

UNIVERSIDAD NACIONAL AGRARIA DE LA SELVA

FACULTAD DE INGENIERÍA EN INFORMATICA Y SISTEMAS



INFORME FINAL DE PRÁCTICAS PRE PROFESIONALES

**DEPURAR E IMPLEMENTAR NUEVOS MODULOS A LOS PROYECTOS DE
BOOKING Y TRADE**

Presentado por: Iomar Igor Alegre Barrera

Asesor: Mg. Rannoverng Yanac Montesino

Periodo de Prácticas Pre Profesionales: 13 de febrero – 13 de mayo

Lugar de las Prácticas Pre Profesionales: HMF Inversiones S.A.C

TINGO MARIA – JUNIO 2023

[Dictamen del asesor]

*{El dictamen deberá indicar la conformidad del asesor respecto a la redacción del informe
(considerar el formato que establece la comisión de prácticas pre profesionales}*

"AÑO DE LA UNIDAD, LA PAZ Y EL DESARROLLO"

CERTIFICADO DE PRÁCTICAS PRE-PROFESIONALES

Mediante la presente se CERTIFICA que:

IOMAR IGOR ALEGRE BARRERA

Realizó sus prácticas pre-profesionales satisfactoriamente en nuestra empresa, desempeñándose como PRACTICANTE DE CONSTRUCCION DE LOS APLICATIVOS WEB TAMPU Y BOOKING en nuestra área de desarrollo de software, desde el 13/02/2023 hasta el 13/05/2023, habiendo cumplido satisfactoriamente las siguientes actividades:

- Desarrollo de nuevos de las aplicaciones web Tampu y Booking con la biblioteca de REACT y framework de laravel.
- Implementación de nuevos módulos a las aplicaciones ya mencionadas.
- Implementación del servicio de smtp2go para el envío de correos electrónicos.
- Despliegue de las aplicaciones Trade y Booking.

Se expide el presente documento a solicitud del interesado.



Tingo Maria, 19 de mayo del 2023

ATENTAMENTE



[Acta de Sustentación de PPP]

{El Acta solo se incluirá después del levantamiento de observaciones después de la sustentación, válido para el empastado final}

DEDICATORIA

Queridos Hugo y Petronila,

A ustedes, mis amados padres, quiero dedicarles este informe de práctica preprofesional. Su amor incondicional, apoyo constante y sacrificio han sido pilares fundamentales en mi camino hacia el crecimiento y la formación profesional.

A través de su ejemplo de trabajo arduo y dedicación, me han enseñado la importancia de la perseverancia y la excelencia en cada proyecto que emprendo. Su confianza en mí ha sido mi mayor motivación para superar los desafíos y alcanzar mis metas.

AGRADECIMIENTOS

- A mis padres por todo el apoyo brindado a lo largo de mi vida y la confianza y orgullo que siempre me expresaron.
- A mis docentes en la Universidad que forjaron en mí una actitud crítica, de cordialidad y participación constante en actividades que nos integren con nuestro entorno.
- Al Ing. Josue S. acuña Cordoba, por las palabras de aliento, motivación y su gran disposición a apoyar a lo largo de todas mis practicas Pre Profesionales y así mismo por brindarme la oportunidad de ser parte de este Proyecto.
- Al Mg. Rannovernrg Yanac Montesino quien me asesoro, por su disposición a apoyarme, orientarme y su gran paciencia ante mis dudas a lo largo del desarrollo de este informe de mis practicas Pre Profesionales.
- Agradecer a cada persona que estuvo en mi camino de estudios preprofesionales y aporto de alguna forma positiva a mi vida. Aunque la comunicación pueda cortarse quiero que sepan que estoy feliz de haberlos conocido, haber compartido experiencias positivas y que hayan hecho esta aventura algo más llevadera. Espero logren sus objetivos y metas. Gracias.

INDICE GENERAL

RESUMEN	20
INTRODUCCIÓN	21
CAPITULO I	22
CARACTERISTICAS GENERALES DE LA ORGANIZACIÓN	22
1.1. Descripción de la Organización	22
1.1. Misión y Visión.....	22
1.1.1. Misión	22
1.1.2. Visión	22
1.2. Organigrama estructural.....	22
MARCO TEORICO	22
2.1. Ingeniería de software	22
2.2. Patrones de Arquitectura de software	22
2.2.1. Patrón de arquitectura MVC.....	22
2.2.2. Trabajo por objetivos	23
2.3. Anydesk	24
2.4. Sistema de Gestión Comercial.....	24
2.4.1. ERP	24
2.5. Interfaz de programación de aplicaciones (API)	25
2.6. Transferencia del estado de presentación (REST).....	26
2.7. Backend	27
2.8. Frontend	27
2.9. FullStack	28
2.10. PHP	28
2.11. Framework Laravel	28
2.12. MySQL	29
2.13. React	29
2.14. GoDaddy.....	29

2.15.	Cpanel.....	29
2.16.	SMTP2Go	30
	CAPITULO III	31
	ACTIVIDADES DESARROLLADAS	31
3.1.	Situación Actual	31
3.1.1.	Problemática	31
3.2.	Objetivos	35
3.3.	Alcance	35
3.4.	Cronograma de actividades.....	35
3.5.	Escenario de trabajo	37
3.5.1.	Dirección de proyecto.....	37
3.5.2.	Equipo de trabajo	37
3.5.3.	Recursos	38
3.6.	Descripción de la arquitectura del proyecto.....	38
3.6.1.	Arquitectura del proyecto	38
3.6.2.	Distribución de carpetas	41
3.7.	Método de trabajo.....	47
3.8.	Construir submódulo de salida (Trade)	48
3.8.1.	Tareas.....	49
3.8.1.1.	Referencias	49
3.8.2.	Ejecución de las tareas.....	51
3.8.2.1.	Backend	51
3.8.2.1.1.	T1 Crear el crud de salida y T2 Crear un endpoint que permita imprimir la salida por el idsalidavehiculo	51
3.8.2.2.	Frontend	54
3.8.2.2.1.	T3. Crear el submódulo de salida.....	54
3.8.3.	Resultados.....	62
3.9.	Construir submódulo de adelantos (Trade)	64

3.9.1.	Tareas.....	65
3.9.1.1.	Referencias	66
3.1.1.	Ejecución de las tareas.....	67
3.1.1.1.	Backend	67
3.1.1.1.1.	T1 Crear el crud de adelantos y T2 Crear un endpoint que permita imprimir la salida por el idadelantovehiculo	67
3.1.1.2.	Frontend	70
3.1.1.2.1.	T3. Crear el submódulo de adelanto.....	70
3.1.1.	Resultados.....	79
3.2.	Construir submódulo de encomiendas (Trade)	81
3.2.1.	Tareas.....	81
3.2.1.1.	Referencia	82
3.2.2.	Ejecución de las tareas.....	83
3.2.2.1.	Backend	84
3.2.2.1.1.	T1. Crear el endpoints de encomienda	84
3.2.2.2.	Frontend	87
3.2.2.2.1.	T2. Crear el submódulo de encomienda	87
3.2.2.2.1.1.	Tab de gestión de encomiendas registradas.....	93
3.2.2.2.1.2.	Tab de gestión de encomiendas por llegar	96
3.2.3.	Resultados.....	100
3.3.	Construir vista pública para la consulta de encomiendas (Trade).....	106
3.3.1.	Tareas.....	107
3.3.1.1.	Referencia	107
3.3.2.	Ejecución de tareas.....	108
3.3.2.1.	Backend	108
3.3.2.1.1.	T1. Se debe rastrear por el número de guía y el código de guía de la boleta de la encomienda	108
3.3.2.2.	Frontend	109
3.3.2.2.1.	T2. Mostrar los estados de una encomienda y resaltar el actual	109

3.3.2.2.2. T3. Al no encontrar una encomienda con la guía y el código mostrar un mensaje de información.....	114
3.3.3. Resultados	114
3.4. Implementar nuevas funcionalidades al submódulo facturación (Trade)	115
3.4.1. Tareas.....	116
3.4.1.1. Referencias	116
3.4.2. Ejecución de las tareas.....	117
3.4.2.1. Backend	117
3.4.2.1.1. T3. Concatenar el nombre del producto y la descripción adicional en el detalle del comprobante al imprimir factura	117
3.4.2.2. Frontend	118
3.4.2.2.1. T1. Al agregar un producto quitar el código en su lugar poner un input donde se pueda ingresar una descripción adicional y T2. Concatenar el detalle del producto con la descripción adicional en la vista de facturación y en el modal de editar factura	119
3.4.3. Resultados	120
3.5. Implementar reportes al submódulo grid (Booking).....	121
3.5.1. Tareas.....	122
3.5.2. Ejecución de tareas.....	122
3.5.2.1. Backend	122
3.5.2.1.1. T1. hacer endpoints donde muestre los reportes pdf de salidas del día y arribos del día 123	
3.5.2.2. Frontend	124
3.5.2.2.1. T2. Mostrar en un modal el pdf de reporte de salidas del día y T3. Mostrar en un modal el pdf de reporte de parte diario donde se listan todos los arribos del día	124
3.5.3. Resultados	127
3.6. Construir vista pública para el registro de huéspedes (Booking)	128
3.6.1. Tareas.....	128
3.6.2. Ejecución de tareas.....	129
3.6.2.1. Frontend	129
3.6.2.1.1. T.1 Construcción de vista pública para el registro de huéspedes	129

3.6.3. Resultados.....	133
3.7. Implementar envío de correo electrónico del enlace de registro de información de reserva (Booking).....	135
3.7.1. Tareas.....	136
3.7.2. Ejecución de tareas.....	136
3.7.2.1. Backend	136
3.7.2.1.1. T1 Implementar servidor smtp y T2 crear endpoint para envío de correo	137
3.7.2.2. Frontend	138
3.7.2.2.1. Cuando el estado de la habitación este en reserva agregar el botón enviar el correo	138
3.7.3. Resultados.....	140
3.8. Desplegar y depurar proyectos en servidor de producción (Booking y Trade).....	142
3.8.1. Despliegue	142
3.8.1.1. Tareas.....	142
3.8.1.2. Ejecución de tareas	142
3.8.1.3. Resultado	146
3.8.2. Depuración	148
3.8.2.1. Tareas.....	149
3.8.2.2. Ejecución de tareas	150
3.8.2.2.1. Identificar y formatear los datos enteros y decimales.....	150
3.8.2.3. Resultados.....	152
CONCLUSIONES.....	153
RECOMENDACIONES.....	155
ANEXOS	156

ÍNDICE DE FIGURAS

Figura 1 Organigrama Estructural HMF Inversiones S.A.C.....	22
Figura 2 Arquitectura MVC.....	23
Figura 3 Módulos que abarca un sistema ERP.....	25
Figura 4 Flujo de trabajo de una API	26
Figura 5 Comunicación REST	26
Figura 6 Servidor backend	27
Figura 7 Tecnologías frontend.....	27
Figura 8 Capos que abraza un desarrollador fullstack.....	28
Figura 9 React componentes	29
Figura 10 Panel Cpanel	30
Figura 11 Regla de tiempo de los proyectos	34
Figura 12 Diagrama frontend	39
Figura 13 Arquitectura backend	40
Figura 14 Arquitectura de los proyectos	41
Figura 15 Distribución de carpetas en visual studio code	42
Figura 16 Distribución de carpetas de las vistas.....	43
Figura 17 Distribución de carpetas de los controladores	43
Figura 18 Distribución de carpetas de los modelos.....	44
Figura 19 Distribución de carpetas de frontend.....	45
Figura 20 Distribución de carpetas de los actions de redux	45
Figura 21 Distribución de carpetas de los reducers de redux	46
Figura 22 Distribución de carpetas de los componentes	46
Figura 23 Objetivo con tareas.....	47
Figura 24 Resultado de una ejecución de un objetivo	47
Figura 25 Entorno de trabajo	48
Figura 26 Flujo de trabajo para el submódulo de gestión de salidas	49
Figura 27 Objetivo: Crear submódulo salida y asignar al módulo Tesorería	49

Figura 28 Vista referencia.....	50
Figura 29 Referencia de impresión de salida.....	50
Figura 30 Fragmento de Código del controlador de salida vehículos	51
Figura 31 Fragmento de Código para listar lo registros de salida	52
Figura 32 Fragmento de código para registrar salida de vehículo	52
Figura 33 Fragmento de Código para actualizar registro de salida vehículo	53
Figura 34 Fragmento de código para eliminar registro de salida vehículo.....	53
Figura 35 Fragmento de código de la impresión de registro de salida vehículo	54
Figura 36 Fragmento de código de los endpoints	54
Figura 37 Distribución de archivos de la vista salida vehículo.....	55
Figura 38 Fragmento de código del componente ExitVehicles	55
Figura 39 Fragmento de código de ExitVehiclesConfig para la configuración del acceso y la ruta	55
Figura 40 Fragmento de Código para listar los registros de las salidas	56
Figura 41 Fragmento de Código del consumo del endpoint de registro de salida y impresión	56
Figura 42 Fragmento de Código para el consumo del endpoint para actualizar registro de salida vehículo ..	57
Figura 43 Fragmento de código del consumo del endpoint para eliminar registro de salida vehículos	57
Figura 44 Fragmento de código del reducer de la vista salida de vehículos	58
Figura 45 Fragmento de código del consumo de los registros de salida vehículos.....	59
Figura 46 Fragmento de código de las acciones de los registros	60
Figura 47 Fragmento de código del botón de crear registro salida vehículo	61
Figura 48 Fragmento de código del modal de registro y edición de registro salida vehículo	61
Figura 49 Fragmento de código para el consumo de los endpoints de registro y actualización de salida vehículos.....	62
Figura 50 Vista de gestión de salida vehículos	62
Figura 51 Vista de modal de registro de salida vehículos.....	63
Figura 52 Vista modal para actualizar registros	63
Figura 53 Vista de pdf generado para la impresión de registro salida vehículo.....	64
Figura 54 Flujo de trabajo para adelanto de vehículos	65

Figura 55 Objetivo: Crear el submódulo de adelanto y asignar al módulo de tesorería	65
Figura 56 Vista de referencia.....	66
Figura 57 Referencia de impresión de adelanto.....	66
Figura 58 Fragmento de Código del controlador de adelanto vehículos	67
Figura 59 Fragmento de Código para listar lo registros de salida	68
Figura 60 Fragmento de código para registrar adelanto de vehículo	68
Figura 61 Fragmento de Código para actualizar registro de adelanto vehículo.....	69
Figura 62 Fragmento de código para eliminar registro de adelanto vehículo.....	69
Figura 63 Fragmento de código de la impresión de registro de adelanto vehículo	70
Figura 64 Fragmento de código de los endpoints	70
Figura 65 Distribución de archivos de la vista adelanto vehículo.....	71
Figura 66 Fragmento de código del componente AdvanceShift	71
Figura 67 Fragmento de código de AdvanceShiftConfig para la configuración del accedo y la ruta.....	71
Figura 68 Fragmento de Código para listar los registros de los adelantos	72
Figura 69 Fragmento de Código del consumo del endpoint de registro de adelanto e impresión	72
Figura 70 Fragmento de Código para el consumo del endpoint para actualizar registro de adelanto vehículo	73
Figura 71 Fragmento de código del consumo del endpoint para eliminar registro de adelanto vehículos	74
Figura 72 Fragmento de código del reducer de la vista adelanto de vehículos	75
Figura 73 Fragmento de código del consumo de los registros de adelanto vehículos.....	76
Figura 74 Fragmento de código de las acciones de los registros	77
Figura 75 Fragmento de código del botón de crear registro adelanto vehículo	78
Figura 76 Fragmento de código del modal de registro y edición de registro adelanto vehículo	78
Figura 77 Fragmento de código para el consumo de los endpoints de registro y actualización de adelanto vehículos.....	79
Figura 78 Vista de gestión de adelanto vehículos	79
Figura 79 Vista de modal de registro de adelanto vehículos.....	80
Figura 80 Vista modal para actualizar registros	80
Figura 81 Vista de pdf generado para la impresión de registro adelanto vehículo.....	80

Figura 82 Flujo de trabajo de encomienda	81
Figura 83 Objetivo: Crear submódulo de encomiendas	82
Figura 84 Referencia para la contrucción de gestion de encomienda	83
Figura 85 Referencia para modal de registro de encomienda	83
Figura 86 Fragmento de código del controlado de encomiendas	84
Figura 87 Fragmento de código de la función de listar encomiendas	84
Figura 88 Fragmento de código para listar los ítems de una encomienda	85
Figura 89 Fragmento de código de registro de encomienda.....	85
Figura 90 Fragmento de código de actualizar encomienda	86
Figura 91 Fragmento de código de la facturación de encomienda	87
Figura 92 Distribución de carpetas del submódulo de encomiendas.....	88
Figura 93 Componente de encomiendas con los tabs de registro de encomiendas y gestión de encomiendas por llegar	88
Figura 94 Se establece los permisos para la ruta	89
Figura 95 Consumo de registros del endpoint de encomiendas	89
Figura 96 Consumo de registros del endpoint de detalle de encomienda	90
Figura 97 Consumo del endpoint para registrar encomienda	90
Figura 98 Consumo de endpoint para actualizar registro de encomienda	91
Figura 99 Consumo de endpoint para actualizar el estado de envio de una encomienda.....	91
Figura 100 Contendor reducer de encomiendas	92
Figura 101 Consumo de los registros de encomienda.....	93
Figura 102 Consumos de las funciones para las acciones de editar, facturar, imprimir y eliminar	94
Figura 103 Fragmento de código de inicialización de datos de encomienda en modal de creación y edición	95
Figura 104 Consumo de función para crear y actualizar registro de encomienda	95
Figura 105 Tab de encomiendas por llegar	96
Figura 106 Consumo de los registros de encomienda.....	97
Figura 107 Consumos de las funciones para las acciones de editar, facturar, imprimir y eliminar	98
Figura 108 Modal de facturación de encomienda.....	99

Figura 109 Fragmento de código del modal de recepción de encomienda	100
Figura 110 Tab registro de encomiendas	101
Figura 111 Tab de encomiendas por llegar	101
Figura 112 Modal de registro y edición de encomienda	102
Figura 113 Formulario y tab del detalle de registro de encomienda	103
Figura 114 Modal de facturación de encomienda.....	104
Figura 115 Modal de entrega de encomienda	104
Figura 116 Comprobante de registro de encomienda.....	105
Figura 117 Comprobante de encomienda facturada	106
Figura 118 Flujo de trabajo de seguimiento de encomienda	107
Figura 119 Objetivo: Construir rastreador de encomiendas	107
Figura 120 Referencia para el tracking de encomiendas.....	108
Figura 121 Fragmento de código de la función para seguimiento de encomienda	109
Figura 122 Registro de la ruta para rastrear encomienda.....	109
Figura 123 Distribución de carpetas de la vista de seguimiento	109
Figura 124 Fragmento de condigo de la vista de seguimiento.....	110
Figura 125 Fragmento de código de la configuración de la ruta y la autorización	110
Figura 126 Fragmento de código para el consumo del endpoint de rastrear encomienda	111
Figura 127 Fragmento de código de redux de la encomienda	111
Figura 128 Fragmento de código del formulario de seguimiento de encomienda	112
Figura 129 Fragmento de código del consumo el endpoint de seguimiento de encomienda	112
Figura 130 Fragmento de código donde se muestra el resultado de la búsqueda	113
Figura 131 Fragmento de código de la información al no encontrar encomienda	114
Figura 132 Vista de seguimiento al no encontra encomienda	115
Figura 133 Vista de seguimiento al encontrar encomienda	115
Figura 134 Flujo de trabajo de la gestión de factura	116
Figura 135 Objetivo: Nuevas funcionalidades al submódulo facturación	116
Figura 136 Referencia para el cambio de facturación	117

Figura 137 Fragmento de código de la consulta de datos para la impresión de factura	118
Figura 138 Fragmento de código de la validación de descripción adicional	118
Figura 139 Fragmento de código del input de descripción adicional	119
Figura 140 Fragmento de código de la concatenación de la descripción adicional	120
Figura 141 Vista de registro de factura con el campo descripción adicional	120
Figura 142 Comprobante de factura con la descripción adicional	121
Figura 143 Flujo de trabajo del submódulo grid de los reportes	122
Figura 144 Objetivo: Implementar reportes a grid.....	122
Figura 145 Fragmento de código del reporte de arribos diarios.....	123
Figura 146 Fragmento de código del reporte de salidas diarias.....	124
Figura 147 Ruta de reportes.....	124
Figura 148 Fragmento de código de la vista de grid.....	125
Figura 149 Fragmento de código del modal de reporte de salidas del día	126
Figura 150 Fragmento de codigo de arrivos del dia	126
Figura 151 Modal de reporte de arribos del día.....	127
Figura 152 Modal de reporte de salidas del día	127
Figura 153 Flujo de trabajo del submódulo de grid.....	128
Figura 154 Objetivo: Construcción de vista pública para registro de huéspedes	129
Figura 155 Fragmento de código de la consulta de la reserva	130
Figura 156 Fragmento de código de conteo de huéspedes	131
Figura 157 Fragmento de código de la asignación de los datos de los huéspedes a GuestRegister	132
Figura 158 Fragmento de código del componente GuestRegister	133
Figura 159 Vista publica de booking.....	134
Figura 160 Componente GuestRegister renderizado	134
Figura 161 Vista cuando no se encuentra la reserva solicitada.....	135
Figura 162 Comprobante que se genera luego de terminar el registro de huéspedes	135
Figura 163 Flujo de trabajo para el envío de correo	136
Figura 164 Objetivo: creación de endpoint para envío de correo.....	136

Figura 165 Endpoints de envío de correo.....	137
Figura 166 Fragmento de código envío correo	137
Figura 167 Fragmento de código del envío de correo usando api de smtp2go	138
Figura 168 Fragmento de código que se agregó a la vista grid	139
Figura 169 Fragmento de código para el consumo del api para el envío del correo	140
Figura 170 Modal de reserva registrada.....	141
Figura 171 Modal de envío de encomienda	141
Figura 172 Correo electrónico que llega al correo	142
Figura 173 Objetivo: Desplegar proyectos	142
Figura 174 Creación de las DB	143
Figura 175 Conexión a la base de datos de manera remota	144
Figura 176 Importación de base de datos	144
Figura 177 Ejecución de las consultas para la creación de las bases de datos	145
Figura 178 Directorio de subdominios	145
Figura 179 Subida de archivos.....	146
Figura 180 Proyecto Trade desplegado	147
Figura 181 Proyecto de Booking (Tampu) desplegado.....	148
Figura 182 Casos de uso Trade	149
Figura 183 Casos de uso Booking	149
Figura 184 Objetivo: Depurar errores	150
Figura 185 Identificación de los datos usando la herramienta del ide	151
Figura 186 Formateo de forma manual de los datos	152

INDICE DE TABLAS

Tabla 1 Cronograma de actividades	35
Tabla 2 Equipo de trabajo.....	37
Tabla 3 Recursos.....	38

RESUMEN

HMF Inversiones S.A.C es una empresa que ofrece servicios de ventas de equipos electrónicos (laptops, impresoras, cámaras web, entre otras cosas), servicio técnico (laptops, impresoras), instalación de cámaras web y desarrollo de soluciones y sistemas de software, utilizando tecnologías de alto nivel. También cuenta con un producto de software ERP, el cual es una aplicación de escritorio que permite a las Mypes y Pymes administrar sus procesos de venta, compra, almacenes, manejo de inventarios, reportes, entre otras cosas.

La empresa actualmente está migrando su sistema de escritorio a dos aplicaciones web con el objetivo de implementar nuevas funciones de forma remota y mejorar la accesibilidad para los clientes. Con estas aplicaciones web, los clientes podrán realizar pedidos, ventas, apertura y cierre de cajas, registro de marcas y categorías sin necesidad de utilizar un pc.

La presente práctica preprofesional tiene como objetivo continuar con la construcción de las aplicaciones web Booking y Trade. A través de reuniones, se asignarán nuevas tareas, se realizarán revisiones y correcciones, lo cual permitirá organizar el trabajo de construcción de acuerdo con las actividades propuestas.

INTRODUCCIÓN

Cada año se crean nuevas empresas con diferentes innovaciones que compiten con las otras que ya llevan algún tiempo en el mercado, para ello la clave de supervivencia de una empresa es la constante actualización de los participantes con ayuda de la tecnología, donde se pueden reducir costos, automatizaciones, entre otras necesidades de la empresa, brindando una experiencia única y accesible a sus clientes.

HMF Inversiones S.A.C, se encuentra migrando su aplicación de escritorio desarrollado con .NET a dos aplicaciones web, una orientada a la gestión de una empresa de transportes (Trade) y otra a la gestión de hoteles (Booking), para que sus clientes puedan mejorar a la accesibilidad al sistema. Para la cual en la presente practica se construyó y desplegaron ambas aplicaciones ya mencionadas.

En el capítulo 1: Se describe las características de la empresa HMF Inversiones S.A.C, empresa donde se desarrolló las prácticas, que se encuentra en la ciudad de Tarapoto. La asistencia fue de forma virtual en horarios de oficina.

En el capítulo 2: Se describe el marco teórico y conceptual que sirvió como base para el desarrollo y entendimiento de la práctica preprofesional.

En el capítulo 3: Se explica el desarrollo de las actividades realizadas durante el periodo de duración de las prácticas preprofesional: construcción y despliegue de las aplicaciones web.

Finalmente se detalla los resultados, conclusiones y anexos.

CAPITULO I
CARACTERISTICAS GENERALES DE LA ORGANIZACIÓN
CAPITULO II
MARCO TEORICO Y CONCEPTUAL

2.1. Ingeniería de software

Según (Roger S. Pressman, n.d.), define la ingeniería de software como una disciplina de cuatro capas: herramientas, métodos, procesos y un enfoque de calidad. Se complementa con la definición en cinco áreas de conocimiento relacionada a la capa de procesos de desarrollo de software: requisitos, diseño, construcción, pruebas y mantenimiento de software, para obtener un producto que cumpla con los estándares de calidad y las expectativas del cliente. (IEEE COMPUTE SOCIETY, n.d.)

2.2. Patrones de Arquitectura de software

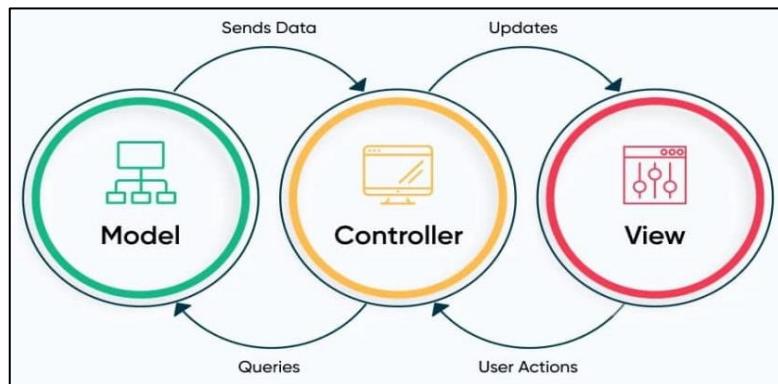
Los patrones de arquitectura describen los diseños de los componentes que forman un sistema, es decir la estructura con la cual estará compuesta el sistema. (Gruhn & Striemer, n.d.)

Según (Roger S. Pressman, n.d.) nos dice que los patrones de arquitectura de software como un proceso de etapas múltiples que se inicia con los requerimientos de información, se sintetizan las representaciones de los datos y la estructura del programa, las características de la interfaz y los detalles del procedimiento.

