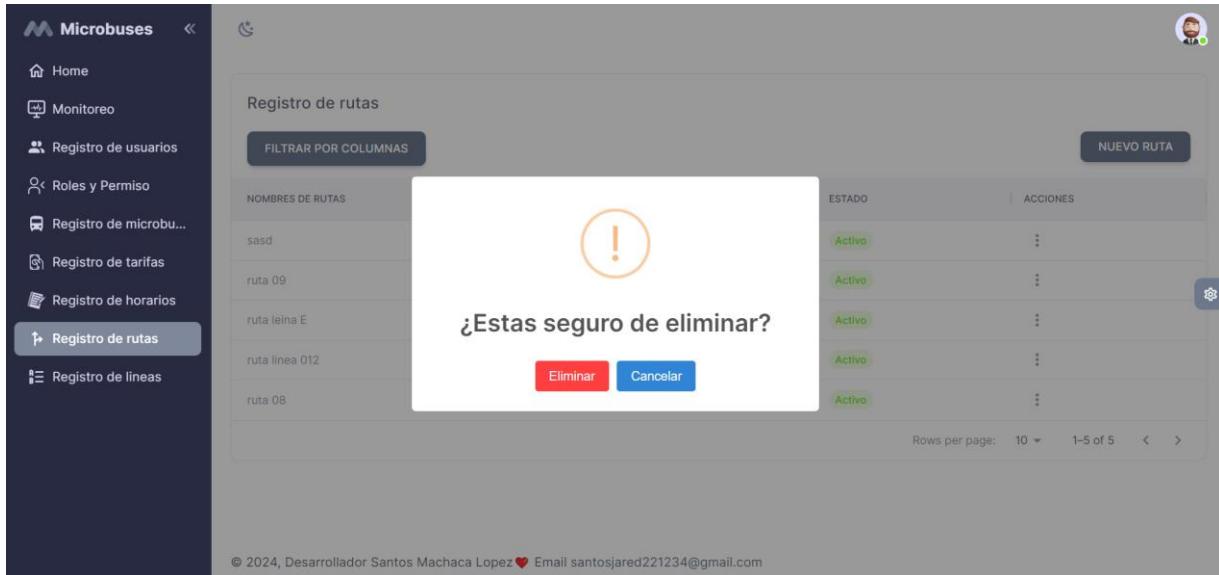
    Complexity is 4 Everything is cool
    async remove(id: string) {
      const delete = await this.roadModel.findOneAndDelete({id},{delete:true},{new:true});
      if(delete){
        return delete;
      }else{
        throw new NotFoundException('tarifa no encontrado');
      }
    }
  }

```

Fuente: Elaboración propia

3.4.16. Eliminar rutas

Figura 70. Eliminar rutas



Fuente: Elaboración propia

Figura 71. Código fuente de Eliminar rutas

```

src > road > road.service.ts > RoadService > remove
export class RoadService {
    Complexity is 4 Everything is cool
    async update(id: string, updateRoadDto: UpdateRoadDto) {
        const errorMessages = await this.isValidData(updateRoadDto)
        if(errorMessages){
            return await this.roadModel.findOneAndUpdate({id},updateRoadDto);
        }
        throw new HttpException(errorMessages, HttpStatus.BAD_REQUEST);
    }

    Complexity is 4 Everything is cool!
    async remove(id: string) {
        const delete = await this.roadModel.findOneAndUpdate({id},{delete:true},{new:true})
        if(delete){
            return delete;
        }else{
            throw new NotFoundException('tarifa no encontrado');
        }
    }

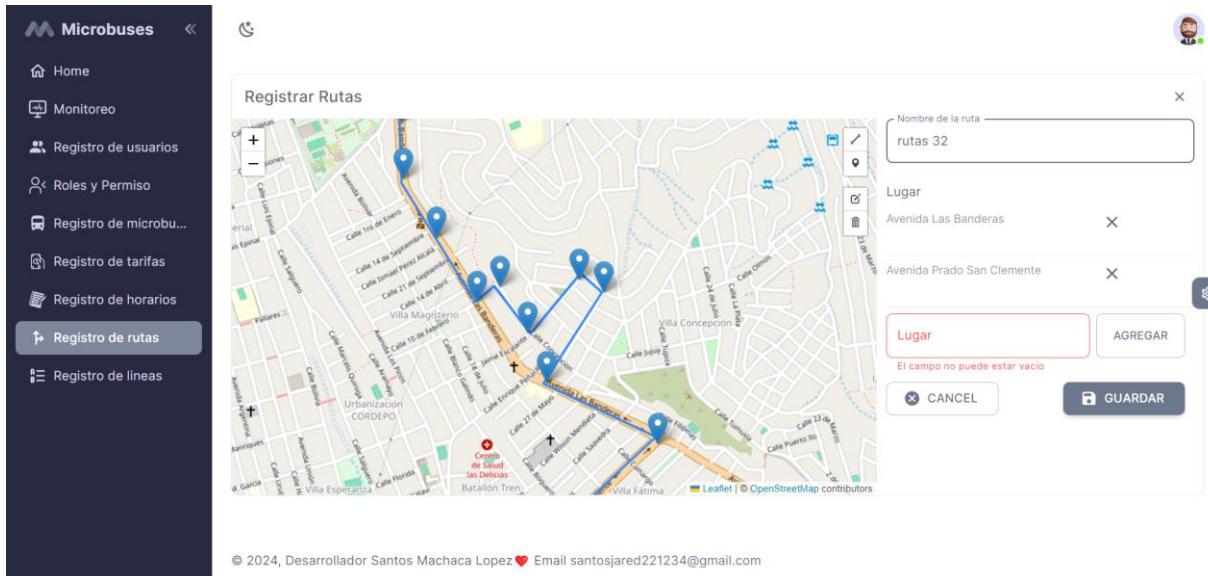
    Complexity is 5 Everything is cool!
    async isValidData(createRoadDto: CreateRoadDto | UpdateRoadDto, update?:boolean){
        const message = {
            name:''
        }
        let isError = false
        if(!createRoadDto.name){
            isError = true
            message.name='el campo nombre de la ruta es requerido'
        }
        if(isError){
            return message
        }
        return null
    }
}

```

Fuente: Elaboración propia

3.4.17. Registro de rutas

Figura 72. Registro de rutas



Fuente: Elaboración propia

Figura 73. Código fuente de Registro de rutas

The screenshot shows a developer's workspace with several tabs open in a browser-based IDE. The tabs include 'EXPLORER', 'horario.schemas M', 'linea.service.ts M', 'rate.service.ts U', 'road.service.ts M X', 'asigned-bus.ts U', 'linea.controller.ts M', 'socket', and 'socket'. The 'road.service.ts' tab is currently active, displaying the following TypeScript code:

```
src > road > road.service.ts > ...
You, 2 weeks ago | author (You)
1 import { HttpException, HttpStatus, Injectable, NotFoundException } from '@nestjs/common';
2 import { CreateRoadDto } from './dto/create-road.dto';
3 import { UpdateRoadDto } from './dto/update-road.dto';
4 import { Road } from './entities/road.entity';
5 import { Model } from 'mongoose';
6 import { RoadDocument } from './schema/road.schema';
7 import { InjectModel } from '@nestjs/mongoose';
8 import { FiltersDto } from 'src/utils/filters.dto';
9
10 You, 2 weeks ago | author (You) | Complexity is 6 It's time to do something...
11 @Injectable()
12 export class RoadService {
13   constructor(@InjectModel(Road.name) private readonly roadModel: Model<RoadDocument>){}
14
15   Complexity is 4 Everything is cool
16   async create(createRoadDto: CreateRoadDto) {
17     const errorMessages = await this.isValidData(createRoadDto);
18     if(errorMessages){
19       throw new HttpException(errorMessages, HttpStatus.BAD_REQUEST);
20     }
21
22   Complexity is 6 It's time to do something...
23   async findAll(filters: FiltersDto) {
24     const { filter, skip, limit } = filters;
25     const regexPattern = new RegExp(filter, 'i');
26     const columnsToSearch = ['name', 'rates'];
27
28     const orConditions = columnsToSearch.map(column => ({
29       [column]: { $regex: regexPattern }
30     }));
31
32     if (skip && limit) {
33       const result = await this.roadModel.find({$or:orConditions}, {delete:false}).skip(skip).limit(limit).exec();
34       const total = await this.roadModel.countDocuments({$or:orConditions}, {delete:false});
35       return { result, total }
36     }
37
38   const result = await this.roadModel.find({$or:orConditions}, {delete:false}).exec();
39
40   }
41 }
```

Fuente: Elaboración propia

3.4.18. Listar tarifas

Figura 74. Listar tarifas



Microbuses

Home

Monitoreo

Registro de usuarios

Roles y Permisos

Registro de microbuses

Registro de tarifas

Registro de horarios

Registro de rutas

Registro de líneas

Registro de tarifas

CERRAR FILTRADO

NOMBRE DE TARIFAS

Modelo

FILTRAR

RESTABLECER

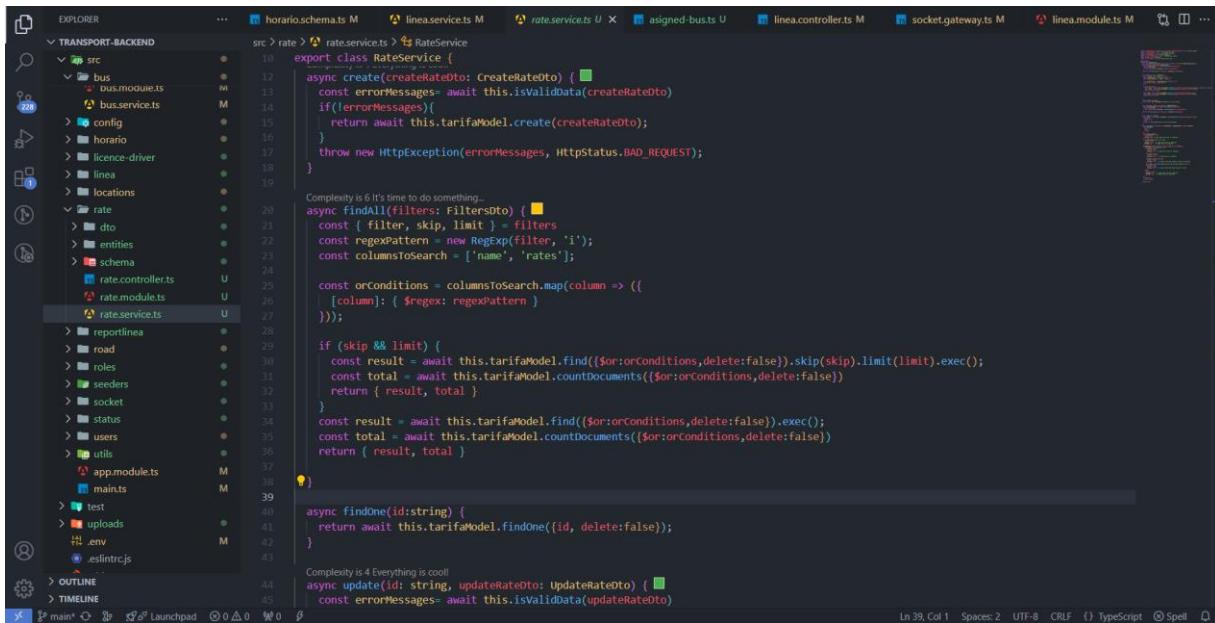
NUEVO TARIFA

NOMBRE DE TARIFAS	FECHA DE CREACIÓN	TARIFAS	ACCIONES
Tarifa 012	25/05/2024	<input type="checkbox"/> ver tarifas	...
Tarifa 12	13/06/2024	<input type="checkbox"/> ver tarifas	...
tarifa card	13/06/2024	<input type="checkbox"/> ver tarifas	...
tarifa con tarjeta	13/06/2024	<input type="checkbox"/> ver tarifas	...
tarifa wikisis	13/06/2024	<input type="checkbox"/> ver tarifas	...

Rows per page: 10 ▾ 1-5 of 5 < >

Fuente: Elaboración propia

Figura 75. Código fuente de Listar tarifas



```
src > rate > RateService
10  export class RateService {
11    ...
12    ...
13    ...
14    ...
15    ...
16    ...
17    ...
18    ...
19    ...
20    ...
21    ...
22    ...
23    ...
24    ...
25    ...
26    ...
27    ...
28    ...
29    ...
30    ...
31    ...
32    ...
33    ...
34    ...
35    ...
36    ...
37    ...
38    ...
39    ...
40    ...
41    ...
42    ...
43    ...
44    ...
45    ...

Complexity is 6 it's time to do something...
async findAll(filters: FiltersDto) {
  const { filter, skip, limit } = filters;
  const regexPattern = new RegExp(filter, 'i');
  const columnsToSearch = ['name', 'rates'];

  const orConditions = columnsToSearch.map(column => ({
    [column]: { $regex: regexPattern }
  }));

  if (skip && limit) {
    const result = await this.tarifaModel.find({$or:orConditions, delete:false}).skip(skip).limit(limit).exec();
    const total = await this.tarifaModel.countDocuments({$or:orConditions, delete:false});
    return { result, total };
  }
  const result = await this.tarifaModel.find({$or:orConditions, delete:false}).exec();
  const total = await this.tarifaModel.countDocuments({$or:orConditions, delete:false});
  return { result, total };
}

Complexity is 4 Everything is cool
async findOne(id:string) {
  return await this.tarifaModel.findOne({id, delete:false});
}

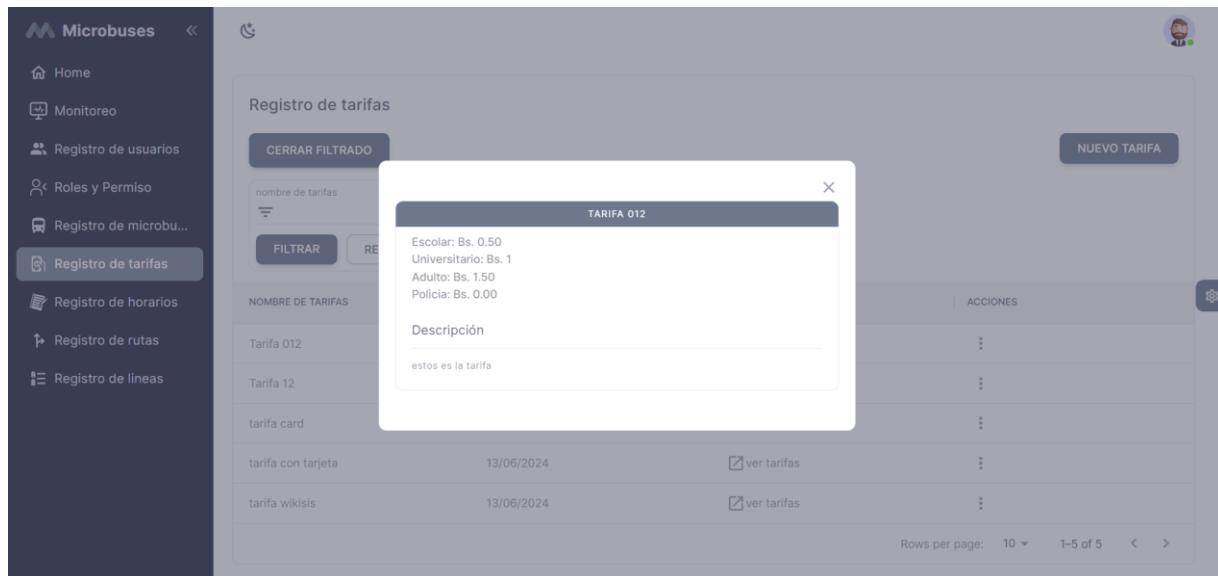
Complexity is 4 Everything is cool
async update(id: string, updateRateDto: UpdateRateDto) {
  const errorMessages = await this.isValidData(updateRateDto);
  if(errorMessages.length > 0) {
    throw new HttpException(errorMessages, HttpStatus.BAD_REQUEST);
  }

  const tarifa = await this.tarifaModel.findById(id);
  if(tarifa) {
    tarifa.name = updateRateDto.name;
    tarifa.rates = updateRateDto.rates;
    tarifa.description = updateRateDto.description;
    tarifa.date = updateRateDto.date;
    tarifa.type = updateRateDto.type;
    tarifa.discount = updateRateDto.discount;
    tarifa.validFrom = updateRateDto.validFrom;
    tarifa.validUntil = updateRateDto.validUntil;
    tarifa.isActive = updateRateDto.isActive;
    tarifa.isDeleted = updateRateDto.isDeleted;
    await this.tarifaModel.save(tarifa);
  }
  return tarifa;
}
```

Fuente: Elaboración propia

3.4.19. Listar tipos de tarifa

Figura 76. Listar tipos de tarifa



Fuente: Elaboración propia

Figura 77. Código fuente de Listar tipos de tarifa



```
src > rate > RateService.ts - RateService
  export class RateService {
    ...
    async create(createRateDto: CreateRateDto) {
      const errorMessages = await this.isValidData(createRateDto);
      if (errorMessages) {
        return await this.tarifaModel.create(createRateDto);
      }
      throw new HttpException(errorMessages, HttpStatus.BAD_REQUEST);
    }
  }

Complexity is 6 It's time to do something...
async findAll(filters: FiltersDto): Promise<Rate[]> {
  const { filter, skip, limit } = filters;
  const regexPattern = new RegExp(filter, 'i');
  const columnsToSearch = ['name', 'rates'];

  const orConditions = columnsToSearch.map(column => ({
    [column]: { $regex: regexPattern }
  }));

  if (skip && limit) {
    const result = await this.tarifaModel.find({$or:orConditions,delete:false}).skip(skip).limit(limit).exec();
    const total = await this.tarifaModel.countDocuments({$or:orConditions,delete:false});
    return { result, total };
  }

  const result = await this.tarifaModel.find({$or:orConditions,delete:false}).exec();
  const total = await this.tarifaModel.countDocuments({$or:orConditions,delete:false});
  return { result, total };
}

async findOne(id:string): Promise<Rate> {
  return await this.tarifaModel.findOne({id, delete:false});
}

Complexity is 4 Everything is cool
async update(id: string, updateRateDto: UpdateRateDto): Promise<Rate> {
  const errorMessages = await this.isValidData(updateRateDto);
  ...
}
```

Fuente: Elaboración propia

3.4.20. Registrar Tarifas

Figura 78. Registrar Tarifas

The screenshot shows the Microbuses application interface. On the left sidebar, there are several navigation items: Home, Monitoreo, Registro de usuarios, Roles y Permisos, Registro de microbuses, Registro de tarifas (which is currently selected and highlighted in blue), Registro de horarios, Registro de rutas, and Registro de líneas.

The main content area displays a "Registro de tarifas" page. It features a search bar with fields for "nombre de tarifas" and "Modelo", and buttons for "FILTRAR" and "RESTABLECER". Below this is a table listing five tarifas:

NOMBRE DE TARIFAS	FECHA DE CREACIÓN
Tarifa 012	25/05/2024
Tarifa 12	13/06/2024
tarifa card	13/06/2024
tarifa con tarjeta	13/06/2024
tarifa wikisis	13/06/2024

To the right, a modal window titled "Registro de Tarifas" is open, titled "Agregar Tarifas". It contains a table for adding new tarifas:

Tipo Tarifa	Tarifa	X
Escolar	Bs. 0.50	X
Universitario	Bs. 1	X
Adulto	Bs. 1.50	X
Tercera Edad	Bs. 1	X

Below the table is a red-bordered input field labeled "Nombre de Tarifa" with the placeholder "el campo Nombre de Tarifa es requerido". There is also a "Descripción" input field, a "CANCELAR" button, and a "GUARDAR" button.

Fuente: Elaboración propia

Figura 79. Código fuente de Registrar Tarifas

```

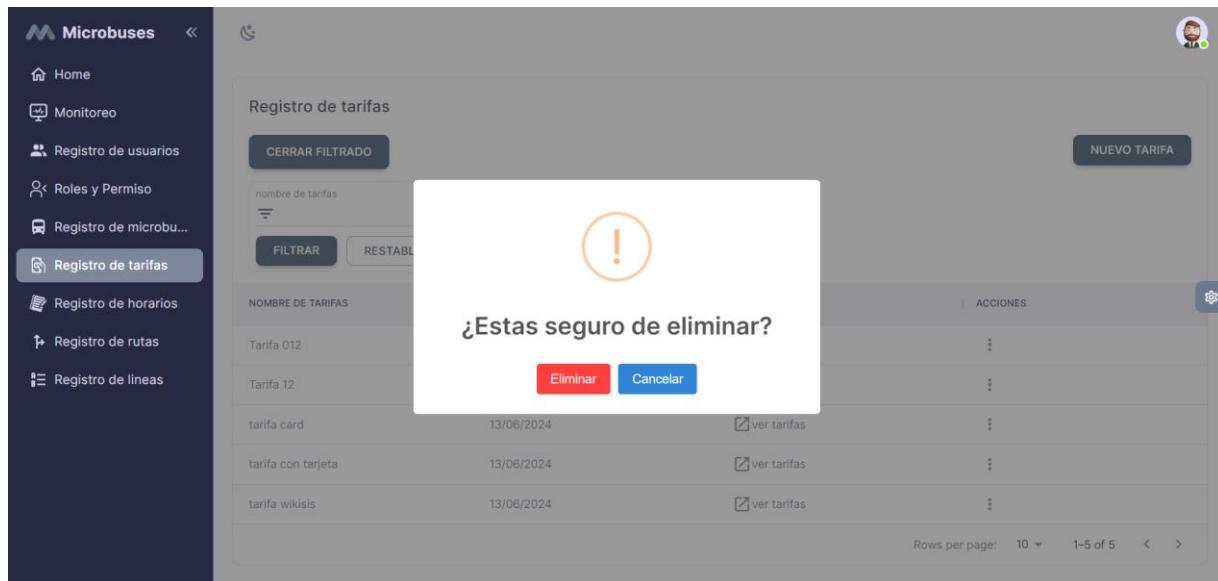
1 import { HttpException, HttpStatus, Injectable, NotFoundException } from '@nestjs/common';
2 import { CreateRateDto } from './dto/create-rate.dto';
3 import { UpdateRateDto } from './dto/update-rate.dto';
4 import { InjectModel } from '@nestjs/mongoose';
5 import { Rate, RateDocument } from './schema/rate.schema';
6 import { Model } from 'mongoose';
7 import { FiltersDto } from 'src/utils/filters.dto';
8
9 Complexity is 13 You must be kidding
10 @Injectable()
11 export class RateService {
12   constructor(@InjectModel(Rate.name) private readonly tarifaModel: Model<RateDocument>){}
13
14   Complexity is 4 Everything is cool
15   async create(createRateDto: CreateRateDto): Promise<Rate> {
16     const errorMessages = await this.isValidData(createRateDto);
17     if(errorMessages){
18       return await this.tarifaModel.create(createRateDto);
19     }
20     throw new HttpException(errorMessages, HttpStatus.BAD_REQUEST);
21
22   Complexity is 6 It's time to do something.
23   async findAll(filters: FiltersDto): Promise<Rate[]> {
24     const { filter, skip, limit } = filters;
25     const regexPattern = new RegExp(filter, 'i');
26     const columnstoSearch = ['name', 'rates'];
27
28     const orconditions = columnstoSearch.map(column => {
29       | (column): { $regex: string } |
30     });
31
32     if (skip && limit) {
33       const result = await this.tarifaModel.find({$or:orconditions}, {delete:false}).skip(skip).limit(limit).exec();
34       const total = await this.tarifaModel.countDocuments({$or:orConditions}, {delete:false});
35       return { result, total }
36     }
37     const result = await this.tarifaModel.find({$or:orConditions}, {delete:false}).exec();
38   }
39
40   Complexity is 10 You must be kidding
41   async delete(id: string): Promise<Rate> {
42     const rate = await this.tarifaModel.findById(id);
43     if (!rate) {
44       throw new NotFoundException(`Rate with id ${id} not found`);
45     }
46     await this.tarifaModel.deleteOne({ _id: id });
47     return rate;
48   }
49
50   Complexity is 10 You must be kidding
51   async update(id: string, updateRateDto: UpdateRateDto): Promise<Rate> {
52     const rate = await this.tarifaModel.findById(id);
53     if (!rate) {
54       throw new NotFoundException(`Rate with id ${id} not found`);
55     }
56     Object.assign(rate, updateRateDto);
57     await this.tarifaModel.save();
58     return rate;
59   }
60
61   Complexity is 10 You must be kidding
62   async findByName(name: string): Promise<Rate> {
63     const rate = await this.tarifaModel.findOne({ name });
64     if (!rate) {
65       throw new NotFoundException(`Rate with name ${name} not found`);
66     }
67     return rate;
68   }
69
70   Complexity is 10 You must be kidding
71   async findByNameAndCard(card: string): Promise<Rate> {
72     const rate = await this.tarifaModel.findOne({ name: card });
73     if (!rate) {
74       throw new NotFoundException(`Rate with name ${card} not found`);
75     }
76     return rate;
77   }
78
79   Complexity is 10 You must be kidding
80   async findByNameAndTarjeta(tarjeta: string): Promise<Rate> {
81     const rate = await this.tarifaModel.findOne({ name: tarjeta });
82     if (!rate) {
83       throw new NotFoundException(`Rate with name ${tarjeta} not found`);
84     }
85     return rate;
86   }
87
88   Complexity is 10 You must be kidding
89   async findByNameAndWikisis(wikisis: string): Promise<Rate> {
90     const rate = await this.tarifaModel.findOne({ name: wikisis });
91     if (!rate) {
92       throw new NotFoundException(`Rate with name ${wikisis} not found`);
93     }
94     return rate;
95   }
96
97   Complexity is 10 You must be kidding
98   async findByNameAndTarjetaWikisis(card: string, tarjeta: string, wikisis: string): Promise<Rate> {
99     const rate = await this.tarifaModel.findOne({ name: card, tarjeta, wikisis });
100    if (!rate) {
101      throw new NotFoundException(`Rate with name ${card}, tarjeta ${tarjeta} and wikisis ${wikisis} not found`);
102    }
103    return rate;
104  }
105}

```

Fuente: Elaboración propia

3.4.21. Eliminar Tarifas

Figura 80. Eliminar Tarifas



Fuente: Elaboración propia

Figura 81. Código fuente de Eliminar Tarifas

```

src > rate > rate.service.ts > RateService > isValidData > createRateDto.rates.map() callback
10   export class RateService {
11     ...
12   }
13   Complexity is 4 Everything is cool!
14   async remove(id: string) { 
15     const delete = await this.tarifaModel.findOneAndUpdate({id},{delete:true},{new:true})
16     if(delete){
17       return delete
18     }else{
19       throw new NotFoundException('tarifa no encontrado')
20     }
21   }
22   Complexity is 13 You must be kidding
23   async isValidData(createRateDto: CreateRateDto | UpdateRateDto, update?:boolean) {
24     const message = {
25       name:'',
26       tipo:'',
27       tarifa:''
28     }
29     let isError = false
30     if(!createRateDto.name){
31       isError = true
32       message.name = 'el campo Nombre de Tarifa es requerido'
33     }
34     if(createRateDto.rates.length === 0){
35       isError = true
36       message.tipo = 'el campo Tipo tarifa es requerido'
37       message.tipo = 'el Campo Tarifa es requerido'
38     }
39     Complexity is 7 It's time to do something...
40     createRateDto.rates.map((values:{tipo:string,tarifa:string}))=>{ 
41       if(values.tarifa && values.tipo){
42         if(!values.tipo){
43           isError = true
44           message.tipo = 'el campo Tipo tarifa es requerido'
45         }
46         if(!values.tarifa){
47           isError = true
48         }
49       }
50     }
51   }
52   ...
53   ...
54   ...
55   ...
56   ...
57   ...
58   ...
59   ...
60   ...
61   ...
62   ...
63   ...
64   ...
65   ...
66   ...
67   ...
68   ...
69   ...
70   ...
71   ...
72   ...
73   ...
74   ...
75   ...
76   ...
77   ...
78   ...
79   ...
80   ...
81   ...
82   ...
83   ...

```

Fuente: Elaboración propia

3.4.22. Listar horarios

Figura 82. Listar horarios

NOMBRE DE HORARIO	LUGAR DE SALIDA	H. PRIM. SALIDA	H. ÚLT. SALIDA	DÍAS	ACCIONES
Horario vuelta 012	villa Santiago	00:00	03:00	<input type="checkbox"/> ver días	⋮
horario de linea F	nueva terminal	02:00	06:02	<input type="checkbox"/> ver días	⋮
horario Ida	Calle salta	00:00	05:00	<input type="checkbox"/> ver días	⋮
ida 012	villa copacabana	00:00	05:00	<input type="checkbox"/> ver días	⋮
Horario vuelta	calle santa tereza	05:00	23:00	<input type="checkbox"/> ver días	⋮

Fuente: Elaboración propia

Figura 83. Código fuente de Listar horarios

```

src > horario > HorarioService.ts
export class HorarioService {
    ...
    async findAll(filters: FiltersDto) {
        const { filter, skip, limit } = filters;
        const regexPattern = new RegExp(filter, 'i');
        const columnstoSearch = ['name', 'place','firstOut','lastOut','days','otherDay'];

        const orConditions = columnstoSearch.map(column => {
            [column]: { $regex: regexPattern }
        });

        if (skip && limit) {
            const result = await this.horarioModel.find({$or:orConditions,delete:false}).skip(skip).limit(limit).exec();
            const total = await this.horarioModel.countDocuments({$or:orConditions,delete:false})
            return { result, total }
        }

        const result = await this.horarioModel.find({$or:orConditions,delete:false}).exec();
        const total = await this.horarioModel.countDocuments({$or:orConditions,delete:false})
        return { result, total }
    }

    findOne(id: string) {
        return this.horarioModel.findOne({id, delete:false});
    }

    Complexity is 4 Everything is cool
    async update(id: string, updateHorarioDto: UpdateHorarioDto) {
        const errorMessages= await this.isValidData(updateHorarioDto)
        if(errorMessages){
            return await this.horarioModel.findOneAndUpdate({_id},updateHorarioDto);
        }
        throw new HttpException(errorMessages, HttpStatus.BAD_REQUEST);
    }

    Complexity is 4 Everything is cool
    async remove(id: string) {
        You, 3 weeks ago * Uncommitted changes
        const delet = await this.horarioModel.findOneAndUpdate({_id},{delete:true},{new:true})
    }
}

```

Fuente: Elaboración propia

3.4.23. Listado de días

Figura 84. Listado de días

Día	Horarios de operación	frecuencia (min)
Lunes	23:00 - 04:00	15
Martes	23:00 - 04:00	15
Miércoles	23:00 - 04:00	15

OPERACION REGULAR DESDE 23:00 A 04:00 DE LA MAÑANA

NOMBRE DE HORARIO
horario 102 lda
horario de prueba
serwrs
sdfsfsdf
asdad
werr
asdadas
asfafa

Descripción

El sistema de registro de horarios es una herramienta eficiente y fácil de usar, diseñada para gestionar y monitorear los tiempos de entrada y salida del personal. Este sistema permite un seguimiento detallado y preciso de las horas trabajadas, facilitando la administración del tiempo y la planificación de recursos.

Fuente: Elaboración propia

Figura 85. Código fuente de Listado de días

```

10 export class HorarioService {
11   ...
12   ...
13   ...
14   ...
15   ...
16   ...
17   ...
18   ...
19   ...
20   ...
21   ...
22   ...
23   ...
24   ...
25   ...
26   ...
27   ...
28   ...
29   ...
30   ...
31   ...
32   ...
33   ...
34   ...
35   ...
36   ...
37   ...
38   ...
39   ...
40   ...
41   ...
42   ...
43   ...
44   ...
45   ...
46   ...
47   ...
48   ...
49   ...
50   ...
51   ...
52   ...
53   ...

```

Complexity is 4 Everything is cool!

```

async update(id: string, updateHorarioDto: UpdateHorarioDto) {
  const errorMessages = await this.isValidData(updateHorarioDto);
  if(errorMessages){
    return await this.horarioModel.findOneAndUpdate({id},updateHorarioDto);
  }
  throw new HttpException(errorMessages, HttpStatus.BAD_REQUEST);
}

async remove(id: string) {
  const delet = await this.horarioModel.findOneAndUpdate({id},{delete:true},{new:true})
}

```

You, 3 weeks ago * Uncommitted changes

Fuente: Elaboración propia

3.4.24. Registro de horario

Figura 86. Registro de horario

NOMBRE DE HORARIO	LUGAR DE SALIDA	H. PRIM. SALIDA	H. ÚLT. SALIDA
Horario vuelta 012	villa Santiago	00:00	03:00
horario de linea F	nueva terminal	02:00	06:00
horario ida	Calle salta	00:00	05:00
ida 012	villa copacabana	00:00	05:00
Horario vuelta	calle santa tereza	05:00	23:00

Fuente: Elaboración propia

Figura 87. Código fuente de Registro de horario

```
src/horario/horario.service.ts > ...
You, 3 weeks ago | 1 author (You)
1 | import { HttpException, HttpStatus, Injectable, NotFoundException } from '@nestjs/common';
2 | import { CreateHorarioDto } from './dto/create-horario.dto';
3 | import { UpdateHorarioDto } from './dto/update-horario.dto';
4 | import { InjectModel } from '@nestjs/mongoose';
5 | import { Horario, HorarioDocument } from './schema/horario.schema';
6 | import { Model } from 'mongoose';
7 | import { FiltersDto } from 'src/utils/filters.dto';
8 |
9 | You, 3 weeks ago | 1 author (You) | Complexity is 11 You must be kidding
10 | @Injectable()
11 | export class HorarioService {
12 |   constructor(@InjectModel(Horario.name) private readonly horarioModel: Model<HorarioDocument>){}
13 |   Complexity is 4 Everything is cool
14 |   async create(createHorarioDto: CreateHorarioDto): Promise<HorarioDocument> {
15 |     const errorMessages: string[] = await this.isEventDataValid(createHorarioDto);
16 |     if(errorMessages.length) {
17 |       return await this.horarioModel.create(createHorarioDto);
18 |     }
19 |     throw new HttpException(errorMessages, HttpStatus.BAD_REQUEST);
20 |   }
21 |
22 |   Complexity is 6 It's time to do something...
23 |   async findAll(filters: FiltersDto): Promise<HorarioDocument[]> {
24 |     const { filter, skip, limit } = filters;
25 |     const regexPattern = new RegExp(filter, 'i');
26 |     const columnsToSearch = ['name', 'place','firstOut','lastOut','days','otherDay'];
27 |
28 |     const orConditions = columnsToSearch.map(column => {
29 |       [column]: { $regex: string } = { $regex: regexPattern };
30 |     });
31 |
32 |     if(skip && limit) {
33 |       const result = await this.horarioModel.find({$or:orConditions,delete:false}).skip(skip).limit(limit).exec();
34 |       const total = await this.horarioModel.countDocuments({$or:orConditions,delete:false})
35 |       return { result, total }
36 |     }
37 |
38 |     const result = await this.horarioModel.find({$or:orConditions,delete:false}).exec();
39 |   }
40 | }
```

Fuente: Elaboración propia

3.4.25. Registro de licencia de conducir

Figura 88. Registro de licencia de conducir

Microbuses

Home

Monitoreo

Registro de usuarios

Roles y Permisos

Registro de microbuses

Registro de tarifas

Registro de horarios

Registro de rutas

Registro de líneas

Registro de usuarios

FILTRAR POR COLUMNAS

NOMBRES Y APELLIDOS	CI	DIRECCIÓN	CELULAR
juan carlos rodriguez gadfdhja@gmail.com	3435354	sdfsdfs	3453535
manuel torrez asdas: asfsdfi@gmail.com	34535	xcsxsd	3453534
ghgvvhgh,m,m,m hjhfpf@gmail.com	9889879	llkhjhg	6676767
dussan canaviri dussan@gmail.com	10531753	Av. las bander...	72381722
Jhony Contreras Vas jhony@gmail.com	1053175	Av. los pinos	72381722
Jheny Benitez Relos Jheny@gmail.com	1385678	Av. Las bande...	72381722

Categoría de la licencia:

Fecha de emisión: dd/mm/aaaa

Fecha de expiración: dd/mm/aaaa

LICENCIA PARA CONDUCIR

NOMBRES:

APELLIDOS:

SEXO:

NACIONALIDAD:

FECHA NAC:

LICENCIA NRO:

CATEGORÍA:

SELECCIONAR FOTO DE LICENCIA FRONTAL

LICENCIA PARA CONDUCIR

NOMBRES:

APELLIDOS:

SEXO:

NACIONALIDAD:

FECHA NAC:

LICENCIA NRO:

CATEGORÍA:

SELECCIONAR FOTO DE LICENCIA TRASERA

NO AGREGAR LICENCIA

Fuente: Elaboración propia

Figura 89. Código fuente de Registro de licencia de conducir

```

src > licence-driver > licence-driver.service.ts > LicenceDriverService
1 import { Injectable } from '@nestjs/common';
2 import { CreateLicenceDriverDto } from './dto/create-licence-driver.dto';
3 import { UpdateLicenceDriverDto } from './dto/update-licence-driver.dto';
4 import { InjectModel } from '@nestjs/mongoose';
5 import { LicenceDriver, LicenceDriverDocument } from './schema/licence-driver.schema';
6 import { Model } from 'mongoose';
7 import * as fs from 'fs';
8
9 Complexity is 7 It's time to do something...
10 @Injectable()
11 export class LicenceDriverService {
12   constructor(@InjectModel('LicenceDriver.name') private readonly licenceDriverModel:Model<LicenceDriverDocument>){}
13
14   Complexity is 7 It's time to do something...
15
16   async create(createLicenceDriverDto: CreateLicenceDriverDto, files: { licenceFront?: Express.Multer.File[], licenceBack?: Express.Multer.
17
18     if(files){
19       const { licenceFront, licenceBack } = files;
20
21       if (licenceFront && licenceFront.length > 0) {
22         const frontImage = licenceFront[0];
23         const frontImagePath = './uploads/${frontImage.originalname}';
24         await fs.promises.rename(frontImage.path, frontImagePath);
25         createLicenceDriverDto.licenceFront = `/uploads/${frontImage.originalname}`;
26       }
27
28       if (licenceBack && licenceBack.length > 0) {
29         const backImage = licenceBack[0];
30         const backImagePath = './uploads/${backImage.originalname}';
31         await fs.promises.rename(backImage.path, backImagePath);
32         createLicenceDriverDto.licenceBack = `/uploads/${backImage.originalname}`;
33       }
34
35     }
36
37     return await this.licenceDriverModel.create(createLicenceDriverDto);
38   }
39
40   async findAll() {
41
42   }
43
44 }

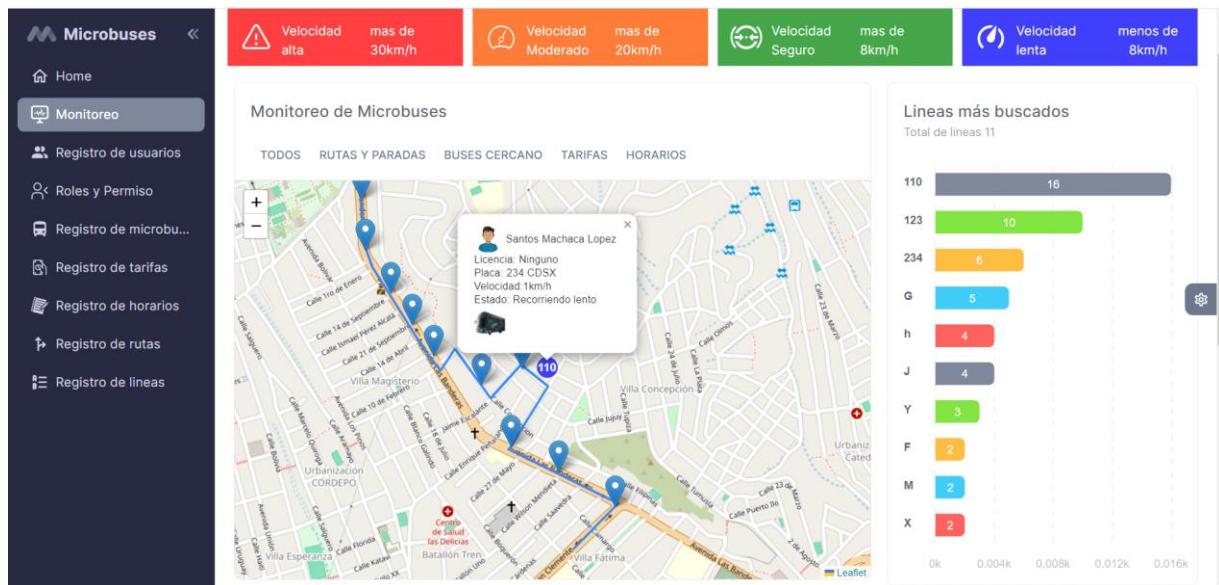
```

Fuente: Elaboración propia

3.5. Implementación de sprint 4

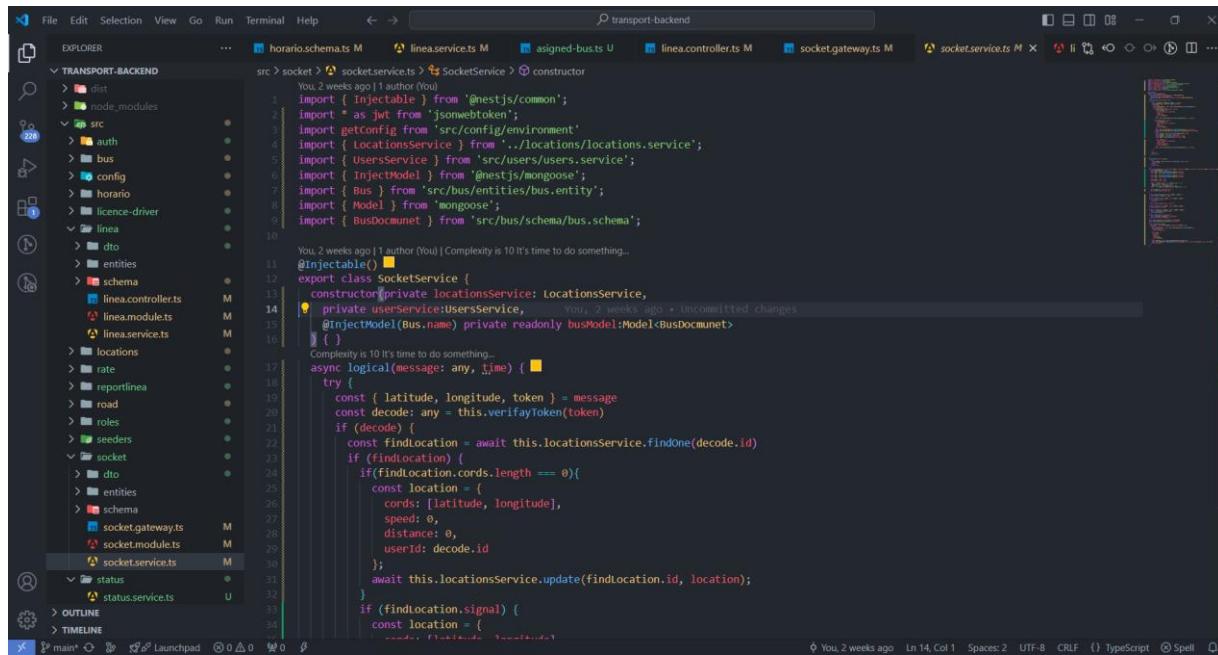
3.5.1. Monitoreo, listar rutas y paradas, ver velocidad de microbuses y ver reporte de las solicitudes

Figura 90. Monitoreo



Fuente: Elaboración propia

Figura 91. Código fuente de Monitoreo



```

src > socket > socket.service.ts > SocketService > constructor
You, 2 weeks ago | 1 author (You) | Complexity is 10 It's time to do something...
import { Injectable } from '@nestjs/common';
import * as jwt from 'jsonwebtoken';
import getConfig from 'src/config/environment';
import { locationsService } from './locations(locations.service';
import { UsersService } from 'src/users/users.service';
import { InjectModel } from '@nestjs/mongoose';
import { Bus } from 'src/bus/entities/bus.entity';
import { Model } from 'mongoose';
import { BusDocument } from 'src/bus/schema/bus.schema';

You, 2 weeks ago | 1 author (You) | Complexity is 10 It's time to do something...
@Injectable()
export class SocketService {
  constructor(private locationsService: locationsService,
    private userService: UsersService,
    @InjectModel(Bus.name) private readonly busModel: Model<BusDocument>) {
  }

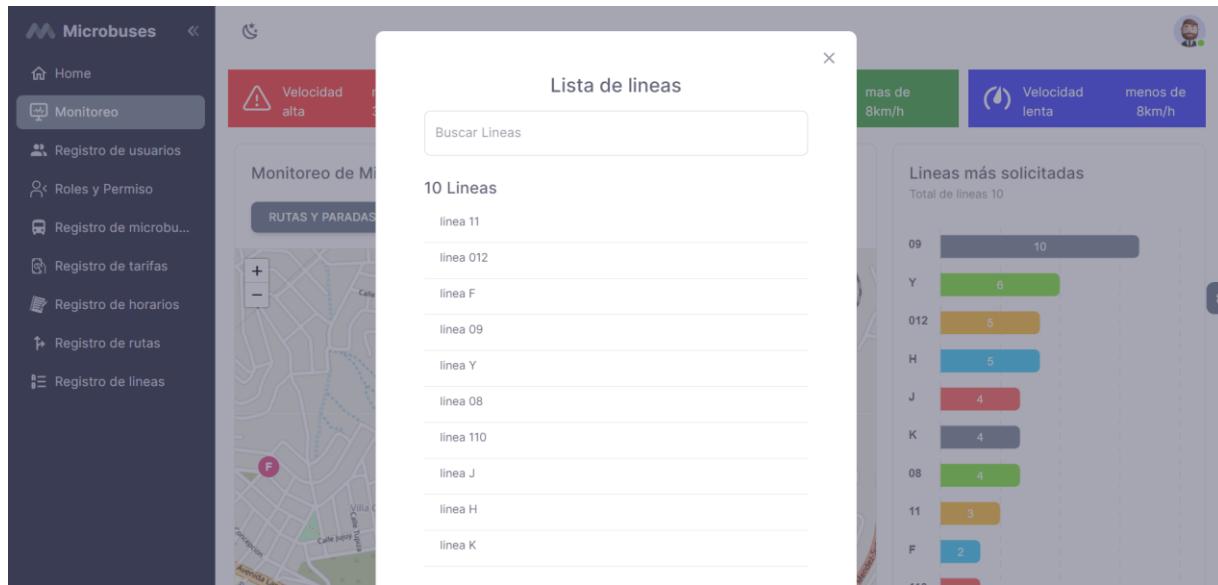
  Complexity is 10 It's time to do something...
  async logical(message: any, time) {
    try {
      const { latitude, longitude, token } = message;
      const decode: any = this.verifyToken(token);
      if (decode) {
        const findlocation = await this.locationsService.findOne(decode.id);
        if (findlocation) {
          if (findlocation.cords.length === 0) {
            const location = {
              cords: [latitude, longitude],
              speed: 0,
              distance: 0,
              userId: decode.id
            };
            await this.locationsService.update(findlocation.id, location);
          }
          if (findlocation.signal) {
            const location = {
              ...
            };
            await this.locationsService.update(findlocation.id, location);
          }
        }
      }
    }
  }
}

```

Fuente: Elaboración propia

3.5.2. Listar líneas

Figura 92. Listar líneas



Fuente: Elaboración propia

Figura 93. Código fuente de Listar líneas