2.2.1. Patrón de arquitectura MVC

Es un patrón de arquitectura de software encargado de separar la lógica de negocio de la interfaz del usuario en tres capas “modelo, vista y controlador” capa uno de ellos con una única responsabilidad, actualmente es el más utilizado en aplicaciones Web, ya que facilita la funcionalidad, mantenibilidad y escalabilidad del sistema, de forma simple y sencilla, a la vez que permite “no mezclar lenguajes de programación en el mismo código”.
. (Eugenio Bahit, n.d.)

Figura 2 Arquitectura MVC



Fuente: (Kinsta, 2023)

La arquitectura MVC se divide en tres niveles (modelo, vista y controlador) cada una encargada de una responsabilidad cumpliendo con el primer principio SOLID:

- **Modelo:** Actúa como intermediario entre el controlador y la base de datos, ejecutando consultas SQL o mediante un ORM.
- **Controlador:** Es el intermediario entre la vista y el modelo, Aquí es donde se consultan los datos el modelo, aplica la lógica del negocio y pasa los datos a la vista.
- **Vista:** Es la parte visible de la aplicación web en los cuales se muestra los datos obtenidos por el modelo de manera grafica.

2.2.2. Trabajo por objetivos

Según (Greenbank, 2001) nos dice que los objetivos en las microempresas son definidos por los propietarios y gerentes de la empresa, tienden a relacionarse con criterios personales más que con criterios empresariales.

Según (RIVERA, n.d.) no dice que, en el contexto del desarrollo de software, el trabajo por objetivos implica establecer metas y objetivos específicos relacionados con el desarrollo, mantenimiento o mejora del software. Estos objetivos pueden incluir la entrega de nuevas características o funcionalidades, la corrección de errores, la mejora del rendimiento o la optimización del código, entre otros.

De acuerdo con lo anterior, se podría concebir

2.3. Anydesk

Anydesk es un sistema que permite a un usuario que proveen un servicio de TI conectarse con sus clientes de manera remota, pudiendo interactuar y transferirse documentos.

Permite al usuario acceder de forma remota al escritorio junto con sus archivos o documentos desde cualquier parte del mundo. AnyDesk tiene una función de libreta de direcciones incorporada, que rastrea conexiones o contactos y permite al usuario ver el estado en línea de esas conexiones. AnyDesk también es accesible en áreas donde el ancho de banda es bajo y la conectividad a Internet es deficiente.(AnyDesk - Javatpoint, n.d.)

2.4. Sistema de Gestión Comercial

Son sistemas que se enfocan en los procesos comerciales de una empresa, facilitando el trabajo y la atención de los clientes.

2.4.1. ERP

Un sistema ERP es un software orientado a una empresa o a un nicho de empresas que brindan bienes o servicios con la finalidad de automatizar y organizar sus procesos principales, con la finalidad de reducir costos y brindar un mejor servicio a sus clientes.

Se refiere a un tipo de software que las organizaciones usan para administrar las actividades comerciales diarias, como la contabilidad, el rendimiento, etc. (Ahed Abugabah & Louis Sanzogni, n.d.)

Figura 3 Módulos que abarca un sistema ERP



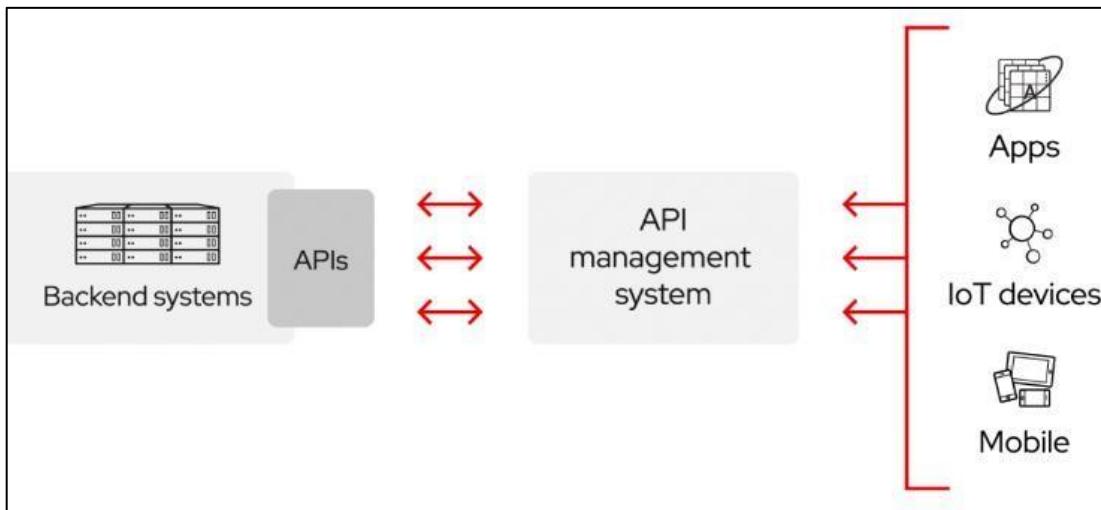
Fuente: Emiral

2.5. Interfaz de programación de aplicaciones (API)

Es un conjunto de definiciones y protocolos que se usa para diseñar e integrar el software de aplicaciones.

Las Apis permiten que sus productos y servicios se comuniquen entre sí, sin necesidad de saber cómo están implementados. Las Apis son un medio simplificado para conectar su propia infraestructura a través del desarrollo de aplicaciones nativas de la nube, pero también permiten compartir los datos con clientes y usuarios externos como se muestra en la figura 4.(MANFRED BORTENSCHLAGER & STEVEN WILLMOTT, n.d.)

Figura 4 Flujo de trabajo de una API

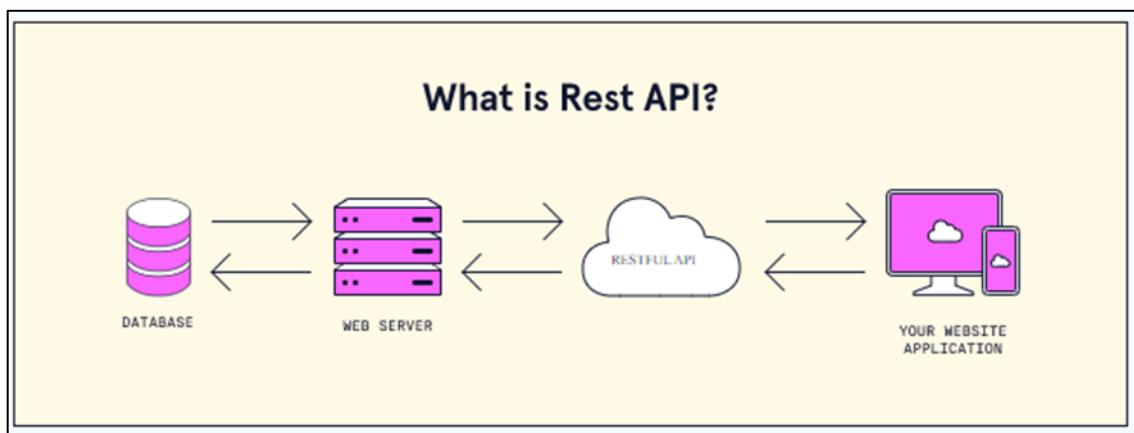


Fuente: (*MANFRED BORTENSCHLAGER & STEVEN WILLMOTT, n.d.*)

2.6. Transferencia del estado de presentación (REST)

Es un estilo arquitectónico para proporcionar estándares de comunicación entre sistemas informáticos en la web, lo que facilita la comunicación de los sistemas entre sí. Los sistemas compatibles con REST, a menudo llamados sistemas RESTful, se caracterizan por no tener estado y separar las preocupaciones del cliente y el servidor como se muestra en la figura 5. (*What Is REST? / Codecademy, n.d.*)

Figura 5 Comunicación REST

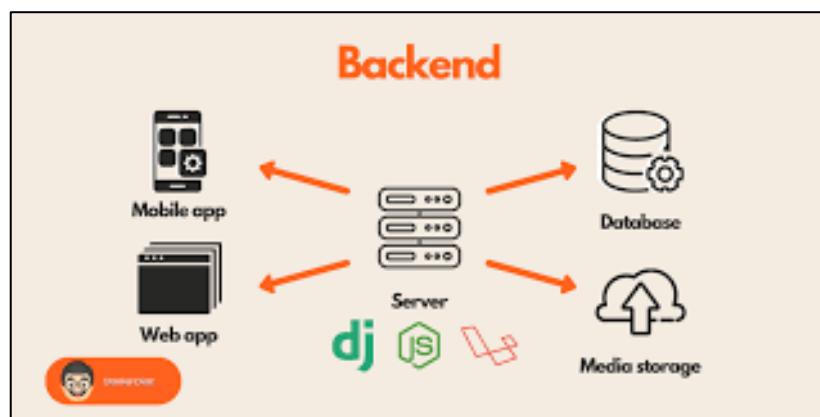


Fuente: (*What Is REST? / Codecademy, n.d.*)

2.7. Backend

Es la parte que se ejecuta en un servidor, es el responsable de tratar los datos en términos general, esto significa que es responsable de recibir las peticiones de los clientes para que se les proporcione datos, para que puedan visualizarlos como se muestra en la figura 6. (Filipova & Vilão, 2018)

Figura 6 Servidor backend



Fuente: (*Geekflare*, n.d.)

2.8. Frontend

El desarrollo Frontend, también conocido como desarrollo del lado del cliente, es la práctica de producir HTML, CSS Y JavaScript para un sitio web o una aplicación web para que un usuario pueda verlos e interactuar directamente con ellos como se muestra en la figura 7. (Gerasimov et al., 2020)

Figura 7 Tecnologías frontend

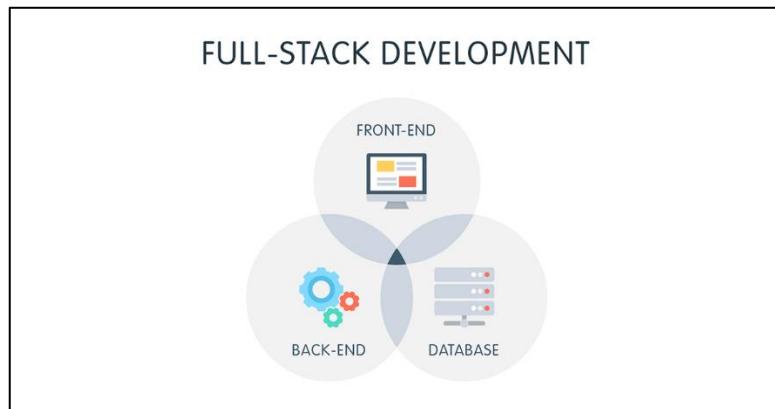


Fuente: Craft Solutions

2.9. FullStack

En el desarrollo de software se denomina fullstack a una persona que maneja uno o más stacks de tecnologías que abarcan desde el frontend y backend, siendo capaces de construir, dar soporte e implementar nuevas funcionalidades a un proyecto como se muestra en la figura 8.

Figura 8 Capos que abraza un desarrollador fullstack



Fuente: revenueriver

2.10. PHP

PHP (acrónimo recursivo de PHP: Hypertext Preprocessor) es un lenguaje de secuencias de comandos de propósito general de código abierto ampliamente utilizado que es especialmente adecuado para el desarrollo web y se puede incrustar en HTML. (*PHP: ¿Qué Es PHP? - Manual*, n.d.)

2.11. Framework Laravel

Laravel es un marco de aplicación web con una sintaxis expresiva y elegante. Un marco web proporciona una estructura y un punto de partida para crear proyectos en el lenguaje de php. (*The PHP Framework For Web Artisans*, n.d.)

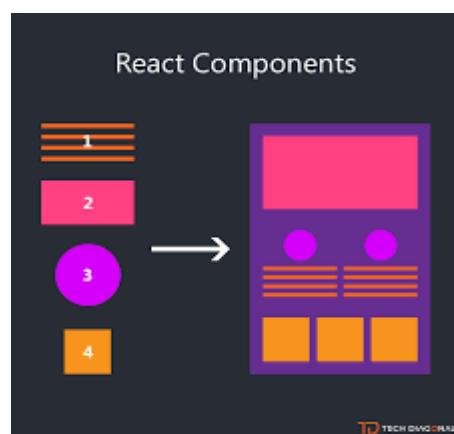
2.12. MySQL

MySQL es la base de datos de código abierto más popular del mundo. Según DB-Engines, MySQL se ubica como la segunda base de datos más popular, detrás de Oracle Database. MySQL impulsa muchas de las aplicaciones más visitadas, incluidas Facebook, Twitter, Netflix, Uber, Airbnb, Shopify y Booking.com. (*What Is MySQL? / Oracle*, n.d.)

2.13. React

React te permite crear interfaces de usuario a partir de piezas individuales llamadas componentes como se muestra en la figura. (*Quick Start – React*, n.d.)

Figura 9 React componentes



Fuente: Medium

2.14. GoDaddy

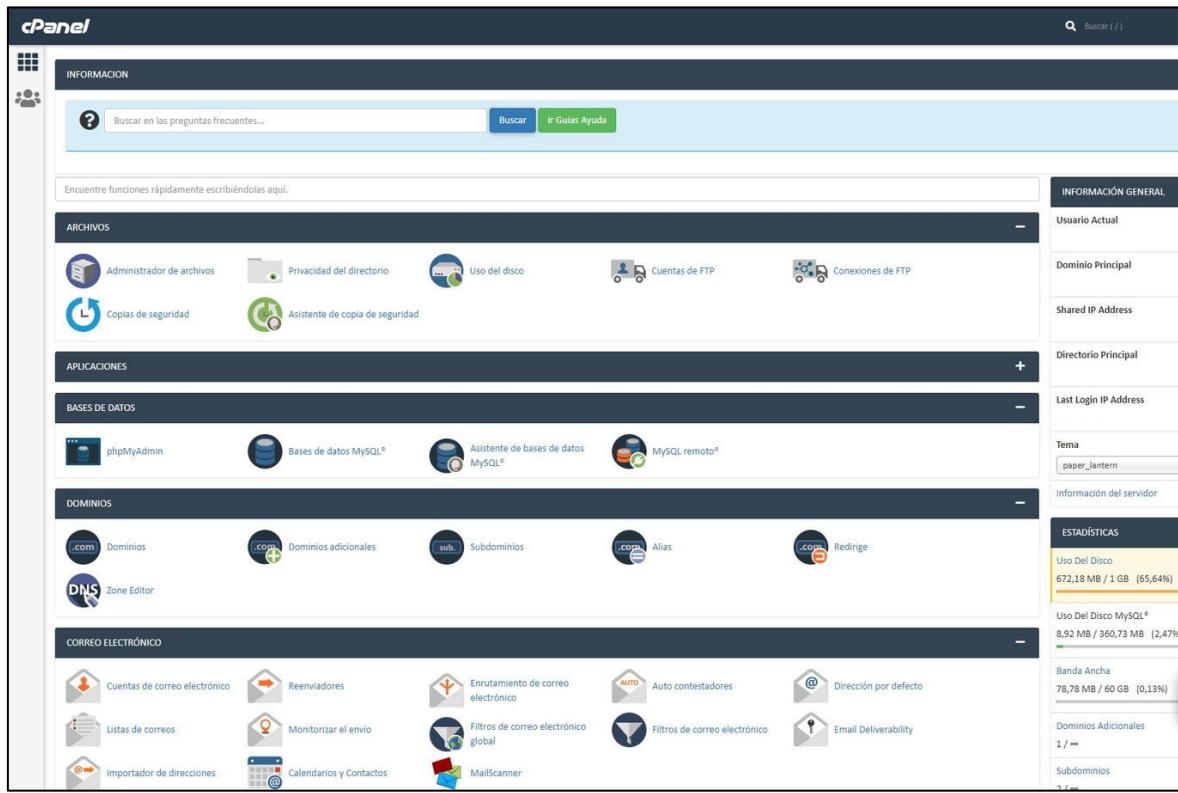
GoDaddy es una empresa a nivel mundial que ofrece servicios de hosting, alquiler de dominios, correos corporativos y marketing. (*GoDaddy PE*, n.d.)

2.15. Cpanel

CPanel es uno de los paneles de control de alojamiento web basados en Linux más populares, que presenta una variedad de módulos como se muestra en la figura 10 que incluyen Archivos, Preferencias, Bases de datos, Aplicaciones web, Dominios, Métricas,

Seguridad, Software, Avanzado y Correo electrónico. La facilidad de uso y una interfaz simple hacen de cPanel una opción popular entre los webmasters. (*What Is CPanel?, n.d.*)

Figura 10 Panel Cpanel



Fuente: Elaboración propia

2.16. SMTP2Go

SMTP2GO es una plataforma de entrega de correo electrónico basada en la nube que ayuda a entregar y rastrear correos electrónicos. Sus características incluyen soporte de tickets, chat en vivo, asistencia de configuración, bucles de retroalimentación, monitoreo de listas negras, detección de trampas de spam, análisis en tiempo real, resúmenes semanales y seguimiento de rebotes/spam. (*SMTP2GO Software, n.d.*)

CAPITULO III

ACTIVIDADES DESARROLLADAS

3.1. Situación Actual

3.1.1. Problemática

La empresa “HMF inversiones S.A.C”, tiene como uno de sus productos, un sistema de gestión comercial de escritorio llamado “KAMAY”. Debido al cambio tecnológico donde todos los servicios se migran a la web, la empresa se ve en la necesidad de realizar la migración de su sistema.

Actualmente la empresa tiene en ejecución dos proyectos webs derivados de un mismo proyecto:

- BOOKING: Orientado a la gestión de los procesos que se realizan en un hotel.
 - Modulo Menú principal: Agrupa submódulos principales para el proceso de reserva de habitaciones.
 - Panel de control (dashboard): Muestra un resumen de las habitaciones y ventas.
 - Reserva / estancia (grid): Encargada de la gestión de reservas y estancias.
 - Modulo Mantenedores: Agrupa submódulos generales de gestión.
 - Comprobantes (vouchers): Encargada de la gestión de comprobantes.
 - Habitación (room): Encargada de la gestión de habitaciones.
 - Clientes / Proveedores (clients_providers): Encargada de la gestión de clientes y proveedores.
 - Clases de habitación (class_room): Encargada de la gestión de clases de habitaciones.

- Categoría / Marca (categories_brand): Encargada de la gestión de categoría y marcas.
- Series (series): Gestión de series.
- Paquetes Turísticos (tourist-packages): Encargada de la gestión de paquetes turísticos.
- Usuarios (users): Encargada de la gestión de usuarios.
- Productos (products): Encargada de la gestión de productos.
- Modulo CPE: Agrupa submódulos para la comunicación con la SUNAT para la facturación electrónica.
 - Gestión CPE (manage_cpe): Encargada de la gestión de CPE.
- Modulo Tesorería: Agrupa submódulos correspondientes a las finanzas del sistema.
 - Administración de caja (manage_cash): Encargada de la gestión de administración de caja.
- Modulo Seguridad: Agrupa submódulos para la seguridad del sistema.
 - Control de acceso (access_control): Encargada de la gestión de control de acceso.
 - Control de menú (menu_control): Gestión de control de menú.
- Modulo Procesos: Agrupa submódulos necesarios para la facturación de los distintos consumos.
 - Facturación (sales): Gestión de facturación.
 - Pedidos (orders): Encargada de la gestión de pedidos.
 - Proformas (proformas): Encargada de la gestión de proformas.
 - Nota de crédito (credit_note): Encargada de la gestión de

notas de créditos.

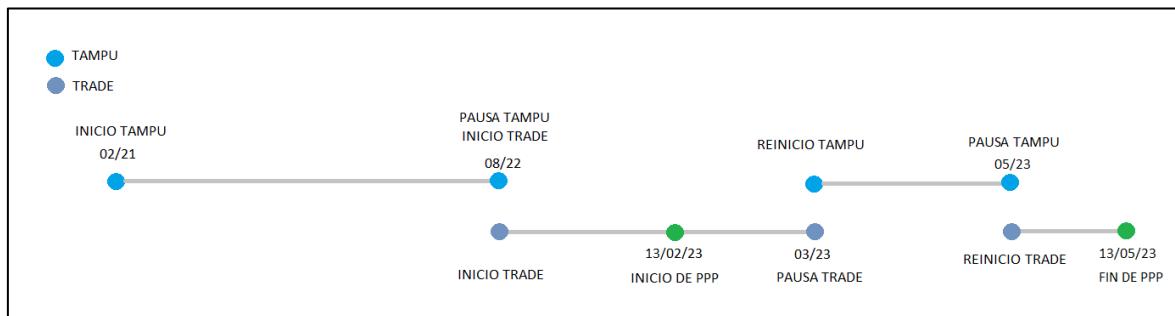
- Modulo Opciones: Agrupa submódulos para la gestión de la empresa.
 - Establecimiento (establishments): Encargada de la gestión de establecimiento.
- TRADE: Orientado en la gestión de los procesos que se realizan en una agencia de transportes.
 - Modulo Menú principal: Agrupa submódulos principales para el proceso de reserva de habitaciones.
 - Panel de control (dashboard): Muestra un resumen de las ventas de pasajes, encomiendas y garaje.
 - Modulo Mantenedores: Agrupa submódulos generales de gestión.
 - Comprobantes (vouchers): Encargada de la gestión de comprobantes.
 - Clientes / Proveedores (clients_providers): Encargada de la gestión de clientes y proveedores.
 - Series (series): Gestión de series.
 - Usuarios (users): Encargada de la gestión de usuarios.
 - Productos (products): Encargada de la gestión de productos.
 - Gestión de conductores (exit_vehicles): Encargada de la gestión de los conductores.
 - Modulo CPE: Agrupa submódulos para la comunicación con la SUNAT para la facturación electrónica.
 - Gestión CPE (manage_cpe): Encargada de la gestión de CPE.
 - Modulo Tesorería: Agrupa submódulos correspondientes a las finanzas del sistema.
 - Administración de caja (manage_cash): Encargada de la

gestión de administración de caja.

- Modulo Seguridad: Agrupa submódulos para la seguridad del sistema.
 - Control de acceso (access_control): Encargada de la gestión de control de acceso.
 - Control de menú (menu_control): Encargada de la gestión de control de menú.
- Modulo Procesos: Agrupa submódulos necesarios para la facturación de los distintos consumos.
 - Facturación (sales): Gestión de facturación.
 - Nota de crédito (credit_note): Encargada de la gestión de notas de créditos.
- Modulo Opciones: Agrupa submódulos para la gestión de la empresa.
 - Establecimiento (establishments): Encargada de la gestión de establecimiento.

El primer proyecto en pasar a construcción fue TAMPU en febrero del 2021 el cual al alcanzar una madurez en el desarrollo se realizó una rama en agosto del 2022 como se muestra en la figura 11.

Figura 11 Regla de tiempo de los proyectos



Fuente: Elaboración propia

3.2. Objetivos

- General

Construir nuevos submódulos, desplegar y depurar los proyectos Booking y Trade.

- **Específicos**

- Implementación de nuevos módulos a las aplicaciones ya mencionadas.
 - Implementación del servicio de smtp2go para el envío de correos electrónicos.
 - Despliegue de las aplicaciones Trade y Booking.

3.3. Alcance

Este trabajo está limitado al alcance que tendrán los módulos de:

- Trade:
 - Moduló de Salida.
 - Moduló de Adelantos.
 - Moduló de encomiendas.
 - Moduló de facturación.
 - Moduló de dashboard.
 - Vista de consulta de encomiendas.
 - Booking:
 - Moduló grid.
 - Vista de registro de huéspedes.

3.4. Cronograma de actividades

Tabla 1 Cronograma de actividades

las reservas (Booking).											
Desplegar y depurar proyectos en servidor de producción (Booking y Trade)											

Fuente: Elaboración propia

3.5. Escenario de trabajo

3.5.1. Dirección de proyecto

El jefe del proyecto es ing. Josue Acuña Cordoba.

3.5.2. Equipo de trabajo

Tabla 2 Equipo de trabajo

Líder del proyecto – Analista programador Senior	
Nombre	Josue Acuña Cordoba
Cargo en la empresa	Gerente general. Ing. en sistemas. Analista programador senior.
Contacto	bodjac@hmail.com
Analista programador	
Nombre	Iomar Igor Alegre Barrera
Cargo en la empresa	Analista programador. Practicante.
Contacto	iomar.alegre@unas.edu.pe

Fuente: Elaboración propia.

3.5.3. Recursos

Durante el tiempo de desarrollo de las prácticas preprofesionales se usaron los siguientes recursos:

Tabla 3 Recursos

Recurso	Descripción
Visual Studio Code	Es un ide que nos facilita el desarrollo a desarrolladores web y JavaScript, con extensiones para admitir casi cualquier lenguaje de programación.
Anydesk	Es un software que nos permite el control remoto a un pc, el propietario proporciona acceso remoto independiente de la plataforma a computadoras personales y otros dispositivos que ejecutan la aplicación host.
HeidiSQL	Es un software de administración de código abierto para MySQL, Microsoft SQL Server y PostgreSQL.
Postman	Es un software que permite realizar peticiones a los endpoint de un api.

Fuente: Elaboración propia.

3.6. Descripción de la arquitectura del proyecto

3.6.1. Arquitectura del proyecto

Ambos proyectos están compuestos por frontend y backend, al ser Tampu una rama de Booking comparte la misma arquitectura.

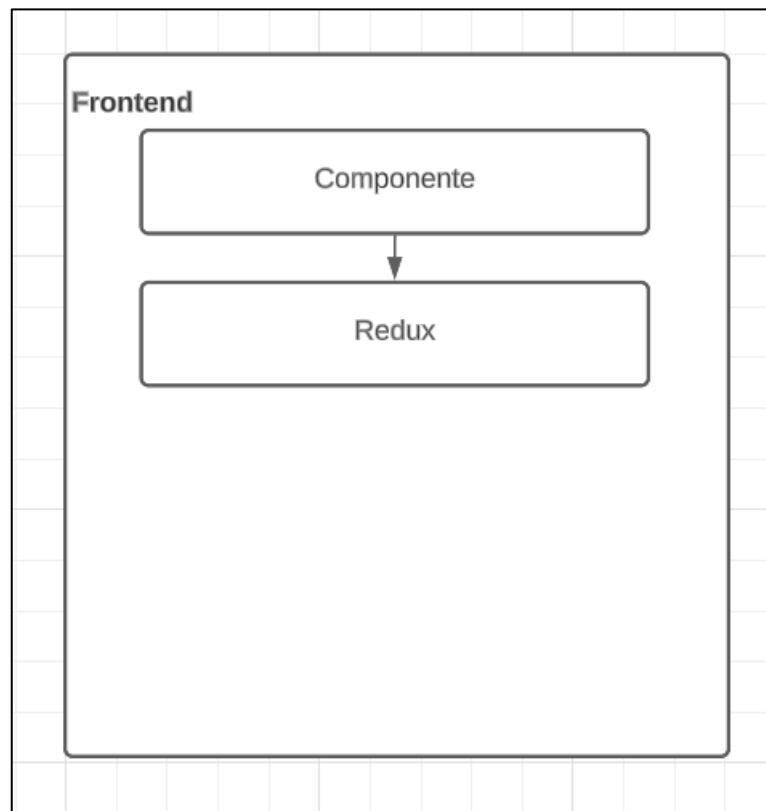
A continuación detallará la arquitectura del proyecto de frontend, backend y forma en las que estas dos arquitecturas trabajan.

3.6.1.1. Frontend

La figura 12 muestra la arquitectura frontend del proyecto el cual está construido en componentes los cuales utilizar redux como una base de datos temporal el cual

permite guardar los datos que son consultados al backend para ser accesible a cualquier componente sin importar la jerarquía.