```

src > linea > linea.service.ts > LineaService > findAll
YOU, 3 days ago · Complexity is 10 It's time to do something...
@Injectable()
export class LineaService {
  constructor(
    @InjectModel(Linea.name) private readonly lineaModel: Model<LineaDocument>,
    @InjectModel(Horario.name) private readonly horarioModel: Model<HorarioDocument>,
    @InjectModel(Tarifa.name) private readonly tarifaModel: Model<TarifaDocument>,
    @InjectModel(Bus.name) private readonly busModel: Model<BusDocument>
  ) {}

  Complexity is 4 Everything is cool!
  async create(createLineaDto: CreateLineaDto): Promise<LineaDocument> {
    const errorMessages = await this.isValidData(createLineaDto);
    if(errorMessages){
      return await this.lineaModel.create(createLineaDto);
    }
    throw new HttpException(errorMessages, HttpStatus.BAD_REQUEST);
  }

  Complexity is 5 Everything is cool!
  async findAll(filters: FiltersDto): Promise<LineaDocument[]> {
    const { filter, skip, limit } = filters;
    const regexPattern = new RegExp(filter, 'i');
    const columnstoSearch = ['name'];

    const orConditions = columnstoSearch.map(column => ({
      [column]: { $regex: regexPattern }
    }));

    const query = { delete: false };
    You, 5 days ago + uncommitted changes

    const populateOptions = [
      { path: 'road', match: { delete: false } },
      { path: 'horario', match: { delete: false } },
      { path: 'rate', match: { delete: false } },
      {
        path: 'buses',
        match: { delete: false },
        populate: [
          ...
        ]
      }
    ];
  }
}

```

You, 5 days ago · Ln 41, Col 37 · Spaces: 2 · UTF-8 · CRLF · {} · TypeScript · Spell · □

Fuente: Elaboración propia

3.5.3. Listar tarifas

Figura 94. Listar tarifas



Fuente: Elaboración propia

Figura 95. Código fuente de Listar tarifas

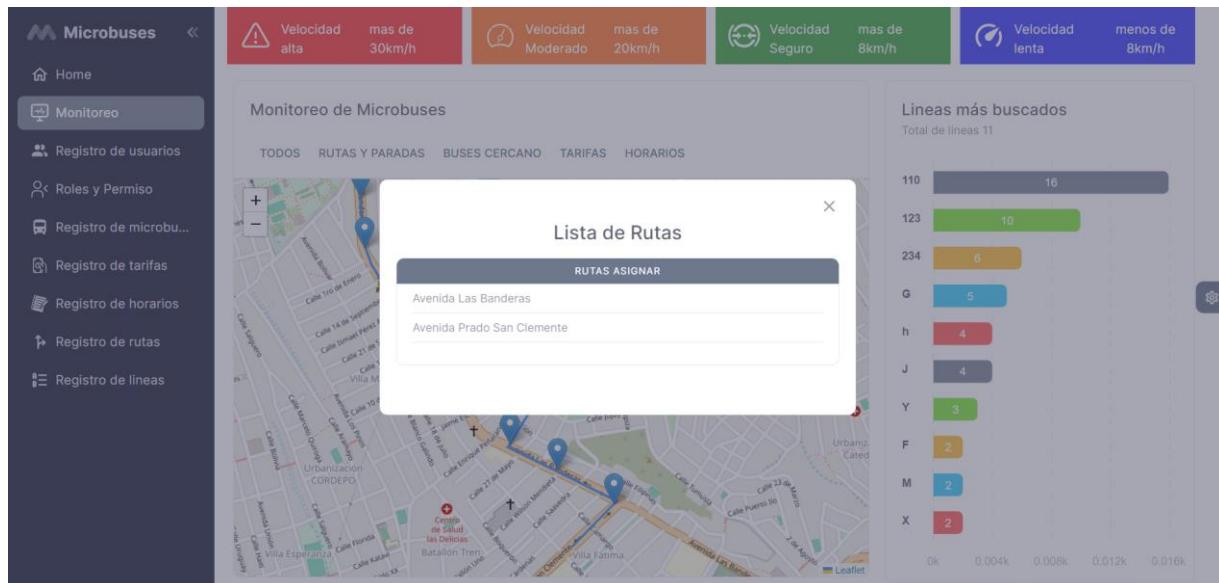
```

src > linea > linea.service.ts > LineaService > findAll
17  export class LineaService {
32    async findAll(filters: FiltersDTO) {
43      const populateOptions = [
44        { path: 'horario', match: { delete: false } },
45        { path: 'rate', match: { delete: false } },
46        {
47          path: 'buses',
48            match: { delete: false },
49            populate: [
50              {
51                path: 'userId',
52                  model: 'Users',
53                  match: { delete: false },
54                  populate: [
55                    { path: 'licenceid',
56                      model: 'LicenceDriver',
57                        match: { delete: false }
58                  }
59                ]
60              }
61            ]
62        };
63      let result;
64      if (!skip && !limit) {
65        result = await this.lineModel.find(query)
66          .populate(populateOptions)
67          .skip(skip)
68          .limit(limit)
69          .exec();
70      } else {
71        result = await this.lineModel.find(query)
72          .populate(populateOptions)
73          .skip(skip)
74          .limit(limit)
75          .exec();
76      }
77    }
78  }
  
```

You, 5 days ago | Ln 77, Col 13 | Spaces: 2 | CRLF | {} TypeScript | ⚡ Spell | 🔍

Fuente: Elaboración propia

Figura 96: Listar rutas de la línea



Fuente: Elaboración propia

Figura 97: Código fuente de Listar rutas de la línea

```

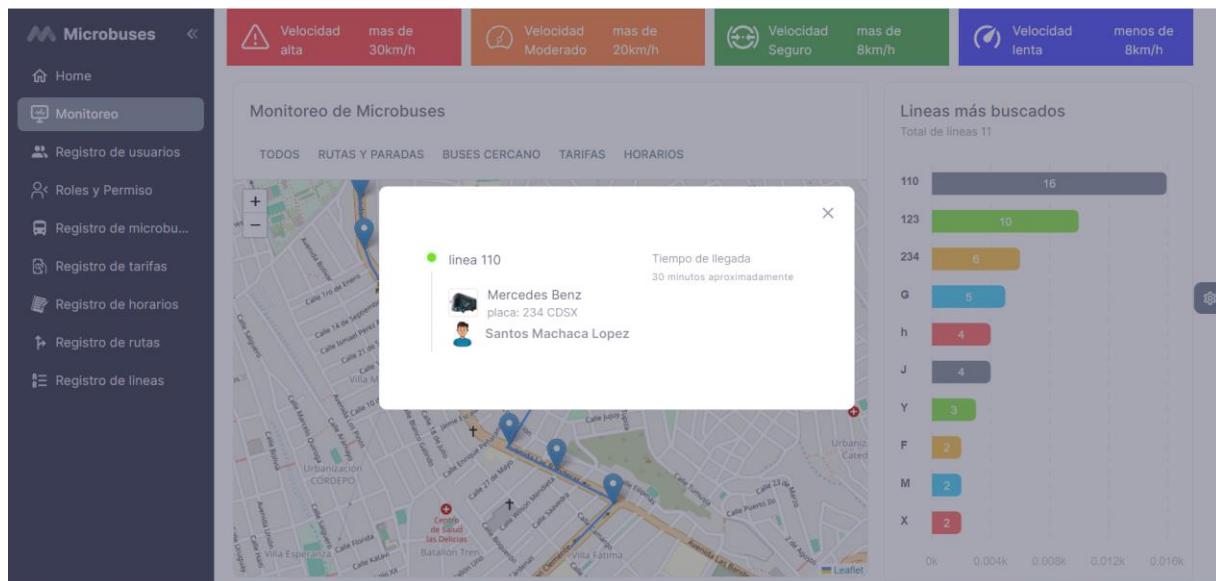
export class LineaService {
  async findOneLinea(id: string) {
    const query = { id: id, delete: false };

    const populateOptions = [
      { path: 'road', match: { delete: false } },
      { path: 'horario', match: { delete: false } },
      { path: 'rate', match: { delete: false } },
      {
        path: 'buses',
        match: { delete: false },
        populate: [
          {
            path: 'userId',
            model: 'Users',
            match: { delete: false },
            populate: [
              { path: 'licenceId', model: 'LicenceDriver', match: { delete: false } }
            ]
          },
          { path: 'locationId', model: 'Locations', match: { delete: false } }
        ]
      }
    ];
  }
}

```

Fuente: Elaboración propia

Figura 98: Ver buses cercanos



Fuente: Elaboración propia

Figura 99: Código fuente de Ver buses cercanos

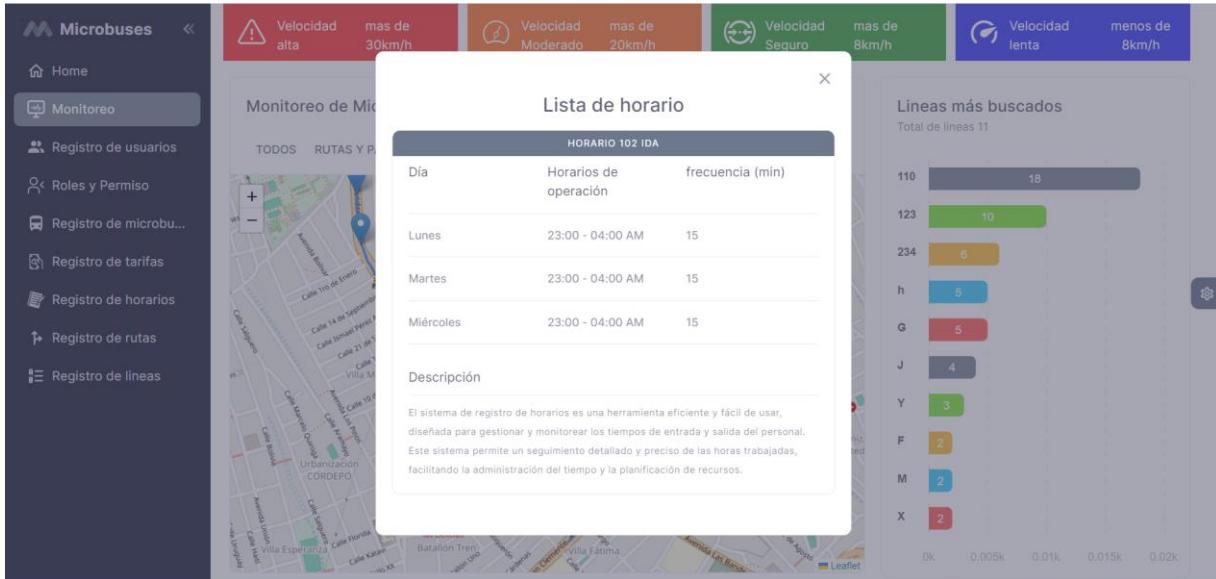
```
src > socket > socket.services > SocketService > nearBus > busAproximated > treeBus.map() callback
  export class SocketService {
    async nearbus(location:{ latitude:number, longitude:number }){
      const busAproximated = treeBus.map((bus)>{
        let localeUser = 0;
        let localeBus = 0;
        let distanceUser = 0;
        let distanceBus = 0;
        let index = 0;
        for (const [lng,lat] of feature.geometry.coordinates){
          if(index>0){
            distanceUser = this.Distance(location, {latitude:lat,longitude:lng})
            distanceBus = this.Distance(this.convertCords(bus.locationId.cords), {latitude:lat,longitude:lng})
          }else{
            const distanceUserB = this.Distance(location, {latitude:lat,longitude:lng})
            const distanceUserBbus = this.Distance(this.convertCords(bus.locationId.cords), {latitude:lat,longitude:lng})
            if(distanceUserB<distanceUser){
              distanceUser = distanceUserB
              localeUser = index
            }
            if(distanceUserBbus<distanceUser){
              distanceBus = distanceUserBbus
              localeBus = index
            }
          }
          index++;
        }
        distance +=distanceBus;
      })
      for(let index = localeBus; index<localeUser;index++){
        distance +=distanceUser;
      }
      return distance;
    }
  }

```

Fuente: Elaboración propia

3.5.4. Listar Horarios

Figura 100. Listar Horarios



Fuente: Elaboración propia

Figura 101. Código fuente de Listar Horarios

```

src > linea > linea.service.ts > LineaService > findAll
17   export class LineaService {
32     async findAll(filters: FiltersDTO) {
43       const populateOptions = [
44         { path: 'horario', match: { delete: false } },
45         { path: 'rate', match: { delete: false } },
46         {
47           path: 'buses',
48             match: { delete: false },
49             populate: [
50               {
51                 path: 'userId',
52                   model: 'Users',
53                     match: { delete: false },
54                     populate: [
55                       { path: 'licenceid',
56                         model: 'LicenceDriver',
57                           match: { delete: false }
58                         }
59                       ],
60                     }
61                   ],
62                 ],
63               ];
64             }
65           ];
66         let result;
67         if (!skip && !limit) {
68           result = await this.lineModel.find(query)
69             .populate(populateOptions)
70             .skip(skip)
71             .limit(limit)
72             .exec();
73         } else {
74           result = await this.lineModel.find(query)
75             .exec();
76         }
77       } You, 5 days ago | Ln 77, Col 13 | Spaces: 2 | UTF-8 | CR/LF | {} TypeScript | ⚡ Spell | ⌂
78     }
    
```

Fuente: Elaboración propia

3.5.5. Estados de conexión de usuarios

Figura 102. Estados de conexión de usuarios

USUARIOS	CI	BUS	PLACA	LINEA	ESTADO DE CONEXION
juan carlos gasdfhja@gmail.com	3435354	Mercedes Benz	HGV 778	X	Desconectado últ. vez nunca se conectó
manuel torrez asfsdf@gmail.com	34535	Nissan	ZSD 789	11	Desconectado últ. vez nunca se conectó
ghgvgvhgh hjhmgt@gmail.com	9889879	Ninguno	—	Ninguno	Desconectado últ. vez nunca se conectó
dussan dussan@gmail.com	10531753	Tata	SAX 897	09	En Linea
Jhony jhony@gmail.com	1053175	Nissan	ZSD 3232	09	Desconectado últ. vez hoy a las 1:17
Jheny Jheny@gmail.com	1385678	Nissan	GHIY 890	F	Baja Señal

Fuente: Elaboración propia

Figura 103. Código fuente de Estados de conexión de usuarios

```

11  export class StatusService {
12    ...
13    ...
14    ...
15    ...
16    ...
17    ...
18    ...
19    ...
20    ...
21    ...
22    ...
23    ...
24    ...
25    ...
26    ...
27    ...
28    ...
29    ...
30    ...
31    ...
32    ...
33    ...
34    ...
35    ...
36    ...
37    ...
38    ...
39    ...
40    ...
41    ...
42    ...
43    ...
44    ...
45    ...
46    ...
47    ...
48    ...
49    ...
50    ...
51    ...
52    ...
53    ...
54    ...
55    ...

```

Complexity is 34 Bloody hell...

async allData(data: any, filter: any) {

Complexity is 29 Bloody hell...

const resultPromises = data.map(async (user) => {

if (!user) {

return null;

Fuente: Elaboración propia

3.6. Seguridad

El presente proyecto es implementado con los framework de Nestjs para la parte backend y Nextjs para la parte frontend, por el cual cuenta con la seguridad de:

- Protección contra la falsificación de solicitudes en sitios cruzados
- Protección de inyección SQL
- SSL/HTTPS.
- Validación de cabecera de host
- Política de remisión
- Seguridad de sesión

3.7. Pruebas

Las pruebas de software son un proceso fundamental en el desarrollo de sistemas, destinado a evaluar y verificar que un programa funcione correctamente, cumpla con los requisitos especificados y esté libre de errores. Este proceso implica la ejecución de un sistema o aplicación con la intención de encontrar fallos y asegurar que el software cumple con sus objetivos de

calidad y funcionalidad. Por el cual en el presente proyecto se emplearon las pruebas de funcionalidad, validación y aceptación, mismas que detallan en los siguientes párrafos.

3.7.1. Pruebas Funcionales

En las siguientes tablas se muestran las pruebas funcionales para cada historia de usuario

Tabla 41. Pruebas Funcionales control de acceso al sistema

Nro.	Variable	Se espera	Se obtiene
1	Sin datos	El sistema no permita ingresar datos vacíos.	El sistema muestra un mensaje de los campos requeridos.
2	Ingreso de credenciales incorrectos	No permitir ingresar al sistema.	No permite ingresar al sistema, muestra un mensaje de datos incorrectos.
3	Ingreso de credenciales correctos	Permitir ingresar al sistema.	Permite ingresar al sistema y muestra los datos del usuario.
4	Cierre de sesión	Destrucción de sesión.	Destruye la sesión y no permite ingresar al sistema.

Fuente: Elaboración propia

Tabla 42. Pruebas Funcionales roles y permisos

Nro.	Variable	Se espera	Se obtiene
1	Editar	Si tiene rol de editar permita editar caso contrario no permita.	Si tiene rol de editar permite realizar ediciones en sistema.
2	Eliminar	Si tiene rol de eliminar permita eliminar caso contrario no permita.	Si tiene rol de eliminar permite eliminar los datos en el sistema.

3	Mostrar	Si tiene de mostrar o leer muestre las interfaces de usuario, caso contrario que no muestre.	Si tiene rol de leer o mostrar la interface muestra la interface al usuario.
4	Crear	Si tiene rol de crear permite crear, caso contrario no.	Si tiene rol de crear permite crear nuevos usuarios y entre otros.

Fuente: Elaboración propia

Tabla 43. prueba funcionales Aplicación GPS

Nro.	Variable	Se espera	Se obtiene
1	Sin datos	No dejar conectar al servidor.	No conecta al servidor muestra el mensaje que no puede conectar con el servidor y campos no válidos.
2	Ingreso de credenciales incorrectos	No permitir conectar al servidor.	No permite conectar al servidor muestra un mensaje de que las credenciales son incorrectos.
3	Ingreso de credenciales correctos	Permitir conectar al servidor y empezar enviar la localización del dispositivo.	Conecta al servidor y empieza enviar la ubicación del dispositivo.
4	Desconectar del servidor	Cerrar sesión y destrucción de la	Cierra la sesión y destruye la ejecución

		ejecución en segundo plano.	en segundo plano de la aplicación.
--	--	-----------------------------	------------------------------------

Fuente: Elaboración propia

Tabla 44. Pruebas funcionales Registrar usuarios

Nro.	Variable	Se espera	Se obtiene
1	Sin datos	El sistema no permite ingresar campos vacíos.	No permite ingresar datos vacíos, muestra un mensaje de que los campos son requeridos.
2	Selección de tipo de archivo	No permitir seleccionar archivos que no sea imagen.	No permite seleccionar archivos que no sea imagen muestra un mensaje de que seleccione una imagen.
3	Duplicidad de datos	El sistema no permite registrar datos duplicados.	No permite registrar datos duplicados muestra un mensaje de dato ya registrados.
4	Validación de campos numéricos y texto	Validar campos de texto y numéricos.	Valida los campos numéricos y texto, y muestra un mensaje de que el campo es numérico o texto.
5	Mostrar lista usuarios	Mostrar la lista de los usuarios actualizada.	Muestra la lista de usuarios actualizada.

Fuente: Elaboración propia

Tabla 45. Pruebas funcionales Registrar microbuses

Nro.	Variable	Se espera	Se obtiene
1	Sin datos	El sistema no permite ingresar campos vacíos.	No permite ingresar datos vacíos, muestra un mensaje de que los campos son requeridos.
2	Selección de tipo de archivo	No permite seleccionar archivos que no sea imagen o pdf.	No permite seleccionar archivos que no sea imagen o pdf, muestra un mensaje de que seleccione una imagen.
3	Duplicidad de datos	El sistema no permite registrar datos duplicados.	No permite registrar datos duplicados muestra un mensaje de dato ya registrados.
4	Validación de campos numéricos y texto	Validar campos de texto y numéricos.	Valida los campos numéricos y texto, y muestra un mensaje de que el campo es numérico o texto.
5	Validación año del modelo	Validar el año del modelo del microbús.	Valida el año del modelo de microbús, muestra un mensaje del año del modelo

			debe ser no menor a 1980.
6	Validación de cantidad de asientos	Validar cantidad de asiento q no sean menores a 6.	Valida la cantidad de asientos, muestra un mensaje de que debe ser la cantidad de asientos no menor a 6.
7	Mostrar lista actualizada	Mostrar lista de minibús actualizada.	Muestra las listas actualizadas del minibús.

Fuente: Elaboración propia

Tabla 46. Pruebas funcionales líneas

Nro.	Variable	Se espera	Se obtiene
1	Sin datos	El sistema no permite ingresar campos vacíos.	No permite ingresar datos vacíos, muestra un mensaje de que los campos son requeridos.
2	Duplicidad de datos	El sistema no permite registrar datos duplicados.	No permite registrar datos duplicados muestra un mensaje de dato ya registrados.
3	Asignar rutas	El sistema permite seleccionar una ruta.	Permite seleccionar rutas, muestra todas las rutas creadas.
4	Asignar horario	El sistema permite seleccionar horarios.	El sistema permite seleccionar horarios,

			muestra una lista de horarios.
5	Asignar tarifa	El sistema permite seleccionar tarifas.	El sistema permite seleccionar horarios, muestra una lista de tarifa.
6	Asignar buses	El sistema permite seleccionar buses.	El sistema permite seleccionar buses, muestra una lista de buses que no están asignados a ninguna línea.
7	Mostrar lista actualizada	Mostrar lista de líneas actualizada.	Muestra las listas actualizadas de líneas.

Fuente: Elaboración propia

Tabla 47. Pruebas funcionales Registrar rutas y pardas

Nro.	Variable	Se espera	Se obtiene
1	Sin datos	El sistema no permite ingresar campos vacíos.	No permite ingresar datos vacíos, muestra un mensaje de que los campos son requeridos.
2	Duplicidad de datos	El sistema no permite registrar datos duplicados.	No permite registrar datos duplicados muestra un mensaje de dato ya registrados.

3	Verificación de cambios	Verificar si se realizaron cambios en el mapa.	Verifica los cambios realizados, muestra una alerta cuando se pretende cerrar el mapa.
----------	-------------------------	--	--

Fuente: Elaboración propia

Tabla 48. Pruebas funcionales Registrar tarifas de los microbuses

Nro.	Variable	Se espera	Se obtiene
1	Sin datos	El sistema no permite ingresar campos vacíos.	No permite ingresar datos vacíos, muestra un mensaje de que los campos son requeridos.
2	Duplicidad de datos	El sistema no permite registrar datos duplicados.	No permite registrar datos duplicados muestra un mensaje de dato ya registrados.
3	Validación de campos numéricos y texto.	Validar campos de texto y numéricos.	Valida los campos numéricos y texto, y muestra un mensaje de que el campo es numérico o texto.
4	Mostrar lista actualizada	Mostrar lista de tarifas actualizada	Muestra lista de tarifas actualizada

Fuente: Elaboración propia

Tabla 49. Pruebas funcionales registrar horarios de salida

Nro.	Variable	Se espera	Se obtiene
------	----------	-----------	------------

1	Sin datos	El sistema no permite ingresar campos vacíos.	No permite ingresar datos vacíos, muestra un mensaje de que los campos son requeridos.
2	Duplicidad de datos	El sistema no permite registrar datos duplicados.	No permite registrar datos duplicados muestra un mensaje de dato ya registrados.
3	Validación de campos numéricos y texto	Validar campos de texto y numéricos.	Valida los campos numéricos y texto, y muestra un mensaje de que el campo es numérico o texto
4	Validación de hora de salida	Validar hora última salida.	Valida la hora de última hora de salida, muestra un mensaje de la hora de última salida debe ser mayor a 1 hora de la hora de salida.

Fuente: Elaboración propia

Tabla 50. Pruebas funcionales registrar licencia de conducir

Nro.	Variable	Se espera	Se obtiene
1	Sin datos	El sistema no permite ingresar campos vacíos.	No permite ingresar datos vacíos, muestra un mensaje de que los

			campos son requeridos.
2	Duplicidad de datos	El sistema no permite registrar datos duplicados.	No permite registrar datos duplicados muestra un mensaje de dato ya registrados.
3	Validación de campos numéricos y texto	Validar campos de texto y numéricos.	Valida los campos numéricos y texto, y muestra un mensaje de que el campo es numérico o texto.
4	Selección de tipo de archivo	No permitir seleccionar archivos que no sea imagen.	No permite seleccionar archivos que no sea imagen muestra un mensaje de que seleccione una imagen.
5	Validación de año de expiración	Validar año de expiración de licencia.	Valida el año de expiración, muestra un mensaje de que debe ingresar un año mayor del año de emisión.

Fuente: Elaboración propia

Tabla 51. Pruebas funcionales Monitoreo

Nro.	Variable	Se espera	Se obtiene
1	Ver la ubicación de microbuses	Mostrar la ubicación de los microbuses.	Muestra la ubicación de los microbuses.

2	Ver alerta de exceso de velocidad	Mostrar una alerta de exceso de velocidad.	El sistema tiene un indicador de rango de velocidad y muestra colores diferentes dependiendo del rango de velocidad.
----------	-----------------------------------	--	--

Fuente: Elaboración propia

Tabla 52. Pruebas funcionales Ver rutas y paradas

Nro.	Variable	Se espera	Se obtiene
1	Visualización de rutas y paradas	Mostrar las rutas y paradas de los microbuses.	Muestra la ruta si tiene asignado rutas y muestra paradas si tiene asignado paradas.
2	Visualización de líneas con las rutas y paradas	Mostrar la línea con sus rutas paradas asignados.	El sistema muestra las líneas con las rutas asignadas.

Fuente: Elaboración propia

Tabla 53. Pruebas funcionales Ver tarifas

Nro.	Variable	Se espera	Se obtiene
1	Visualización de tarifas de las líneas	Mostrar tarifas de las líneas seleccionada.	El sistema muestra una lista de tarifas de la línea seleccionado.

Fuente: Elaboración propia

Tabla 54. Pruebas funcionales ver estado de conexión

Nro.	Variable	Se espera	Se obtiene
1	Visualización estados de conexión	Mostrar los estados de conexión de usuario.	El sistema muestra en una tabla los estados de conexión de los

			usuarios con los buses y líneas asignados.
2	Filtros en las columnas	El sistema permita realizar filtros por las columnas.	El sistema realiza filtros por las columnas.

Fuente: Elaboración propia

Tabla 55. Pruebas funcionales ver velocidad de microbuses

Nro.	Variable	Se espera	Se obtiene
1	Visualización de velocidad de desplazamiento de los microbuses	Mostrar las velocidades de desplazamiento de los microbuses.	El sistema muestra la velocidad de los microbuses que se desplazan.
2	Visualización de datos de chofer	Mostrar los datos de los usuarios.	El sistema muestra una foto del usuario y sus datos y foto de minibús al hacer click en la ubicación.

Fuente: Elaboración propia

Tabla 56. Pruebas funcionales ver reporte de las solicitudes de las líneas

Nro.	Variable	Se espera	Se obtiene
1	Visualización de reporte de los microbuses	Mostrar reportes de solicitudes de las líneas.	El sistema muestra un reporte en estadísticas de los 10. microbuses más solicitadas.

Fuente: Elaboración propia

3.8. Pruebas de aceptación

Las pruebas de aceptación son un tipo de pruebas de software realizadas para determinar si un sistema cumple con los requisitos especificados y si está listo para su entrega o implementación.

Estas pruebas se centran en verificar que el software funcione según lo esperado desde la perspectiva del usuario final. De tal manera en el presente proyecto se especifican detallada las pruebas de aceptación en tablas.

Tabla 57. Pruebas de aceptación control de acceso al sistema

Nombre de la clase	Nombre de prueba de caja negra	Descripción	Resultado
Control de acceso al sistema	Iniciar sesión	Se verifica que no se pueda autenticar si los campos están vacíos.	✓
		Se verifica que no se puede acceder al sistema si los datos del usuario o contraseña no son válidos.	✓

Fuente: Elaboración propia

Tabla 58. Pruebas de aceptación roles y permisos

Nombre de la clase	Nombre de prueba de caja negra	Descripción	Resultado
Roles y permisos	Crear	Se verifica que no se puede registrar datos nulos o vacíos.	✓
		Se verifica que no permite registrar datos duplicados.	✓
	Editar	Se verifica que no se pueda actualizar los datos con campos vacíos.	✓
		Se verifica que no se pueda actualizar los datos si no tiene permisos para editar.	✓
	Eliminar	Se verifica que no se puede eliminar datos si no tiene permisos para eliminar.	✓
	Listar	Se verifica que actualiza la lista de roles cuando se realizan las acciones de editar, eliminar o actualizar.	✓
	Asignar rol	Se verifica que asigna el rol al usuario.	✓

Fuente: Elaboración propia

Tabla 59. Pruebas de aceptación aplicación GPS

Nombre de la clase	Nombre de prueba de caja negra	Descripción	Resultado
Aplicación GPS	Aplicación GPS	Se verifica que no se pueda autenticar si los campos están vacíos.	✓
		Se verifica que no se puede acceder al sistema si los datos del usuario o contraseña no son válidos.	✓
		Se verifica que envía la ubicación del dispositivo y comienza a ejecutarse en segundo plano si el usuario es autenticado.	✓

Fuente: Elaboración propia

Tabla 60. Pruebas de aceptación registrar usuarios

Nombre de la clase	Nombre de prueba de caja negra	Descripción	Resultado
Registro de usuario	Crear	Se verifica que no se puede registrar datos nulos o vacíos.	✓
		Se verifica que no permite registrar datos duplicados.	✓
		Se verifica que el tipo de archivo para subir como foto del usuario sea de tipo imagen.	✓
	Editar	Se verifica que no se pueda actualizar los datos con campos vacíos.	✓
		Se verifica que no se pueda actualizar los datos si no tiene permisos para editar.	✓
	Eliminar	Se verifica que no se puede eliminar datos si no tiene permisos para eliminar.	✓
	Listar	Se verifica que actualiza la lista de roles cuando se realizan las acciones de editar, eliminar o actualizar.	✓

Fuente: Elaboración propia

Tabla 61. Pruebas de aceptación registrar usuarios

Nombre de la clase	Nombre de prueba de caja negra	Descripción	Resultado
Registro de usuario	Crear	Se verifica que no se puede registrar datos nulos o vacíos.	✓
		Se verifica que no permite registrar datos duplicados.	✓
		Se verifica que el tipo de archivo para subir como foto del usuario sea de tipo imagen.	✓
	Editar	Se verifica que no se pueda actualizar los datos con campos vacíos.	✓
		Se verifica que no se pueda actualizar los datos si no tiene permisos para editar.	✓
	Eliminar	Se verifica que no se puede eliminar datos si no tiene permisos para eliminar.	✓
	Listar	Se verifica que actualiza la lista de roles cuando se realizan las acciones de editar, eliminar o actualizar.	✓

Fuente: Elaboración propia

Tabla 62. Pruebas de aceptación registrar microbús

Nombre de la clase	Nombre de prueba de caja negra	Descripción	Resultado
Registro de microbús	Crear	Se verifica que no se puede registrar datos nulos o vacíos.	✓
		Se verifica que no permite registrar datos duplicados.	✓
		Se verifica que el tipo de archivo para subir como foto del usuario sea de tipo imagen.	✓
	Editar	Se verifica que no se pueda actualizar los datos con campos vacíos.	✓
		Se verifica que no se pueda actualizar los datos si no tiene permisos para editar.	✓

	Eliminar	Se verifica que no se puede eliminar datos si no tiene permisos para eliminar.	✓
	Listar	Se verifica que actualiza la lista de roles cuando se realizan las acciones de editar, eliminar o actualizar.	✓

Fuente: Elaboración propia

Tabla 63. Pruebas de aceptación registrar líneas

Nombre de la clase	Nombre de prueba de caja negra	Descripción	Resultado
Registrar líneas	Crear	Se verifica que no se puede registrar datos nulos o vacíos.	✓
		Se verifica que no permite registrar datos duplicados.	✓