Figura 12 Diagrama frontend

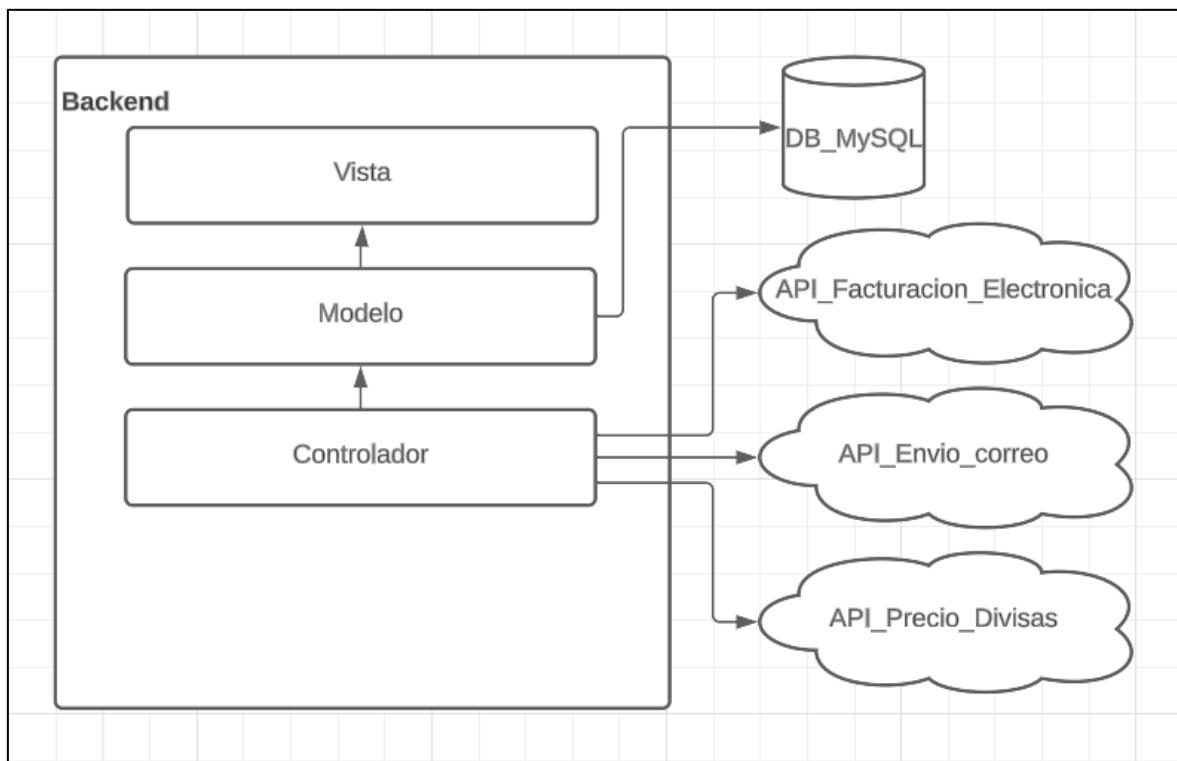


Fuente: Elaboración propia

3.6.1.2. Backend

En la figura 13 se muestra las relaciones entre las distintas capas del frontend y cómo se relacionan con las capas del backend, también se muestra el método de comunicación y los distintos servicios que terceros que se comunican el backend.

Figura 13 Arquitectura backend



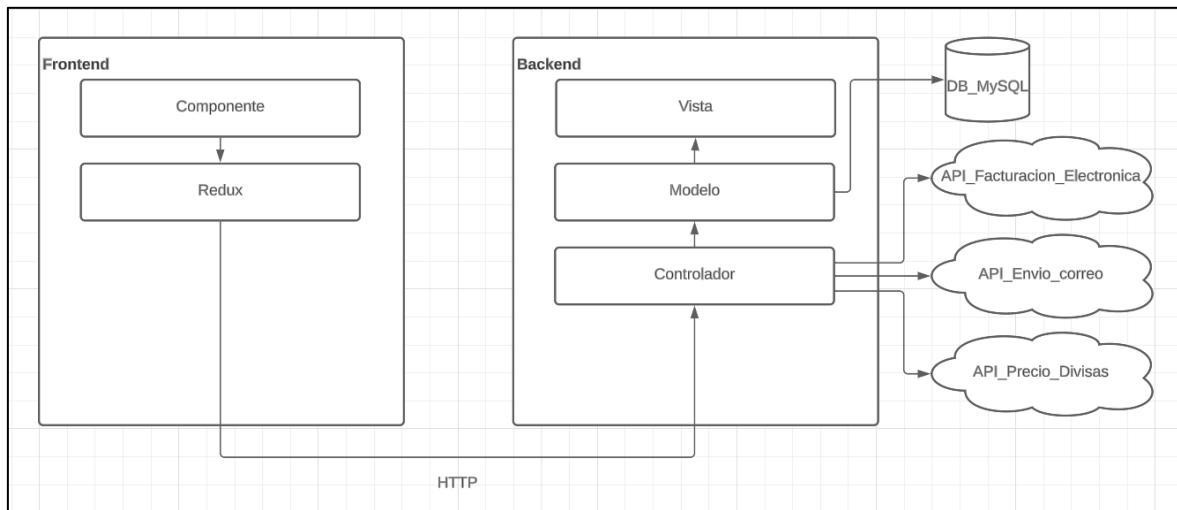
Fuente: Elaboración propia

3.6.1.3. Arquitectura

La comunicación entre ambas arquitecturas se realiza a través de http, el frontend consume los endpoints del api del backend.

Las peticiones que llegan desde el servicio de Frontend son recibidos por el controlador backend, que será el encargado de dirigir la solicitud a los modelos y a la base de datos o a los servicios que el frontend solicite como se muestra en la figura 14.

Figura 14 Arquitectura de los proyectos



Fuente: Elaboración propia

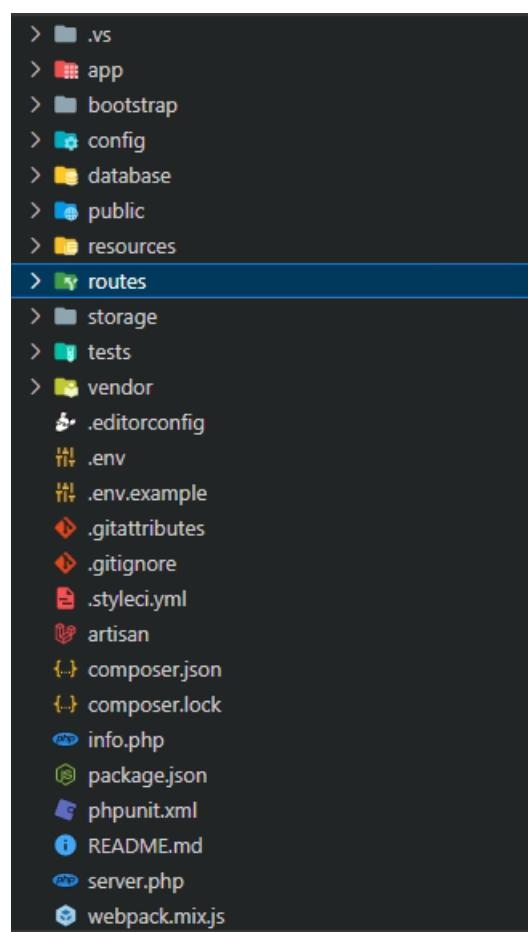
3.6.2. Distribución de carpetas

La distribución de carpetas en la misma para ambos proyectos para el backend y el frontend.

3.6.2.1. Backend

A continuación, se presenta en la figura 15 la distribución de carpetas del proyecto de backend generados por laravel 8 en el IDE visual studio code.

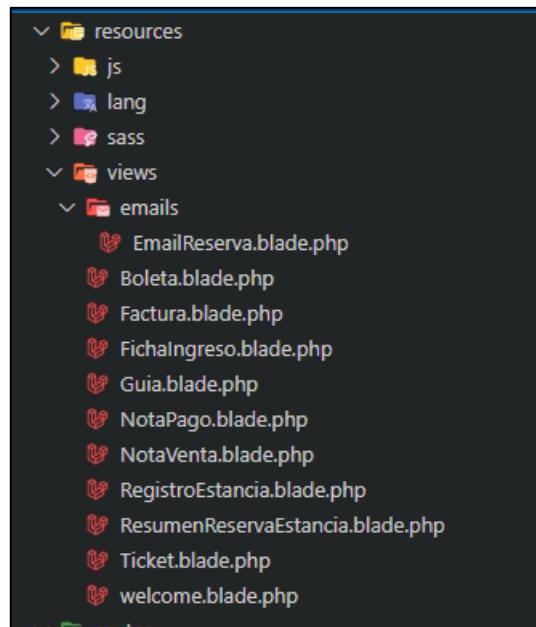
Figura 15 Distribución de carpetas en visual studio code



Fuente: Elaboración propia

A continuación, se presenta en la figura 16 la distribución de carpetas de las vistas, en su totalidad son utilizadas para generar reportes pdf.

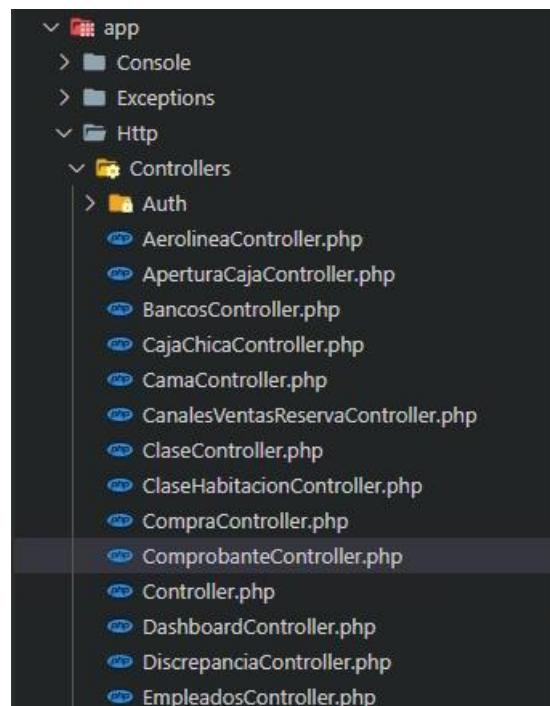
Figura 16 Distribución de carpetas de las vistas



Fuente: Elaboración propia

A continuación, se presenta en la figura 17 la distribución de carpetas de los controladores.

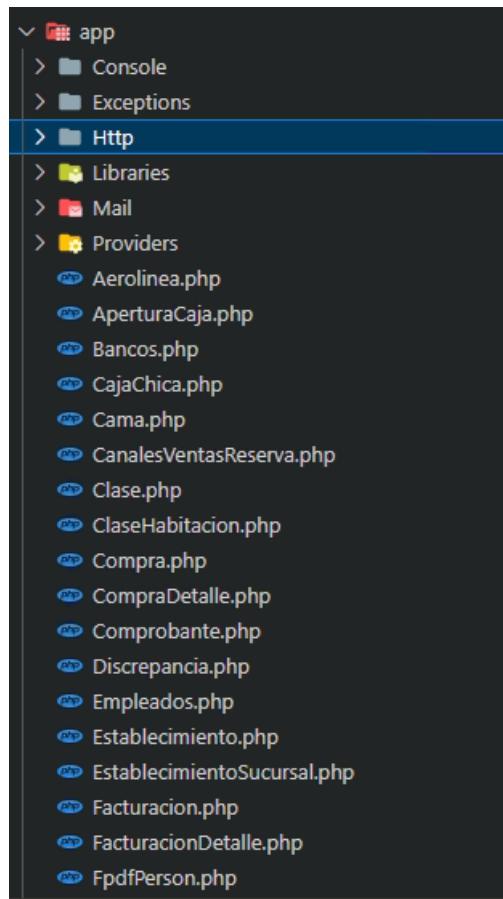
Figura 17 Distribución de carpetas de los controladores



Fuente: Elaboración propia

A continuación, se presenta en la figura 18 la distribución de carpetas de los modelos.

Figura 18 Distribución de carpetas de los modelos

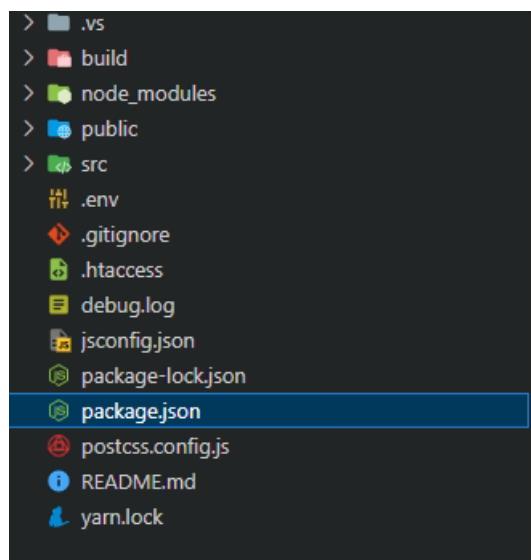


Fuente: Elaboración propia

3.6.2.2. Frontend

A continuación, se presenta en la figura 19 la distribución de carpetas del proyecto de frontend generado con React versión 16 en el IDE visual studio code.

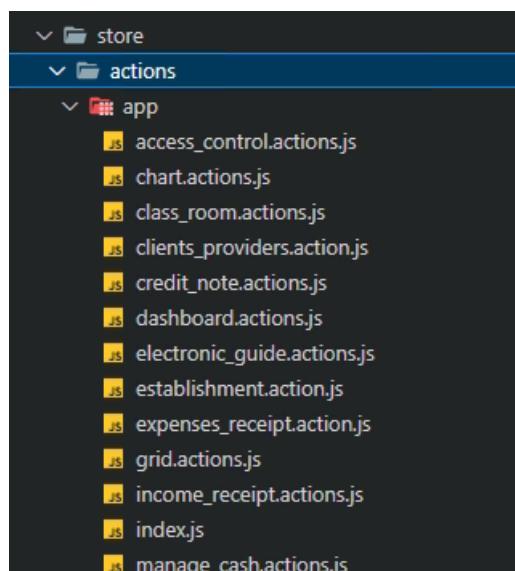
Figura 19 Distribución de carpetas de frontend



Fuente: Elaboración propia

A continuación, se presenta en la figura 20 la distribución de carpetas de los actions de redux.

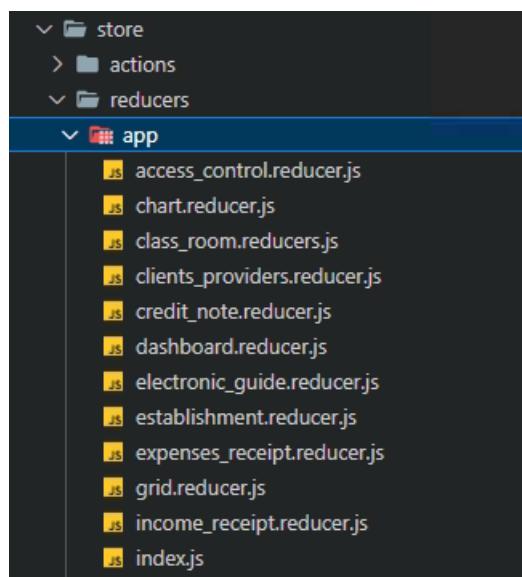
Figura 20 Distribución de carpetas de los actions de redux



Fuente: Elaboración propia

A continuación, se presenta en la figura 21 la distribución de carpetas de los reducers de redux.

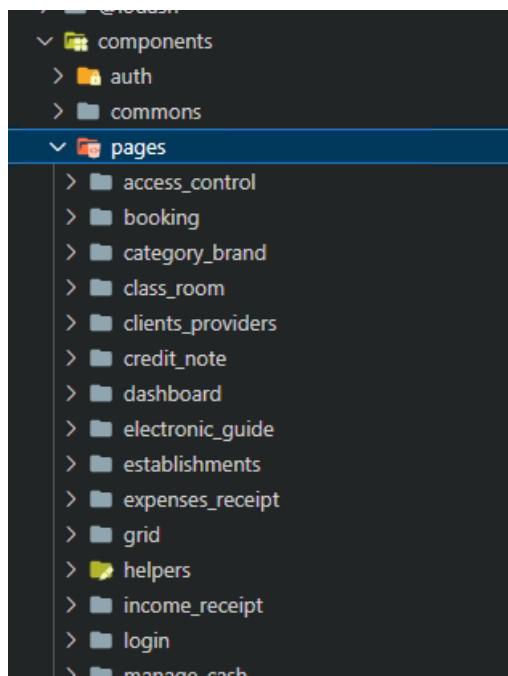
Figura 21 Distribución de carpetas de los reducers de redux



Fuente: Elaboración propia

A continuación, se presenta en la figura 22 la distribución de carpetas de los componentes.

Figura 22 Distribución de carpetas de los componentes



Fuente: Elaboración propia

3.7. Método de trabajo

Se realiza una reunión por meet con el líder del proyecto para definir las tareas y disipar dudas. El resultado de la reunión es el objetivo con la lista de tareas a desarrollar, como se muestra en las figuras 23 y 24.

Figura 23 Objetivo con tareas

O. Nuevas funcionalidades al submódulo grid							
T1. Agrega botón para enviar correo de registro de huéspedes en el modal de habitación							
T2. cuando el estado de la habitación este en reserva preparar el botón para consumir el endpoint que permitirá el envío del correo con la información de la reserva al usuario contacto.							
T3. Hacer que el ancho de la grilla sea responsive para pantallas con resolución de 1920 x 1080.							
T4. hacer endpoints donde muestre los reportes pdf de salidas del dia y arrivos del dia							
T5. Mostrar en un modal el pdf de reporte de salidas del día.							
T6. Mostrar en un modal el pdf de reporte de parte diario donde se listan todos los arrivos del dia.							

Fuente: Elaboración propia

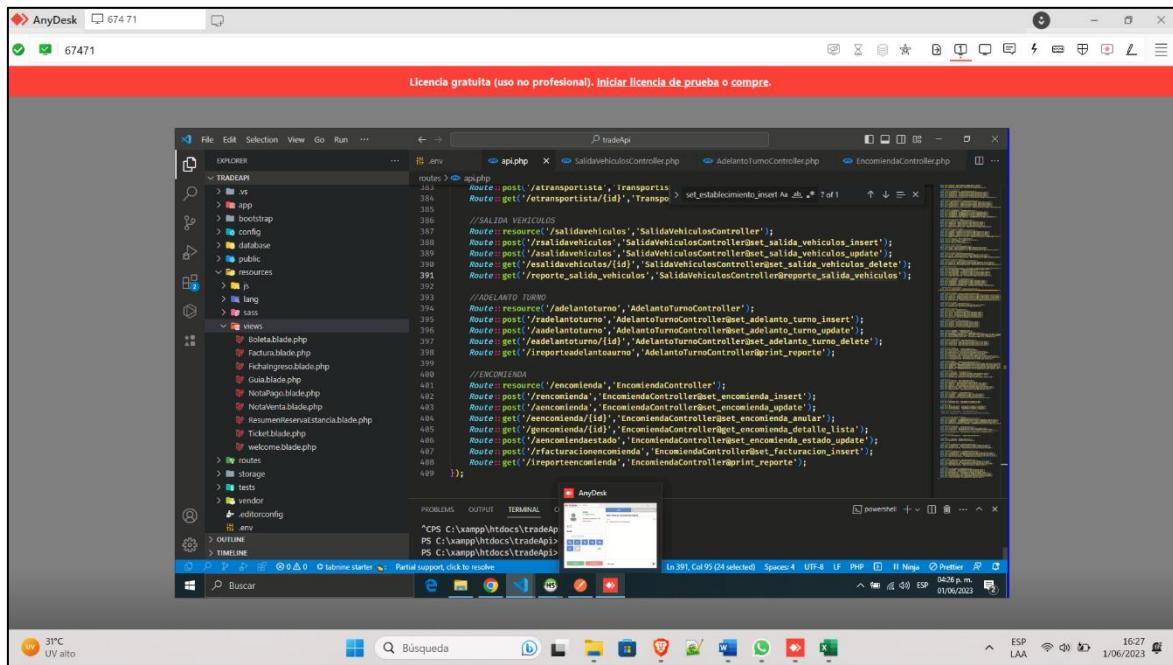
Figura 24 Objetivo con tareas

O. Creacion del endpoint para envio de correo sobre reservas							
T1 Implementar servidor smtp							
T2 Crear endpoint para el envio							
T3 El correo debe mostrar la imagen del hotel, una breve descripción de contacto y el link de registro de huéspedes							

Fuente: Elaboración propia

La figura 24 muestra un reporte donde se describe de manera corta en donde se realizaron los cambios, estos reportes solo se realizan al corregir errores y no al implementar nuevas funcionalidades.

Figura 25 Entorno de trabajo



Fuente: Elaboración propia

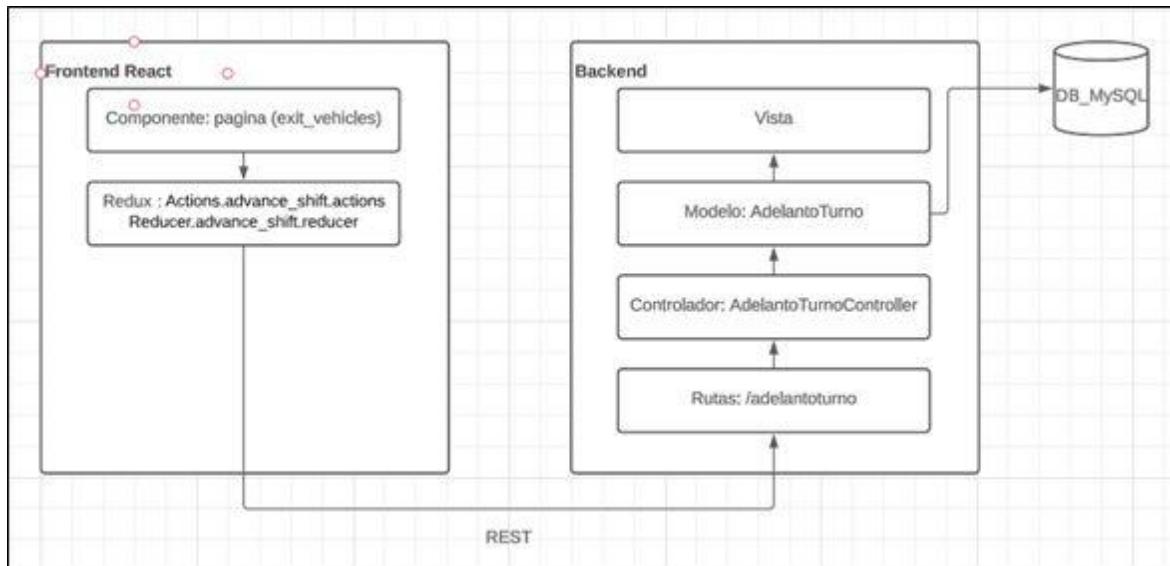
En la figura 25 muestra la conexión mediante anydesk al pc de desarrollo donde se encuentran los proyectos backend y frontend desplegados en servidores locales.

3.8. Construir submódulo de salida (Trade)

El submódulo de salida será el encargado de gestionar las salidas programadas de los conductores, donde registrarán el conductor y el monto a cobrar por encomiendas a oficina, estos registros serán visibles para los usuarios que se encuentren en la misma sucursal.

Para iniciar la construcción se tendrá en cuenta el diagrama de flujo de trabajo de la figura (26) donde se describen los archivos a interactuar en el frontend y backend.

Figura 26 Flujo de trabajo para el submódulo de gestión de salidas



Fuente: Elaboración propia

3.8.1. Tareas

En la figura 26 se muestra el objetivo a alcanzar y las tareas a realizar.

Figura 27 Objetivo: Crear submódulo salida y asignar al módulo Tesorería

O. Crear el submodulo de salida y asignarla al modulo tesoreria		
T1. Crear el crud de salida		
crear: el correlativo será incrementable por dia, el correlativo comienza en 1 cada dia,		
luego de registrar, mostrar el pdf de la salida para imprimir		
editar: permitir actualizar idtransportista y monto		
Listar: listar las salida de los vehiculos que tienen el estado activo, nombre del chofer,		
placa de cehiculo que condu activo=0		eliminado = -1
Eliminar: Cambiar de estado a -1		
T2. Crear un endpoint que permita imprimir la salida por el idsalidavehiculo		
T3. Crear el submódulo de salida		
crear un motal para crear, editar la salida, el modal se desplegará luego de pulsar el		
boton de registrar o en la accion de editar de la tabla		
en la tabla se listaran los registros de salidas, adicional tendra las acciones de editar,		
eliminar y imprimir		
Luego de cada registro o actualización recargar automaticamente los datos de la tabla		

Fuente: Elaboración propia

3.8.1.1. Referencias

En la figura 27 se muestra una vista como referencia para los botones de registrar y la tabla donde se mostrarán los datos.

Figura 28 Vista referencia

Mantenedores de conductor						INICIO / GESTION DE CONDUCTOR
Gestion de conductor						
Registrar						Reporte
Acciones	Razón Social	RUC	Dirección	Marca / Placa		
<input checked="" type="checkbox"/> 	ADOLFO VARGAS RUIZ	11111111	-	- / AZP-212		
<input checked="" type="checkbox"/> 	ALAN RODRIGUEZ SAAVEDRA	11111111	-	- / M6G-837		
<input checked="" type="checkbox"/> 	ALENCAR REATEGUI TELLO	11111111	-	- / BCI-569		
<input checked="" type="checkbox"/> 	AMADEO BARADALES GONZAL...	11111111	-	- / AWR-258		
<input checked="" type="checkbox"/> 	CARLOS ALBERTO LOPEZ SAN...	11111111	-	- / F5I-952		
<input checked="" type="checkbox"/> 	DANIEL SANCHEZ HILDEBRANTD	11111111	-	- / S1C-051		
<input checked="" type="checkbox"/> 	DIOGENES CORAL PANDURO	11111111	-	- / M6Z-965		
<input checked="" type="checkbox"/> 	EDGARDO RUIZ LOZANO	11111111	-	- / P2V-060		

Copyright © 2020 [ACCORD Software](#). Todos los derechos reservados. Version 1.0.0

Fuente: Elaboración propia

En la figura 28 se muestra una impresión que servirá como referencia para imprimir un registro de salida.

Figura 29 Referencia de impresión de salida



Fuente: HMF inversiones

3.8.2. Ejecución de las tareas

Las tareas se dividieron en backend y frontend.

3.8.2.1. Backend

3.8.2.1.1. T1 Crear el crud de salida y T2 Crear un endpoint que permita imprimir la salida por el idsalidavehiculo

Se realizo la creación del controlador donde se crearán las funciones para el crud como se muestra en la figura 30.

Figura 30 Fragmento de Código del controlador de salida vehículos

```
namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\SalidaVehiculos;
use Codedge\Fpdf\Fpdf;
use Illuminate\Support\Facades\Validator;
use Illuminate\Support\Facades\DB;

class SalidaVehiculosController extends Controller
{
    /**
     * Handle the incoming request.
     *
     * @param \Illuminate\Http\Request $request
     * @return \Illuminate\Http\Response
     */
    public function index(Request $request)
    {
```

Fuente: Elaboración propia

Luego de la creación del controlador se construyó las funciones para (listar, crear, editar, eliminar, imprimir) como se muestra en las figuras 31, 32, 33, 34 y 35.

En la figura 31 se muestra el fragmento de código para listar los registros de encomiendas en la tabla de salidas.