		Se verifica que el tipo de archivo para subir como foto del usuario sea de tipo imagen.	✓
	Editar	Se verifica que no se pueda actualizar los datos con campos vacíos.	✓
		Se verifica que no se pueda actualizar los datos si no tiene permisos para editar.	✓
	Eliminar	Se verifica que no se puede eliminar datos si no tiene permisos para eliminar.	✓
	Listar	Se verifica que actualiza la lista de roles cuando se realizan las acciones de editar, eliminar o actualizar.	✓
	Asignar ruta	Se verifica que asigna la ruta a la línea.	✓
	Asignar tarifas	Se verifica que asigna las tarifas seleccionadas a la línea.	✓
	Asignar horarios	Se verifica que asigna los horarios seleccionados a la línea.	✓
	Asignar buses	Se verifica que asigna los microbuses que no estén asignados a ninguna línea.	✓

Fuente: Elaboración propia

Tabla 64. Pruebas de aceptación registrar rutas y paradas

Nombre de la clase	Nombre de prueba de caja negra	Descripción	Resultado
Registrar rutas y paradas	Crear	Se verifica que no se puede registrar datos nulos o vacíos.	✓
		Se verifica que no permite registrar datos duplicados.	✓
		Se verifica que el tipo de archivo para subir como foto del usuario sea de tipo imagen.	✓
	Editar	Se verifica que no se pueda actualizar los datos con campos vacíos.	✓
		Se verifica que no se pueda actualizar los datos si no tiene permisos para editar.	✓
	Eliminar	Se verifica que no se puede eliminar datos si no tiene permisos para eliminar.	✓
	Listar	Se verifica que actualiza la lista de roles cuando se realizan las acciones de editar, eliminar o actualizar.	✓

Fuente: Elaboración propia

Tabla 65. Pruebas de aceptación registrar tarifas de los microbuses

Nombre de la clase	Nombre de prueba de caja negra	Descripción	Resultado
Registrar tarifas de los microbuses	Crear	Se verifica que no se puede registrar datos nulos o vacíos.	✓
		Se verifica que no permite registrar datos duplicados.	✓
		Se verifica que el tipo de archivo para subir como foto del usuario sea de tipo imagen.	✓
	Editar	Se verifica que no se pueda actualizar los datos con campos vacíos.	✓
		Se verifica que no se pueda actualizar los datos si no tiene permisos para editar.	✓

	Eliminar	Se verifica que no se puede eliminar datos si no tiene permisos para eliminar.	✓
	Listar	Se verifica que actualiza la lista de roles cuando se realizan las acciones de editar, eliminar o actualizar.	✓

Fuente: Elaboración propia

Tabla 66. Pruebas de aceptación registrar horarios de salida

Nombre de la clase	Nombre de prueba de caja negra	Descripción	Resultado
registrar horarios de salida	Crear	Se verifica que no registra campos vacíos.	✓
		Se verifica que no permite registrar datos duplicados.	✓
		Se verifica que valida datos numéricos y texto.	✓
	Editar	Se verifica que no se pueda actualizar los datos con campos vacíos.	✓
		Se verifica que no se pueda actualizar los datos si no tiene permisos para editar.	✓
	Eliminar	Se verifica que no se puede eliminar datos si no tiene permisos para eliminar.	✓
	Listar	Se verifica que actualiza la lista de roles cuando se realizan las acciones de editar, eliminar o actualizar.	✓

Fuente: Elaboración propia

Tabla 67. Pruebas de aceptación Monitoreo

Nombre de la clase	Nombre de prueba de caja negra	Descripción	Resultado
Monitoreo	Listar	Se verifica que monitorea la ubicación de los microbuses, sus velocidades.	✓

Fuente: Elaboración propia

Tabla 68. Pruebas de aceptación ver rutas y paradas

Nombre de la clase	Nombre de prueba de caja negra	Descripción	Resultado
Visualización de rutas y paradas	Listar	Se verifica que muestra las rutas y paradas de la línea seleccionada.	✓

Fuente: Elaboración propia

Tabla 69. Pruebas de aceptación ver rutas y tarifas

Nombre de la clase	Nombre de prueba de caja negra	Descripción	Resultado
visualización de tarifas	Listar	Se verifica que muestra las tarifas de la línea seleccionada.	✓

Fuente: Elaboración propia

Tabla 70. Pruebas de aceptación ver estados de conexión

Nombre de la clase	Nombre de prueba de caja negra	Descripción	Resultado
Visualización de estados de conexión	Listar	Se verifica que muestra los estados de conexión de los usuarios.	✓

Fuente: Elaboración propia

Tabla 71. Pruebas de aceptación ver velocidad de microbuses

Nombre de la clase	Nombre de prueba de caja negra	Descripción	Resultado
visualización de velocidad de microbuses	Listar	Se verifica que muestra la velocidad de desplazamiento de los microbuses.	✓

Fuente: Elaboración propia

Tabla 72. Pruebas de aceptación ver reporte de solicitudes de líneas

Nombre de la clase	Nombre de prueba de caja negra	Descripción	Resultado
visualización de reporte de solicitudes de líneas	Listar	Se verifica que muestra el reporte de las líneas más solicitadas.	✓

Fuente: Elaboración propia

CONCLUSIONES

Al finalizar la implementación del sistema de control y monitoreo vehicular, se puede concluir que se alcanzaron y cumplieron las metas establecidas en el trabajo dirigido. Por lo tanto, ha sido subsanado el problema identificado en el control de microbuses con las siguientes conclusiones:

- ✓ Se logró establecer toda la fundamentación teórica necesaria de sustento al presente proyecto basado en el marco referencial, que facilita la lectura al investigador para una idea más clara sobre el sistema de control vehicular desarrollado.
- ✓ Se realizó un diagnóstico adecuado de la situación actual del control y monitoreo vehicular de los microbuses, mediante la aplicación de entrevistas a los choferes de los microbuses que trabajan con una línea y al encargado de trabajo dirigido, por otro lado, también se aplicó las encuestas a la población que ayudaron a determinar los requerimientos del sistema.
- ✓ La metodología scrum sirvió para planificar y diseñar el sistema en base a los requerimientos determinados, que sustenta la documentación de la estructura y funcionalidad del sistema.
- ✓ Se utilizó el lenguaje de programación TypeScript para el desarrollo del sistema de control y monitoreo vehicular, java para el desarrollo de la aplicación GPS. Gestor de base de datos mongo DB para el almacenamiento de datos, las tecnologías como framework nextjs para la parte frontend, nestjs para la parte backend y OpenstreetMap para la implementación de visualización de mapas, y las herramientas de Visual Code y Android Studio. con los cuales se alcanzó a lograr la obtención del sistema propuesto.
- ✓ Se llevaron a cabo pruebas exhaustivas de funcionalidad y aceptación que demostraron el correcto funcionamiento del software. Estas pruebas fueron fundamentales para verificar que el sistema cumple con todos los requisitos establecidos y funciona de manera óptima en diversos escenarios. Además, los resultados obtenidos respaldan la eficacia y confiabilidad del sistema.

Por lo tanto, se cumplió con el objetivo establecido como solución al problema identificado de control vehicular en la fundación software libre de la carrera de Ingeniería de Sistemas de la Universidad Autónoma Tomás Frías.

RECOMENDACIONES

Se tiene las siguientes recomendaciones para el mejoramiento y optimización del rendimiento del presente proyecto desarrollado.

- ✓ Se recomienda considerar la implementación de hardware GPS de mayor precisión y fiabilidad, que pueda proporcionar datos más exactos de velocidad y ubicación.
- ✓ Se recomienda evaluar la integración de servicios GPS pagos que ofrezcan mejores prestaciones y precisión en tiempo real, reduciendo el margen de error en los cálculos.
- ✓ Se recomienda aplicar otras operaciones a parte aplicar la fórmula de Haversine para obtener una distancia casi exacta y la misma velocidad, o en otro caso integrar servicios de pago. Tomar en cuenta que los cálculos de distancias, velocidades y precisión de ubicación no son tan precisos debido a que se implementó con herramientas gratuitas como haversine que tiene un error hasta 20 km, pero este error se puede optimizar aplicando algoritmos y cálculos extras, sin embargo el consumo de recursos es mayor dependiendo a la cantidad de usuarios conectados, por otra parte, la velocidad puede no ser exacta por las curvas y dependiendo del tiempo de actualizaciones del envío de ubicaciones.
- ✓ Se recomienda explorar la incorporación de inteligencia artificial para el análisis predictivo del tráfico y la optimización de rutas.
- ✓ Se recomienda establecer un equipo de soporte técnico dedicado que pueda resolver problemas rápidamente y mantener el sistema funcionando sin interrupciones.
- ✓ Se recomienda fortalecer las medidas de seguridad para proteger la información sensible de los usuarios y asegurar el cumplimiento de las normativas de privacidad.

REFERENCIAS BIBLIOGRÁFICAS

- Álvarez, A. (06 de julio de 2020). *net mind*. Obtenido de Historias de Usuario: qué son, reglas y consejos.: <https://netmind.net/es/historias-de-usuario-reglas/>
- aplyca. (14 de Octubre de 2022). *aplyca*. Obtenido de Nextjs: <https://www.aplyca.com/blog/nextjs-el-futuro-web-que-es-nextjs>
- arimetrics. (2022). *arimetrics*. Obtenido de Qué es JavaScript: <https://www.arimetrics.com/glosario-digital/javascript>
- arimetrics. (2022). *arimetrics*. Obtenido de Qué es CSS: <https://www.arimetrics.com/glosario-digital/css>
- arimetrics. (2022). *codigo qr*. Obtenido de <https://www.arimetrics.com/glosario-digital/codigo-qr>
- Astera. (2023). *Astera*. Obtenido de Qué es la integridad de datos en una base de datos: <https://www.astera.com/es/type/blog/data-integrity-in-a-database/>
- aws. (2023). *¿Que es Java?* Obtenido de <https://www.ibm.com/mx-es/topics/java>
- Barrios, J. C. (2023). *OpenWebinars*. Obtenido de Qué es TypeScript: <https://openwebinars.net/blog/que-es-typescript/>
- bizneo. (30 de agosto de 2023). *planilla de pagos*. Obtenido de <https://www.bizneo.com/blog/planilla-de-pago/>
- Borovsky, D. (10 de octubre de 2023). *Desarrollo de sistema web - perú*. Obtenido de https://es.linkedin.com/pulse/desarrollo-de-sistema-web-per%C3%BA-denisa-borovskoy?trk=article-ssr-frontend-pulse_more-articles_related-content-card
- Carlos, J. (2022). *ZAPtest*. Obtenido de ¿Qué son las pruebas funcionales?: <https://www.zaptest.com/es/que-son-las-pruebas-funcionales-tipos-ejemplos-lista-de-comprobacion-y-aplicacion>
- casiopea. (17 de Febrero de 2015). *casiopea*. Obtenido de Autobus: <https://wiki.ead.pucv.cl/Autobus>
- Ce-entel. (2023). *concepto de planilla de pagos*. Obtenido de <https://ce.entel.cl/articulos/planilla-de-sueldos/>
- chakray. (2023). *chakray*. Obtenido de Lenguajes de programación: tipos y características: <https://www.chakray.com/es/lenguajes-programacion-tipos-caracteristicas/>
- Chernyak, A. Z. (2023). *Zaptest*. Obtenido de prueba de caja blanca: <https://www.zaptest.com/es/pruebas-de-caja-blanca-que-es-como-funciona-retos-metricas-herramientas-y-mas>
- Coelho, F. (2023). *Significados* . Obtenido de Metodología: <https://www.significados.com/metodologia/>

concepto. (20203). *concepto*. Obtenido de ¿Qué es un lenguaje de programación?:
<https://concepto.de/lenguaje-de-programacion/>

concepto. (12 de noviembre de 2023). *Sistema*. Obtenido de <https://concepto.de/sistema/>

Crea System. (2023). *¿que es un sistema web?* Obtenido de <https://www.creasytem.net/posts/que-es-un-sistema-web>

Diego. (25 de agosto de 2022). *Medium*. Obtenido de Introducción a Nestjs:
<https://medium.com/@diego.coder/introducci%C3%B3n-a-nestjs-88f1ca90df35>

DOS IDEAS. (29 de agosto de 2011). Obtido de Backlog Del Sprint:
https://dosideas.com/wiki/Backlog_Del_Sprint

Educacion Activa. (2024). *Cálculo de distancia entre dos puntos con coordenadas*. Obtenido de <https://educacionactiva.org/calcular-distancia-entre-dos-puntos-con-coordenadas/>

encyclopedia online. (2024). *Medios de transporte*. Obtenido de <https://encyclopediaonline.com/es/medios-de-transporte/>

esri. (2023). *esri*. Obtenido de Cómo se usan los mapas para aplicar SIG:
<https://resources.arcgis.com/es/help/getting-started/articles/026n000000t00000.htm>

Euroinnova. (2023). *Euroinnova*. Obtenido de ¿Qué es un tester?:
<https://www.euroinnova.edu.es/blog/actividades-de-un-tester>

factorial. (5 de octubre de 2023). *¿que son los permisos laborales?* Obtenido de <https://factorialhr.es/blog/tipos-permisos-laborales-retribuidos/>

franko, J. (2023). *Android Studio: ¿Qué es y cómo funciona?* Obtenido de <https://jaimefranko.com/android-studio-que-es-y-como-funciona/>

free, j. f. (29 de noviembre de 2023). *coursera*. Obtenido de ¿Qué es una historia de usuario?:
<https://www.coursera.org/mx/articles/what-is-user-story>

Friend, B. A. (2023). *BEAGILEMYFRIEND*. Obtenido de En qué consiste el Sprint Backlog:
<https://beagilemyfriend.com/en-que-consiste-el-sprint-backlog/>

Gonzales, J. (15 de Marzo de 2023). *Alura* . Obtenido de Styled Components y Material UI. ¿Qué son y cómo utilizarlas en tu proyecto React?: <https://www.aluracursos.com/blog/styled-components-material-ui-que-son-como-utilizarlas>

Granda, D. D. (2023). *Platzi*. Obtenido de ¿Qué es JavaScript y para qué sirve?:
https://platzi.com/clases/1814-basico-javascript/26290-que-es-javascript/?utm_source=google&utm_medium=cpc&utm_campaign=20290685455&utm_adgroup=&utm_content=&&gad_source=1&gclid=CjwKCAiAgeeBhBAEiwAoDDhn4NOmlfrh9gDfafCvOYJdF2_mNZi7-z8xJTDcG2QCI0Cxa89R9NxH

Gudiña, V. (2023). *Transporte Publico*. Obtenido de <https://definicion.de/transporte-publico/>

Hammer, J. C. (1993). *Reingenieria*.

IBM. (s.f.). Obtenido de <https://www.ibm.com/mx-es/topics/java>

IBM. (s.f.). *¿Que es Java?* Obtenido de <https://www.ibm.com/mx-es/topics/java>

IBM. (2023). *¿Que es Java?* Obtenido de <https://www.ibm.com/mx-es/topics/java>

IBM. (2023). *¿Qué es una prueba de software?* Obtenido de <https://www.ibm.com/mx-es/topics/software-testing#:~:text=Las%20pruebas%20de%20software%20son,y%20la%20mejora%20del%20rendimiento.>

IBM. (2023). *IBM.* Obtenido de *¿Qué es Docker?:* <https://www.ibm.com/mx-es/topics/docker>

ICARIA. (julio de 4 de 2023). *ICARIA.* Obtenido de *Qué es un tester de software:* <https://icariatechnology.com/tester-de-software-funciones/>

ilimit. (18 de Marzo de 2022). *ilimit.* Obtenido de *Metodología SCRUM: qué es y cómo implementarlo:* <https://www.ilimit.com/blog/metodologia-scrum/>

Inesdi. (27 de Enero de 2022). *expertos.* Obtenido de *mongoDB:* <https://www.inesdi.com/blog/mongodb-que-es-caracteristicas-para-que-sirve/>

ionos. (17 de Febrero de 2021). *ionos.* Obtenido de *Que es swagger:* <https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/que-es-swagger/>

ionos. (17 de Julio de 2021). *ionos.* Obtenido de *¿Qué es CSS?:* <https://www.ionos.es/digitalguide/paginas-web/diseno-web/que-es-css/>

ionos. (2023). *ionos.* Obtenido de *¿Qué es un servidor? :* <https://www.ionos.es/digitalguide/servidores/know-how/que-es-un-servidor-un-concepto-dos-definiciones/>

Itequia. (26 de Enero de 2023). *Itequia.* Obtenido de *Descubre Nest.js, el Framework de desarrollo de Backend para Frontend:* <https://itequia.com/es/descubre-nestjs/>

kasperkey. (2023). *biometrica facial.* Obtenido de *El reconocimiento facial es una categoría de seguridad biométrica. Otras formas de software biométrico incluyen el reconocimiento de voz, el reconocimiento de huellas digitales y el reconocimiento de retina o iris. La tecnología se utiliza principalmente*

KeepCoding. (30 de noviembre de 2022). *KeepCoding.* Obtenido de *¿Qué es TypeScript? :* <https://keepcoding.io/blog/typescript/>

Keepcoding. (2 de Diciembre de 2023). *Keepcoding.* Obtenido de *¿Qué es un developer?:* <https://keepcoding.io/blog/que-es-un-developer/>

kimaldi. (12 de noviembre de 2023). *control de personal.* Obtenido de https://www.kimaldi.com/blog/control_de_acceso_y_presencia/control_de_personal/

kinsta. (19 de Diciembre de 2022). *kinsta.* Obtenido de *¿Qué es Next.js?:* <https://kinsta.com/es/base-de-conocimiento/next-js/>

- kodigo. (1 de Diciembre de 2023). *kodigo*. Obtenido de el perfil digital más demandado:
<https://kodigo.org/que-es-un-software-developer-el-perfil-digital-mas-demandado/>
- La Rioja. (octubre de 2023). *Administración de los recursos humanos*. Obtenido de
<https://mexico.unir.net/economia/noticias/administracion-recursos-humanos-funciones/>
- linkedin. (11 de abril de 2023). *linkedin*. Obtenido de Que son las pruebas de aceptación:
<https://es.linkedin.com/pulse/que-son-las-pruebas-de-aceptaci%C3%B3n-andes-training>
- MacNeil, C. (3 de enero de 2023). *asana*. Obtenido de Product Owner: funciones y responsabilidades clave: <https://asana.com/es/resources/product-owner>
- Maria, A. (2017). *Reingeniería*.
- Marín, E. M. (2024). *Distancia Recorrida y Desplazamiento*. Obtenido de
<https://significado.com/distancia-recorrida-desplazamiento/>
- Marte, C. (13 de mayo de 2020). *Ambit*. Obtenido de Claves y características de la reingeniería de procesos o BPR: <https://www.ambit-bst.com/blog/claves-y-caracter%C3%ADsticas-de-la-reingenier%C3%ADA-de-procesos-o-bpr>
- Martín González Soto. (2023). *¿Qué es el GPS y cómo funciona?* Obtenido de <https://nanova.org/que-es-el-gps-y-como-funciona/>
- Martines, J. (19 de Junio de 2023). *Asana*. Obtenido de metodología scrum:
<https://asana.com/es/resources/what-is-scrum>
- Mediotrasporte. (2023). *autobus*. Obtenido de <https://mediosdetransporte.org/autobus/>
- Molina, D. (15 de diciembre de 2021). *iebs*. Obtenido de Qué es un Product Owner y diferencias con el Product Manager: <https://www.iebschool.com/blog/diferencias-product-owner-product-manager-marketing-estrategico/>
- Navarro, A. (2024). *Qué es y para qué sirve Android Studio*. Obtenido de
<https://androidayuda.com/programar-mensajes-en-whatsapp/>
- Oracle. (2023). *Oracle*. Obtenido de Contenedor: <https://www.oracle.com/mx/cloud/cloud-native/container-registry/what-is-docker/>
- pablo. (25 de Noviembre de 2022). *By Orange*. Obtenido de HTML5: todo sobre el último estándar de HTML: <https://blog.orange.es/consejos-y-trucos/html5/#>
- Paessler. (2023). *Paessler*. Obtenido de ¿Qué es un servidor?: <https://www.paessler.com/es/it-explained/server>
- Páez, G. (2020). *Tipos de transporte*. Obtenido de <https://economipedia.com/definiciones/tipos-de-transporte.html>
- Paula. (12 de Mayo de 2022). *iebs*. Obtenido de Definición y características del Scrum Master:
<https://www.iebschool.com/blog/definicion-y-caracteristicas-del-scrum-master-agile-scrum/>

- Piarc. (2024). *CONTROL DEL TRÁNSITO URBANO*. Obtenido de <https://rno-its.piarc.org/es/conceptos-basicos-its-tecnologias-its-control-del-transito/control-del-transito-urbano>
- Pimentel, C. (22 de marzo de 2023). *Evalart*. Obtenido de Cómo utilizar pruebas de integridad en los procesos de reclutamiento: <https://evalart.com/es/blog/como-utilizar-pruebas-de-integridad-en-los-procesos-de-reclutamiento/>
- por, P. (2023). *Android Studio: ¿Qué es y cómo funciona?* Obtenido de <https://jaimefranko.com/android-studio-que-es-y-como-funciona/>
- Proware. (octubre de 2023). *control de personal*. Obtenido de <https://www.proware.com.co/blog/importancia-control-de-personal/>
- Quinteros, J. (2020). *Sistemas Computacionales*. 3.
- Radigan. (2023). *Atlassian*. Obtenido de Backlog del producto: qué es y cómo crearlo : <https://www.atlassian.com/es/agile/scrum/backlogs>
- Raeburn, A. (22 de agosto de 2022). *asana*. Obtenido de ¿Qué es un product backlog?: <https://asana.com/es/resources/product-backlog>
- Raeburn, A. (12 de julio de 2023). *asana*. Obtenido de ¿Qué es un Scrum Master y cuál es su función?: <https://asana.com/es/resources/scrum-master>
- Robledano, A. (28 de Octubre de 2019). *OpenWebinars*. Obtenido de Qué es MongoDB: <https://openwebinars.net/blog/que-es-mongodb/>
- services. (2023). *reconocimiento facial*. Obtenido de <https://aws.amazon.com/es/what-is/facial-recognition/>
- significados. (2023). *En las matemáticas, la distancia entre dos puntos del espacio euclídeo equivale a la longitud del segmento de la recta que los une, expresado numéricamente. En espacios más complejos, como los definidos en la geometría no euclidiana, el «camino más corto».* Obtenido de <https://significadosweb.com/concepto-de-distancia-recorrida-en-fisica-que-es-definicion-significado-y-ejemplos/>
- Skeleton . (2 de Agosto de 2023). *Skeleton* . Obtenido de Crea Aplicaciones Web Impresionantes con Material UI, Guía para Principiantes : <https://bluuweb.dev/05-react/material-ui.html>
- soluciones. (septiembre de 2023). *Enclopedia de negocios*. Obtenido de <https://www.shopify.com/es/enciclopedia/administracion-de-recursos-humanos>
- sphera. (8 de junio de 2022). *¿Que es el permiso de trabajo?* Obtenido de <https://sphera.com/glosario-es/que-es-el-permiso-de-trabajo/?lang=es>
- SQA. (23 de marzo de 2022). *SQA*. Obtenido de Pruebas de caja negra : <https://es.linkedin.com/pulse/pruebas-de-caja-negra-sqa-s-a>
- tester. (27 de noviembre de 2023). *tester*. Obtenido de Pruebas de aceptación de usuario: <https://www.wearetesters.com/investigacion-ux/pruebas-aceptacion-usuario-uat-que-son/>

testing. (2023). *testing*. Obtenido de Pruebas de integración de software: qué son, niveles y tipos:
<https://www.testingit.com.mx/blog/pruebas-de-integracion-de-software>

wiki. (2023). *Acerca de OpenStreetMap*. Obtenido de
https://wiki.openstreetmap.org/wiki/ES:Acerca_de_OpenStreetMap

wikipedia. (20 de julio de 2022). *wikipedia*. Obtenido de Pruebas de caja blanca:
https://es.wikipedia.org/wiki/Pruebas_de_caja_blanca

wikipedia. (2023). *Transporte Publico*. Obtenido de
https://es.wikipedia.org/wiki/Transporte_p%C3%BAblico

wikipedia. (2023 de Septiembre de 2023). *wikipedia*. Obtenido de swagger:
[https://es.wikipedia.org/wiki/Swagger_\(software\)](https://es.wikipedia.org/wiki/Swagger_(software))

wikipedia. (17 de Noviembre de 2023). *wikipedia*. Obtenido de prueba de caja negra:
[https://es.wikipedia.org/wiki/Caja_negra_\(sistemas\)](https://es.wikipedia.org/wiki/Caja_negra_(sistemas))

wikipedia. (2023). *wikipedia*. Obtenido de Pruebas funcionales:
https://es.wikipedia.org/wiki/Pruebas_funcionales

wikipedia. (17 de noviembre de 2023). *wikipedia*. Obtenido de Transporte público:
https://es.wikipedia.org/wiki/Transporte_p%C3%BAblico

wikipedia. (17 de noviembre de 2023). *wikipedia*. Obtenido de Cambiar a la tabla de contenidos:
https://es.wikipedia.org/wiki/Autob%C3%A9s_urbano

wikipedia. (31 de Octubre de 2023). *wikipedia*. Obtenido de Cambiar a la tabla de contenidos:
https://es.wikipedia.org/wiki/Sistema_de_informaci%C3%B3n_geogr%C3%A1fica

wikipedia. (31 de Octubre de 2023). *wikipedia*. Obtenido de Sistema de información geográfica :
https://es.wikipedia.org/wiki/Sistema_de_informaci%C3%B3n_geogr%C3%A1fica

wikipedia. (2024). *GPS*. Obtenido de <https://es.wikipedia.org/wiki/GPS>

wikipedia. (2024). *OpenStreetMap*. Obtenido de <https://es.wikipedia.org/wiki/OpenStreetMap>

wikipedia. (s.f.). *autobus*. Obtenido de https://es.wikipedia.org/wiki/Autob%C3%A9s_urbano

wikipedia. (s.f.). *OpenStreetMap*. Obtenido de <https://es.wikipedia.org/wiki/OpenStreetMap>

wikipedia. (s.f.). *Pruebas de software*. Obtenido de
https://es.wikipedia.org/wiki/Pruebas_de_software

wikipedia. (3 de Octubre de 2023). *wikipedia*. Obtenido de Metodología:
<https://es.wikipedia.org/wiki/Metodolog%C3%ADa>

Workana. (2021). *Workana*. Obtenido de ¿Qué es HTML 5?: <https://i.workana.com/glosario/que-es-html-5/>

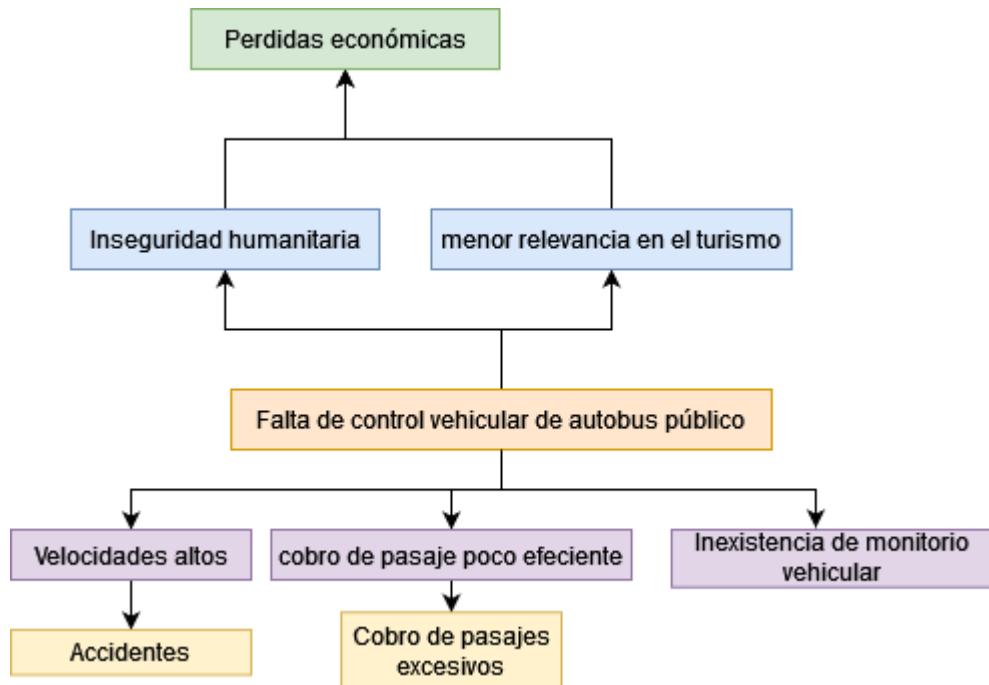
zapata. (24 de agosto de 2023). *Control de asistencia biométrico*. Obtenido de
<https://www.buk.cl/blog/control-de-asistencia-biometrico>

BIBLIOGRAFÍA

- García, L. (2019). *Desarrollo de un Sistema de Monitoreo Vehicular utilizando IoT*. Tesis de Licenciatura, Universidad Católica de Chile.
- Pressman, R. S. (2014). *Ingeniería del Software: Un Enfoque Práctico*. McGraw-Hill.
- Rodríguez, M. (2020). *Implementación de Metodologías Ágiles en el Desarrollo de Software*. Tesis de Maestría, Universidad San Mayor Andrés de La Paz.
- Sommerville, I. (2016). *Software Engineering*. Pearson Education.
- Visual Studio Code. (2022). *Visual Studio Code User Guide*. Microsoft Corporation.
- Android Studio. (2022). *Android Developer Guide*. Google LLC.

ANEXOS

Anexo 1. Árbol de Problemas



Anexo 2. Guía de entrevista para conductor

GUIA DE ENTREVISTA N°1	
Objetivo: obtener información detallada sobre el funcionamiento y proceso actual de sistema vehicular de la ciudad de Potosí	
Entrevistador (es): Univ. Santos machaca Lopez	Entrevistado(a): Adolfo Mamani Quispe (Conductor de la línea J)
Lugar: Nueva terminal de la ciudad de Potosí	Fecha: 25 de marzo de año 2024
Hora inicio: 10:30 AM	Hora de finalización: 11:30 AM
CUESTIONARIO	
1) ¿Usted utiliza un sistema de control vehicular que monitoree en tiempo real la ubicación de minibús?	
SI() NO()	
2) ¿Has trabajado anteriormente con dispositivos de seguimiento GPS o sistemas similares?	
SI() NO()	
3) ¿Si utilizo o utiliza un sistema de monitoreo explica el funcionamiento del sistema?	
4) ¿Qué le parecería la implementación de un sistema de control vehicular y monitoreo en tiempo real?	

- 5) ¿Qué desearía agregar al sistema de control y monitoreo vehicular?
- 6) ¿Qué medidas tomas para garantizar la seguridad de la carga y de los pasajeros durante los viajes?
- 7) ¿Tienes experiencia en la utilización de dispositivos de comunicación para reportar incidencias o coordinar rutas?
- 8) ¿Qué harías en caso de emergencia mientras estás en ruta?
- 9) ¿Cómo mantienes registros precisos de tus viajes y actividades relacionadas con el vehículo?
- 10) ¿Tiene experiencia en el uso de los dispositivos GPS?

Anexo 3. Guía de entrevista para administrador de sindicato

GUIA DE ENTREVISTA N°3	
Objetivo: obtener información detallada sobre el funcionamiento y proceso actual de sistema vehicular de la ciudad de Potosí	
Entrevistador (es): Univ. Santos machaca Lopez	Entrevistado(a): Miguel Checa Torrez (administrador del sindicato 27 de mayo)
Lugar: Nueva terminal de la ciudad de Potosí	Fecha: 25 de marzo de año 2024
Hora inicio: 10:30 AM	Hora de finalización: 11:30 AM
CUESTIONARIO	
1) ¿Usted utiliza un sistema de control vehicular que monitoree en tiempo real la ubicación de minibús?	
SI() NO()	
2) ¿Has trabajado anteriormente con dispositivos de seguimiento GPS o sistemas similares?	
SI() NO()	
3) ¿Si utilizo o utiliza un sistema de monitoreo explica el funcionamiento del sistema?	
4) ¿Qué le parecería la implementación de un sistema de control vehicular y monitoreo en tiempo real?	
5) ¿Qué desearía agregar al sistema de control y monitoreo vehicular?	

6) ¿Cómo es el control de los vehículos y los conductores?

7) ¿Tienes experiencia en la utilización de dispositivos de comunicación para reportar incidencias o coordinar rutas?

8) ¿Cómo se garantiza la seguridad a los ciudadanos o pasajeros?

Anexo 4. Encuesta a la población

Encuesta a la población
1. ¿Tiene conocimiento de sistemas de control y monitoreo vehicular de los microbuses? ¿Cuáles son?
2. ¿Le gustaría que se implemente un sistema para el monitoreo vehicular de los microbuses?
Si(<input type="checkbox"/>) No(<input type="checkbox"/>) No sé(<input type="checkbox"/>)
3. ¿Por qué le gustaría que se implemente un sistema de monitoreo para los microbuses?
4. ¿Cree usted que daría seguridad a los ciudadanos la implementación de un sistema de monitoreo y control de los microbuses?
Si(<input type="checkbox"/>) No(<input type="checkbox"/>) No sé(<input type="checkbox"/>)
5. ¿Crees que un sistema de monitoreo y control podría mejorar la puntualidad y eficiencia del servicio de minibuses?
Si(<input type="checkbox"/>) No(<input type="checkbox"/>) No sé(<input type="checkbox"/>)
6. ¿Qué características te gustaría que incluir el sistema de monitoreo y control para mejorar tu experiencia como pasajero?
7. ¿Qué funciones consideras esenciales en un sistema de monitoreo y control de minibuses? (ej. seguimiento en tiempo real, cámaras de seguridad, registro de velocidad, etc.)
8. ¿Qué tipo de información te gustaría poder acceder a través del sistema de monitoreo y control? (ej. ubicación del vehículo, comportamiento del conductor, estado mecánico del vehículo, etc.)

Anexo 5. Cronograma de actividades

Anexo 6. Respuestas de la entrevista al conductor

GUIA DE ENTREVISTA N°2	
Objetivo: obtener información detallada sobre el funcionamiento y proceso actual de sistema vehicular de la ciudad de Potosí.	
Entrevistador (es): Univ. Santos Machaca Lopez	Entrevistado(a): Adolfo Mamani Quispe (Conductor de la línea J)
Lugar: Cuarto Anillo Av. Beni de la ciudad de Santa Cruz de la Sierra	Fecha: 04 de marzo de año 2024
Hora inicio: 14:30	Hora de finalización: 15:30
CUESTIONARIO	
1) ¿Usted utiliza un sistema de control vehicular que monitoree en tiempo real la ubicación de minibús?	SI() NO(X)
2) ¿Has trabajado anteriormente con dispositivos de seguimiento GPS o sistemas similares?	SI() NO(X)
3) ¿Si utilizo o utiliza un sistema de monitoreo explica el funcionamiento del sistema?	R. No, aun no existe el sistema de monitoreo de vehículos en nuestra ciudad.
4) ¿Qué le parecería la implementación de un sistema de control vehicular y monitoreo en tiempo real?	

R. Sería de gran ventaja para los pasajeros, porque ya obtendrían sus datos del vehículo en tiempo real, el viaje es más seguro y el tiempo que va llegar a si destino.

5) ¿Qué desearía agregar al sistema de control y monitoreo vehicular?

R. Se podría agregar cámaras de cabina para detectar sus signos fatiga, distracciones esto ayudaría a alertas en tiempo real, el reconocimiento facial para monitorear su estado de alerta

6) ¿Qué medidas tomas para garantizar la seguridad de la carga y de los pasajeros durante los viajes?

R. Primeramente, se inspecciona el mantenimiento del vehículo, Capacitación del conductor y saber los primeros auxilios y concentración en el viaje y trazando la ruta con GPS.

7) ¿Tienes experiencia en la utilización de dispositivos de comunicación para reportar incidencias o coordinar rutas?

R. Solo utilizados nuestros teléfonos (celular) para reportar incidencias el caso de rutas la ubicación de GPS.

8) ¿Qué harías en caso de emergencia mientras estás en ruta?

R. El vehículo debería tener sus propios botiquines de emergencias, martillos, y también el chófer debe saber de los primeros auxilios qué se presenta en los viajes.

9) ¿Cómo mantienes registros precisos de tus viajes y actividades relacionadas con el vehículo?

R. En planillas llenando manualmente todos los datos de los pasajeros y tanto del vehículo queda registrado en las planillas.

10) ¿Tiene experiencia en el uso de los dispositivos GPS?

R. Si, el sistema GPS solo muestra el movimiento del pasajero y no traza las rutas del del vehículo de las líneas que trabajamos dentro de nuestra ciudad

Anexo 7. Respuesta de la guía de entrevista administrador

GUIA DE ENTREVISTA N°3	
Objetivo: obtener información detallada sobre el funcionamiento y proceso actual de sistema vehicular de la ciudad de Potosí	
Entrevistador (es): Univ. Santos machaca Lopez	Entrevistado(a): Miguel Checa Torrez (administrador del sindicato 27 de mayo)
Lugar: Nueva terminal de la ciudad de Potosí	Fecha: 25 de marzo de año 2024
Hora inicio: 10:30 AM	Hora de finalización: 11:30 AM
CUESTIONARIO	
1) ¿Usted utiliza un sistema de control vehicular que monitoree en tiempo real la ubicación de minibús?	SI() NO(X) 2) ¿Has trabajado anteriormente con dispositivos de seguimiento GPS o sistemas similares? SI() NO(X) 3) ¿Si utilizo o utiliza un sistema de monitoreo explica el funcionamiento del sistema? R. No, no utilizo ningún sistema de control vehicular ni en caso de emergencias.
4) ¿Qué le parecería la implementación de un sistema de control vehicular y monitoreo en tiempo real?	

R. Sería altamente beneficioso que monitoree la velocidad y la ubicación para la mejora de la seguridad de los pasajeros.

5) ¿Qué desearía agregar al sistema de control y monitoreo vehicular?

R. Se podría agregar cámaras de cabina para detectar sus signos fatiga, distracciones esto ayudaría a alertas en tiempo real, el reconocimiento facial para monitorear su estado de alerta.

6) ¿Cómo es el control de los vehículos y los conductores?

R. Mediante una planilla manual tanto del vehículo y conductor y la hora que sale de la parada.

7) ¿Tienes experiencia en la utilización de dispositivos de comunicación para reportar incidencias o coordinar rutas?

R. No, tengo experiencia del dispositivo de dispositivos para reportar incidencias.

8) ¿Cómo se garantiza la seguridad a los ciudadanos o pasajeros?

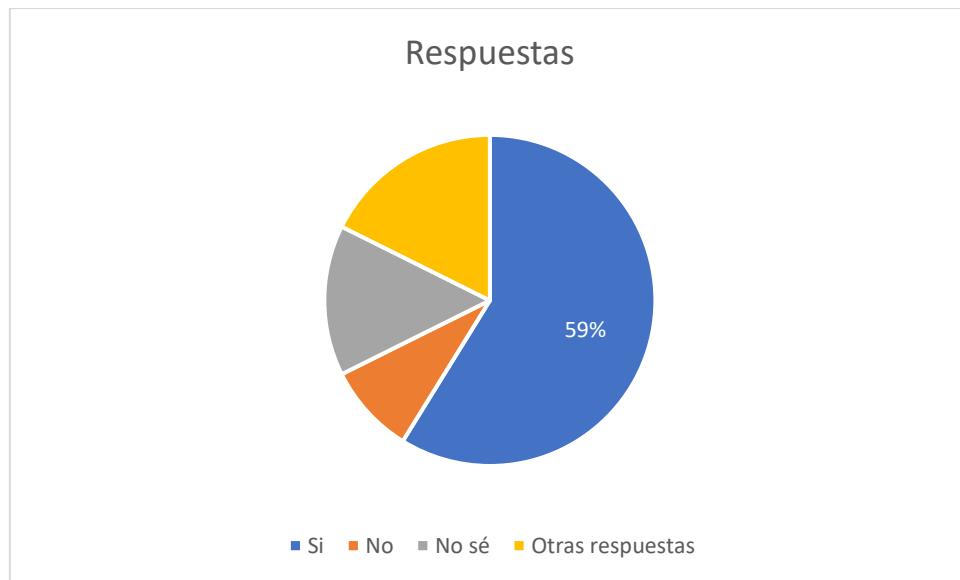
R. Se garantiza mediante la presencia policial, cámaras de vigilancia esto forma la seguridad de los ciudadanos de diversas amenazas.

Anexo 8. Respuestas de las encuestas

Respuestas	Cantidad
Si	200
No	30
No sé	50
Otras respuestas	60

Fuente: Elaboración propia

Anexo 9. Grafica de las respuestas



Fuente: Elaboración propia

Anexo 10. Obtención de costos por COCOMO

Cálculo de cuenta total

Cálculo Cuenta Total			
Parámetros	Nº de Parámetros	Peso	Cuenta
Número de Entradas	90	6	540
Número de Salidas	30	5	150
Número de Peticiones	35	5	175
Número de Archivos	5	7	35
Número de Interfaz Externa	15	4	60
Algoritmos	15	9	135
Cuenta Total			1095

Fuente: Elaboración propia

Cálculo de factor de complejidad

Calculo Factor de Complejidad		RESPUESTA
Nº	INTERROGANTE	
1	Requiere el sistema de copias de seguridad	2
2	Se requiere comunicación de datos	5
3	Existe funciones de procedimiento distribuido	5
4	Es critico el funcionamiento	4
5	Se ejecutara el sistema en un entorno operativo existente y fuertemente utilizado	2
6	Requiere el sistema entrada de datos interactivos	5
7	Requiere de datos interactivos que la transacción de entrada se lleve a cabo sobre múltiples pantallas	5
8	Se actualizaran los archivos maestros de forma interactivo	1
9	Son complejos las E/S los archivos o las peticiones	4
10	Es complejo el procesamiento interno	5
11	Se ha diseñado el código para ser reutilizable	4
12	Esta incluido en el diseño la conversión y la instalación	4
13	Se ha diseñado el sistema para soportar múltiples instalaciones en diferentes organizaciones	5
14	Se ha diseñado la aplicación para facilitar los cambios y para ser fácilmente utilizado por el usuario	4
	TOTAL	55
	Fi	4

Fuente: Elaboración propia

Cálculo de Pc(nominal)

Cálculo PC(Nominal)	
Pc(nominal) = Cuenta_total*(0,65+0,01*Prom(Fi))	
Pc(nominal)=	754,768
Donde el Pc(real) se obtiene restando el porcentaje de reutilización que en este caso para el proyecto es de 35%	