Figura 31 Fragmento de Código para listar los registros de salida

```
public function index(Request $request)
{
    $vehiculos = DB::table('salida_vehiculos as sv')
        ->join('transportista as t','t.idtransportista','=','sv.idtransportista')
        ->select('sv.*','t.razonsocial','t.placa')
        ->where('sv.estado','!=',-1)
        ->get();
    $json = array();
    if(!empty($vehiculos)){
        $json = array(
            "status"=>200,
            "mensaje"=>"total_registros".count($vehiculos),
            "detalles"=>$vehiculos
    }
}
```

Fuente: Elaboración propia

En la figura 32 se muestra el fragmento de código para registrar salida de vehículo.

Figura 32 Fragmento de código para registrar salida de vehículo

```
public function set_salida_vehiculos_insert(Request $request){
    $json = array();
    $validator = Validator::make($request->all(), [
        'idtransportista' => 'required',
        'monto' => 'required',
        'idusers' => 'required'
    ]);
    if ($validator->fails()) {
        return response()->json($validator->errors()->toJson(), 400);
    }
    $data=array(
        'idtransportista'=>$request->input('idtransportista'),
        'monto'=>$request->input('monto'),
        'fecha'=>date("Y-m-d H:i:s"),
        'idusers'=>$request->input('idusers')
    );
    $salidaTemp=DB::table('salida_vehiculos')
        ->whereDate('fecha',date("Y-m-d"))
        ->orderBy('idsalidavehiculos', 'desc')
```

Fuente: Elaboración propia

En la figura 33 se muestra el fragmento de código para actualizar registro de salida vehículo.

Figura 33 Fragmento de Código para actualizar registro de salida vehículo

```
public function set_salida_vehiculos_update(Request $request){
    $id = $request->input("idsalidavehiculos");
    $json = array();
    //Recoger datos
    $datos = array(
        'idtransportista'=>$request->input("idtransportista"),
        'monto'=>$request->input('monto'),
    );
    if(!empty($datos)){
        $validar = DB::table('salida_vehiculos')
            ->where('idsalidavehiculos', '=', $id)
            ->get();

        if (!$validar->count() == 0){
            $menu = SalidaVehiculos::where("idsalidavehiculos", $id)->update($datos);
            $json = array(
                "status"=>200,
                "mensaje"=>"Registro exitoso, ha sido actualizado",
                "data"=>
            );
        }
    }
}
```

Fuente: Elaboración propia

En la figura 34 se muestra el fragmento de código para eliminar un registro de salida.

Figura 34 Fragmento de código para eliminar registro de salida vehículo

```
public function set_salida_vehiculos_delete($idsalidavehiculos){
    $json = array();
    $validar = DB::table('salida_vehiculos')
        ->where('idsalidavehiculos', '=', $idsalidavehiculos)
        ->get();

    if (!$validar->count() == 0){
        SalidaVehiculos::where("idsalidavehiculos", $idsalidavehiculos)->update(['estado'=>-1]);
        $json = array(
            "status"=>200,
            "mensaje"=>"Registro anulado",
        );
    }
}
```

Fuente: Elaboración propia

En la figura 35 se muestra el fragmento de código que permitirá generar el pdf para la impresión del registro de salida.

Figura 35 Fragmento de código de la impresión de registro de salida vehículo

```
public function get_salida_vehiculos_print($id){
    $vehiculos = DB::table('salida_vehiculos as sv')
        ->join('transportista as t','t.idtransportista','=','sv.idtransportista')
        ->join('users as u','u.id','=','sv.idusers')
        ->select('sv.*','t.razonsocial','t.placa','u.name')
        ->where([
            ['sv.idsalidavehiculos','=',$id]
        ])
    ->first();
    $fpdf= new Fpdf('P','mm',array(80,350));
    $fpdf->AddPage();
    $fpdf->SetFont('Courier', 'B', 10);
    $textypost=5;
    $fpdf->setXY(5,$textypost);
    $fpdf->Cell(0,0,utf8_decode("CONTROL DE SALIDA Nº: ".str_pad($vehiculos->correlativo, 7, "0", $textypost+=4;
    $fpdf->setXY(2,$textypost);
    $fpdf->SetFont('Courier', 'B', 8);
    $fpdf->Cell(0, 0, utf8_decode("FECHA EMISIÓN: ") . date('d/m/Y', strtotime(explode(' ', $vehiculo
    $textypost+=3;
```

Fuente: Elaboración propia

Luego se realizó los endpoints necesarios para llamar a cada función del controlador y la impresión del registro, como se muestra en la figura 36.

Figura 36 Fragmento de código de los endpoints

```
//SALIDA VEHICULOS
Route::resource('/salidavehiculos','SalidaVehiculosController'); //listar
Route::post('/rsalidavehiculos','SalidaVehiculosController@set_salida_vehiculos_insert'); //crear
Route::post('/asalidavehiculos','SalidaVehiculosController@set_salida_vehiculos_update'); //actualizar
Route::get('/esalidavehiculos/{id}','SalidaVehiculosController@set_salida_vehiculos_delete'); //eliminar
Route::get('/salidavehiculos/{id}','SalidaVehiculosController@get_salida_vehiculos_print'); //imprimir
```

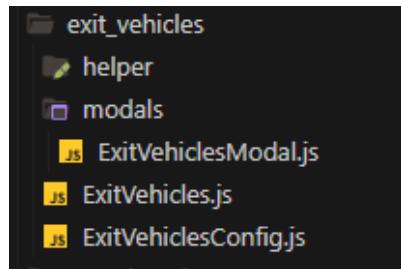
Fuente: Elaboración propia

3.8.2.2. Frontend

3.8.2.2.1. T3. Crear el submódulo de salida

Se creo la carpeta de “exit_vehicles” en el directorio de carpetas como se muestra en la figura 37.

Figura 37 Distribución de archivos de la vista salida vehículo



Fuente: Elaboración propia

Se inicializó el componente de “ExitVehicles” el cual se mostrará en la vista como de muestra en la figura 38, se el archivo “ExitVehiclesConfig” se establece los permisos para la ruta y su url como se muestra en la figura 39.

Figura 38 Fragmento de código del componente ExitVehicles

```
const ExitVehicles = (props) => {
  const [managementDriverModal, setManagementDriverModal] = useState(false);
  const [managementDriverSelect, setManagementDriverSelect] = useState(null);
  const [rows, setRows] = useState(null);
  useEffect(() => {
    props.getManagementDriver();
  }, []);
  useEffect(() => {
    props.getExitVehicles();
  }, [props.crud_exit_vehicles]);
```

Fuente: Elaboración propia

Figura 39 Fragmento de código de ExitVehiclesConfig para la configuración del acceso y la ruta

```
export const ExitVehiclesConfig = {
  auth: AuthRoles.admin,
  routes: [
    {
      path: `${process.env.PUBLIC_URL}/exit_vehicles`,
      component: React.lazy(() => import('./ExitVehicles'))
```

Fuente: Elaboración propia

Luego se construyó los actions y reducer necesarios para redux para el consumo y almacenamiento de los datos obtenidos desde el api.

En la figura 40 se muestra el consumo de los registros de las salidas mediante el endpoint construido en el api del backend.

Figura 40 Fragmento de Código para listar los registros de las salidas

```
export function getExitVehicles() {
  const request = axios.get(
    `${process.env.REACT_APP_API_URL}/api/salidavehiculos`
  );

  return (dispatch) =>
    request.then((response) => {
      if (parseInt(response.data.status) === 404) {
        if (localStorage.getItem("access_token")) {
          console.log(response.data.detalle);
          localStorage.removeItem("access_token");
          delete axios.defaults.headers.common["Authorization"];
          return dispatch(Actionss.logoutUser());
        }
        return;
      }
      return dispatch({
        type: GET_LIST_EXIT_VEHICLES,
        payload: response.data.detalles === null ? [] : response.data.detalles,
      });
    });
}
```

Fuente: Elaboración propia

En la figura 41 se muestra el consumo del endpoint para registrar una salida, también se muestra que cuando el backend responde al frontend con un código 200, el frontend realiza automáticamente el consumo del endpoint para la impresión del registro de la factura.

Figura 41 Fragmento de Código del consumo del endpoint de registro de salida y impresión

```
export function saveExitVehicles(form) {
  const request = axios.post(
    `${process.env.REACT_APP_API_URL}/api/rsalidavehiculos`,
    form
  );
  return (dispatch) =>
    request.then((response) => [
      if (parseInt(response.data.status) === 404) {
        if (localStorage.getItem("access_token")) { ... }
      }
      return;
    ])
    if (response.data.status === 200) {
      toast.success(response.data.mensaje);
      window.open(
        `${process.env.REACT_APP_API_URL}/api/salidavehiculos/${response.data.detalles.idsalidavehiculos}`,
        "_blank"
      );
    } else {
    }
}
```

Fuente: Elaboración propia

En la figura 42 se muestra el consumo del endpoint para actualizar un registro de salida vehículos.

Figura 42 Fragmento de Código para el consumo del endpoint para actualizar registro de salida vehículo

```
export function updateExitVehicles(form) {
  const request = axios.post(
    `${process.env.REACT_APP_API_URL}/api/asalidavehiculos`,
    form
  );

  return (dispatch) =>
    request.then((response) => {
      if (parseInt(response.data.status) === 404) { ... }
      if (response.data.status === 200) { ... }
      } else { ... }
    }
    dispatch({
      type: CRUD_EXIT_VEHICLES,
      payload: response.data.detalles ? response.data.detalles : null,
    });
}
```

Fuente: Elaboración propia

En la figura 42 se muestra el consumo del endpoint para eliminar registro de salida vehículos.

Figura 43 Fragmento de código del consumo del endpoint para eliminar registro de salida vehículos

```
export function deleteExitVehicles(form) {
  const request = axios.get(
    `${process.env.REACT_APP_API_URL}/api/esalidavehiculos/${form.idsalidavehiculos}`,
    form
  );

  return (dispatch) =>
    request.then((response) => {
      if (parseInt(response.data.status) === 404) { ... }
      if (response.data.status === 200) { ... }
      } else { ... }
    }
    console.log(response.data.detalles);
    dispatch({
      type: CRUD_EXIT_VEHICLES,
      payload: response.data.detalles ? response.data.detalles : null,
    });
}
```

Fuente: Elaboración propia

Luego de construir las funciones necesarias para el consumo de los endpoints del api del backend, se construyó el contenedor reducer para redux como se muestra en la figura 44, el cual servirá para almacenar los datos de manera temporal para ser consultados desde cualquier componente, sin la necesidad de pasar los datos por los props.

Figura 44 Fragmento de código del reducer de la vista salida de vehículos

```
const initialState = {
  list_exit_vehicles: null,
  list_exit_report_vehicles: null,
  crud_exit_vehicles: null,
};
const exitVehicles = function (state = initialState, action) {
  switch (action.type) {
    case Actions.GET_LIST_EXIT_VEHICLES: {
      return {
        ...state,
        list_exit_vehicles: action.payload,
      };
    }
    case Actions.CRUD_EXIT_VEHICLES: {
      return {
        ...state,
        crud_exit_vehicles: action.payload,
      };
    }
  }
};
```

Fuente: Elaboración propia

Luego de construyó la vista en el componente en el cual se consultaron los registros con la función “getExitVehicles” de las salidas como se muestra en la figura 45.

Figura 45 Fragmento de código del consumo de los registros de salida vehículos

```
const ExitVehicles = (props) => {
  const [managementDriverModal, setManagementDriverModal] = useState(false);
  const [managementDriverSelect, setManagementDriverSelect] = useState(null);
  const [rows, setRows] = useState(null);
  useEffect(() => {
    props.getManagementDriver();
  }, []);
  useEffect(() => {
    props.getExitVehicles();
  }, [props.crud_exit_vehicles]);
  useEffect(() => {
    if (props.list_exit_vehicles) {
      setRows(
        props.list_exit_vehicles
          .filter((p) => p.idusers == props.userRole.id)
          .map((p) => {
            p.id = p.idsalidavehiculos;
            return p;
          })
      );
    }
  }, [props.list_exit_vehicles]);
```

Fuente: Elaboración propia

En la figura 46 se muestran las acciones que van a tener cada registro una vez listados en el table.

Figura 46 Fragmento de código de las acciones de los registros

```
headerName: "Acciones",
headerAlign: "center",
align: "center",
width: 150,
disableClickEventBuddling: true,
renderCell: ({ row }) => [
    //Editar Salida
    const onClickEdit = () => {
        setManagementDriverModal(true);
        setManagementDriverSelect(row);
    };

    //Eliminar Salida
    const onClickDelete = () => {
        props.deleteExitVehicles(row);
    };

    //imprimir salida
    const onClickPrint = () => {
        window.open(
            `${process.env.REACT_APP_API_URL}/api/salidavehiculos/${row.idsalidavehiculos}`,
            "_blank"
        );
    };
]
```

Fuente: Elaboración propia

Para realizar la creación y edición de un registro, se utiliza el mismo componente que contiene un modal, al ser una edición se cargaran los datos del registro para ser modificados, para crear un nuevo registro se abrirá el modal luego de pulsar el botón registrar, en la figura 47 se muestra la llamada al modal, para el editar se ejecuta la función “onClickEdit” que se muestra en la figura 46 se realiza el llamado al modal de edición.

Figura 47 Fragmento de código del botón de crear registro salida vehículo

```
<div className="card-body">
  <div className="row d-flex justify-content-between">
    <div className="col-1">
      <button
        onClick={() => {
          setManagementDriverSelect(null);
          setManagementDriverModal(true);
        }}
        className="btn btn-primary"
      >
        Registrar
      </button>
    </div>
  </div>
<div
  style={{ height: 420, width: "100%" }}>
```

Fuente: Elaboración propia

En la figura 48 se muestra fragmento del código del componente modal para la carga de datos de un registro para su edición o para su creación en la figura 49 se muestra el consumo de los endpoints del api a través de las funciones “saveExitVehicle” para registrar una salida y “updateExitVehicle” para la edición de un registro.

Figura 48 Fragmento de código del modal de registro y edición de registro salida vehículo

```
9   import * as Actions from "store/actions/app";
10  const DriverModal= (props)=>{
11    const initialsStateValues={
12      idsalidavehiculos: null,
13      idtransportista: null,
14      monto:null,
15      idusers:props.userId
16    }
17    const [values, setValues] = useState(initialsStateValues);
18    useEffect(()=>{
19      if(props.statusModal){
20        if(props.driverSelect){
21
22          setValues(props.driverSelect);
23        }else{
24          setValues(initialsStateValues);
25        }
26      }
```

Fuente: Elaboración propia

Figura 49 Fragmento de código para el consumo de los endpoints de registro y actualización de salida vehículos

```
        }
        return false;
    }
    function handleSubmit(e){
        e.preventDefault();
        if(validateForm())return;
        if(props.driverSelect){
            props.updateExitVehicles(values);
        }else{
            props.saveExitVehicles(values);
        }
        setValues(initialsStateValues);
        props.hideModal();
    }
    const handleInputChange=(e)=>{
```

Fuente: Elaboración propia

3.8.3. Resultados

Se logró el objetivo de construir la gestión de salida de vehículos, en la figura 50 se muestra el table con los registros y sus acciones “editar, imprimir y eliminar”, en la figura 51 y 52 se muestra el modal que sirve para la creación y edición de un registro de salida vehículo, en la figura 53 se muestra el pdf a imprimir generado de un registro.

Figura 50 Vista de gestión de salida vehículos

Mantenedor de salida de vehiculos						INICIO / GESTION DE SALIDA
Salida de vehiculos						
Registrar						
Acciones	Chofer	Vehiculo	Monto		Fecha	
  	ADOLFO VARGAS RUIZ	AZP-212	10		2023-03-09 09:16:13	

Fuente: Elaboración propia

Figura 51 Vista de modal de registro de salida vehículos

Registrar conductor

Transportista:

Seleccionar

Monto:

Guardar

Fuente: Elaboración propia

Figura 52 Vista modal para actualizar registros

Actualizar conductor

Transportista:

ADOLFO VARGAS RUIZ

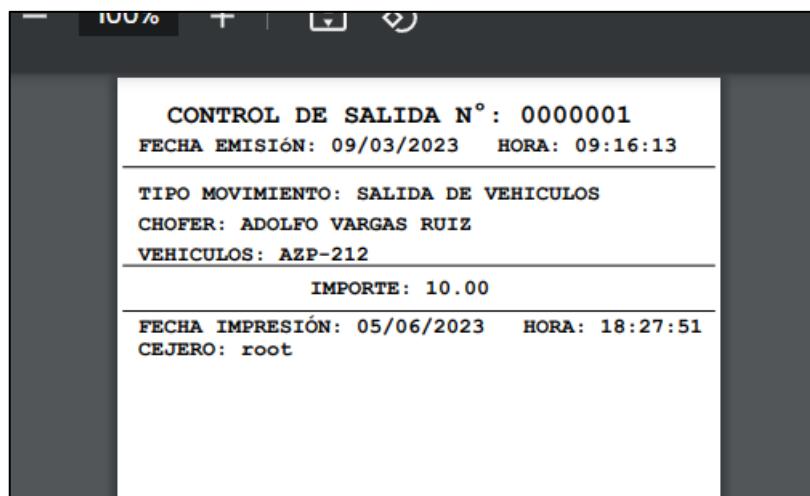
Monto:

10

Actualizar

Fuente: Elaboración propia

Figura 53 Vista de pdf generado para la impresión de registro salida vehículo



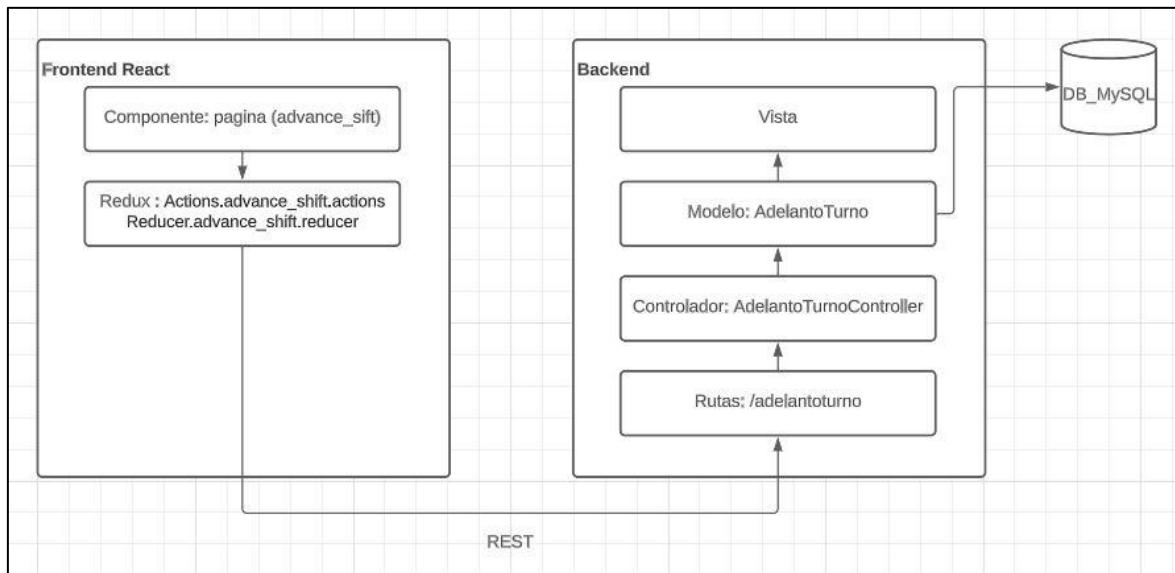
Fuente: Elaboración propia

3.9. Construir submódulo de adelantos (Trade)

El submódulo de salida será el encargado de gestionar los adelantos “salidas que no estaban programadas” de los conductores, donde se registrará el conductor y el monto a cobrar por encomiendas a oficina, estos registros serán visibles para los usuarios que se encuentren en la misma sucursal.

Para iniciar la construcción se tendrá en cuenta el diagrama de flujo de trabajo de la figura 54 donde se describen los archivos a interactuar en el frontend y backend.

Figura 54 Flujo de trabajo para adelanto de vehículos



Fuente: Elaboración propia

3.9.1. Tareas

En la figura 55 se muestra el objetivo a alcanzar y las tareas a realizar.

Figura 55 Objetivo: Crear el submódulo de adelanto y asignar al módulo de tesorería

O. Crear el submodule de adelanto
T1. Crear el crud de adelanto
crear: el correlativo será incremental por dia, el correlativo comienza en 1 cada dia, luego de registrar, mostrar el pdf de la salida para imprimir
editar: permitir actualizar idtransportista y monto
Listar: listar los adelantos de los vehiculos que tienen el estado activo "nombre del chofer, placa de vehiculo que conduce el monto y fecha"
Eliminar: Cambiar de estado a -1
T2. Crear un endpoint que permita imprimir la salida por el idadelantovehiculo
T3. Crear el submodule de salida y signarla al modulo tesoreria
crear un modal para crear, editar la salida, el modal se desplegará luego de pulsar el boton de registrar o en la accion de editar de la tabla
en la tabla se listaran los registros de salidas, adicional tendra las acciones de editar, eliminar y imprimir
Luego de cada registro o actualización recargar automaticamente los datos de la tabla

Fuente: Elaboración propia

3.9.1.1. Referencias

En la figura 56 se muestra una vista como referencia para los botones de registrar y la tabla donde se mostrarán los datos.

Figura 56 Vista de referencia

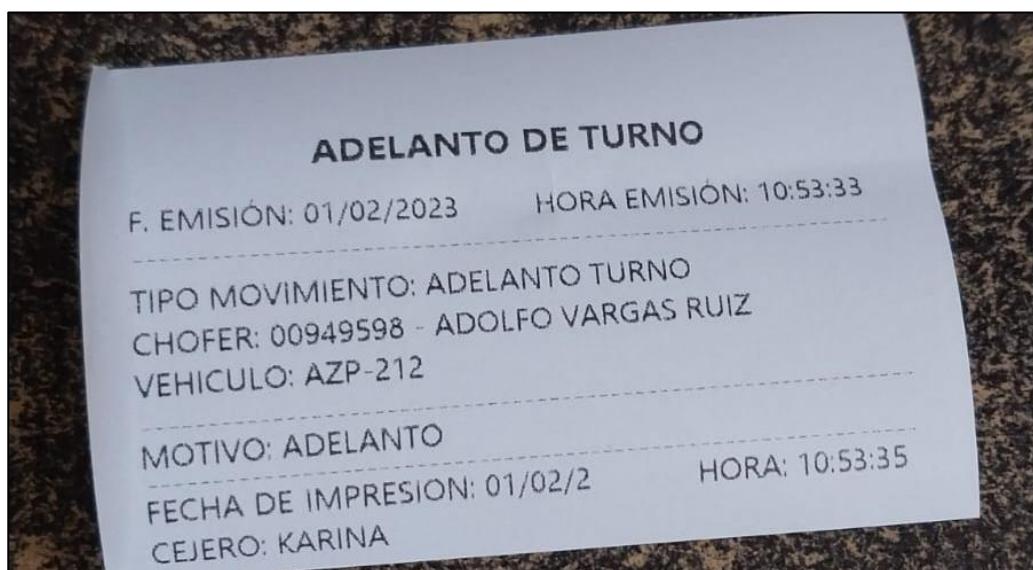
Acciones	Razón Social	RUC	Dirección	Marca / Placa
	ADOLFO VARGAS RUIZ	11111111	-	- / AZP-212
	ALAN RODRIGUEZ SAAVEDRA	11111111	-	- / M6G-837
	ALENCAR REATEGUI TELLO	11111111	-	- / BCI-569
	AMADEO BARADALES GONZAL...	11111111	-	- / AWR-258
	CARLOS ALBERTO LOPEZ SAN...	11111111	-	- / F5I-952
	DANIEL SANCHEZ HILDEBRANTO	11111111	-	- / S1C-051
	DIogenes CORAL PANDURO	11111111	-	- / M6Z-965
	EDGARDO RUIZ LOZANO	11111111	-	- / P2V-060

Copyright © 2020 [ACCORD Software](#). Todos los derechos reservados. Version 1.0.0

Fuente: Elaboración propia

En la figura 28 se muestra una impresión que servirá como referencia para imprimir un registro de adelanto.

Figura 57 Referencia de impresión de adelanto



Fuente: HMF inversiones

3.1.1. Ejecución de las tareas

Las tareas se dividieron en backend y frontend.

3.1.1.1. Backend

3.1.1.1.1. T1 Crear el crud de adelantos y T2 Crear un endpoint que permita imprimir la salida por el idadelantovehiculo

Se realizo la creación del controlador donde se crearán las funciones para el crud como se muestra en la figura 58.

Figura 58 Fragmento de Código del controlador de adelanto vehículos

```
use Illuminate\Http\Request;
use App\AdelantoTurno;
use Codedge\Fpdf\Fpdf\Fpdf;
use App\NumerosEnLetras;
use Illuminate\Support\Facades\Validator;
use Illuminate\Support\Facades\DB;

class AdelantoTurnoController extends Controller
{
    /**
     * Handle the incoming request.
     *
     * @param \Illuminate\Http\Request $request
     * @return \Illuminate\Http\Response
     */
}
```

Fuente: Elaboración propia

Luego de la creación del controlador se construyó las funciones para (listar, crear, editar, eliminar, imprimir) como se muestra en las figuras 59, 60, 61, 62 y 63.

Figura 59 Fragmento de Código para listar los registros de salida

```
'public function index(Request $request)
{
    $adelantos = DB::table('adelanto_turno as at')
        ->join('transportista as t','t.idtransportista','=','at.idtransportista')
        ->select('at.*','t.razonsocial','t.placa')
        ->where('at.estado','!=',-1)
        ->get();
    $json = array();
    if(!empty($adelantos)){
        $json = array(
            "status"=>200,
            "mensaje"=>"total_registros".count($adelantos),
            "detalles"=>$adelantos
        );
    }
}
```

Fuente: Elaboración propia

En la figura 60 se muestra el fragmento de código para registrar adelanto de vehículo.

Figura 60 Fragmento de código para registrar adelanto de vehículo

```
'public function set_adelanto_turno_insert(Request $request){
    $json = array();
    $validator = Validator::make($request->all(), [
        'idtransportista' => 'required',
        'motivo' => 'required',
        'idusers' => 'required'
    ]);
    if ($validator->fails()) {
        return response()->json($validator->errors()->toJson(), 400);
    }
    $data=array(
        'idtransportista'=>$request->input('idtransportista'),
        'motivo'=>$request->input('motivo'),
        'fecha'=>date("Y-m-d H:i:s"),
        'idusers'=>$request->input('idusers')
    );
    $adelantoTemp=DB::table('adelanto_turno')
        ->whereDate('fecha',date("Y-m-d"))
        ->orderBy('idadelantoturno', 'desc')
        ->first();
    if($adelantoTemp){
        $data["correlativo"] = $adelantoTemp->correlativo+1;
    }else{
        $data["correlativo"] = 1;
    }
    $idelantoturno = DB::table('adelanto_turno')->insertGetId($data);
    $data['idadelantoturno']=$idelantoturno;
}
```

Fuente: Elaboración propia

En la figura 61 se muestra el fragmento de código para actualizar registro de adelanto vehículo.

Figura 61 Fragmento de Código para actualizar registro de adelanto vehículo

```
public function set_adelanto_turno_update(Request $request){
    $id = $request->input("idadelantoturno");
    $json = array();
    //Recoger datos
    $datos = array(
        'idtransportista'=>$request->input("idtransportista"),
        'motivo'=>$request->input('motivo'),
    );
    if(!empty($datos)){
        $validar = DB::table('adelanto_turno')
            ->where('idadelantoturno', '=', $id)
            ->get();

        if (!$validar->count() == 0){
            $menu = AdelantoTurno::where("idadelantoturno", $id)->update($datos);
            $json = array(
                "status"=>200,
                "mensaje"=>"Registro exitoso, ha sido actualizado",
                "detalles"=>$validar[0]
            );
        }
    }
}
```

Fuente: Elaboración propia

En la figura 62 se muestra el fragmento de código para eliminar registro de adelanto vehículo.