Pc(real)=Pc(nominal)-Pc(nominal)*0,35=	490,6

Fuente: Elaboración propia

Estimación de costos Cocomo

TECNICA DE ESTIMACIÓN DE COSTOS: COCOMO

Aplicando el **COCOMO** como técnica de estimación de costos, considerando al proyecto como un proyecto semiacoplado se tiene:

Tipo de proyecto	ab	bb	cb	db	
Empotrado	3.6	1.2	2.5	0.32	Del texto

Fuente: Elaboración propia

Cálculo de costo Sia

$$\text{Esfuerzo} = 3.6 * (\text{PC(real)})^{1.2}$$

$$\text{Esfuerzo}= 6097,8$$

$$\text{Duración} = 2.5 * (\text{Esfuerzo})^{0.32} = 40,7$$

Donde la duración del proyecto es de 35 semanas, 9 meses

Donde la duración del proyecto es de 35 semanas, 4 meses y medio

Cálculo del costo del SIA

Para realizar el cálculo del desarrollo del SIA se debe primero establecer un conjunto de métricas como:

Productividad= Capacidad de desarrollo por unidad de tiempo

Donde la productividad al tratarse de un equipo de desarrollo se aplicará el $VE(a+4*m+b)/6$ porque $N=3$

$$\text{Productividad}= 4$$

Donde:

Linea Base: (**FUENTE SUELDOS INSTITUCIONES POTOSINAS**)

(AAPOS: 3500 GAMP:6500 SEPSA:7000)

$$VE(\text{costo}) = (3500+4*6500+7000)/6 = 6083,3$$

Costo por Pc = $VE(\text{costo})/\text{Productividad}$

$$\text{Costo por Pc} = 1520,8$$

$$\text{Finalmente el Costo del SIA=} \quad 746119,5 \text{ Bs.}$$

Fuente: Elaboración propia

Anexo 11. Diccionario de datos

NOMBRE DEL ARCHIVO: Autenticación			
CAMPO	TAMAÑO	TIPO DE DATO	RELACIÓN
email	30	String	
password	30	String	
delete		Boolean	
id	30	String	Llave Primaria

Fuente: Elaboración propia

NOMBRE DEL ARCHIVO: Buses			
CAMPO	TAMAÑO	TIPO DE DATO	RELACIÓN
Marca	30	String	
Modelo		Number	
Tipo	30	String	
Placa	10	String	
Cantidad		Number	
Foto	100	String	
Ruat	100	String	
Estados	30	String	
idUser	30	String	Llave Foránea
idLocalizacion	30	String	Llave Foránea
delete		Boolean	
id	30	String	Llave Primaria

Fuente: Elaboración propia

NOMBRE DEL ARCHIVO: Horarios			
CAMPO	TAMAÑO	TIPO DE DATO	RELACIÓN
Nombre_Horario	30	String	
Lugar_Salida	30	String	
Hora_Salida	30	String	
Hora_Ult_Salida	30	String	
Dias		Array	

Otro_Dia	30	String	
Descripcion		Text	
Delete		Boolean	
Id	30	String	Llave Primaria

Fuente: Elaboración propia

NOMBRE DEL ARCHIVO: Licencias_Conductor			
CAMPO	TAMAÑO	TIPO DE DATO	RELACIÓN
Categoría	30	String	
Fecha_Emision		Date	
Fecha_Expiracion		Date	
Foto_Delante	30	String	
Foto_Atras	30	String	
delete		Boolean	
id	30	String	Llave Primaria

Fuente: Elaboración propia

NOMBRE DEL ARCHIVO: Lineas			
CAMPO	TAMAÑO	TIPO DE DATO	RELACIÓN
Nombre_Linea	30	String	
Cantidad_Solicitados		Number	
idRutas	30	String	Llave Foránea
idHorario		Array	Llave Foránea
idTarifas		Array	Llave Foránea
idBuses		Array	Llave Foránea
delete		Boolean	
Id	30	String	Llave Primaria

Fuente: Elaboración propia

NOMBRE DEL ARCHIVO: Localizaciones			
CAMPO	TAMAÑO	TIPO DE DATO	RELACIÓN
Coordenadas		Array	
Velocidad		Number	

Distancia		Number	
Estados	10	String	
id	30	String	Llave Primaria

Fuente: Elaboración propia

NOMBRE DEL ARCHIVO: Tarifas			
CAMPO	TAMAÑO	TIPO DE DATO	RELACIÓN
Tarifa		Array	
Nombre_Tarifa	30	String	
Descripcion		Text	
Fecha_Creacion		Date	
delete		Boolean	
id	30	String	Llave Primaria

Fuente: Elaboración propia

NOMBRE DEL ARCHIVO: Rutas			
CAMPO	TAMAÑO	TIPO DE DATO	RELACIÓN
Nombre_Ruta	30	String	
GeoJson		Json	
Zoom		Number	
Coordenadas		Array	
delete		Boolean	
id	30	String	Llave Primaria

Fuente: Elaboración propia

NOMBRE DEL ARCHIVO: Roles y Permisos			
CAMPO	TAMAÑO	TIPO DE DATO	RELACIÓN
Nombre_Rol	30	String	
Permito		Array	
Delete		Boolean	
Id	30	String	Llave Primaria

Fuente: Elaboración propia

NOMBRE DEL ARCHIVO: Usuarios			
CAMPO	TAMAÑO	TIPO DE DATO	RELACIÓN
Nombre_Usuario	30	String	
Apellidos	30	Number	
Genero	30	String	
Ci	15	String	
Celular	20	String	
Direccion	30	String	
Pais	30	String	
Email	30	String	
Foto	30	String	
Estado_Conexion	10	String	
Ult_Conexion		Date	
idRol		Array	Llave Foránea
idLicencia	30	String	Llave Foránea
delete		Boolean	
id	30	String	Llave Primaria

Fuente: Elaboración propia

MANUAL DE USUARIO

NOMBRES Y APELLIDOS	CI	DIRECCIÓN	CELULAR	GÉNERO	NACIONALIDAD	ROL	ACCIONES
juan carlos rodriguez gasdfhja@gmail.com	3435354	sdfsdfs	3453535	Masculino	Bolivia	Niguno	⋮
manuel torrez asdas	34535	xcsxsd	3453534	Masculino	Bolivia	Niguno	⋮
ghggvhgh ,m,m,m hjhmgf@gmail.com	9889879	llkhjg	6676767	Masculino	Afganistán	Niguno	⋮
dussan canaviri dussan@gmail.com	10531753	Av. las bander...	72381722	Masculino	Bolivia	Niguno	⋮
Jhony Contreras Vas jhony@gmail.com	1053175	Av. los pinos	72381722	Masculino	Bolivia	Niguno	⋮
Jheny Benitez Relos Jheny@gmail.com	1385678	Av. Las bande...	72381722	Femenina	Bolivia	Niguno	⋮

Rows per page: 10 ⋮ 1-6 of 6 < >

© 2024, Desarrollador Santos Machaca Lopez ❤ Email santosjared221234@gmail.com

El botón filtrar por columnas al hacer click en ella se despliega una pequeña columna para insertar datos y filtrar mediante ella, el botón nuevo usuario despliega un formulario para insertar los datos del usuario.

Registro de usuarios

FILTRAR POR COLUMNAS

NOMBRES Y APELLIDOS	CI	DIRECCIÓN	CELULAR
juan carlos rodriguez gasdfhja@gmail.com	3435354	sdfsdfs	3453535
manuel torrez asdas	34535	xcsxsd	3453534
ghggvhgh ,m,m,m hjhmgf@gmail.com	9889879	llkhjg	6676767
dussan canaviri dussan@gmail.com	10531753	Av. las bander...	72381722
Jhony Contreras Vas jhony@gmail.com	1053175	Av. los pinos	72381722
Jheny Benitez Relos Jheny@gmail.com	1385678	Av. Las bande...	72381722

© 2024, Desarrollador Santos Machaca Lopez ❤ Email santosjared221234@gmail.com

Agregar Nuevo Usuario

👤

👁

+ AGREGAR LICENCIA

Al hacer click en el botón de agregar licencia se despliega nuevos campos para agregar la licencia de conducir del usuario.

The screenshot shows the Microbuses application's user registration interface. On the left, a sidebar menu includes options like Home, Monitoreo, Registro de usuarios, Roles y Permisos, Registro de microbuses, Registro de tarifas, Registro de horarios, Registro de rutas, and Registro de líneas. The main area displays a table of registered users with columns for NOMBRES Y APELLIDOS, CI, DIRECCIÓN, and CELULAR. Below the table is a copyright notice: © 2024, Desarrollador Santos Machaca Lopez Email santosjared221234@gmail.com. To the right, a modal window titled 'Registro de usuarios' contains fields for 'SELECCIONAR FOTO DE LICENCIA FRONTAL' and 'SELECCIONAR FOTO DE LICENCIA TRASERA'. It also includes a 'NO AGREGAR LICENCIA' link, a 'CANCEL' button, and a 'GUARDAR' button, which is highlighted with a red border.

El botón cancelar permite cancelar el registro de usuario, al hacer click en ella se cierra el formulario y se restablece el formulario, y el botón guardar es para guardar datos del usuario.

The screenshot shows the Microbuses application's user profile details. The sidebar menu is identical to the previous screenshot. The main area features a modal window titled 'Licencia de conducir' containing a placeholder user image, the name 'ghggvhgh ,m,m,m', and details such as Cédula de Identidad: 9889879, Correo Electrónico: hjhhgfg@gmail.com, and Estado de conexión: Nunca. Below this is a section for 'Licencia de Conducir' with fields for Fecha emisión: 28/05/2024, Fecha expiración: 27/06/2027, and Categoría: A, accompanied by a small image of a driver's license. To the right of the modal is a table titled 'NUEVO USUARIO' with columns for NACIONALIDAD, ROL, and ACCIONES. The table lists several users from Bolivia and Afghanistan, each with a three-dot menu icon. At the bottom of the page are pagination controls: Rows per page: 10, 1–6 of 6, and navigation arrows.

Este modal se despliega cuando se hace click en los 3 puntos en la columna de opciones, muestra los detalles del usuario.

Registro de usuarios

FILTRAR RESTABLECER

USUARIOS	CI	DIRECCIÓN	CELULAR	GÉNERO	NACIONALIDAD	ROL	AACCIONES
juan carlos rodríguez gasdfhja@gmail.com	3435354	sdfsdfs	3453535	Masculino	Bolivia	Niguno	⋮
manuel torrez asdas asfadsf@gmail.com	34535	xcsxsd	3453534	Masculino	Bolivia	Niguno	⋮
ghggvghg,m,m,m jhjhgf@gmail.com	9889879	llkhhjg	6676767	Masculino	Afganistán	Niguno	⋮
dussan canaviri dussan@gmail.com	10531753	Av. las bande...	72381722	Masculino	Bolivia	Niguno	⋮
Jhony Contreras Vas jhony@gmail.com	1053175	Av. los pinos	72381722	Masculino	Bolivia	Niguno	⋮
Jheny Benitez Relos Jheny@gmail.com	1385678	Av. Las bande...	72381722	Femenina	Bolivia	Niguno	⋮

Rows per page: 10 1-6 of 6 < >

Este campo se habilita cuando se hace click en el botón de filtrar por columnas, estos campos sirven para realizar filtrados por la columna. El botón filtrar filtra los datos al hacer click, de los campos introducidos, y el botón restablecer restablecerá a su estado inicial de las tablas y resetea los campos de filtrado.

Lista de Microbuses

FILTRAR POR COLUMNAS NUEVO MICROBUS

MARCA	MODELO	TIPO	PLACA	ASIENTOS	CHOFER	RUAT	ESTADO	AACCIONES
Mercedes Benz	2006	Bus	HGV 778	28	Juan carlos rodríguez 3435354	Abrir	Activo	⋮
Tata	1990	Microbus	SAX 897	32	dussan canaviri 10531753	Abrir	Activo	⋮
Nissan	1990	Microbus	ZSD 3232	16	Jhony Contreras Vas 1053175	Sin Docu...	En Manteni...	⋮
Nissan	1990	Microbus	ZSD 789	34	manuel torrez asdas 34535	Sin Docu...	Inactivo	⋮
Nissan	1990	Microbus	xdfg 678	16	No asignado	Sin Docu...	Proceso	⋮
Nissan	1990	Microbus	GHIY 890	32	Jheny Benitez Relos 1385678	Abrir	Activo	⋮
Foton	1990	Microbus	dfg78	16	No asignado	Sin Docu...	Activo	⋮
Nissan	1990	Microbus	uyi909	16	No asignado	Sin Docu...	Activo	⋮

Rows per page: Speaker (Realtek(R) Audio): 52% < >

Este campo se habilita cuando se hace click en los botones de registro de microbuses de la barra lateral, los botones filtrar por columnas son para habilitar campos de filtrado, y el botón nuevo microbús, al hacer click en ella desplegará un formulario para introducir los datos de los microbuses.

The screenshot shows the 'Lista de Microbuses' (List of Microbuses) page. At the top, there are buttons for 'CERRAR FILTRADO' (Close Filter), 'NUEVO MICROBUS' (New Microbus), and a gear icon. Below these are two buttons: 'FILTRAR' (Filter) and 'RESTABLECER' (Reset). A red box highlights the 'FILTRAR' button and its associated input fields. The main table has columns: Marca, Modelo, Tipo, Placa, Asientos, Chofer, Ruat, Estado, and Acciones. The table contains several rows of microbus data, each with a small profile picture, name, ID, and status.

Este campo se habilita cuando se hace click en el botón de filtrar por columnas, estos campos sirven para realizar filtrados por la columna. El botón filtrar filtra los datos al hacer click, de los campos introducidos, y el botón restablecer restablecerá a su estado inicial de las tablas y resetea los campos de filtrado.

The screenshot shows the 'Nuevo Microbus' (New Microbus) form. It includes fields for 'Marca' (Nissan), 'Tipo de Vehículo' (Microbus), 'Modelo' (1990), 'Placa del Vehículo' (empty), 'Cantidad de asientos' (16), 'Estado de Vehículo' (Activo), and a file upload field for 'Subir documento pdf de Ruat'. At the bottom are 'CANCEL' and 'GUARDAR' buttons, with the 'GUARDAR' button highlighted by a red box.

El formulario se habilita al hacer click en el botón de nuevo microbús, es para la inserción de los datos de los microbuses, el botón cancelar cancela el registro de datos, cierra el formulario y restablece los campos, y el botón guardar es para guardar los datos del formulario.

The screenshot shows a list of buses with columns: MARCA, MODELO, TIPO, PLACA, ASIENTOS, CHOFER, RUAT, ESTADO, and ACCIONES. The 'ACCIONES' column contains three buttons: 'Abrir' (Open), 'Activo' (Active), and a more options menu. The 'Detalles' button is highlighted with a red box.

MARCA	MODELO	TIPO	PLACA	ASIENTOS	CHOFER	RUAT	ESTADO	ACCIONES
Mercedes Benz	2006	Bus	HGV 778	28	juan carlos rodriguez 3435354	Abrir	Activo	...
Tata	1990	Microbus	SAX 897	32	dussan canaviri 10531753	Abrir	Activo	...
Nissan	1990	Microbus	ZSD 3232	16	Jhony Contreras Vas 1053175	Sin Docu...	En M...	...
Nissan	1990	Microbus	ZSD 789	34	manuel torrez asdas 34555	Sin Docu...	Inactivo	...
Nissan	1990	Microbus	xdfg 678	16	No asignado	Sin Docu...	Proceso	...
Nissan	1990	Microbus	GHIY 890	32	Jheny Benitez Relos 1385678	Abrir	Activo	...
Foton	1990	Microbus	dfg78	16	No asignado	Sin Docu...	Activo	...
Nissan	1990	Microbus	uyi909	16	No asignado	Sin Docu...	Activo	...

Estos campos se habilitan al hacer click en los tres puntos, que son para ver los detalles del chofer, editar y eliminar los buses.

The modal window displays the following information:

- Driver: Jhony Contreras Vasquez
- Cédula de Identidad: 1053175
- Correo Electrónico: jhony@gmail.com
- Estado de conexión: 13/06/2024
- DESASIGNAR CHOFER (highlighted with a red box)
- Licencia de Conducir
- Fecha emisión: 18/06/2024
- Fecha expiración: 01/06/2027
- Categoría: A
- Image of a driver's license card

El modal es visible cuando se hacer click en el botón de detalles que esta en la columna de acciones.

The screenshot shows the 'Lista de usuarios' (List of users) modal open over a table of microbus registrations. The modal has a search bar labeled 'Bucar usuarios' and displays '1 Usuarios de 1' user found, with a preview of the user's profile picture and ID. The background table lists various microbuses with columns for MARCA, MODELO, AÑO, TIPO, PLACA, CANTIDAD DE ASIENTOS, ESTADO, and ACCIONES.

El modal es visible cuando se hace click en botón asignar chofer que está en la columna acciones, es para asignar chofer, muestra a todos los choferes no asignados, y al hacer click en uno de ellos se asignará el chofer.

The screenshot shows the 'Registro de tarifas' (Registration of tariffs) page. It features a 'FILTRAR POR COLUMNAS' (Filter by columns) button, a 'NUEVO TARIFA' (New tariff) button, and a table listing five tariff entries. Each entry includes fields for NOMBRE DE TARIFAS, FECHA DE CREACIÓN, TARIFAS, and ACCIONES. The URL at the bottom is 'localhost:4001/minibus/tarifas/'.

Este campo se habilita cuando se hace click en los botones de registro de tarifas de la barra lateral, los botones filtrar por columnas son para habilitar campos de filtrado, y el botón nuevo tarifa, al hacer click en ella desplegará un formulario para introducir los datos de tarifas.

Este campo se habilita cuando se hace click en el botón de filtrar por columnas, estos campos sirven para realizar filtrados por la columna. El botón filtrar filtra los datos al hacer click, de los campos introducidos, y el botón restablecer restablecerá a su estado inicial de las tablas y resetea los campos de filtrado.

El formulario se muestra al hacer click en el boton nuevo tarifa, permite el registro de las trifas, el botón cancelar cancela el registro de tarifa al hacer click en ella se cierra el formulario y se resetea. El botón guardar es para guardar la información introducida de las tarifas.

Registro de tarifas

FILTRAR POR COLUMNAS

NOMBRE DE TARIFAS

TARIFA 012

Escolar: Bs. 0.50
Universitario: Bs. 1
Adulto: Bs. 1.50
Policia: Bs. 0.00

Descripción

estos es la tarifa

ACCIONES

NUEVO TARIFA

Rows per page: 10 1-5 of 5 < >

© 2024, Desarrollador Santos Machaca Lopez Email santosjared221234@gmail.com

El modal es visible cuando se hace click en el botón ver tarifas, muestra las tarifas creadas.

Registro de horario

FILTRAR POR COLUMNAS

NOMBRE DE HORARIO LUGAR DE SALIDA H. PRIM. SALIDA H. ÚLT. SALIDA DÍAS ACCIONES

Horario vuelta 012	villa Santiago	00:00	03:00	<input checked="" type="checkbox"/> ver días	⋮
horario de linea F	nueva terminal	02:00	06:02	<input checked="" type="checkbox"/> ver días	⋮
horario Ida	Calle salta	00:00	05:00	<input checked="" type="checkbox"/> ver días	⋮
ida 012	villa copacabana	00:00	05:00	<input checked="" type="checkbox"/> ver días	⋮
Horario vuelta	calle santa tereza	05:00	23:00	<input checked="" type="checkbox"/> ver días	⋮

NUEVO HORARIO

Rows per page: 10 1-5 of 5 < >

© 2024, Desarrollador Santos Machaca Lopez Email santosjared221234@gmail.com

Este campo se habilita cuando se hace click en los botones de registro de horarios de la barra lateral, los botones filtrar por columnas son para habilitar campos de filtrado, y el botón nuevo tarifa, al hacer click en ella desplegará un formulario para introducir los datos de tarifas. Las filas ver días, al hacer click en ella se mostrará un modal con los días asignados.

OPERACION REGULAR DESDE 23:00 A 04:00 DE LA MAÑANA

Día	Horarios de operación	frecuencia (min)
Lunes	23:00 - 04:00	15
Martes	23:00 - 04:00	15
Miércoles	23:00 - 04:00	15

Descripción

Este sistema de registro de horarios es una herramienta eficiente y fácil de usar, diseñada para gestionar y monitorear los tiempos de entrada y salida del personal. Este sistema permite un seguimiento detallado y preciso de las horas trabajadas, facilitando la administración del tiempo y la planificación de recursos.

El modal días es visible cuando se hace click en los botones ver días que está en la columna de días, muestra los días de horario.

Registro de Horarios

NOMBRE DE HORARIO	LUGAR DE SALIDA	H. PRIM. SALIDA	H. ULTIMA SALIDA
Horario vuelta 012	villa Santiago	00:00	03:00
horario de linea F	nueva terminal	02:00	06:00
horario Ida	Calle salta	00:00	05:00
ida 012	villa copacabana	00:00	05:00
Horario vuelta	calle santa tereza	05:00	23:00

Registro de Horario