Figura 62 Fragmento de código para eliminar registro de adelanto vehículo

```
public function set_adelanto_turno_delete($idadelantoturno){
    $json = array();
    $validar = DB::table('adelanto_turno')
        ->where('idadelantoturno', '=', $idadelantoturno)
        ->get();

    if (!$validar->count() == 0){
        AdelantoTurno::where("idadelantoturno", $idadelantoturno)->update(['estado'=>-1]);
        $json = array(
            "status"=>200,
            "mensaje"=>"Registro anulado",
            "detalles"=>$validar[0]
        );
    }else{
        $json = array(
            "status"=>400,
            "mensaje"=>"No se ha encontrado el registro"
        );
    }
}
```

Fuente: Elaboración propia

En la figura 63 se muestra el fragmento de código que permitirá generar el pdf para la impresión del registro de adelanto.

Figura 63 Fragmento de código de la impresión de registro de adelanto vehículo

```
public function get_adelanto_turno_print($id){
    $adelanto = DB::table('adelanto_turno as at')
        ->join('transportista as t','t.idtransportista','=','at.idtransportista')
        ->join('users as u','u.id','=',$at.idusers')
        ->select('at.*','t.razonsocial','t.placa','u.name')
        ->where('at.idadelantoturno','=',$id)
        ->first();
    $fpdf= new Fpdf('P','mm',array(80,350));
    $fpdf->AddPage();
    $fpdf->SetFont('Courier', 'B', 10);
    $textypost=5;
    $fpdf->setXY(5,$textypost);
    $fpdf->Cell(0,0,utf8_decode("ADELANTO DE TURNO N°: ".str_pad($adelanto->correlativo, 7, "0", STR_PAD_LEFT)),0
    $textypost+=4;
    $fpdf->setXY(2,$textypost);
    $fpdf->SetFont('Courier', 'B', 8);
    $fpdf->Cell(0, 0, utf8_decode("FECHA EMISIÓN: ") . date('d/m/Y', strtotime(explode(' ', $adelanto->fecha)[0])));
    $textypost+=3;

    $fpdf->Line(1,$textypost,79,$textypost);
    $textypost += 2;
    $fpdf->setXY(2, $textypost);
    $fpdf->Multicell(0, 3, utf8_decode("TIPO MOVIMIENTO: ADELANTO TURNO"));
    $textypost += 4;
    $fpdf->Cell(0, 0, utf8_decode("ADELANTO TURNO"));
```

Fuente: Elaboración propia

Luego se realizó los endpoints necesarios para llamar a cada función del controlador y la impresión del registro, como se muestra en la figura 36.

Figura 64 Fragmento de código de los endpoints

```
//ADELANTO TURNO
Route::resource('/adelantoturno', 'AdelantoTurnoController'); //listar
Route::post('/radelantoturno', 'AdelantoTurnoController@set_adelanto_turno_insert'); //crear
Route::post('/aadelantoturno', 'AdelantoTurnoController@set_adelanto_turno_update'); //actualizar
Route::get('/adelantoturno/{id}', 'AdelantoTurnoController@set_adelanto_turno_delete'); //eliminar
Route::get('/adelantoturno/{id}', 'AdelantoTurnoController@get_adelanto_turno_print'); //imprimir
```

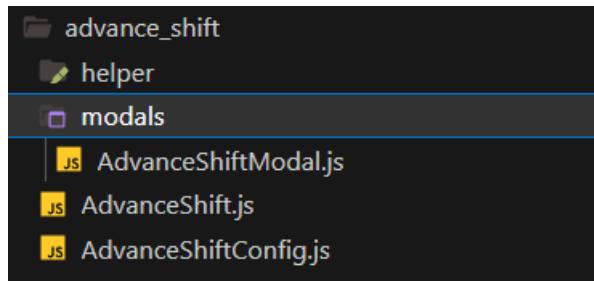
Fuente: Elaboración propia

3.1.1.2. Frontend

3.1.1.2.1. T3. Crear el submódulo de adelanto

Se creo la carpeta de “advance_shift” en el directorio de carpetas como se muestra en la figura 65.

Figura 65 Distribución de archivos de la vista adelanto vehículo



Fuente: Elaboración propia

Se inicializó el componente de “AdvanceShift” el cual se mostrará en la vista como de muestra en la figura 66, se el archivo “AdvanceShiftConfig” se establece los permisos para la ruta y su url como se muestra en la figura 67.

Figura 66 Fragmento de código del componente AdvanceShift

```
7 import * as Actions from "store/actions/app";
8 import withDragDropContext from "./helper/withDnDContext";
9 import Loader from "../helpers/loader/Loader";
10 import AdvanceShiftModal from "./modals/AdvanceshiftModal";
11
12 const AdvanceShift = (props) => {
13   const [managementDriverModal, setManagementDriverModal] = useState(false);
14   const [managementDriverSelect, setManagementDriverSelect] = useState(null);
15   const [rows, setRows] = useState(null);
16   useEffect(() => {
```

Fuente: Elaboración propia

Figura 67 Fragmento de código de AdvanceShiftConfig para la configuración del acceso y la ruta

```
2 import { AuthRoles } from "components/auth";
3
4 export const AdvanceShiftConfig = {
5   auth: AuthRoles.admin,
6   routes: [
7     {
8       path: `${process.env.PUBLIC_URL}/advance_shift`,
9       component: React.lazy(() => import('./AdvanceShift'))
10    }
11  ],
12};
```

Fuente: Elaboración propia

Luego se construyó los actions y reducer necesarios para redux para el consumo y almacenamiento de los datos obtenidos desde el api.

En la figura 68 se muestra el consumo de los registros de los adelantos mediante el endpoint construido en el api del backend.

Figura 68 Fragmento de Código para listar los registros de los adelantos

```
export function getAdvanceShift() {
  const request = axios.get(
    `${process.env.REACT_APP_API_URL}/api/adelantoturno`
  );

  return (dispatch) =>
    request.then((response) => {
      if (parseInt(response.data.status) === 404) { ... }
      return dispatch({
        type: GET_LIST_ADVANCE_SHIFT,
        payload: response.data.detalles === null ? [] : response.data.detalles,
      });
    });
}
```

Fuente: Elaboración propia

En la figura 69 se muestra el consumo del endpoint para registrar un adelanto, también se muestra que cuando el backend responde al frontend con un código 200, el frontend realiza automáticamente el consumo del endpoint para la impresión del registro.

Figura 69 Fragmento de Código del consumo del endpoint de registro de adelanto e impresión

```
export function saveAdvanceShift(form) {
  const request = axios.post(
    `${process.env.REACT_APP_API_URL}/api/adelantoturno`,
    form
  );

  return (dispatch) =>
    request.then((response) => {
      if (parseInt(response.data.status) === 404) { ... }
      if (response.data.status === 200) [
        toast.success(response.data.mensaje),
        window.open(
          `${process.env.REACT_APP_API_URL}/api/adelantoturno/${response.data.detalles.idadelantoturno}`,
          "_blank"
        );
      ] else {
        toast.error(response.data.mensaje);
      }

      dispatch({
        type: CRUD_ADVANCE_SHIFT,
        payload: response.data.detalles ? response.data.detalles : null,
      });
    });
}
```

Fuente: Elaboración propia

En la figura 70 se muestra el consumo del endpoint para actualizar un registro de adelanto vehículos.

Figura 70 Fragmento de Código para el consumo del endpoint para actualizar registro de adelanto vehículo

```
export function updateAdvanceShift(form) {
  const request = axios.post(
    `${process.env.REACT_APP_API_URL}/api/aadelantoturno`,
    form
  );

  return (dispatch) =>
    request.then((response) => {
      if (parseInt(response.data.status) === 404) { ...
      }
      if (response.data.status === 200) {
        toast.success(response.data.mensaje);
      } else {
        toast.error(response.data.mensaje);
      }
      /* getMenuControl(); */
      return dispatch({
        type: CRUD_ADVANCE_SHIFT,
        payload: response.data.detalles ? response.data.detalles : n
      });
    });
}
```

Fuente: Elaboración propia

En la figura 42 se muestra el consumo del endpoint para eliminar registro de adelanto vehículos.

Figura 71 Fragmento de código del consumo del endpoint para eliminar registro de adelanto vehículos

```
export function deleteAdvanceShift(form) {
  const request = axios.get(
    `${process.env.REACT_APP_API_URL}/api/eadelantoturno/${form.idadelantoturno}`,
    form
  );

  return (dispatch) =>
    request.then((response) => [
      if (parseInt(response.data.status) === 404) { ... }
    ])
      if (response.data.status === 200) {
        toast.success(response.data.mensaje);
      } else {
        toast.error(response.data.mensaje);
      }
      console.log(response.data.detalles);
      /* getMenuControl(); */
      dispatch({
        type: CRUD_ADVANCE_SHIFT,
        payload: response.data.detalles ? response.data.detalles : null,
      });
}
```

Fuente: Elaboración propia

Luego de construir las funciones necesarias para el consumo de los endpoints del api del backend, se construyó el contenedor reducer para redux como se muestra en la figura 72, el cual servirá para almacenar los datos de manera temporal para ser consultados desde cualquier componente, sin la necesidad de pasar los datos por los props.

Figura 72 Fragmento de código del reducer de la vista adelanto de vehículos

```
const initialState = {
  list_advance_shift:null,
  crud_advance_shift:null,
};
const advanceShift = function (state = initialState, action) {
  switch (action.type) {
    case Actions.GET_LIST_ADVANCE_SHIFT: {
      return {
        ...state,
        list_advance_shift: action.payload
      };
    }
    case Actions.CRUD_ADVANCE_SHIFT:{ 
      return{
        ...state,
        crud_advance_shift: action.payload
      };
    }
    default: {
      return state;
    }
  }
}
```

Fuente: Elaboración propia

Luego se construyó la vista en el componente en el cual se consultaron los registros con la función “getAdvanceShift” de los adelantos como se muestra en la figura 73.

Figura 73 Fragmento de código del consumo de los registros de adelanto vehículos

```
const AdvanceShift = (props) => {
  const [managementDriverModal, setManagementDriverModal] = useState(false);
  const [managementDriverSelect, setManagementDriverSelect] = useState(null);
  const [rows, setRows] = useState(null);
  useEffect(() => {
    props.getManagementDriver();
  }, []);
  useEffect(() => {
    props.getAdvanceShift();
  }, [props.crud_advance_shift]);
  useEffect(() => {
    if (props.list_advance_shift) {
      setRows(
        props.list_advance_shift
          .filter((p) => p.idusers == props.userRole.id)
          .map((p) => {
            p.id = p.idadelantoturno;
            return p;
          })
      );
    }
  }, [props.list_advance_shift]);
};
```

Fuente: Elaboración propia

En la figura 74 se muestran las acciones que van a tener cada registro una vez listados en el table.

Figura 74 Fragmento de código de las acciones de los registros

```
const columns = [
  {
    field: "",
    headerName: "Acciones",
    headerAlign: "center",
    align: "center",
    width: 150,
    disableClickEventBubbling: true,
    renderCell: ({ row }) => {
      //Editar Order y abrir modal
      const onClickEdit = () => {
        setManagementDriverModal(true);
        setManagementDriverSelect(row);
      };

      //imprimir orden
      const onClickPrint = () => {
        window.open(
          `${process.env.REACT_APP_API_URL}/api/adelantoturno/${row.idadelantoturno}`,
          "_blank"
        );
      };

      //Eliminar orden
      const onClickDelete = () => {
        props.deleteAdvanceShift(row);
      };
    }
  }
];
```

Fuente: Elaboración propia

Para realizar la creación y edición de un registro, se utiliza el mismo componente que contiene un modal, al ser una edición se cargaran los datos del registro para ser modificados, para crear un nuevo registro abrirá el modal luego de pulsar el botón, registrar en la figura 75 se muestra la llamada al modal, para el editar se ejecuta la función “onClickEdit” que se muestra en la figura 76 se realiza el llamado al modal de edición.

Figura 75 Fragmento de código del botón de crear registro adelanto vehículo

```
<div className="card-body">
  <div className="row d-flex justify-content-between">
    <div className="col-1">
      <button
        onClick={() => {
          setManagementDriverSelect(null);
          setManagementDriverModal(true);
        }}
        className="btn btn-primary"
      >
        Registrar
      </button>
    </div>
  </div>
```

Fuente: Elaboración propia

En la figura 76 se muestra fragmento del código del componente modal para la carga de datos de un registro para su edición o para su creación en la figura 77 se muestra el consumo de los endpoints del api a través de las funciones “saveAdvanceShift” para registrar una salida y “updateAdvanceShift” para la edición de un registro.

Figura 76 Fragmento de código del modal de registro y edición de registro adelanto vehículo

```
import { useActions } from "store/actions/app";
const AdvanceShiftModal = (props) => {
  const initialsStateValues = {
    idadelantoturno: null,
    idtransportista: null,
    motivo: null,
    idusers: props.userId
  };
  const [values, setValues] = useState(initialsStateValues);
  useEffect(() => {
    if (props.statusModal) {
      if (props.driverSelect) {
        setValues(props.driverSelect);
      } else {
        setValues(initialsStateValues);
      }
    }
  });
}
```

Fuente: Elaboración propia

Figura 77 Fragmento de código para el consumo de los endpoints de registro y actualización de adelanto vehículos

```
function handleSubmit(e){  
    e.preventDefault();  
    if(validateForm())return;  
    if(props.driverSelect){  
        props.updateAdvanceShift(values);  
    }else{  
        props.saveAdvanceShift(values);  
    }  
    setValues(initialsStateValues);  
    props.hideModal();  
}  
const handleInputChange = (e) => {
```

Fuente: Elaboración propia

3.1.1. Resultados

Se logró el objetivo de construir la gestión de adelanto de vehículos, en la figura 78 se muestra el table con los registros y sus acciones “editar, imprimir y eliminar”, en la figura 79 y 80 se muestra el modal que sirve para la creación y edición de un registro de adelanto vehículo, en la figura 81 de muestra el pdf a imprimir generado de un registro.

Figura 78 Vista de gestión de adelanto vehículos

Mantenedor de adelanto de turno						INICIO / GESTIÓN DE ADELANTO DE TURNO
Adelanto de turno						
Registrar						Reporte
Acciones	Chofer	Vehículo	Motivo		Fecha	

Fuente: Elaboración propia

Figura 79 Vista de modal de registro de adelanto vehículos

Registrar conductor

Transportista:

Seleccionar

Motivo:

Guardar

Fuente: Elaboración propia

Figura 80 Vista modal para actualizar registros

Actualizar conductor

Transportista:

FRANZ VILLANUEVA PINEDO

Motivo:

200

Guardar

Fuente: Elaboración propia

Figura 81 Vista de pdf generado para la impresión de registro adelanto vehículo



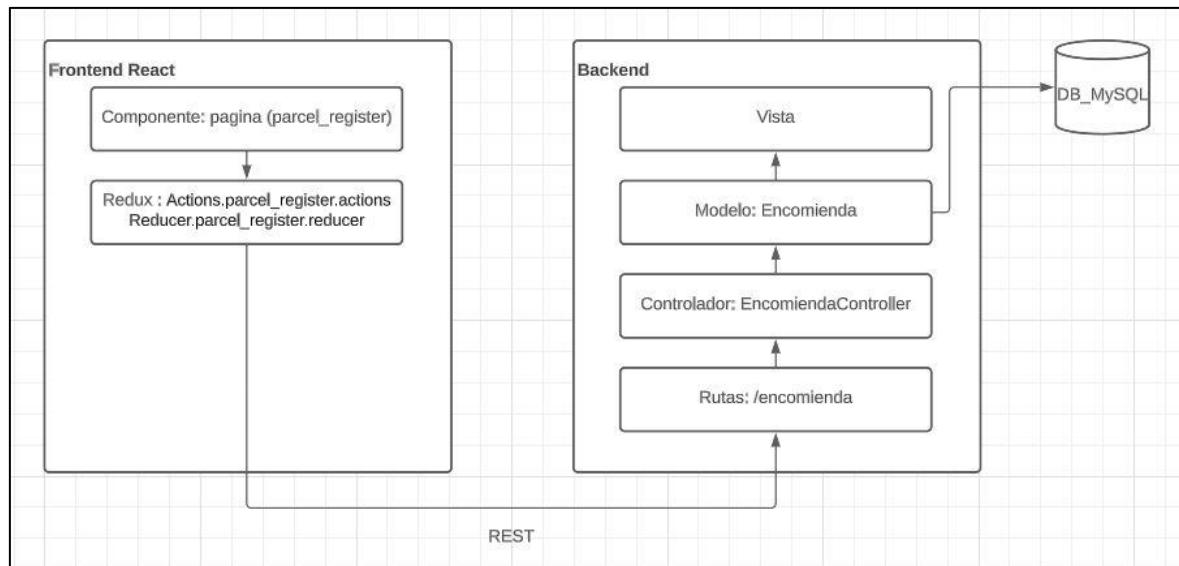
Fuente: Elaboración propia

3.2. Construir submódulo de encomiendas (Trade)

El submódulo de encomiendas será la encargada de gestionar las encomiendas por usuario y sucursal, donde se registrará la sucursal destino de la encomienda, conductor, remitente, destinatario, clave de encomienda, dirección, condición de entrega, observación y los ítems enviados con su descripción, unidad de medida, cantidad y precio por el servicio de encomienda.

Para iniciar la construcción se tendrá en cuenta el diagrama de flujo de trabajo de la figura 82 donde se describen los archivos a interactuar en el frontend y backend.

Figura 82 Flujo de trabajo de encomienda



Fuente: Elaboración propia

3.2.1. Tareas

En la figura 83 se muestra el objetivo a alcanzar y las tareas a realizar.

Figura 83 Objetivo: Crear submódulo de encomiendas

O. Crear el submódulo de encomiendas y asignarla al modulo de procesos			
T1. Crear el endpoints de encomienda			
crear:validar que la caja fue aperturada, luego de registrar, mostrar el pdf de la salida para imprimir			
editar: permitir actualizar idtransportista y monto			
Listar: listar los adelantos de los vehiculos que tienen el estado activo "Acciones, Comprobante, Remitente, DNI/RUC, Reg. Por, Transito, Fecha, Estado de encomienda"			
Actualizar estado: permitir actualizar el estado de una encomienda			
Eliminar: Cambiar de estado a -1			
Imprimir: Para imprimir la encomienda por el idencomienda			
Facturar: Permitira facturar una encomienda			
T2. Crear el submódulo de encomienda, crear dos tabs			
Tabs de Registro de encomienda			
se gestionara las encomiendas registrada por el ususario			
tendra las acciones de editar, facturar, eliminar y imprimir			
solo el usuario que registro la encomienda podra, facturar			
y eliminar			
La encomiendas tendran los siguientes estados (Origen, transito, destino, entregado)			
Editar: solo el usuario que registro la encomienda podar editarla el			
resto solo podra ver los datos de la encomienda			
Luego de cada registro o actualización recargar automaticamente los			
datos de la tabla			
las encomiendas facturadas solo tendran opcion de imprimir emcomienda			
crear un matal para crear, editar la encomeinda			
Tabs de Encomiendas por llegar			
se gestionara las encomiendas que tienen como destino la sucursal del usuario			
permitir facturar la encomienda si el tipo de envio es contra entrega			
las encomiendas facturadas solo tendran opcion de imprimir emcomienda			

Fuente: Elaboración propia

3.2.1.1. Referencia

En la figura 84 se muestra una referencia de la vista de gestión de encomiendas a construir.

Figura 84 Referencia para la contrucción de gestión de encomienda

The screenshot shows a search interface for managing packages. At the top left is a blue 'Agregar' button. To its right is a dropdown menu with 'Búsqueda' crossed out. On the far right is a search bar labeled 'Ingresar criterio de búsqueda'. Below the search bar is a table header with columns: 'Acciones', 'DNI/RUC', 'Nombre/Razón Social', 'Alias', 'Dirección', 'Teléfono', and 'Estado'. The 'Nombre/Razón Social' column has a note: 'COMPROBANTE, NOMBRE REMITENTE, D NI/RUC, USUARIO QUE LO REGISTRO, TRANSITO, FECHA DE REGISTRO, ESTADO'. The main area below the table header contains the text 'Sin filas'.

Fuente: Elaboración propia

En la figura 85 se muestra una referencia del modal para el registro y la edición de un registro de encomienda.

Figura 85 Referencia para modal de registro de encomienda

The screenshot shows a modal dialog titled 'REGISTRO DE CARGO - ENCOMIEDA'. The dialog has two tabs: 'Registro' (selected) and 'Listado'. At the top right are window control buttons. Below the tabs are input fields for 'Nº': CE01 and '0000006'. The main form contains several groups of fields: 'Destino' (with a dropdown arrow), 'Conductor' (text input), 'Placa' (text input), 'Remitente' (text input), 'Consignado a' (text input), 'Dirección' (text input), 'Descripción' (text input), 'Unidad' (dropdown: UNIDAD /BIEN/), 'Cantidad' (text input: 0.00), and 'Precio' (text input: 0.00). Below these are tables for 'Cantidad' and 'Descripción' (U.M., Precio U., Sub Total). At the bottom are dropdowns for 'Condición' and 'Entrega', and a text input for 'Observación'. A green 'Grabar' button is located at the bottom right.

Fuente: Elaboración propia

3.2.2. Ejecución de las tareas

Las tareas se dividieron en backend y frontend.

3.2.2.1. Backend

3.2.2.1.1. T1. Crear el endpoints de encomienda

Se realizo la creación del controlador donde se crearán las funciones para los endpoints como se muestra en la figura 86.

Figura 86 Fragmento de código del controlador de encomiendas

```
DiscrepanciaController.php  
EmpleadosController.php  
EncomiendaController.php  
EstablecimientoController.php  
EstablecimientoSucursalController.php  
FacturacionController.php  
  
11  use Illuminate\Support\Facades\Validator;  
12  use Illuminate\Support\Facades\DB;  
13  
14  class EncomiendaController extends Controller  
15  {  
16      public function index(Request $request)  
17  }
```

Fuente: Elaboración propia

Luego de la creación del controlador se construyó las funciones para (listar, crear, editar, eliminar, imprimir) como se muestra en las figuras 87, 89 y 90.

En el fragmento de código de la figura 87 se muestra la función para listar los registros de encomiendas en la tabla del frontend.

Figura 87 Fragmento de código de la función de listar encomiendas

```
public function index(Request $request)  
{  
    $encomiendas=DB::table('encomienda as e')  
    ->select('e.*',DB::raw('sc.serie_comprobante,pr.razon_social_nombre as remitente_n  
    ->join('serie_comprobante as sc','sc.serie_comprobante_id','=','e.serie_comprobant  
    ->join('persona as pr','pr.persona_id','=','e.idremitente')  
    ->join('persona as pc','pc.persona_id','=','e.idconsignado')  
    ->join('transportista as t','t.idtransportista','=','e.idtransportista')  
    ->join('establecimiento_sucursal as esi','esi.establecimiento_sucursal_id','=','e.  
    ->join('establecimiento_sucursal as esf','esf.establecimiento_sucursal_id','=','e.  
    ->join('users as u','u.id','=','e.idusuario')  
    ->orderBy('e.id_encomienda', 'desc')  
    ->where('e.estado','!=',-1)  
    ->get();  
    $json = array();  
    if(!empty($encomiendas))  
    {  
        $json = array(  
            "status"=>200,  
            "mensaje"=>"total_registros".count($encomiendas),  
            "detalles"=>$encomiendas  
    }  
}
```

Fuente: Elaboración propia

En la figura 88 se muestra el fragmento de código para listar los datos de la una encomienda y su detalle, necesarios para el modal de edición.

Figura 88 Fragmento de código para listar los ítems de una encomienda

```
public function get_encomienda_detalle_lista($idencomienda){
    $json = array();
    $encomienda=Encomienda::where("id_encomienda", $idencomienda)->first();
    $detalle=EncomiendaDetalle::where("id_encomienda", $idencomienda)->get();

    if(!empty($encomienda)){
        $json = array(
            "status"=>200,
            "mensaje"=>"total_registros".count($detalle),
            "detalles"=> array(
                "encomienda" => $encomienda,
                "detalle" => $detalle
            )
        );
    }else{
        $json = array(
            "status"=>400,
            "mensaje"=>"No hay ningún curso registrado",
        );
    }
}
```

Fuente: Elaboración propia

En la figura 89 se muestra el fragmento de código para registrar la encomienda y su detalle.

Figura 89 Fragmento de código de registro de encomienda

```
public function set_encomienda_insert(Request $request){
    $encomienda = $request->input('encomienda');
    $detalle = $request->input('detalle');
    $idserie = $encomienda[ 'serie_comprobante_id' ];
    $correlativo = $encomienda[ 'correlativo' ];

    $encomienda[ "fecharegistro" ]=date("Y-m-d H:i:s");
    $idencomienda = DB::table('encomienda')->insertGetId($encomienda);
    //ACTUALIZAMOS LA SERIE DEL COMPROBANTE
    $affected = DB::table( 'serie_comprobante' )
        ->where( 'serie_comprobante_id' , $idserie )
        ->update([ 'correlativo' => $correlativo ]);
    foreach($detalle as $value){
        $value[ 'id_encomienda' ]=$idencomienda;
        $valueT=array(
            "id_encomienda"=>$idencomienda,
            "idunidadmedida"=>$value[ 'idunidadmedida' ],
            "descripcion"=>$value[ 'descripcion' ],
            "cantidad"=>$value[ 'cantidad' ],
            "precio"=>$value[ 'precio' ],
            "subtotal"=>$value[ 'subtotal' ]
        );
        DB::table('encomienda_detalle')->insert($valueT);
    }
}
```

Fuente: Elaboración propia

En la figura 90 se muestra el fragmento de código para actualizar un registro de encomienda y su detalle.

Figura 90 Fragmento de código de actualizar encomienda

```
public function set_encomienda_update(Request $request){
    $encomienda = $request->input('encomienda');
    $idencomienda=$encomienda['id_encomienda'];
    $detalle = $request->input('detalle');
    if(!empty($encomienda)){
        $validar = DB::table('encomienda')
            ->where('id_encomienda', '=', $idencomienda)
            ->get();
        if (!$validar->count()==0){
            Encomienda::where("id_encomienda", $idencomienda)->update($encomienda);
            EncomiendaDetalle::where("id_encomienda", $idencomienda)->delete();
            foreach($detalle as $value){
                $valueT=array(
                    "id_encomienda"=>$idencomienda,
                    "idunidadmedida"=>$value['idunidadmedida'],
                    "descripcion"=>$value['descripcion'],
                    "cantidad"=>$value['cantidad'],
                    "precio"=>$value['precio'],
                    "subtotal"=>$value['subtotal']
                );
                DB::table('encomienda_detalle')->insert($valueT);
            }
        }
    }
}
```

Fuente: Elaboración propia

En la figura 91 se muestra el fragmento de código para realizar la facturación directamente desde las acciones de la tabla donde se listan las encomiendas.

Figura 91 Fragmento de código de la facturación de encomienda

```
public function set_facturacion_insert(Request $request){
    $facturacion = $request->input('facturacion');
    $detalle = $request->input('detalle');
    $idserie = $facturacion['serie_comprobante_id'];
    $correlativo = $facturacion['correlativo'];
    $id_encomienda= $facturacion['id_encomienda'];
    $json = array();

    $idfacturacion = DB::table('facturacion')->insertGetId([
        'fecha_emision'=>$facturacion['fecha_emision'],
        'persona_id'=>$facturacion['persona_id'],
        'tipo_moneda_id'=>$facturacion['tipo_moneda_id'],
        'tipo_operacion_id'=>$facturacion['tipo_operacion_id'],
        'tipo_pago_id'=>$facturacion['tipo_pago_id'],
        'tipo_forma_pago_id'=>$facturacion['tipo_forma_pago_id'],
        'serie_comprobante_id'=>$facturacion['serie_comprobante_id'],
        'numero_comprobante'=>$facturacion['numero_comprobante'],
        'valor_venta'=>$facturacion['valor_venta'],
        'descuento_global'=>$facturacion['descuento_global'],
        'tasa_cambio'=>$facturacion['tasa_cambio'],
        'gravadas'=>$facturacion['gravadas'],
        'exoneradas'=>$facturacion['exoneradas'],
        'inafectas'=>$facturacion['inafectas'],
        'gratuitas'=>$facturacion['gratuitas'],
        'totaligv'=>$facturacion['totaligv'],
        'totalisc'=>$facturacion['totalisc'],
```

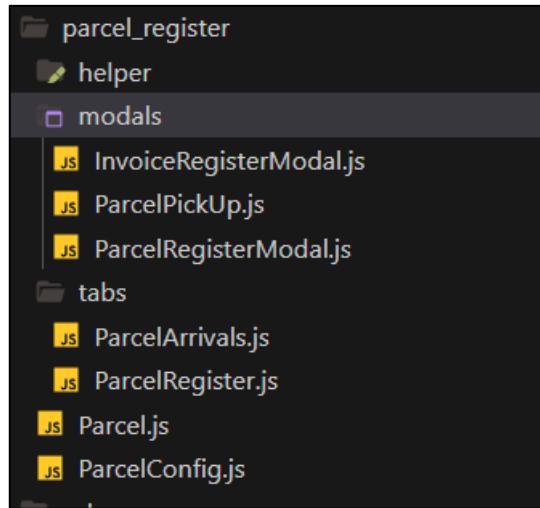
Fuente: Elaboración propia

3.2.2.2. Frontend

3.2.2.2.1. T2. Crear el submódulo de encomienda

En la figura 92 se muestra la distribución de carpetas del submódulo de encomiendas.

Figura 92 Distribución de carpetas del submódulo de encomiendas



Fuente: Elaboración propia

En la figura 93 se muestra el fragmento de código de la vista de gestión de encomiendas, donde habrá dos tabs para la gestión de las encomiendas, por defecto se mostrará el tab de encomiendas registradas.

Figura 93 Componente de encomiendas con los tabs de registro de encomiendas y gestión de encomiendas por llegar

```
import ParcelRegister from "./tabs/ParcelRegister";
import ParcelArrivals from "./tabs/ParcelArrivals";
const Parcel = (props) => {
  const [key, setKey] = useState("ParcelRegister");

  return (
    <React.Fragment>
      <Content
        title="Procesos"
        section="Encomiendas"
        content={
          <div className="row">
            <div className="col-12">
              <div className="card">
                <Tabs activeKey={key} onSelect={(k) => setKey(k)}>
                  <Tab eventKey="ParcelRegister" title="Registro de encomiendas">
                    {key === "ParcelRegister" ? (
                      <ParcelRegister tabKey="ParcelRegister" />
                    ) : (
                      <></>
                    )}
                  </Tab>
                  <Tab eventKey="ParcelArrivals" title="Encomiendas por llegar">
                    {key === "ParcelArrivals" ? (
                      <ParcelArrivals tabKey="ParcelArrivals" />
                    ) : (
                      <></>
                    )}
                  </Tab>
                </Tabs>
              </div>
            </div>
          </div>
        }
      </Content>
    </React.Fragment>
  );
}
```

Fuente: Elaboración propia

En la figura 94 se muestra el archivo “ParcelConfig” se establece los permisos para la ruta y su url.

Figura 94 Se establece los permisos para la ruta

```
import { AuthRoles } from "components/auth";

export const ParcelConfig = {
  auth: AuthRoles.admin,
  routes: [
    {
      path: `${process.env.PUBLIC_URL}/parcel_register`,
      component: React.lazy(() => import("./Parcel")),
    },
  ],
};
```

Fuente: Elaboración propia

Luego se construyó los actions y reducer necesarios para redux para el consumo y almacenamiento de los datos obtenidos desde el api.

En la figura 95 se muestra el consumo de los registros de las encomiendas mediante el endpoint construido en el api del backend.

Figura 95 Consumo de registros del endpoint de encomiendas

```
export const PARCEL_SEARCH_LOADER = "[PARCEL REGISTER] PARCEL SEARCH LOADER";
export function getParcelRegister() {
  const request = axios.get(`${process.env.REACT_APP_API_URL}/api/encomienda`);

  return (dispatch) =>
    request.then((response) => {
      if (parseInt(response.data.status) === 404) { ... }
      return dispatch({
        type: GET_LIST_PARCEL_REGISTER,
        payload: response.data.detalles === null ? [] : response.data.detalles,
      });
    });
}
```

Fuente: Elaboración propia

En la figura 96 se muestra el consumo del endpoint de detalle de encomienda, el cual se consumirá para cargar los datos al editar un registro.

Figura 96 Consumo de registros del endpoint de detalle de encomienda

```
// export function getParcelRegisterDetail(form) {
//   const request = axios.get(
//     `${process.env.REACT_APP_API_URL}/api/gencomienda/${form.id_encomienda}`
//   );

//   return (dispatch) =>
//     request.then((response) => {
//       if (parseInt(response.data.status) === 404) { ... }
//     })
//     return dispatch({
//       type: GET_LIST_PARCEL_REGISTER_DETAIL,
//       payload: response.data.detalles === null ? [] : response.data.detalles,
//     });
// }
```

Fuente: Elaboración propia

En la figura 97 se muestra el consumo del endpoint para registrar encomienda.

Figura 97 Consumo del endpoint para registrar encomienda

```
export function saveParcelRegister(form) {
  const request = axios.post(
    `${process.env.REACT_APP_API_URL}/api/rencomienda`,
    form
  );

  return (dispatch) =>
    request.then((response) => {
      if (parseInt(response.data.status) === 404) { ... }
      if (response.data.status === 200) {
        toast.success(response.data.mensaje);
        window.open(
          `${process.env.REACT_APP_API_URL}/api/encomienda/${response.data.detalles.id_encomienda}`,
          "_blank"
        );
      } else {
        toast.error(response.data.mensaje);
      }

      dispatch({
        type: CRUD_PARCEL_REGISTER,
        payload: response.data.detalles
      });
    });
}
```

Fuente: Elaboración propia

En la figura 98 se muestra el consumo de endpoint para actualizar registro de encomienda.

Figura 98 Consumo de endpoint para actualizar registro de encomienda

```
export function updateParcelRegister(form) {
  const request = axios.post(
    `${process.env.REACT_APP_API_URL}/api/aencomienda`,
    form
  );
  return (dispatch) =>
    request.then((response) => {
      if (parseInt(response.data.status) === 404) { ... }
      if (response.data.status === 200) {
        toast.success(response.data.mensaje);
      } else {
        toast.error(response.data.mensaje);
      }
      /* getMenuControl(); */
      return dispatch({
        type: CRUD_PARCEL_REGISTER,
        payload: response.data.detalles ? response.data.detalles : null,
      });
    });
}
```

Fuente: Elaboración propia

En la figura 99 se muestra el consumo de endpoint para actualizar el estado de envío de una encomienda.

Figura 99 Consumo de endpoint para actualizar el estado de envío de una encomienda

```
export function updateStatusParcelRegister(form) {
  const request = axios.post(
    `${process.env.REACT_APP_API_URL}/api/aencomiendaestado`,
    form
  );

  return (dispatch) =>
    request.then((response) => {
      if (parseInt(response.data.status) === 404) { ... }
      if (response.data.status === 200) {
        toast.success(response.data.mensaje);
      } else {
        toast.error(response.data.mensaje);
      }
      /* getMenuControl(); */
      return dispatch({
        type: CRUD_PARCEL_REGISTER,
        payload: response.data.detalles ? response.data.detalles : null,
      });
    });
}
```

Fuente Elaboración propia

Luego de construir las funciones necesarias para el consumo de los endpoints del api del backend, se construyó el contenedor reducer para redux como se muestra en la figura 100, el cual servirá para almacenar los datos de manera temporal para ser consultados desde cualquier componente, sin la necesidad de pasar los datos por los props.

Figura 100 Contendor reducer de encomiendas

```
import { actions } from '../../../../../actions/app';
const initialState = {
    list_parcel_register: null,
    list_parcel_register_detail: null,
    crud_parcel_register: null,
    parsel_search: null,
    loader_parsel_search: false,
};
const parcelRegister = function (state = initialState, action) {
    switch (action.type) {
        case Actions.GET_LIST_PARCEL_REGISTER: {
            return {
                ...state,
                list_parcel_register: action.payload,
            };
        }
        case Actions.GET_LIST_PARCEL_REGISTER_DETAIL: {
            return {
                ...state,
                list_parcel_register_detail: action.payload,
            };
        }
        case Actions.CRUD_PARCEL_REGISTER: {
            return {
                ...state,
                crud_parcel_register: action.payload,
                list_parcel_register_detail: null,
            };
        }
        case Actions.PARCEL_SEARCH: {
            return {
                ...state,
                parsel_search: action.payload,
            };
        }
        case Actions.PARCEL_SEARCH_LOADER: {
            return {
                ...state,
                loader_parsel_search: true,
            };
        }
    }
}
```

Fuente: Elaboración propia

3.2.2.2.1.1. Tab de gestión de encomiendas registradas

Luego se construyó la vista en el componente en el cual se consultaron los registros con la función “getParcelRegister” de los adelantos como se muestra en la figura 101, también se muestra el recorrido de la lista “list_parcel_register” en cual tienes todos los registros de las encomiendas registradas de los usuarios y se filtrar para solo mostrar los registros que ingreso el usuario actual, se realiza un mapeo a esta lista se formatea del estado de la encomienda ya que este se guarda con un entero.

Figura 101 Consumo de los registros de encomienda

```
useEffect(() => {
  console.log("user_ID", props.userRole.id);
  props.getParcelRegister();
  props.getAllSeries(props.userRole.id);
  props.getAllUnitMeasures();
}, [props.crud.Parcel_register]);
useEffect(() => {
  if (props.list_parcel_register) {
    setRows(
      props.list_parcel_register
        .filter((p) => p.idusuario == props.userRole.id)
        .map((p) => {
          p.id = p.id_encomienda;
          if (p.estado == 0) p.estadoText = "ORIGEN";
          if (p.estado == 1) p.estadoText = "TRANSITO";
          if (p.estado == 2) p.estadoText = "DESTINO";
          if (p.estado == 3) p.estadoText = "ENTREGADO";
          return p;
        })
    );
  }
}
```

Fuente: Elaboración propia

En la figura 102 se muestra el consumo de las funciones para las acciones de editar, facturar, imprimir y eliminar los registros de encomiendas.

Figura 102 Consumos de las funciones para las acciones de editar, facturar, imprimir y eliminar

```
//Editar Order y abrir modal
const onClickEdit = () => {
  if (props.openings) {
    if (
      props.openings.filter(
        opening =>
          opening.id_usuario == props.userRole.id && opening.estado == 0
      ).length == 0
    ) {
      setManagementDriverModal(true);
      setManagementDriverSelect(row);
      props.getParcelRegisterDetail(row);
    }
  } else { ... }
};

//imprimir orden
const onClickPrint = () => {
  window.open(
    `${process.env.REACT_APP_API_URL}/api/encomienda/${row.id_encomienda}`,
    "_blank"
  );
};

//Eliminar orden
const onClickDelete = () => {
  props.deleteParcelRegister(row);
};
const onClickFacturacion = () => {
  if (props.openings) {
    if (
      props.openings.filter(
        opening =>
          opening.id_usuario == props.userRole.id && opening.estado == 0
      ).length == 0
    ) {
      setManagementDriverModal(true);
      setManagementDriverSelect(row);
      props.getParcelRegisterDetail(row);
    }
  } else { ... }
};
```

Fuente: Elaboración propia

En la figura 103 se muestra el fragmento de código del modal de creación, el cual muestra a inicializar los datos de encomienda o cargar los datos de un registro para editar.

Figura 103 Fragmento de código de inicialización de datos de encomienda en modal de creación y edición

```
);
useEffect(() => {
  if (props.statusModal) {
    if (props.driverSelect) {
      console.log(
        "idselect",
        props.driverSelect.idremitente + "-" + props.driverSelect.idconsignado
      );
      props.getSearchSelectGrid(
        props.driverSelect.idremitente + "-" + props.driverSelect.idconsignado
      );
      setValues(props.driverSelect);
    } else {
      setValues(initialsStateValues);
    }
  }
}, [props.statusModal]);
```

Fuente: Elaboración propia

En la figura 104 se muestra el consumo de la función “saveParcelRegister” para el registro de una encomienda y el consumo de la función “updateParcelRegister” para la edición de un registro.

Figura 104 Consumo de función para crear y actualizar registro de encomienda

```
];
function handleSubmit(e) {
  e.preventDefault();
  if (validateForm()) return;

  const form = {
    encomienda: { ...values, subtotal: sumSubTotalData() },
    detalle: dataArray,
  };

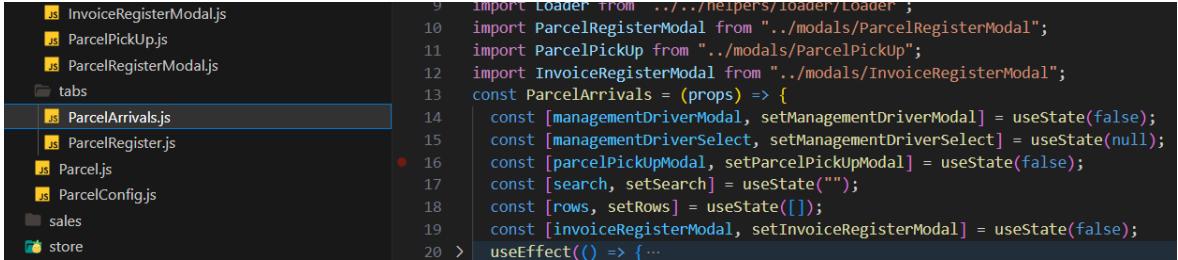
  if (props.driverSelect) {
    props.updateParcelRegister(form);
  } else {
    props.saveParcelRegister(form);
  }
  setValues(initialsStateValues);
  props.hideModal();
}
```

Fuente: Elaboración propia

3.2.2.2.1.2. Tab de gestión de encomiendas por llegar

En la figura 105 se muestra el componente donde se realizará la gestión de las encomiendas por llegar a la sucursal del usuario.

Figura 105 Tab de encomiendas por llegar



The screenshot shows a code editor with a file tree on the left and a code editor on the right. The file tree includes files like InvoiceRegisterModal.js, ParcelPickUp.js, ParcelRegisterModal.js, tabs, ParcelArrivals.js (which is selected), ParcelRegister.js, Parcel.js, ParcelConfig.js, sales, and store. The code editor contains a snippet of JavaScript:

```
9 import Loader from '../../../../../Helpers/Loader/Loader';
10 import ParcelRegisterModal from '../modals/ParcelRegisterModal';
11 import ParcelPickUp from '../modals/ParcelPickUp';
12 import InvoiceRegisterModal from '../modals/InvoiceRegisterModal';
13 const ParcelArrivals = (props) => {
14   const [managementDriverModal, setManagementDriverModal] = useState(false);
15   const [managementDriverSelect, setManagementDriverSelect] = useState(null);
16   const [parcelpickupModal, setParcelPickupModal] = useState(false);
17   const [search, setSearch] = useState("");
18   const [rows, setRows] = useState([]);
19   const [invoiceRegisterModal, setInvoiceRegisterModal] = useState(false);
20   useEffect(() => { ... })
```

Fuente: Elaboración propia

Luego se consultaron los registros con la función “getParcelRegister” de los adelantos como se muestra en la figura 106, también se muestra el recorrido de la lista “list_parcel_register” en cual tienes todos los registros de las encomiendas registradas de los usuarios y se filtrar para solo mostrar los registros que tienen como destino la sucursal del usuario, se realiza un mapeo a esta lista se formatea del estado de la encomienda ya que este se guarda con un entero.

Figura 106 Consumo de los registros de encomienda

```
useEffect(() => {
  console.log("user_ID", props.userRole.id);
  props.getParcelRegister();
  props.getAllSeries(props.userRole.id);
  props.getAllUnitMeasures();
}, [props.crud.Parcel_register]);
useEffect(() => {
  if (props.list.Parcel_register) {
    setRows(
      props.list.Parcel_register
        .filter(
          (p) =>
            p.id_usuario != props.userRole.id &&
            p.id_sucursal_inicio != props.userRole.sucursal_id
        )
        .map((p) => {
          p.id = p.id_encomienda;
          if (p.estado == 0) p.estadoText = "ORIGEN";
          if (p.estado == 1) p.estadoText = "TRANSITO";
          if (p.estado == 2) p.estadoText = "DESTINO";
          if (p.estado == 3) p.estadoText = "ENTREGADO";
          return p;
        })
    );
  }
}, [props.list.Parcel_register]);
```

Fuente: Elaboración propia

En la figura 107 se muestra el consumo de las funciones para las acciones de editar, facturar, imprimir y eliminar los registros de encomiendas.

Figura 107 Consumos de las funciones para las acciones de editar, facturar, imprimir y eliminar

```
//Editar Order y abrir modal
const onClickEdit = () => {
  if (props.openings) {
    if (
      props.openings.filter(
        opening =>
          opening.id_usuario == props.userRole.id && opening.estado == 0
      ).length == 0
    ) {
      setManagementDriverModal(true);
      setManagementDriverSelect(row);
      props.getParcelRegisterDetail(row);
    }
  } else { ... }
};

//imprimir orden
const onClickPrint = () => {
  window.open(
    `${process.env.REACT_APP_API_URL}/api/encomienda/${row.id_encomienda}`,
    "_blank"
  );
};

//Eliminar orden
const onClickDelete = () => {
  props.deleteParcelRegister(row);
};
const onClickFacturacion = () => {
  if (props.openings) {
    if (
      props.openings.filter(
        opening =>
          opening.id_usuario == props.userRole.id && opening.estado == 0
      ).length == 0
    ) { ... }
  }
};
```

Fuente: Elaboración propia

En la figura 108 se muestra un fragmento del código para la facturación de un registro de encomienda.

Figura 108 Modal de facturación de encomienda

```
return (
  <div>
    <Modal
      size="lg"
      show={props.statusModal}
      onHide={props.hideModal}
      centered
    >
      <form onSubmit={handleSubmit}>
        <Modal.Header closeButton>
          <Modal.Title>
            Facturar encomienda
            {parcel.id_encomienda
              ? " / Guía " + props.leftZero(parcel.id_encomienda, 7)
              : ""}
            {parcel.codigo ? " / Código " + parcel.codigo : ""}
          </Modal.Title>
        </Modal.Header>
        <Modal.Body>
          <Container>
            <Row>
              <Col className="d-flex justify-content-center">
                <p className="h2 mx-3 text-primary">TOTAL A PAGAR</p>
                <p className="h2 mx-3 text-success">
                  {parseFloat(parcel.subtotal).toFixed(2)}
                </p>
              </Col>
            </Row>
            <Row>
              <Col xs={5}>
                <label className="form-label">DNI/RUC/OTRO:</label>
                <InputGroup className="mb-3">
                  <AsyncTypeahead
                    id="customer-typeahead"
                    allowNew={false}
                    <!-- Other AsyncTypeahead props -->
```

Fuente: Elaboración propia

Figura 109 Fragmento de código del modal de recepción de encomienda

```
rade > parcel_register > modals > ParcelPickUp.js > ...
114     show={props.statusModal}
115     onHide={props.hideModal}
116     centered
117   >
118     <form onSubmit={handleSubmit}>
119       <Modal.Header closeButton>
120         <Modal.Title>
121           | Registrar recepción de encomienda - {values.codigo}
122         </Modal.Title>
123       </Modal.Header>
124       <Modal.Body>
125         <Container>
126           <Row>
127             <Col sm={8} className="form-inline my-2 my-lg-0 d-flex">
128               <label className="form-label mr-2" style={{ width: "25%" }}>
129                 Remitente:
130               </label>
131               <input
132                 className="form-control"
133                 value={values.remitente_nombre}
134                 disabled={true}
135                 style={{ width: "70%" }}
136                 type="text"
137               />
138             </Col>
139             <Col sm={4} className="form-inline my-2 my-lg-0 d-flex">
140               <label className="form-label mr-2">DNI/RUC:</label>
141               <input
142                 className="form-control"
143                 style={{ width: "55%" }}
144                 value={values.remitente_documento}
145                 disabled={true}
146                 type="text"
147               />
148             </Col>
149           </Row>
```

Fuente: Elaboración propia

3.2.3. Resultados

Se concluyó con las tareas y se logró el objetivo como resultado se obtuvo las siguientes vistas funcionales.

Se construyó la gestión de encomiendas registradas como se muestra en la figura 110.

Figura 110 Tab registro de encomiendas

The screenshot shows a web-based application for managing packages. At the top, there's a header with the title 'Encomiendas' and a 'INICIO / PROCESOS' link. Below the header, there are two tabs: 'Registro de encomiendas' (selected) and 'Encomiendas por llegar'. A blue button labeled 'Registrar' is visible. On the right, there's a search bar with the placeholder 'Ingresar criterio de búsqueda:'. The main area displays a table with the following data:

Acciones	Comprobante	R	DNI/RUC	Reg. por	Transito	Fecha	Estado
	CT01-0000001	J...	73141097	tarapoto	Tarapoto - Lamas	2023-05-23 16:11:57	Origen

Fuente: Elaboración propia

Se construyó el tab de gestión de encomiendas por llegar como se muestra en la figura 111.

Figura 111 Tab de encomiendas por llegar

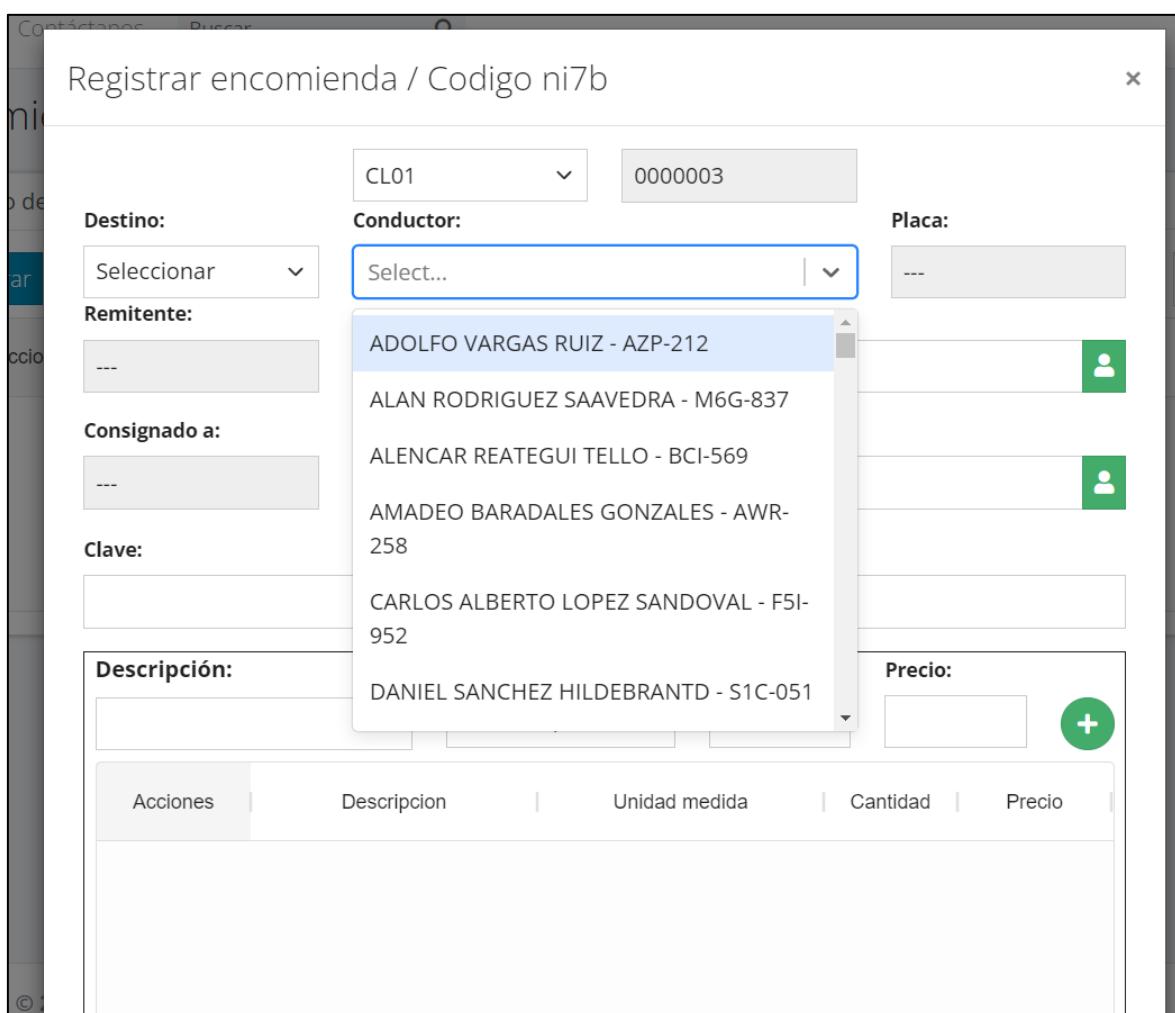
This screenshot shows the 'Encomiendas por llegar' tab. The interface is similar to Figure 110, with a header, tabs, and a search bar. The table data is as follows:

Acciones	Comprobante	R	DNI/RUC	Reg. por	Transito	Fecha	Estado
	CT01-0000002	JC	73141097	tarapoto	Tarapoto - Lamas	2023-06-06 11:10:33	Entregado
	CT01-0000001	JC	73141097	tarapoto	Tarapoto - Lamas	2023-05-23 16:11:57	Origen

Fuente: Elaboración propia

Se construyó el modal para registrar la encomienda como se muestra en la figura 112 y en el registro de encomienda también se registra los productos que contiene la encomienda como se muestra en la figura 113.

Figura 112 Modal de registro y edición de encomienda



Fuente: Elaboración propia

Figura 113 Formulario y tab del detalle de registro de encomienda

Descripción:	Unidad:	Cantidad:	Precio:															
<input type="text"/>	UNIDAD (SERVICIO) <input type="button" value="▼"/>	<input type="text"/>	<input type="text"/>															
<table border="1"><thead><tr><th>Acciones</th><th>Descripción</th><th>Unidad medida</th><th>Cantidad</th><th>Precio</th></tr></thead><tbody><tr><td></td><td>mochila</td><td>69</td><td>1.00</td><td>10.00</td></tr><tr><td colspan="5"> </td></tr></tbody></table>				Acciones	Descripción	Unidad medida	Cantidad	Precio		mochila	69	1.00	10.00					
Acciones	Descripción	Unidad medida	Cantidad	Precio														
	mochila	69	1.00	10.00														
Condición:	Entrega a:	Total:																
<input type="button" value="PAGADO"/> <input type="button" value="▼"/>	<input type="button" value="AGENCIA"/> <input type="button" value="▼"/>	10.00																
Observación: <input type="text"/>																		
<input type="button" value="Actualizar"/>																		

Fuente: Elaboración propia

Se construyo el formulario para facturar las encomiendas directamente de la table donde se listan los registros de las encomiendas como se muestra en la figura 114.