DÍAS

Lunes Martes Miércoles
 Jueves Viernes Sábado
 Domingo Feriados Otro

Hora primera Salida: 00:00 Hora última Salida: 00:00

Lugar de salida: Nombre de horario:

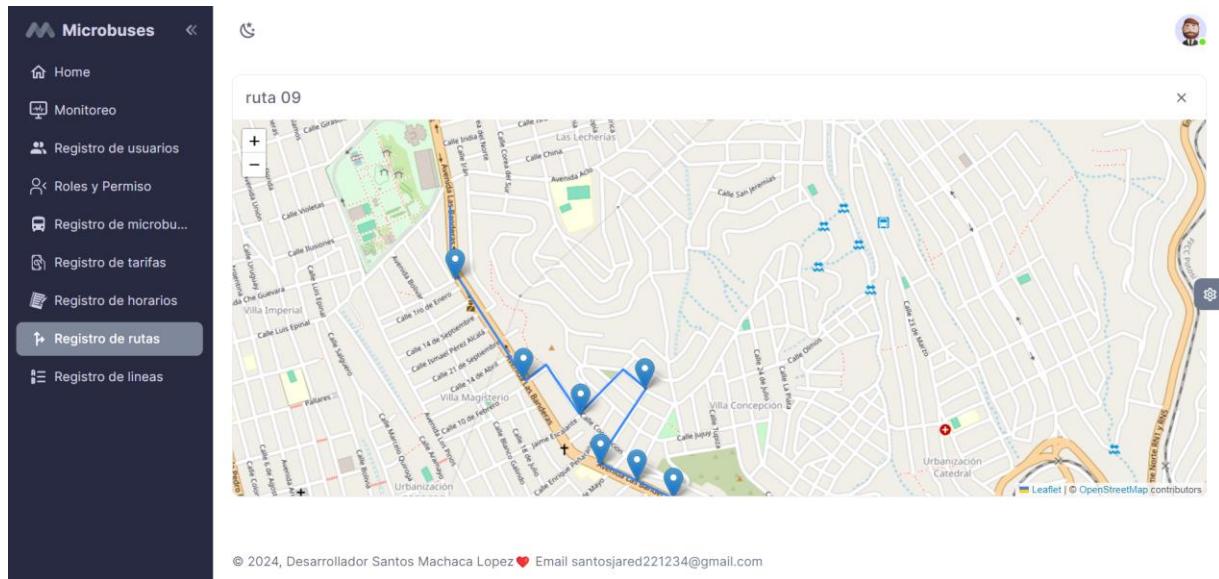
Descripción:

CANCEL **GUARDAR**

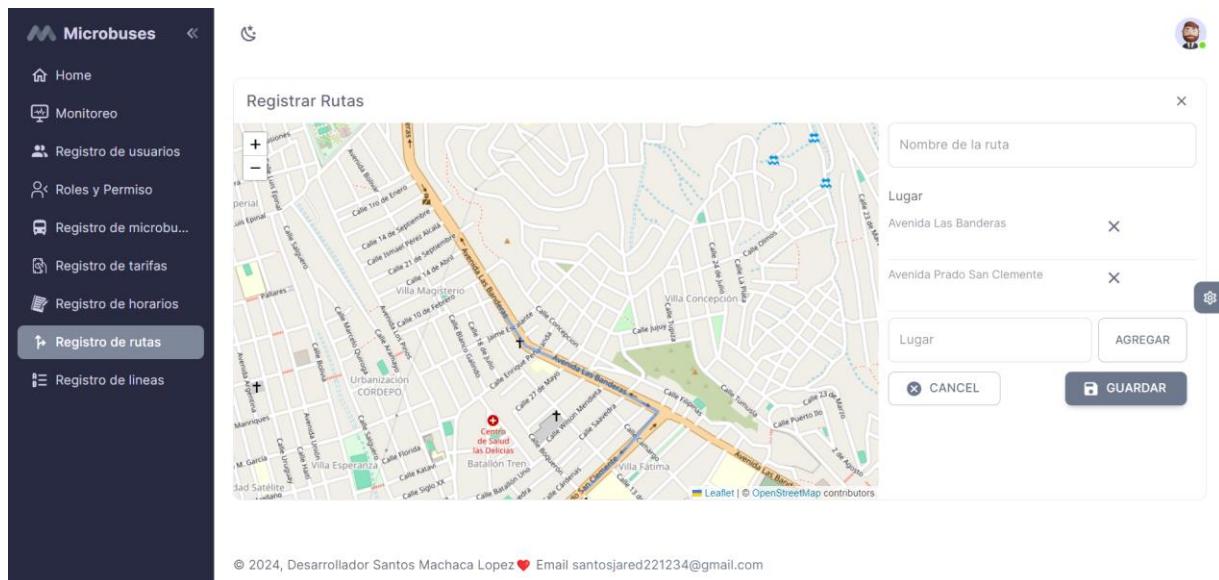
El formulario es visible cuando se hace click en el botón nuevo horario, es para registro de horario de salidas de los microbuses, el botón cancelar, al hacer click en ella se cancela el registro de horario, se cierra el formulario y se restablece el formulario y el botón guardar al hacer click en ella se guarda los datos.

Este campo se habilita cuando se hace click en el botón de filtrar por columnas, estos campos sirven para realizar filtrados por la columna. El botón filtrar filtra los datos al hacer click, de los campos introducidos, y el botón restablecer restablecerá a su estado inicial de las tablas y resetea los campos de filtrado.

Este campo se habilita cuando se hace click en los botones de registro de tarifas de la barra lateral, los botones filtrar por columnas son para habilitar campos de filtrado, y el botón nuevo tarifa, al hacer click en ella desplegará un formulario para introducir los datos de tarifas.

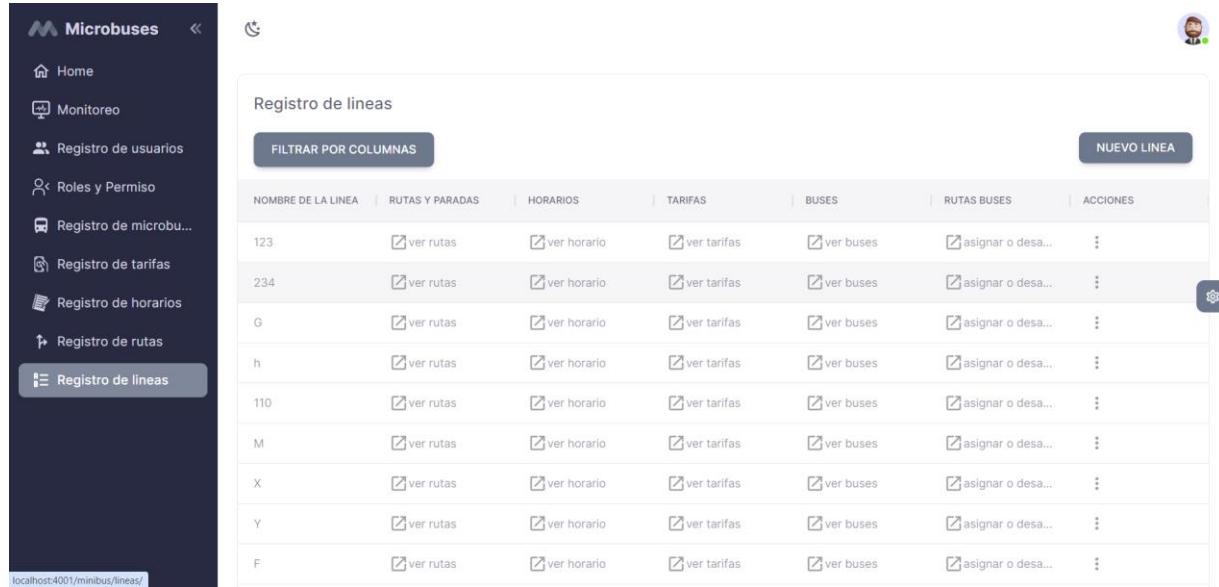


La interfaz se muestra la ruta dibujada en el registro de ruta, este interface se abre cuando se hace click en botón mostrar rutas.



La interfaz registrar ruta se abre cuando se hace click en el botón nuevo ruta, en esta interface se puede dibujar rutas, el botón finish es para terminar el dibujado, el botón delete las point es para borrar el ultimo punto y el botón cancel es para cancelar la acción, el botón con icono linea recta es para dibujar ruta, el botón con icono rombo es para ubicar las paradas.

El botón de cancelar es para cancelar el registro al hacer click en ella se reseteará el formulario se cerrará, el botón guardar es para guardar la información generada.



NOMBRE DE LA LINEA	RUTAS Y PARADAS	HORARIOS	TARIFAS	BUSES	RUTAS BUSES	ACCIONES
123	<input type="checkbox"/> ver rutas	<input type="checkbox"/> ver horario	<input type="checkbox"/> ver tarifas	<input type="checkbox"/> ver buses	<input type="checkbox"/> asignar o desa...	⋮
234	<input type="checkbox"/> ver rutas	<input type="checkbox"/> ver horario	<input type="checkbox"/> ver tarifas	<input type="checkbox"/> ver buses	<input type="checkbox"/> asignar o desa...	⋮
G	<input type="checkbox"/> ver rutas	<input type="checkbox"/> ver horario	<input type="checkbox"/> ver tarifas	<input type="checkbox"/> ver buses	<input type="checkbox"/> asignar o desa...	⋮
h	<input type="checkbox"/> ver rutas	<input type="checkbox"/> ver horario	<input type="checkbox"/> ver tarifas	<input type="checkbox"/> ver buses	<input type="checkbox"/> asignar o desa...	⋮
110	<input type="checkbox"/> ver rutas	<input type="checkbox"/> ver horario	<input type="checkbox"/> ver tarifas	<input type="checkbox"/> ver buses	<input type="checkbox"/> asignar o desa...	⋮
M	<input type="checkbox"/> ver rutas	<input type="checkbox"/> ver horario	<input type="checkbox"/> ver tarifas	<input type="checkbox"/> ver buses	<input type="checkbox"/> asignar o desa...	⋮
X	<input type="checkbox"/> ver rutas	<input type="checkbox"/> ver horario	<input type="checkbox"/> ver tarifas	<input type="checkbox"/> ver buses	<input type="checkbox"/> asignar o desa...	⋮
Y	<input type="checkbox"/> ver rutas	<input type="checkbox"/> ver horario	<input type="checkbox"/> ver tarifas	<input type="checkbox"/> ver buses	<input type="checkbox"/> asignar o desa...	⋮
F	<input type="checkbox"/> ver rutas	<input type="checkbox"/> ver horario	<input type="checkbox"/> ver tarifas	<input type="checkbox"/> ver buses	<input type="checkbox"/> asignar o desa...	⋮

Este campo se habilita cuando se hace click en los botones de registro de líneas de la barra lateral, los botones filtrar por columnas son para habilitar campos de filtrado, y el botón nuevo línea, al hacer click en ella desplegará un formulario para introducir los datos de tarifas.

La columna rutas y paradas es para ver las rutas y paradas de la línea, la columna horarios es para ver los horarios de la salida de la línea, la columna tarifas es para ver las tarifas de las líneas, la columna buses es para ver los buses asignados o no asignados a la línea.

Asignar o desasignar rutas a la linea 123

Rutas asignadas			
NRO	NOMBRE DE LA R...	PREVIA	DESASIGNAR
1	ruta de la linea...	<input checked="" type="checkbox"/> previa	>
2	ruta ida 012	<input checked="" type="checkbox"/> previa	>
3	lufa	<input checked="" type="checkbox"/> previa	>

Rutas no asignadas			
ASIGNAR	NRO	NOMBRE DE LA R...	PREVIA
<	1	yo	<input type="checkbox"/> previa
<	2	rutas las band...	<input type="checkbox"/> previa
<	3	x	<input type="checkbox"/> previa
<	4	rutas 32	<input type="checkbox"/> previa
<	5	rutas asignar	<input type="checkbox"/> previa

Rows per page: 10 ▾ 1-3 of 3 < >

Rows per page: 10 ▾ 1-5 of 5 < >

© 2024, Desarrollador Santos Machaca Lopez ❤ Email santosjared221234@gmail.com

Esta interfaz se despliega cuando se hace click en el botón ver rutas, el botón desasignar ruta desasigna la ruta. El botón asignar ruta aparece cuando no tiene asignado rutas, al hacer click en ella se muestra un menú de rutas.

ruta de la linea 012

Rutas

Avenida Las Banderas

El modal es visible cuando se hace click en botón asignar rutas, el campo buscar ruta permite la búsqueda de la ruta, al hacer click en una de las rutas se asignará la ruta.

NRO	NOMBRE DEL HO...	HORARIO	DESASIGNAR
1	serwrs	<input checked="" type="checkbox"/> ver horario	

Rows per page: 10 ▾ 1-1 of 1 < >

ASIGNAR	NRO	NOMBRE DEL HO...	HORARIO
	1	horario 102 ida	<input checked="" type="checkbox"/> ver horario
	2	horario de pru...	<input checked="" type="checkbox"/> ver horario
	3	sdfsfsdf	<input checked="" type="checkbox"/> ver horario
	4	sdfsf	<input checked="" type="checkbox"/> ver horario
	5	asdad	<input checked="" type="checkbox"/> ver horario
	6	werr	<input checked="" type="checkbox"/> ver horario
	7	asdadas	<input checked="" type="checkbox"/> ver horario
	8	asfafa	<input checked="" type="checkbox"/> ver horario
	9	asadads	<input checked="" type="checkbox"/> ver horario

La interfce de asignar o desasignar horarios a la linea g se despliega cuando se hace click ver horarios de einterface horarios.

Día	Horarios de operación	frecuencia (min)
Feriados	23:00 - 05:06	30
Jueves	23:00 - 05:06	30
Domingo	23:00 - 05:06	30

Este modal es visible cuando se hace click en el botón ver horario, en la parte izquierda muestra todos los horarios asignados y en la parte derecha muestra todos los horarios no asignados. Sin embargo, se puede asignar o desasignar los horarios de acuerdo a lo que indican los botones.

Asignar o desasignar tarifas a la linea G

NRO	NOMBRE DE LA T...	TARIFAS	DESASIGNAR
1	tarifa prepago	<input type="checkbox"/> ver tarifa	

Rows per page: 10 1-1 of 1

ASIGNAR	NRO	NOMBRE DE LA T...	TARIFAS
	1	tarifas2	<input type="checkbox"/> ver tarifa
	2	tarifaps	<input type="checkbox"/> ver tarifa
	3	tarifas10	<input type="checkbox"/> ver tarifa
	4	asdasd	<input type="checkbox"/> ver tarifa
	5	asdad	<input type="checkbox"/> ver tarifa
	6	qwee	<input type="checkbox"/> ver tarifa
	7	asadsa	<input type="checkbox"/> ver tarifa
	8	sdfsf	<input type="checkbox"/> ver tarifa
	9	qwewe	<input type="checkbox"/> ver tarifa

Las interfaces para asignar o desasignar rutas se despliegan cuando se hace click en el botón de asignar 'ver tarifa' para rutas.

Asignar o desasignar tarifas a la linea G

NRO	NOMBRE DE LA T...	TARIFAS
1	tarifa prepago	<input checked="" type="checkbox"/> ver tarifa

TARIFA PREPAGO

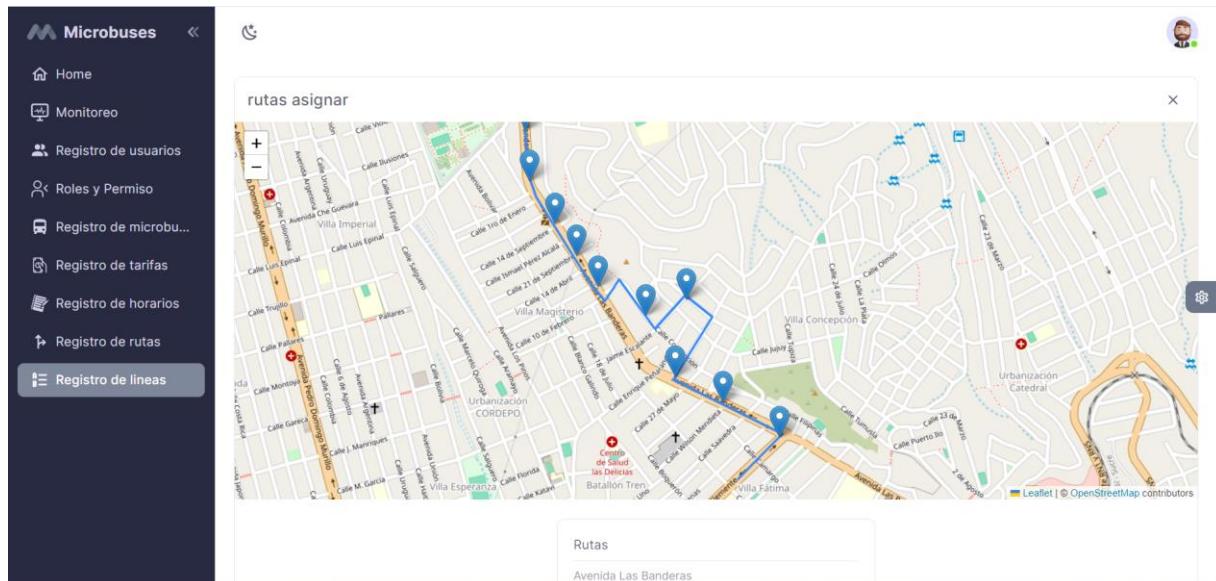
Escolar: Bs. 0.50
Universitario: Bs. 1
Adulto: Bs. 1.50
Tercera Edad: Bs. 1

NRO	NOMBRE DE LA T...	TARIFAS
1	tarifas2	<input type="checkbox"/> ver tarifa
2	tarifaps	<input type="checkbox"/> ver tarifa
3	tarifas10	<input type="checkbox"/> ver tarifa
4	asdasd	<input type="checkbox"/> ver tarifa
5	asdad	<input type="checkbox"/> ver tarifa
6	qwee	<input type="checkbox"/> ver tarifa
7	asadsa	<input type="checkbox"/> ver tarifa
8	sdfsf	<input type="checkbox"/> ver tarifa
9	qwewe	<input type="checkbox"/> ver tarifa

Este modal es visible cuando se hace click en el botón 'ver tarifa', en la parte izquierda muestra todas las tarifas asignadas y en la parte derecha muestra todas las tarifas no asignadas. Sin embargo, se puede asignar o desasignar las tarifas de acuerdo a lo que indican los botones.

Este modal es visible cuando se hace click en el botón ver buses, en la parte izquierda muestra todos los buses asignados y en la parte derecha muestra todos los buses no asignados. Sin embargo, se puede asignar o desasignar buses de acuerdo a lo que indican los botones.

Esta interface sirve para asignar o desasignar rutas a los buses que están asignadas a la linea



Esta interfaz se muestra cuando se hace click en los botones vista previa de la rutas

NOMBRE DE LA LINEA	RUTAS Y PARADAS	HORARIOS	TARIAS
11	<button>ASIGNAR</button>	<input type="checkbox"/> ver horario	
012	<input checked="" type="checkbox"/> sasd	<input type="checkbox"/> ver horario	
F	<input checked="" type="checkbox"/> ruta 09	<input type="checkbox"/> ver horario	
09	<input checked="" type="checkbox"/> ruta 09	<input type="checkbox"/> ver horario	
Y	<input checked="" type="checkbox"/> ruta 09	<input type="checkbox"/> ver horario	
08	<input checked="" type="checkbox"/> sasd	<input type="checkbox"/> ver horario	
110	<input checked="" type="checkbox"/> ruta 09	<input type="checkbox"/> ver horario	
J	<input checked="" type="checkbox"/> sasd	<input type="checkbox"/> ver horario	
H	<input checked="" type="checkbox"/> ruta 09	<input type="checkbox"/> ver horario	

Este formulario se muestra al hacer click en el botón nuevo línea, es para registrar líneas, el botón cancelar es para cancelar el registro de la línea, al hacer click en ella se cierra el formulario y se resetea, el botón guardar guarda los datos del introducidos en el formulario.

NOMBRE DE LA LINEA	RUTAS Y PARADAS	HORARIOS	TARIFAS	BUSES	RUTAS BUSES	ACCIONES
123	<input type="checkbox"/> ver rutas	<input type="checkbox"/> ver horario	<input type="checkbox"/> ver tarifas	<input type="checkbox"/> ver buses	<input type="checkbox"/> asignar o desa...	⋮
234	<input type="checkbox"/> ver rutas	<input type="checkbox"/> ver horario	<input type="checkbox"/> ver tarifas	<input type="checkbox"/> ver buses	<input type="checkbox"/> asignar o desa...	⋮
G	<input type="checkbox"/> ver rutas	<input type="checkbox"/> ver horario	<input type="checkbox"/> ver tarifas	<input type="checkbox"/> ver buses	<input type="checkbox"/> asignar o desa...	⋮
h	<input type="checkbox"/> ver rutas	<input type="checkbox"/> ver horario	<input type="checkbox"/> ver tarifas	<input type="checkbox"/> ver buses	<input type="checkbox"/> asignar o desa...	⋮
110	<input type="checkbox"/> ver rutas	<input type="checkbox"/> ver horario	<input type="checkbox"/> ver tarifas	<input type="checkbox"/> ver buses	<input type="checkbox"/> asignar o desa...	⋮
M	<input type="checkbox"/> ver rutas	<input type="checkbox"/> ver horario	<input type="checkbox"/> ver tarifas	<input type="checkbox"/> ver buses	<input type="checkbox"/> asignar o desa...	⋮
X	<input type="checkbox"/> ver rutas	<input type="checkbox"/> ver horario	<input type="checkbox"/> ver tarifas	<input type="checkbox"/> ver buses	<input type="checkbox"/> asignar o desa...	⋮
Y	<input type="checkbox"/> ver rutas	<input type="checkbox"/> ver horario	<input type="checkbox"/> ver tarifas	<input type="checkbox"/> ver buses	<input type="checkbox"/> asignar o desa...	⋮
F	<input type="checkbox"/> ver rutas	<input type="checkbox"/> ver horario	<input type="checkbox"/> ver tarifas	<input type="checkbox"/> ver buses	<input type="checkbox"/> asignar o desa...	⋮

Este campo se habilita cuando se hace click en el botón de filtrar por columnas, estos campos sirven para realizar filtrados por la columna. El botón filtrar filtra los datos al hacer click, de los campos introducidos, y el botón restablecer restablecerá a su estado inicial de las tablas y reseteará los campos de filtrado.



Conecitar con el sistema web Microbuses

Descripción

Tu dispositivo móvil se comportará como un aparato GPS, al conectar se compartirá la ubicación actual de tu dispositivo. Para conectar tu teléfono con el sistema web de microbuses inserte los campos proporcionado por el sistema web de microbuses.

Email —

dussan@gmail.com

Contraseña —

.....

Servidor —

<https://polite-regions-mate.loca.lt>

CONECTAR

EDITAR CAMPOS



La aplicación GPS es para conectar el dispositivo con el sistema de microbuses, mediante ella se envía la ubicación de dispositivo.

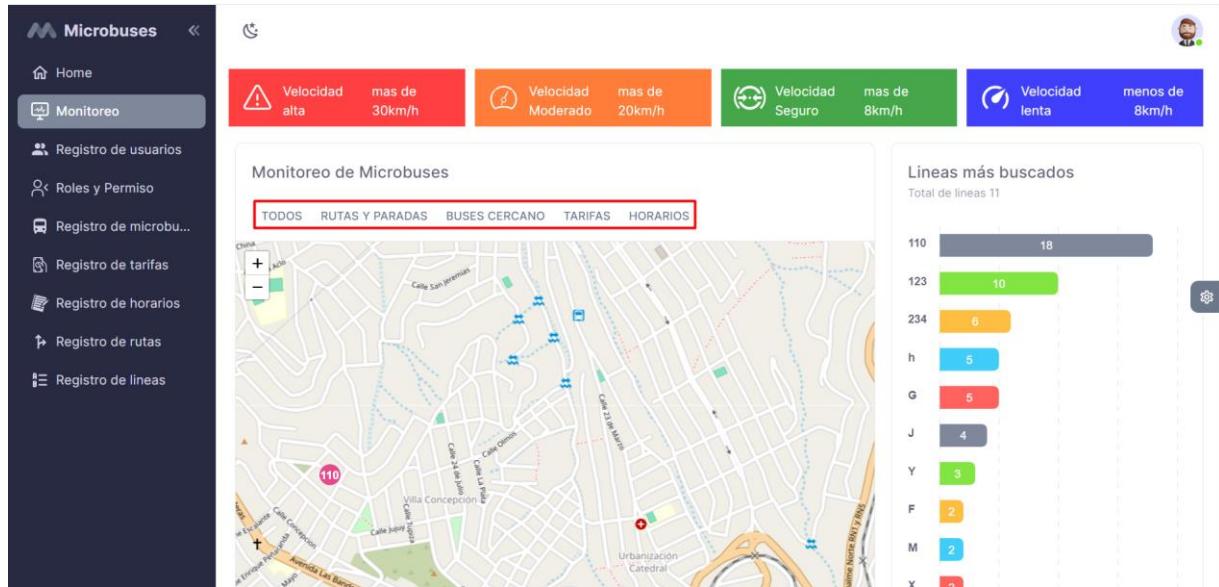
The screenshot shows the Microbuses application's monitoring interface. On the left is a dark sidebar with navigation options: Home, Monitoreo (selected), Registro de usuarios, Roles y Permiso, Registro de microbuses, Registro de tarifas, Registro de horarios, Registro de rutas, and Registro de líneas. The main area has a header with four speed filters: 'Velocidad alta mas de 30km/h' (red), 'Velocidad Moderado mas de 20km/h' (orange), 'Velocidad Seguro mas de 8km/h' (green), and 'Velocidad lenta menos de 8km/h' (blue). Below the filters is a map titled 'Monitoreo de Microbuses' showing a grid of streets with bus routes. To the right of the map is a bar chart titled 'Lineas más buscados' with the following data:

Línea	Cantidad
110	18
123	10
234	6
h	5
G	5
J	4
Y	3
F	2
M	2
X	1

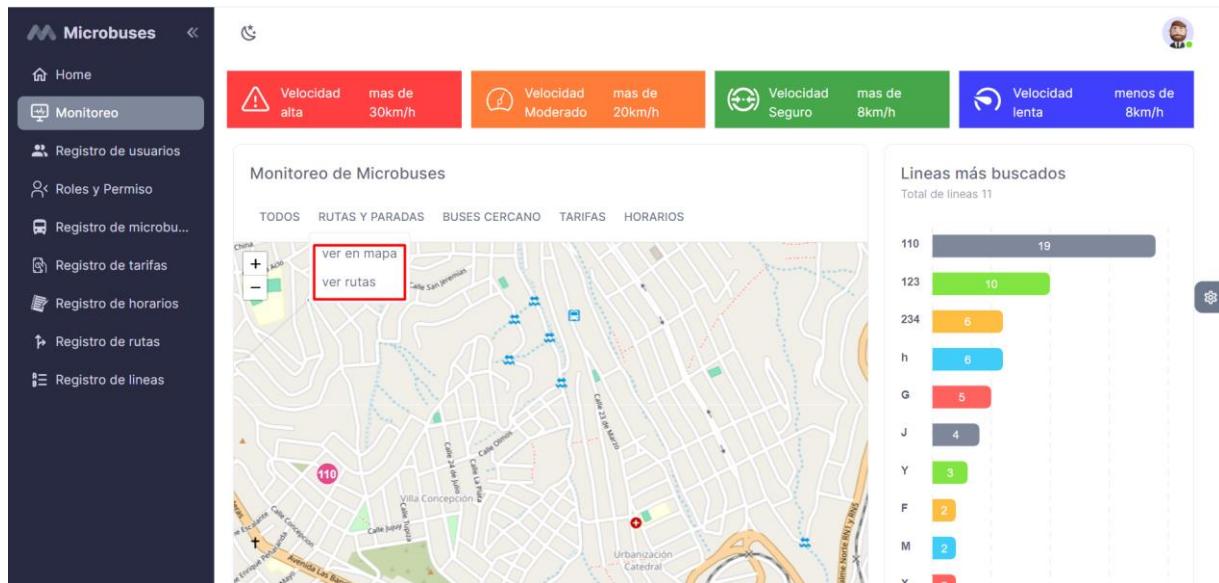
La interfaz monitoreo se despliega cuando se hace click en el botón de monitoreo que esta la barra lateral, en esta interface se muestra en tiempo real la ubicación de los microbuses, y su velocidad y entre otra información, en la parte derecha muestra una gráfica de las líneas más solicitadas. El botón rutas y paradas es para ver las rutas de las líneas. Al hacer click en ella se despliega un modal para seleccionar las líneas registradas.

This screenshot shows the same monitoring interface as above, but with a modal window open over the map. The modal is titled 'Lista de líneas' and contains a search bar 'Buscar Lineas'. Below it is a list of 10 registered lines: linea 11, linea 012, linea F, linea 09, linea Y, linea 08, linea 110, linea J, linea H, and linea K. The list is enclosed in a red box. The background map and other interface elements are visible behind the modal.

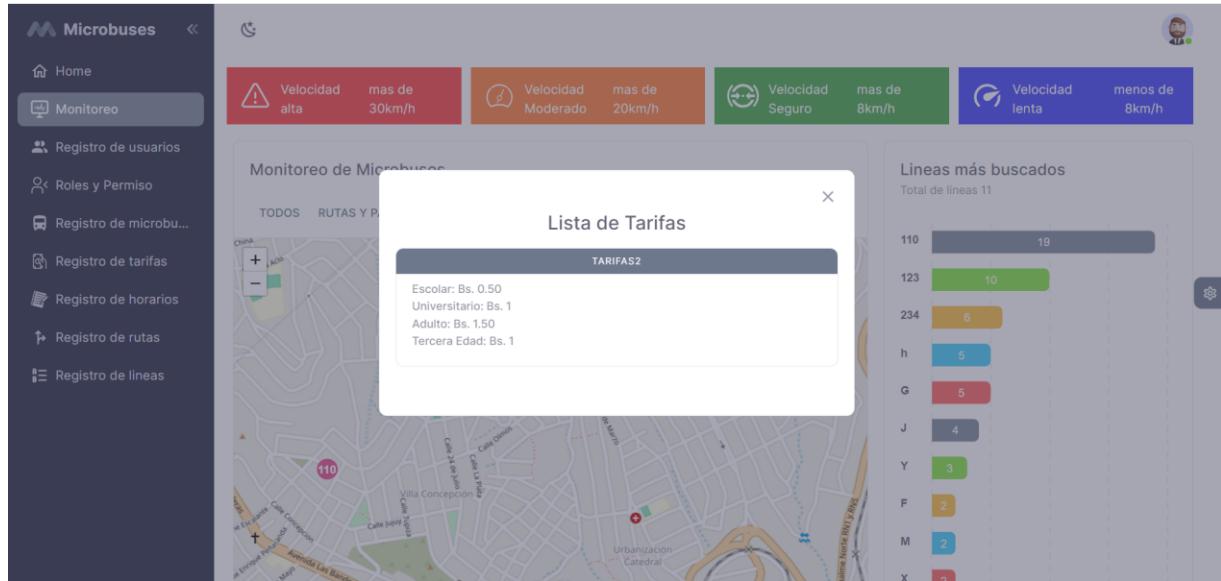
Este modal se muestra al hacer click en el botón rutas y paradas, el campo buscar líneas es para buscar las líneas, y al seleccionar en las líneas se mostrará las rutas de la linea



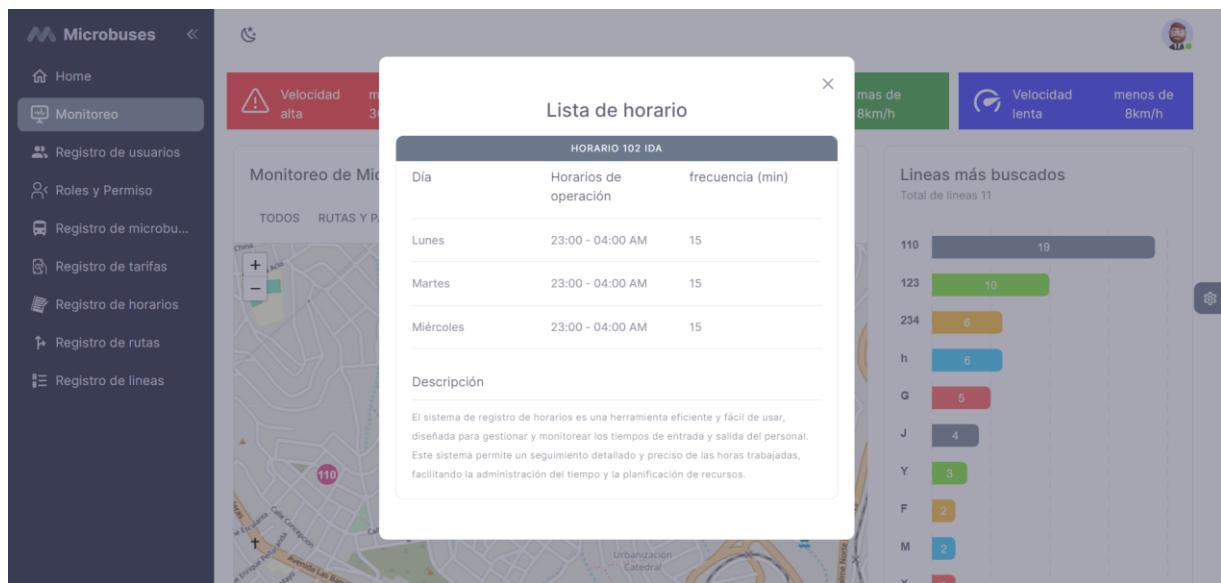
El botón todos es para ver todas las líneas, el botón tarifas es para ver las tarifas de la línea, y horarios es para ver los horarios de salida de los microbuses, el botón ver paradas y rutas es para ver las rutas en mapa y lugares donde pasa el microbus.



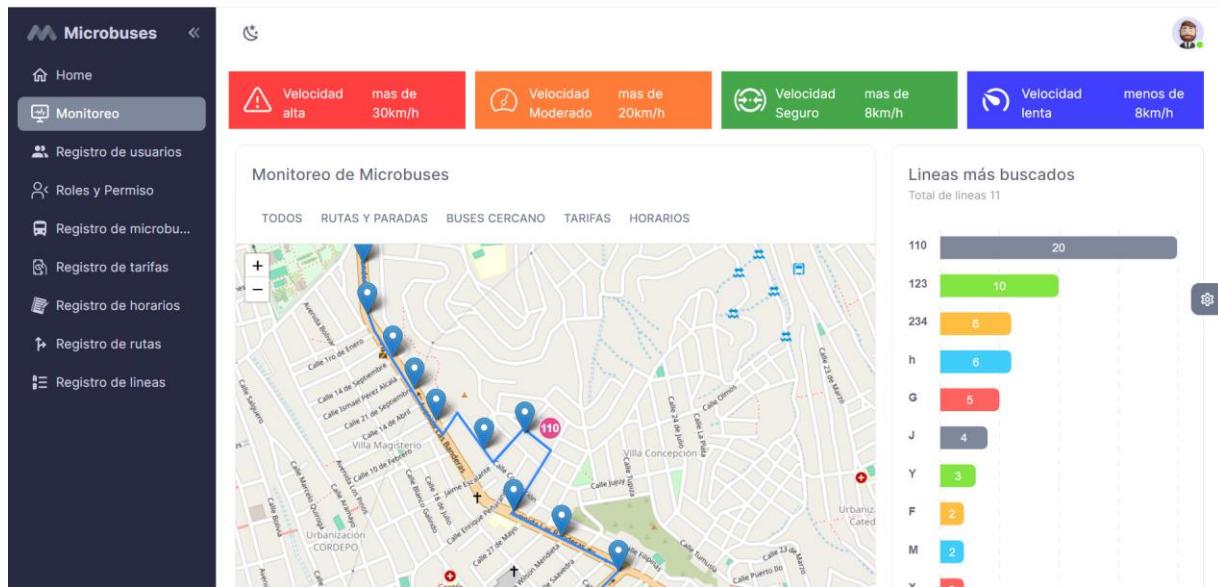
Estos botones permiten ver los reportes de las rutas, tanto en mapas y en una tabla.



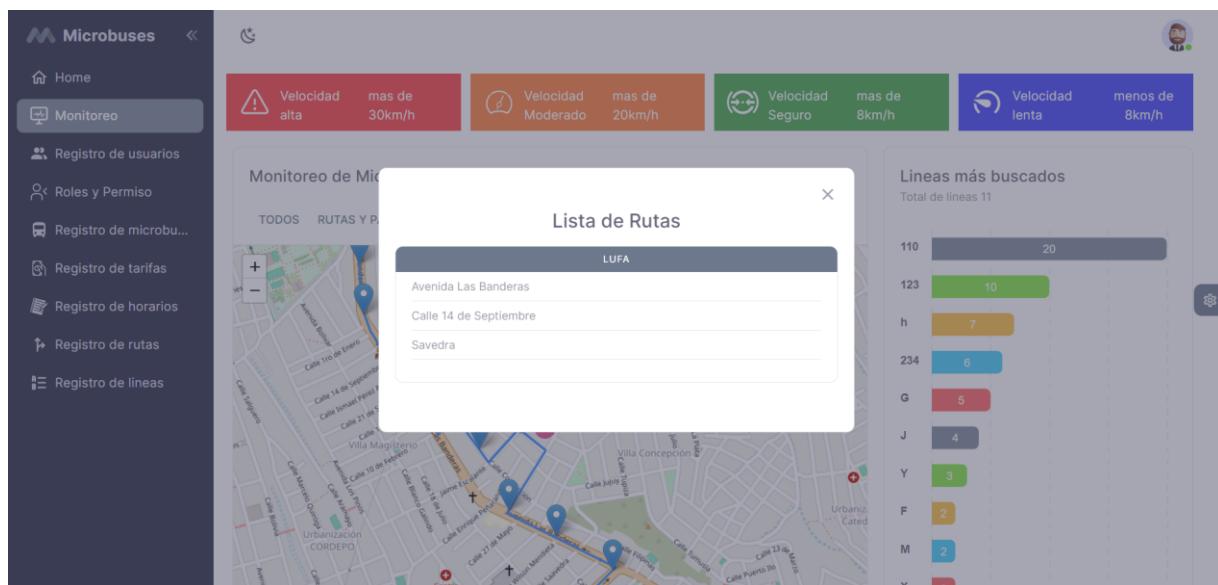
El modal es visible cuando se hace click en el botón tarifas, muestra las tarifas asignado a la línea.



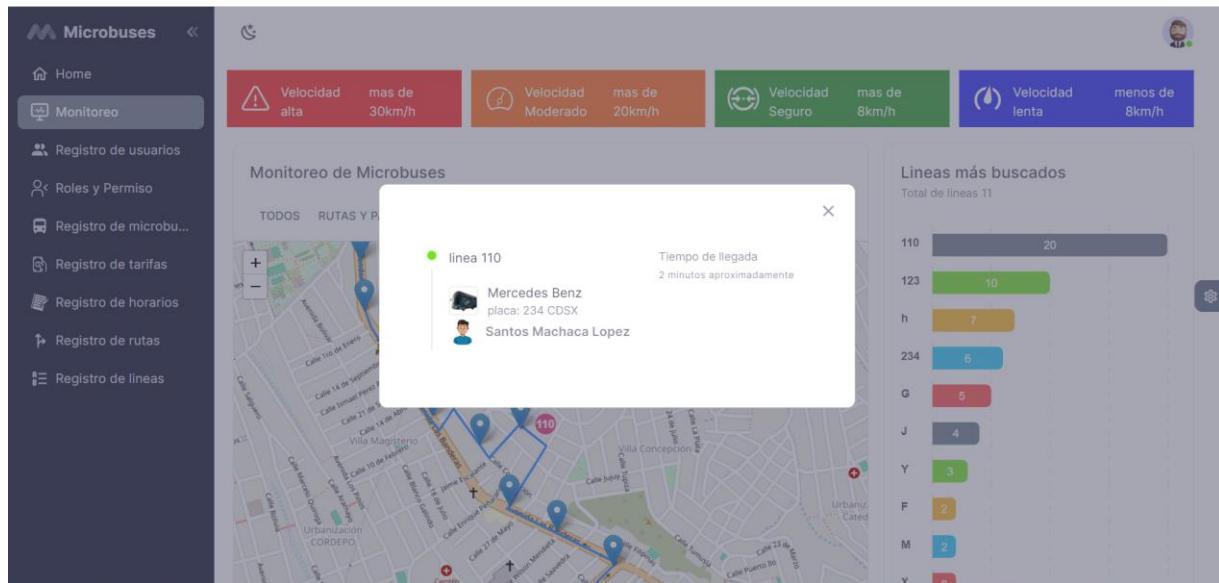
El modal listo de horario es visible cuando se hace click en el botón horario, muestra los horarios de salida de los microbuses, y los días



Este interface se despliega cuando se hace click en ver rutas en mapas.



Este modal se despliega cuando se hace click eb ver rutas.



Este modal es visible cuando se hace click en el botón buses cercano y muestra los buses que están cercanos.

The screenshot shows the Microbuses monitoring dashboard with a red box highlighting the filter section. The filter section includes fields for 'Usuarios', 'CI', 'Bus', 'Placa', 'Línea', and 'Estado de conexión'. Below the filter section is a table titled 'USUARIOS' with columns for 'USUARIOS', 'CI', 'BUS', 'PLACA', 'LINEA', and 'ESTADO DE CONEXION'. The table lists five users: juan carlos, manuel torrez, ghggvghg, dussan, and Jhony. The 'ESTADO DE CONEXION' column shows their connection status: Desconectado (ultimo contacto), Desconectado (ultimo contacto), Desconectado (ultimo contacto), En Linea (ultimo contacto), and Desconectado (ultimo contacto).

USUARIOS	CI	BUS	PLACA	LINEA	ESTADO DE CONEXION
juan carlos gaodthja@gmail.com	3435354	Mercedes Benz	HGV 778	X	Desconectado ult. vez nunca se conectó
manuel torrez asfdsfghmail.com	34535	Nissan	ZSD 789	11	Desconectado ult. vez nunca se conectó
ghggvghg hjhgfgh@gmail.com	9889879	Ninguno	—	Ninguno	Desconectado ult. vez nunca se conectó
dussan dussan@gmail.com	10531753	Tata	SAX 897	09	En Linea ultimo contacto
Jhony jhony@gmail.com	1053175	Nissan	ZSD 3232	09	Desconectado ultimo contacto 13/06/2024 a las 1:17

Este campo se habilita cuando se hace click en el botón de filtrar por columnas, estos campos sirven para realizar filtrados por la columna. El botón filtrar filtra los datos al hacer click, de los campos introducidos, y el botón restablecer restablecerá a su estado inicial de las tablas y reseteará los campos de filtrado.