Figura 114 Modal de facturación de encomienda

The screenshot shows a modal window titled "Facturar encomienda / Guía 0000003 / Código BgOP". At the top, it displays "TOTAL A PAGAR 10.00". The form fields include:

- DNI/RUC/OTRO: Buscar cliente
- Nombre/ Razon social:
- Dirección:
- Comprobante: BOLETA DE VENTA ELECTRONICA Serie: BT01 Número: 0000008
- Tipo de pago: -Seleccione- Forma de pago: -Seleccione- Fecha de emisión: 06/06/2023
- Moneda: SOLES Número de referencia:
- Observación:
- Cobrar

Fuente: Elaboración propia

Se construyó en modal para la entrega de la encomienda como se muestra en la figura 115.

Figura 115 Modal de entrega de encomienda

The screenshot shows a modal window titled "Registrar recepción de encomienda - Ne5e". The form fields include:

Remitente:	JOSE FERNANDO VILCA MENESSES	DNI/RUC:	73141097
Consignado:	JOSE FERNANDO VILCA MENESSES	DNI/RUC:	73141097
Condición:	CONTRAENTREGA	Entrega:	AGENCIA

Buttons: Origen (Tarapoto) → Transito → Destino (Lamas)

Total: 20.00 Clave: Entregar

Fuente: Elaboración propia

Se construyó los endpoints encargados de generar el reporte pdf para la impresión de los comprobantes como se muestra en las figuras 116 y 117.

Figura 116 Comprobante de registro de encomienda

The document is a PDF titled "Comprobante de registro de encomienda" (Delivery Order Receipt). It features a logo for "EMPRESA DE TRANSPORTES Y TURISMO 'LAMAS TOURS S.A.'" with a stylized bus graphic. Below the logo, company details are listed: RUC: 20531513453, DOMICILIO FISCAL: JR. ALFONSO UGARTE, and TELF: 042522283. The document is divided into sections: "CARGO" (Cargo) with code CT01-0000001, "PAGADO" (Paid), "AGENCIA" (Agency), and "Destino" (Destination) set to "Tarapoto-Lamas". It includes transaction details: FECHA: 06/06/2023, HORA: 11:09:16. The "Remitente" (Sender) is JOSE FERNANDO VILCA MENESES, DNI/RUC: 73141097. The "Consignatario" (Recipient) is also JOSE FERNANDO VILCA MENESES, DNI/RUC: 73141097. The "Dirección" (Address) is listed but blank. A table shows the item details: CANT. (Quantity), DESCRIPCIÓN (Description), U.M. (Unit of Measure), P.U. (Price per Unit), and SUB TOTAL (Subtotal). One item is listed: 01 mochila (1.00 units, ZZ price, 10.00 subtotal). The total amount is S/ 10.00. An "Observación" (Observation) section is present but empty.

CANT.	DESCRIPCIÓN	U.M.	P.U.	SUB TOTAL
1.00	01 mochila	ZZ	10.00	10.00

TOTAL: S/ 10.00

Observación:

Fuente: Elaboración propia

Figura 117 Comprobante de encomienda facturada



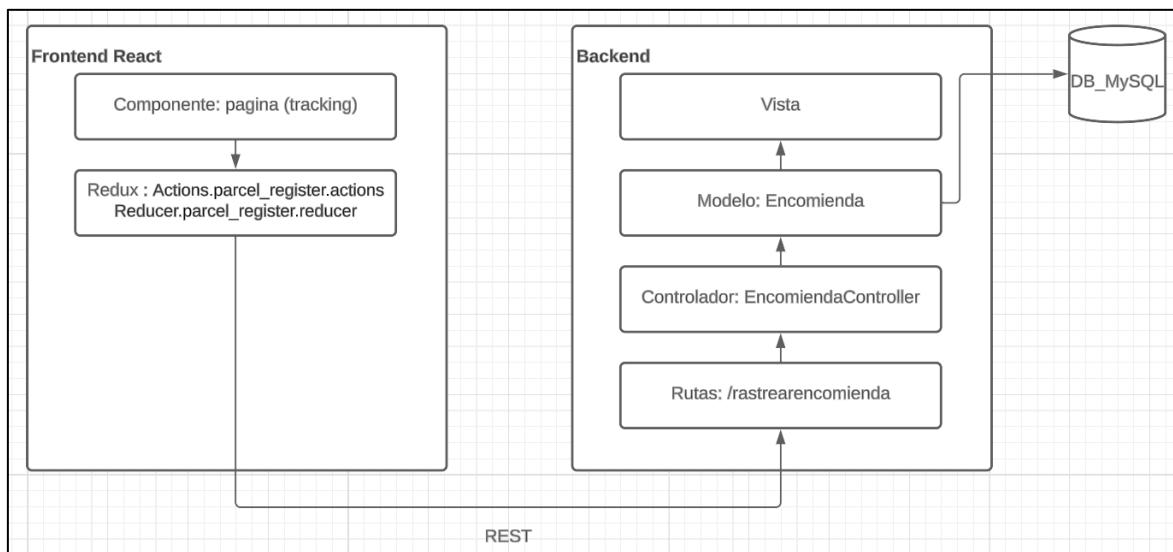
Fuente: Elaboración propia

3.3. Construir vista pública para la consulta de encomiendas (Trade)

La vista de consulta de encomiendas será una página publica la cual permitirá a los clientes hacer ver el estado y un resumen de su encomienda.

Para iniciar la construcción se tendrá en cuenta el diagrama de flujo de trabajo de la figura 118 donde se describen los archivos a interactuar en el frontend y backend.

Figura 118 Flujo de trabajo de seguimiento de encomienda



Fuente: Elaboración propia

3.3.1. Tareas

En la figura 119 se muestra el objetivo a alcanzar y las tareas a realizar.

Figura 119 Objetivo: Construir rastreador de encomiendas

O. Construcción de vista publica de rastreador de encomiendas					
T1. Se debe rastrear por el numero de guia y el codigo de guia de la boleta de la encomiendas					
T2. Mostrar los estados de una encomienda y resaltar el actual					
T3. Al no encontrar una encomienda con la guia y el codigo mostrar un mensaje de información					

Fuente: Elaboración propia.

3.3.1.1. Referencia

En la figura 120 se muestra una referencia para la vista de rastreo de encomiendas.

Figura 120 Referencia para el tracking de encomiendas



Fuente: HMF inversiones.

3.3.2. Ejecución de tareas

Las tareas se dividieron en backend y frontend.

3.3.2.1. Backend

3.3.2.1.1. T1. Se debe rastrear por el número de guía y el código de guía de la boleta de la encomienda.

Para el rastreo de la encomienda se construyó una nueva función en el controlador de encomienda el cual permite filtrar las encomiendas por el número de guía y el código de guía que se imprimían en el comprobante de encomienda, como se muestra en la figura 121 y luego se registró la ruta como se muestra en la figura 122.

Figura 121 Fragmento de código de la función para seguimiento de encomienda

```
public function get_encomienda_tracking(Request $request){  
    $nroGuia=$request->input('nroGuia');  
    $codGuia=$request->input('codGuia');  
    $encomienda=DB::table('encomienda as e')  
        ->select('e.*',DB::raw('sc.serie_comprobante,pr.razon_social_nombre as remitente_nombre,pr.numero_documento as numero_documento'))  
        ->join('serie_comprobante as sc','sc.serie_comprobante_id','=',$e->serie_comprobante_id')  
        ->join('persona as pr','pr.persona_id','=',$e->idremitente')  
        ->join('persona as pc','pc.persona_id','=',$e->idconsignado')  
        ->join('transportista as t','t.idtransportista','=',$e->idtransportista)  
        ->join('establecimiento_sucursal as esi','esi.establecimiento_sucursal_id','=',$e->id_sucursal_inicio')  
        ->join('establecimiento_sucursal as esf','esf.establecimiento_sucursal_id','=',$e->id_sucursal_fin')  
        ->join('users as u','u.id','=',$e->idusuario')  
        ->orderBy('e.id_encomienda', 'desc')  
        ->where([['e.estado','!=',-1],  
            ['e.id_encomienda','=',$nroGuia],
```

Fuente: Elaboración propia

Figura 122 Registro de la ruta para rastrear encomienda

```
Route::post('/rastrearencomienda', 'EncomiendaController@get_encomienda_tracking');
```

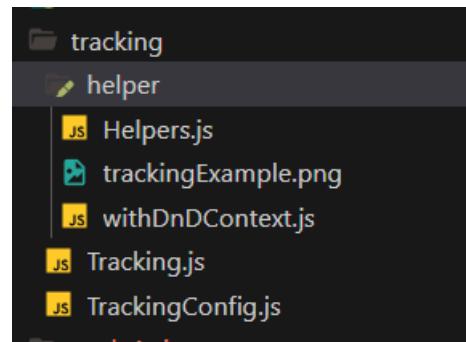
Fuente: Elaboración propia

3.3.2.2. Frontend

3.3.2.2.1. T2. Mostrar los estados de una encomienda y resaltar el actual

En la figura 123 se muestra la distribución de archivos para la vista de seguimiento de encomienda.

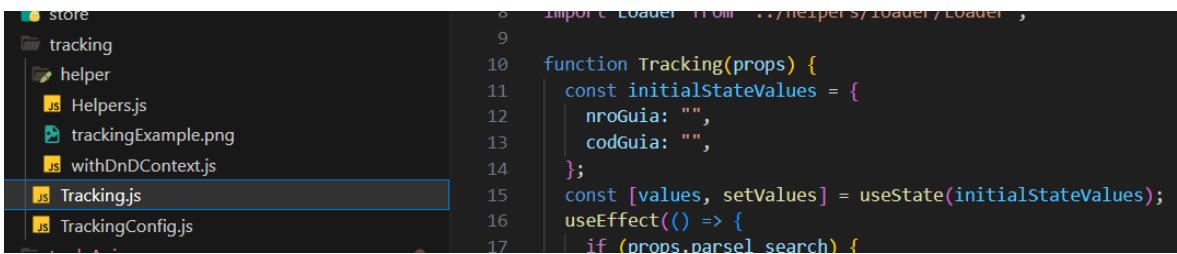
Figura 123 Distribución de carpetas de la vista de seguimiento



Fuente: Elaboración propia

Se crea el componente de la vista como se muestra en la figura 124.

Figura 124 Fragmento de código de la vista de seguimiento



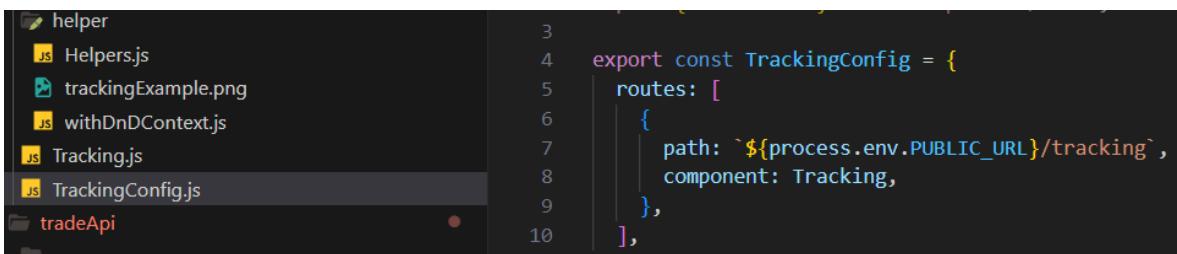
```
store
  tracking
    helper
      JS Helpers.js
      trackingExample.png
      JS withDnDContext.js
    JS Tracking.js
    JS TrackingConfig.js

  8  import { Loader } from '../helpers/loader/Loader';
  9
 10 function Tracking(props) {
 11   const initialStateValues = {
 12     nroGuia: '',
 13     codGuia: '',
 14   };
 15   const [values, setValues] = useState(initialStateValues);
 16   useEffect(() => {
 17     if (props.parseSearch) {
```

Fuente: Elaboración propia

En la figura 125 se muestra la configuración de la ruta de la vista, en este caso no se muestra en la configuración el atributo “auth” que define la protección a la ruta.

Figura 125 Fragmento de código de la configuración de la ruta y la autorización



```
helper
  JS Helpers.js
  trackingExample.png
  JS withDnDContext.js
  JS Tracking.js
  JS TrackingConfig.js
  tradeApi
    ...
  VS

  3
  4  export const TrackingConfig = {
  5    routes: [
  6      {
  7        path: `${process.env.PUBLIC_URL}/tracking`,
  8        component: Tracking,
  9      },
 10    ],
 11  },
```

Fuente: Elaboración propia

Luego se creó la función para consumir el endpoint de “rastrearencomienda” y se agregó el consumo al actions y reducir ya creados de módulo de encomienda como se muestra en las figuras 126 y 127.

Figura 126 Fragmento de código para el consumo del endpoint de rastrear encomienda

```
export function searchParcelRegister(form) {
  const request = axios.post(
    `${process.env.REACT_APP_API_URL}/api/rastrearencomienda`,
    form
  );

  return (dispatch) => {
    dispatch(searchParcelLoader(true));
    return request.then((response) => {
      if (response.data.status === 200) { ...
      } else { ...
      }
      dispatch(searchParcelLoader(false));
      return dispatch({
        type: PARCEL_SEARCH,
        payload: response.data.detalles ? response.data.detalles : null,
      });
    });
  };
}
```

Fuente: Elaboración propia

Figura 127 Fragmento de código de redux de la encomienda

```
}
case Actions.PARCEL_SEARCH: {
  return {
    ...state,
    parcel_search: action.payload,
  };
}
case Actions.PARCEL_SEARCH_LOADER: {
```

Fuente: Elaboración propia

Luego se construyó el formulario donde se ingresa el número de guía y código de guía como se muestra en la figura 128, en la figura 129 se muestra el consumo de la función para realizar la consulta al endpoint del backend.

Figura 128 Fragmento de código del formulario de seguimiento de encomienda

```
</Card.Header>
<Card.Body>
  <form onSubmit={handleSubmit}>
    <Row>
      <Col sm={12}>
        <label className="form-label">Número de guía:</label>
        <input
          className="form-control"
          value={values.nroGuia}
          id="nroGuia"
          onChange={(e) => {
            if (e.target.value.length <= 7)
              setValues({ ...values, nroGuia: e.target.value });
          }}
          type="text"
        />
      </Col>
    </Row>
    <Row className="mt-2">
      <Col sm={12}>
        <label className="form-label">Código de guía:</label>
        <input
          className="form-control"
          value={values.codGuia}
          id="codGuia"
          onChange={(e) => {
            if (e.target.value.length <= 4)
              setValues({ ...values, codGuia: e.target.value });
          }}
          type="text"
        />
      </Col>
    </Row>
  </form>
</Card.Body>
```

Fuente: Elaboración propia

Figura 129 Fragmento de código del consumo el endpoint de seguimiento de encomienda

```

  }
  function handleSubmit(e) {
    e.preventDefault();
    if (validateForm()) return;
    props.searchParcelRegister(values);
    focusForm("card-data");
  }
  const focusForm = (idForm) => {
```

Fuente: Elaboración propia

Luego de realizar la consulta de recuperar los datos directamente de recuperar “parsel_search” que lo almacena y se pinta los datos en la vista como se muestra en la figura 130.

Figura 130 Fragmento de código donde se muestra el resultado de la búsqueda

```
sm={12}
className=" d-flex justify-content-center align-items-center">
>
<button
  className={
    " btn  " +
    (props.parsel_search.estado == 0
      ? "bg_button_tracking"
      : "border border-dark")
  }
  data-toggle="tooltip"
  data-placement="bottom"
  title={props.parsel_search.fecharegistro}
  style={{ borderRadius: "10px" }}
>
  Origen
  <br /> ({props.parsel_search.sucursalinicio})
</button>

<i
  className="fas fa-arrow-right"
  style={{ color: "#A1A1A1" }}></i>
<button
  className={
    " btn  " +
    (props.parsel_search.estado == 1
      ? "bg_button_tracking"
      : "border border-dark")
  }
  data-placement="bottom"
  title={props.parsel_search.fechatransporte}
  style={{ height: "100%", borderRadius: "10px" }}>
```

Fuente: Elaboración propia

3.3.2.2.2. T3. Al no encontrar una encomienda con la guía y el código mostrar un mensaje de información

Si luego de realizar la consulta el reducer “parcel_search” sigue vacío significa que no encontró ninguna encomienda con el número de guía y código de guía, por lo tanto, se mostrará un mensaje de información y una imagen de referencia de como ubicar los catos de la guía como se muestra en la figura 131.

Figura 131 Fragmento de código de la información al no encontrar encomienda

```
>
  <Loader
    show={props.loader_parsel_search}
    center={true}
    isbd={false}
  />
  {!props.parsel_search ? (
    <Card.Body className="d-flex align-items-center justify-content-center">
      <Col>
        <Row>
          <Col sm={12}>
            <p className="h4 text-center">
              Puedes consultar los datos rastreo en la parte inferior
              de su guia:
            </p>
          </Col>
        </Row>
        <img
          src={TrackingExample}
          width="100%"
          alt="ejemplo rastreo"
          className="border border-dark"
        />
      </Col>
    </Card.Body>
```

Fuente: Elaboración propia

3.3.3. Resultados

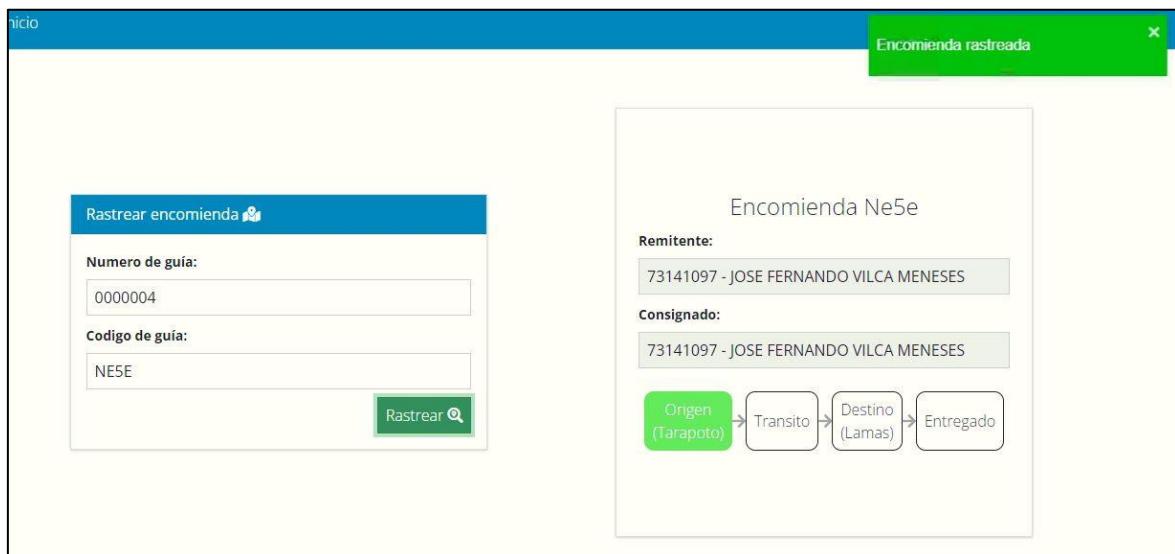
Se concluyó con las tareas y se logró el objetivo como resultado se obtuvo las siguientes vistas funcionales que se muestran en la figura 132 y 133.

Figura 132 Vista de seguimiento al no encontrar encomienda



Fuente: Elaboración propia

Figura 133 Vista de seguimiento al encontrar encomienda



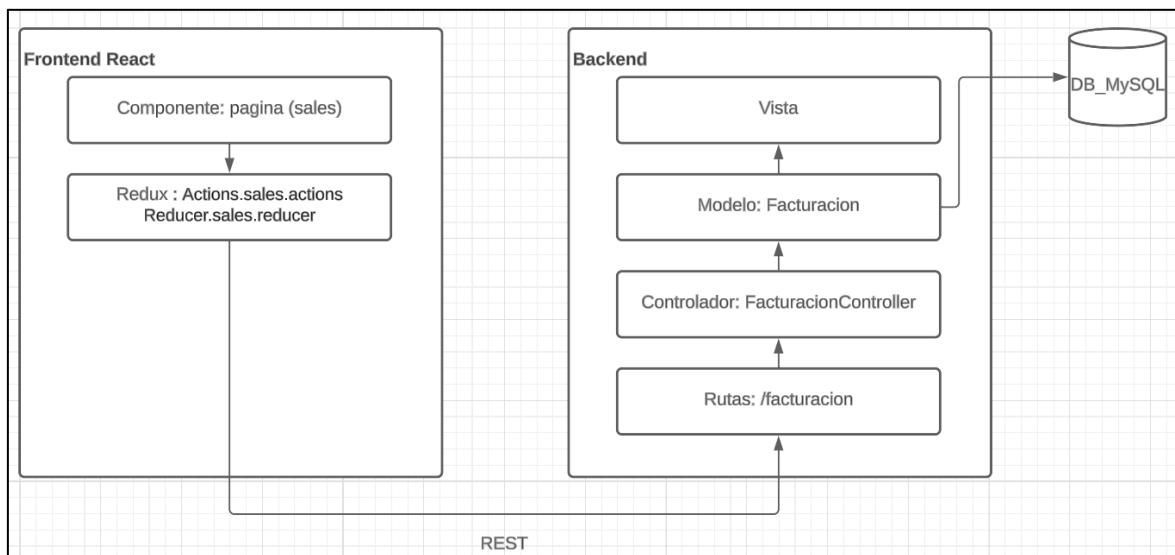
Fuente: Elaboración propia

3.4. Implementar nuevas funcionalidades al submódulo facturación (Trade)

El submódulo de facturación es el encargado de la facturación de los servicios brindados por la empresa, se realizará una modificación en el cual se agregará un campo de descripción adicional a los productos.

El submódulo de facturación tiene el siguiente diagrama de flujo de trabajo de la figura 134 donde se describen los archivos a interactuar en el frontend y backend.

Figura 134 Flujo de trabajo de la gestión de factura



Fuente: Elaboración propia

3.4.1. Tareas

En la figura 135 se muestra el objetivo a alcanzar y las tareas a realizar.

Figura 135 Objetivo: Nuevas funcionalidades al submódulo facturación

O. Nuevas funcionalidades al submódulo facturación
T1. Al agregar un producto quitar el código en su lugar poner un input donde se pueda ingresar un descripción adicional
T2. Concatenar el detalle del producto con la descripción adicional en la vista de facturación y en el modal de editar factura
T3. Concatenar el nombre del producto y la descripción adicional en el detalle del comprobante al imprimir factura

Fuente: Elaboración propia

3.4.1.1. Referencias

En la figura 136 se muestran los cambios a realizar en el frontend.

Figura 136 Referencia para el cambio de facturación

The screenshot shows a software application window titled "Ventas". At the top, there are tabs for "Registro" and "Listado de ventas", with "Listado de ventas" being the active tab. Below the tabs, there's a section for "Ingreso de producto" with fields for "Código:" (with a crossed-out value), "Descripción del producto:" (with a dropdown arrow), "Precio:", "Cantidad:", and "SubTotal:". To the right of these fields is a green "+" button. Below this is a table with columns: "Acciones", "#", "Cantidad", "UM", "Descripción", "Precio", "Descuento", and "Importe". A single row is visible, showing a delete icon, the number 1, the quantity 1, the unit "UNIDAD (BIENES)", the description "LUNA ROJA", and the price 32.00. An arrow points from the text "Descripción adicional" at the top to the "+ Descripción adicional" entry in the table row.

Fuente: Elaboración propia

3.4.2. Ejecución de las tareas

3.4.2.1. Backend

3.4.2.1.1. T3. Concatenar el nombre del producto y la descripción adicional en el detalle del comprobante al imprimir factura

Como se muestra en el fragmento de código de la figura 137 con controlador de factura ya consultaba todos los datos del campo detalle de la factura, lo que se realizó es validar su el campo de “detalle_encomienda” este lleno para concatenar la descripción adicional de lo contrario se imprimirá solo el nombre del producto facturado como se muestra en la figura 138.

Figura 137 Fragmento de código de la consulta de datos para la impresión de factura

```
/*IMPRIMIR COMPROBANTES*/
public function pdf_imprime_comprobante($tipeid,$id){
    //return $idempresa.'-'.$id;
    /*CONSULTA PARA COMPROBANTE FACTURA, BOLETA, NOTA DE VENTA, TICKET*/
    $nota_credito = NotaCredito::where("facturacion_id",$id)->first();

    $facturacion = DB::table('facturacion as f')
        ->select('f.*',DB::raw('p.razon_social_nombre,p.numero_documento,p.direccion,tm.codigo,tp.tipo_pago'))
        ->join('persona as p','p.persona_id','=','f.persona_id')
        ->join('tipo_moneda as tm','tm.tipo_moneda_id', '=', 'f.tipo_moneda_id')
        ->join('tipo_operacion as top','top.tipo_operacion_id', '=', 'f.tipo_operacion_id')
        ->join('tipo_pago as tp','tp.tipo_pago_id', '=', 'f.tipo_pago_id')
        ->join('tipo_forma_pago as tfp','tfp.tipo_forma_pago_id' , '=', 'f.tipo_forma_pago_id')
        ->join('serie_comprobante as sc','sc.serie_comprobante_id' , '=', 'f.serie_comprobante_id')
        ->join('comprobante as c','c.comprobante_id','=', 'sc.comprobante_id')
        ->where('f.facturacion_id','','=',$id)
        ->first();
    /* return json_encode($facturacion); */
    $facturacionDetalle = DB::table('facturacion_detalle as fd')
        ->select('fd.*',DB::raw('p.denominacion,um.abreviatura,ed.descripcion as detalle_encomienda'))
        ->join('producto as p','p.producto_id','=', 'fd.producto_id')
        ->join('unidad_medida as um','um.unidad_medida_id','=', 'fd.unidad_medida_id')
        ->leftJoin('encomienda_detalle as ed','ed.id_encomienda_detalle','=', 'fd.id_encomienda_detalle')
        ->where('fd.facturacion_id','','=',$id)
        ->get();
    // $establecimiento = Establecimiento::where("establecimiento_id",$idempresa)->first();
}
```

Fuente: Elaboración propia

Figura 138 Fragmento de código de la validación de descripción adicional

```
$descontar=0;
$totalPagar=0;
foreach($facturacionDetalle as $rM){
    $count++;
    $total+=floatval($rM->sub_total);
    $descontar=$rM->descuento;
    if($rM->detalle_encomienda=="")$rM->detalle_encomienda=$rM->denominacion;
    $fpdf->SetXY(2, $textypost);
    $fpdf->MultiCell(0, 3, '*' . utf8_decode(strtoupper(($rM->producto_id==2)?$rM->detalle_encomienda:$y3 = $fpdf->GetY();
    $textypost = $y3 + 2;
    $fpdf->SetXY(2, $textypost);
    $fpdf->Cell(0, 0, str_pad(number_format(floatval($rM->cantidad), 2, '.', ','), (8 - strlen(floatval($textypost += 3;
}
$fpdf->Line(1, $textypost, 79, $textypost);
$textypost += 2;
```

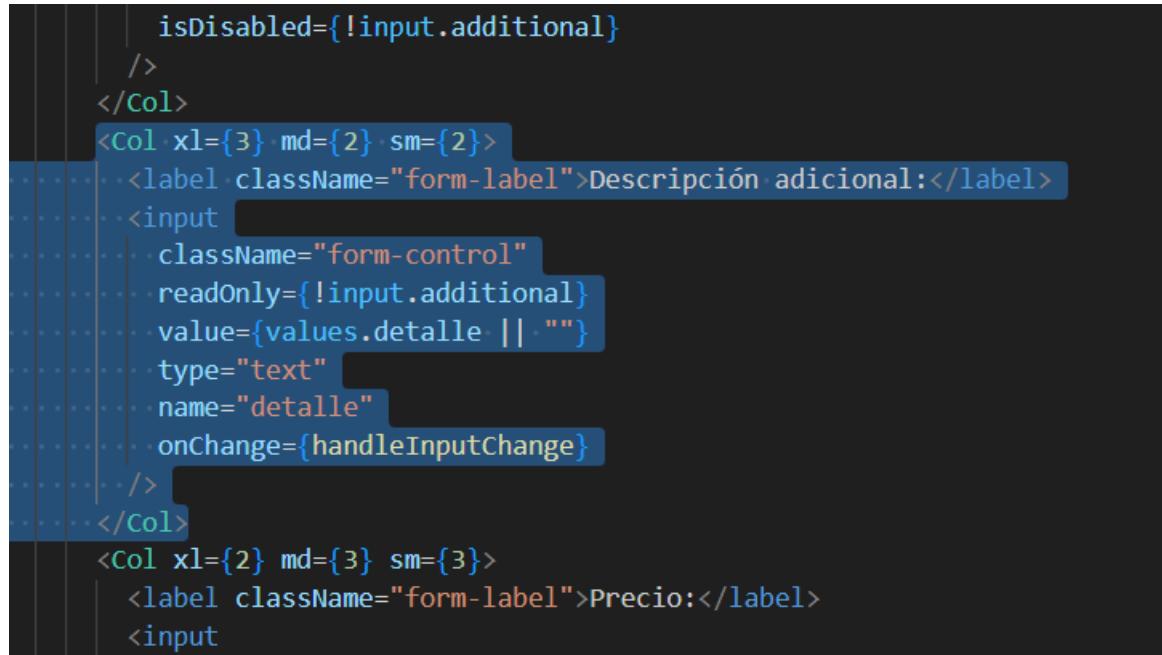
Fuente: Elaboración propia

3.4.2.2. Frontend

3.4.2.2.1. T1. Al agregar un producto quitar el código en su lugar poner un input donde se pueda ingresar una descripción adicional y T2. Concatenar el detalle del producto con la descripción adicional en la vista de facturación y en el modal de editar factura

Se agrego el campo de descripción adiciona al tab de registro de factura como se muestra en la figura 139.

Figura 139 Fragmento de código del input de descripción adicional



```
|    |  isDisabled={!input.additional}
|    |  />
|  </Col>
|  <Col xl={3} md={2} sm={2}>
|    <label className="form-label">Descripción adicional:</label>
|    <input
|      className="form-control"
|      readOnly={!input.additional}
|      value={values.detalle || ""}
|      type="text"
|      name="detalle"
|      onChange={handleInputChange}
|    />
|  </Col>
|  <Col xl={2} md={3} sm={3}>
|    <label className="form-label">Precio:</label>
|    <input
```

Fuente: Elaboración propia

En la figura 140 se muestra el fragmento de código de la validación para concatenar la descripción adicional para los registros que se mostrarán en la tabla de detalles de la factura que se muestra en la figura 141, lo mismo se realizó en el modal de editar.

Figura 140 Fragmento de código de la concatenación de la descripción adicional

```
        },
        field: "denominacion",
        headerName: "Descripción",
        headerAlign: "center",
        width: 180,
        disableClickEventBuddling: true,
        renderCell: ({ row }) => {
          if (row.detalle && row.detalle.length > 0) {
            return row.denominacion + " - " + row.detalle;
          } else {
            return row.denominacion;
          }
        },
        flex: 1,
      },
```

Fuente: Elaboración propia

3.4.3. Resultados

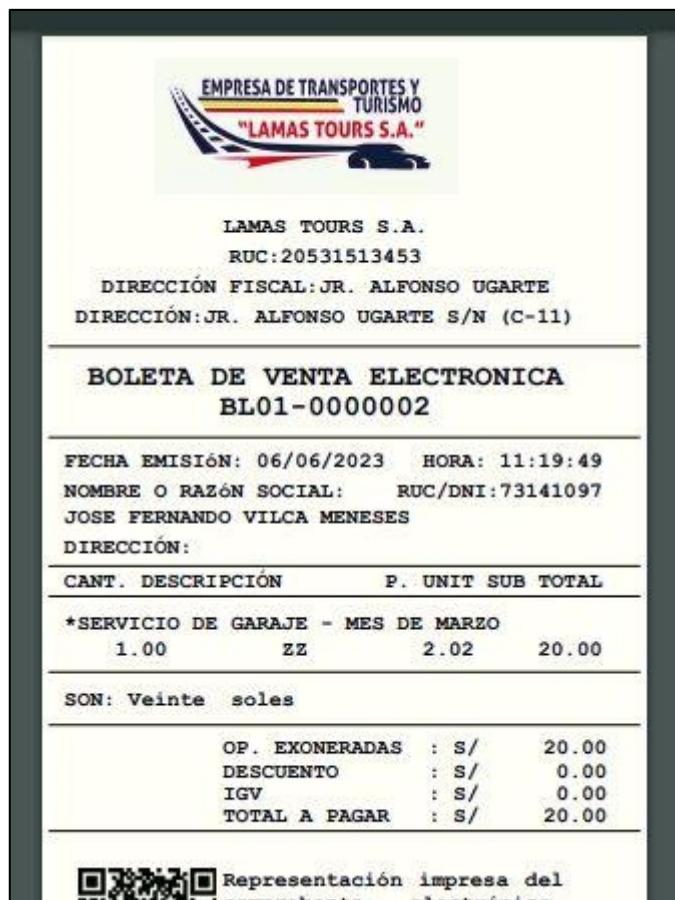
Se concluyó con las tareas y se logró el objetivo como resultado se obtuvo las siguientes vistas funcionales que se muestran en la figura 141 y 142.

Figura 141 Vista de registro de factura con el campo descripción adicional

The screenshot shows a software application window titled 'Ventas'. At the top, there are tabs for 'Registro' and 'Listado de ventas', with 'Listado de ventas' being the active tab. In the top right corner, there is a link 'INICIO / PROCESOS'. The main area contains a form for entering product details. The form has fields labeled 'Descripción del producto:' (with a dropdown arrow), 'Descripción adicional:', 'Precio:', 'Cantidad:', and 'SubTotal:'. Below the form is a table with columns: 'Acciones', '#', 'Cantidad', 'UM', 'Descripción', 'Precio', and 'Importe'. A green circular button with a '+' sign is located to the right of the form. The table has several rows, with the first row being highlighted in light blue.

Fuente: Elaboración propia

Figura 142 Comprobante de factura con la descripción adicional



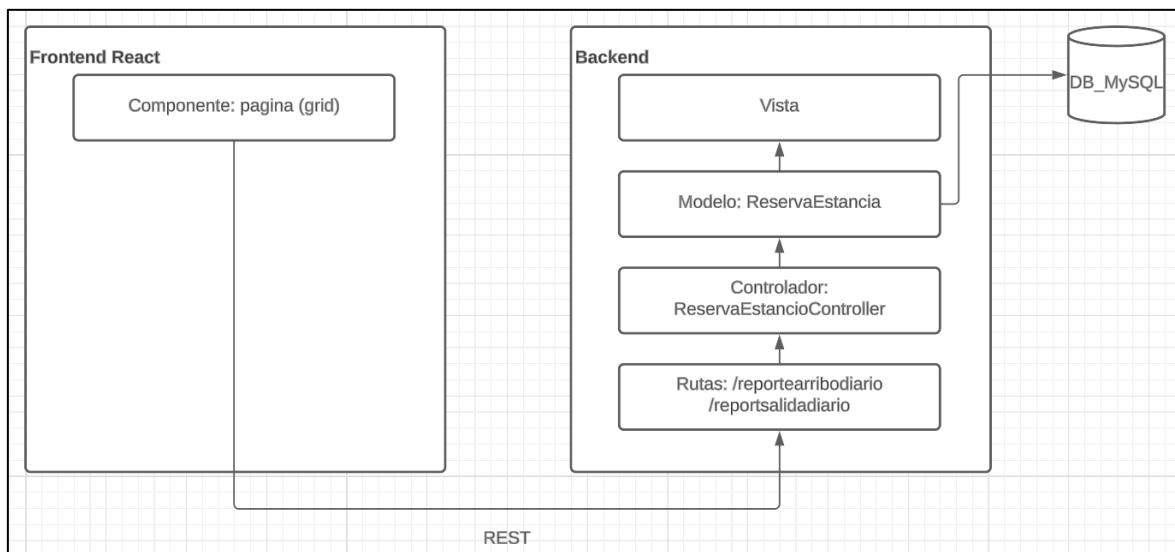
Fuente: Elaboración propia

3.5. Implementar reportes al submódulo grid (Booking)

El sub modulo grid es el encargado del registro de reserva, estancias, consumos, por habitación. Se implementará serán nuevas funcionalidades de reportes y envío de correo electrónico.

En la figura 143, se muestra de flujo de trabajo de los reportes a implementar, no se realiza ningún almacenamiento en redux, ya que los reportes pdfs a mostrar se consumen directamente de los endpoints.

Figura 143 Flujo de trabajo del submódulo grid de los reportes



Fuente: Elaboración propia

3.5.1. Tareas

En la figura 144 se muestra el objetivo y las tareas para implementar reportes al submódulo grid.

Figura 144 Objetivo: Implementar reportes a grid

O. Implementar reportes al submódulo grid
T1. hacer endpoints donde muestre los reportes pdf de salidas del dia y arrivos del dia
arrivos del dia.
T2. Mostrar en un modal el pdf de reporte de salidas del día.
T3. Mostrar en un modal el pdf de reporte de parte diario donde se listan todos los arribos del día

Fuente: Elaboración propia

3.5.2. Ejecución de tareas

Las tareas se dividieron en backend y frontend.

3.5.2.1. Backend

3.5.2.1.1. T1. hacer endpoints donde muestre los reportes pdf de salidas del día y arribos del día

Se agregaron las funciones como se muestra en las figuras 145 y 146 para generar los pdfs de los reportes y se agregó al controlador de “ReservaEstanciaController” el cual es el encargado de la reservas y estancias, posterior se creó las rutas como se muestra en la figura 146.

Figura 145 Fragmento de código del reporte de arribos diarios

```
public function get_reporte_diario_arribos(){
    $detalle= DB::table('reserva_estancia as re')
        ->rightJoin('reserva_estancia_huesped as reh','reh.reserva_estancia_id','=','re.reserva_estancia_id')
        ->leftJoin('reserva_estancia_vuelo as rev','re.reserva_estancia_id','=','rev.reserva_estancia_id')
        ->join('reserva_estancia_habitacion as reha','reha.reserva_estancia_id','=','re.reserva_estancia_id')
        ->join('persona as p','p.persona_id','=','re.persona_id')
        ->join('habitacion as ha','reha.habitacion_id','=','ha.habitacion_id')
        ->select('re.codigo_reserva', 'ha.numero_habitacion', 'p.razon_social_nombre', 'p.telefono', 'reh.persona_id')
        ->groupBy('reh.persona_id')
        ->groupBy('reha.reserva_estancia_habitacion_id')
        ->where([
            ['re.fecha_checkin','=',null]
        ])
        ->whereDay(['re.fecha_llegada',date("d")])
        ->orderBy('re.fecha_llegada','desc')
        ->get();

    for($i=0;$i<count($detalle);$i++){
        $detalle[$i]->pasajeroRZ=DB::table('persona')
            ->where('persona_id',$detalle[$i]->persona_id)
            ->get()[0]->razon_social_nombre;
    }
    $fpdf= new FpdfPerson();
    $fpdf->AddPage();

    $fpdf->SetFont('Courier', 'B', 14);
    $textypos=8;
    $fpdf->setXY(5,$textypos);

    $fpdf->Cell(0,0,"ARRIVOS DEL DIA",0,1,"C");
    $fpdf->Image("http://cdn.onlinewebfonts.com/svg/img_154778.png",70,($textypos-4),-2700);
    $textypos+= 8;
```

Fuente: Elaboración propia

Figura 146 Fragmento de código del reporte de salidas diarias

```
public function get_reporte_diario_salidas(){
    $detalle= DB::table('reserva_estancia as re')
    ->rightJoin('reserva_estancia_huesped as reh','reh.reserva_estancia_id','=','re.reserva_estancia_id')
    ->leftJoin('reserva_estancia_vuelo as rev','re.reserva_estancia_id','=','rev.reserva_estancia_id')
    ->join('reserva_estancia_habitacion as reha','reha.reserva_estancia_id','=','re.reserva_estancia_id')
    ->join('persona as p','p.persona_id','=','re.persona_id')
    ->join('habitacion as ha','reha.habitacion_id','=','ha.habitacion_id')
    ->select('re.codigo_reserva', 'ha.numero_habitacion', 'p.razon_social_nombre', 'p.telefono', 'rev.aereolinea',
    'reh.persona_id')
    ->groupBy('reh.persona_id')
    ->groupBy('reha.reserva_estancia_habitacion_id')
    /* ->where([
    |   ['re.fecha_checkin','!=',null]
    ]) */
    ->whereDay('re.fecha_salida',date("d"))
    ->orderBy('re.fecha_salida','desc')
    ->get();

    for($i=0;$i<count($detalle);$i++){
        $detalle[$i]->pasajeroRZ=DB::table('persona')
        ->where('persona_id',$detalle[$i]->persona_id)
        ->get()[0]->razon_social_nombre;
    }
    $fpdf= new FpdfPerson();
    $fpdf->AddPage();

    $fpdf->SetFont('Courier', 'B', 14);
    $textypos=8;
    $fpdf->setXY(5,$textypos);

    $fpdf->Cell(0,0,"SALIDAS DEL DIA",0,1,"C");
}
```

Fuente: Elaboración propia

Figura 147 Ruta de reportes

```
Route::get('/reportearrivediario', 'ReservaEstanciaController@get_reporte_diario_arribos');
Route::get('/reportesalidadiario', 'ReservaEstanciaController@get_reporte_diario_salidas');
```

Fuente: Elaboración propia

3.5.2.2. Frontend

3.5.2.2.1. T2. Mostrar en un modal el pdf de reporte de salidas del día y T3. Mostrar en un modal el pdf de reporte de parte diario donde se listan todos los arribos del día

En la figura 148 se muestra el fragmento de código donde se agregaron los botones de los reportes de parte diario y salidas.

Figura 148 Fragmento de código de la vista de grid

```
<div className="row">
  <div className="col-12">
    <button
      type="button"
      onClick={() => {
        | this.setState({ dailyReportModal: true });
      }}
      className="btn btn-warning mr-2"
    >
      Parte diario
    </button>
    <button
      type="button"
      className="btn btn-light"
      onClick={() => {
        | this.setState({ exitReportModal: true });
      }}
    >
      Salidas
    </button>
  </div>
</div>
```

Fuente: Elaboración propia

En los modales se muestra una etiqueta iFrame en cual realiza el consumo de los endpoints como se muestra en las figuras 149 y 150.

Figura 149 Fragmento de código del modal de reporte de salidas del día

```
{ ...props }
size="xl"
aria-labelledby="contained-modal-title-vcenter"
centered
>
<Modal.Header closeButton>
  <Modal.Title id="contained-modal-title-vcenter">
    <i className="fas fa-door-open mr-2"/></i>Salidas del día
  </Modal.Title>
</Modal.Header>
<Modal.Body>
  <div className="col-12">
    <div className="col-12">
      {
        (props.show)?
          <div width="100%" height="800px" className="position-relative">
            <Loader show={frameLoading} center={true} />
            <iframe width="100%" onLoad={()=>{setFrameLoading(false)}} height="100%" />
          </div>
        :
        </>
      }
    </div>
  </div>
</Modal.Body>
```

Fuente: Elaboración propia

Figura 150 Fragmento de código de arrivals del dia

```
<Modal
  { ...props }
  size="xl"
  aria-labelledby="contained-modal-title-vcenter"
  centered
>
<Modal.Header closeButton>
  <Modal.Title id="contained-modal-title-vcenter">
    <i className="fas fa-plane-departure mr-2"/></i>Arrivados del día
  </Modal.Title>
</Modal.Header>
<Modal.Body>
  <div className="col-12">
    <div className="col-12">
      {
        (props.show)?
          <div width="100%" height="800px" className="position-relative">
            <Loader show={frameLoading} center={true} />
            <iframe width="100%" onLoad={()=>{setFrameLoading(false)}} height="100%" />
          </div>
        :
        </>
      }
    </div>
  </div>
</Modal.Body>
```

Fuente: Elaboración propia

3.5.3. Resultados

Se concluyó con las tareas y se logró el objetivo como resultado, se construyeron las apis para los reportes y se construyó las siguientes vistas de los reportes como se muestra en las figuras 145 y 146.

Figura 151 Modal de reporte de arribos del día

The screenshot shows a modal window titled "Arribos del día". The header bar includes a back arrow, the title, and standard window controls. Below the header is a toolbar with icons for download, print, and other functions. The main content area is a table titled "ARRIVOS DEL DIA". The table has five columns: "Código Reserva", "Habitación", "Contacto", "Vuelos", and "Pasajeros". The data in the table is as follows:

Código Reserva	Habitación	Contacto	Vuelos	Pasajeros
E40900TV4	201-F	JOSE FERNANDO VILCA MENESSES		JOSE FERNANDO VILCA MENESSES
V6YBUX7GP	202-I	NEY RUIZ ISUIZA		JOSE FERNANDO VILCA MENESSES
V6YBUX7GP	202-I	NEY RUIZ ISUIZA		NEY RUIZ ISUIZA
E40900TV4	201-F	JOSE FERNANDO VILCA MENESSES		NEY RUIZ ISUIZA

Fuente: Elaboración propia

Figura 152 Modal de reporte de salidas del día

The screenshot shows a modal window titled "Salidas del día". The header bar includes a back arrow, the title, and standard window controls. Below the header is a toolbar with icons for download, print, and other functions. The main content area is a table titled "SALIDAS DEL DIA". The table has five columns: "Código Reserva", "Habitación", "Contacto", "Vuelos", and "Pasajeros". The data in the table is as follows:

Código Reserva	Habitación	Contacto	Vuelos	Pasajeros
E40900TV4	201-F	JOSE FERNANDO VILCA MENESSES		JOSE FERNANDO VILCA MENESSES
V6YBUX7GP	202-I	NEY RUIZ ISUIZA		JOSE FERNANDO VILCA MENESSES
V6YBUX7GP	202-I	NEY RUIZ ISUIZA		NEY RUIZ ISUIZA
E40900TV4	201-F	JOSE FERNANDO VILCA MENESSES		NEY RUIZ ISUIZA

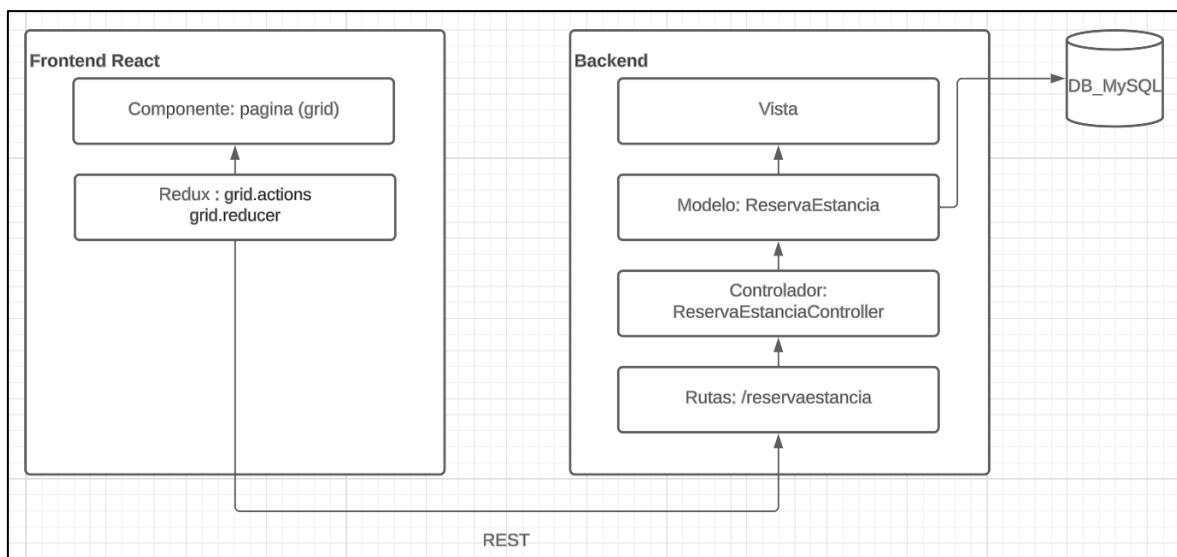
Fuente: Elaboración propia

3.6. Construir vista pública para el registro de huéspedes (Booking)

La vista de registro de huéspedes será publica para el registro de los huéspedes de una reserva, se registrarán los nombres, nacionalidad, documento de identidad, numero de documento de identidad, etc.

En la figura 143, se muestra de flujo de trabajo de los reportes a implementar.

Figura 153 Flujo de trabajo del submódulo de grid



Fuente: Elaboración propia

3.6.1. Tareas

En la figura 154 se muestra el objetivo y las tareas para implementar la nueva vista de registro de huéspedes.

Figura 154 Objetivo: Construcción de vista pública para registro de huéspedes

O. Construcción de vista pública para el registro de huéspedes
T.1 Construcción de vista pública para el registro de huéspedes con las siguientes pautas
.Se debe pasar el id de reserva por url para mostrar la información de reserva
.Si no existe id de reserva muestra un texto informativo
.Mostrar los datos generales de la reserva "fecha, días, usuario que la registró"
adultos y niños mostrar los card donde se introducirán los datos de los huéspedes
.Debajo de mostrar los datos generales, de acuerdo con el número de huéspedes
sistema si el huésped es contacto solo mostrar la información del huésped los demás se tendrán que validar
.Al listar los huéspedes se deben cargar todos los huéspedes registrados desde la grilla de booking
.Los huéspedes que no son contacto deben registrar su "nacionalidad, nombre, edad"
y tener un botón de registrar, luego de registrar los datos de los huéspedes podrán ser editados.
.Si la reserva tiene 3 huéspedes y en la primera visita solo se registraron dos, la página debe cargar los huéspedes ya registrados y permitir registrar el que falta
.Agregar un botón luego de los datos de los huéspedes con el texto "Registrar reserva"
el cual guardará todos los datos de la estancia y solo permitirá ver la información de la estancia y los huéspedes para futuras visitas

Fuente: Elaboración propia

3.6.2. Ejecución de tareas

En este caso los endpoint ya fueron creados y consumidos desde los actions del grid, también ya fue creada la vista pública de registro de huéspedes, lo que faltaría para concluir con el objetivo es toda la lógica del frontend, el cual se tiene que realizar la maquetación de la vista para mostrar y registrar los datos de los huéspedes.

Las tareas solo se realizaron en el frontend.

3.6.2.1. Frontend

3.6.2.1.1. T.1 Construcción de vista pública para el registro de huéspedes

En la figura 155 se muestra el fragmento de código donde se recupera el id de la reserva de la url y se pasa a la función "getResumeByBooking" el cual consulta con el endpoint del backend para traer los datos.

Figura 155 Fragmento de código de la consulta de la reserva

```
const Booking = (props) => {
  const [huespedes, setHuespedes] = useState([]);
  useEffect(() => {
    if (!props.match.params.reserva_estancia_id) {
      return;
    }
    props.getResumeByBooking(props.match.params.reserva_estancia_id);
    props.getAllTypeDocuments();
    //return props.getResumeByBooking(null);
  }, []);
  useEffect(() => {
```

Fuente: Elaboración propia

En la figura 156 se muestra el tratamiento de los huéspedes, luego se realiza un recorrido de esta lista como se muestra en la figura 157 desde para pasar los al componente “GuestRegister” el cual será en el cargado de mostrar los datos de los huéspedes en un card y del registro de los datos como se muestra en la figura 158.

Figura 156 Fragmento de código de conteo de huéspedes

```
useEffect(() => {
  if (props.resume && props.resume.resumen.length > 0) [
    var huedes = props.resume.huespedes.filter(
      (h) => parseInt(h.menor) === 0
    );
    var huedesMenos = props.resume.huespedes.filter(
      (h) => parseInt(h.menor) === 1
    );
    let tHuespedes = [];
    for (let i = 0; i < props.resume.resumen[0].numero_adultos; i++) {
      tHuespedes.push({
        key: i,
        isChildren: false,
        data: huedes[i] ? huedes[i] : null,
      });
    }
    let index = tHuespedes.length;
    for (let i = 0; i < props.resume.resumen[0].numero_ninos; i++) {
      tHuespedes.push({
        key: index,
        isChildren: true,
        data: huedesMenos[i] ? huedesMenos[i] : null,
      });
      index++;
    }
    console.log(tHuespedes);
    setHuespedes(tHuespedes);
  ]
}, [props.resume]);
```

Fuente: Elaboración propia

Figura 157 Fragmento de código de la asignación de los datos de los huéspedes a GuestRegister

```
        </Row>
        {huespedes.map((huesped) => {
            return (
                <GuestRegister
                    key={"huesped" + huesped.key}
                    isChildren={huesped.isChildren}
                    data={huesped.data}
                    reserva_estancia_id={
                        props.resume.resumen[0].reserva_estancia_id
                    }
                />
            );
        })
    <Row className="d-flex justify-content-end mb-3">
        {parseInt(props.resume.resumen[0].estado_registro) === 1 ? (
```

Fuente: Elaboración propia

Figura 158 Fragmento de código del componente GuestRegister

```
<form onSubmit={handleSubmit} id={props.key}>
  <div className="card-header">
    {props.isChildren ? "Registrar menor" : "Registrar huesped"}
  </div>
  <div className="card-body">
    <Row>
      <Col sm={12}>
        <Row>
          <label for="staticEmail" className="col-sm-3 col-form-label">
            Nacionalidad:
          </label>
          <div className="col-sm-9">
            <input
              type="text"
              className="form-control"
              id="staticEmail"
              name="nacionalidad"
              onChange={handleInputChange}
              value={values.nacionalidad}
              disabled={
                !props.data || parseInt(props.data.estado_registro) === 0
                ? false
                : true
              }
            />
          </div>
        </Row>
      </Col>
    </Row>
    <Row className="mt-3">
      <Col sm={6}>
        <Row>
          <label for="staticEmail" className="col-sm-3 col-form-label">
            Tipo documento:
          </label>
```

Fuente: Elaboración propia

3.6.3. Resultados

Se concluyó con las tareas y se logró el objetivo como resultado, se construyeron las apis para los reportes y se construyó las siguientes vistas de los reportes como se muestra en las figuras 159, 160, 161 y 162.

Figura 159 Vista publica de booking

Tampu - ACCORD Software

localhost:3000/booking/6

Gmail YouTube Maps Noticias react-select-search... Alerts - Bootstrap v... RGB Color Codes C... Best JSON Format... 192.168.1.40

Incógnito

empu Registro de huéspedes

Contacto: HMF INVERSIONES S.A.C. RUC/DNI: 20600306970

Fecha de llegada: 2023-06-15 Fecha de salida: 2023-06-15 Habitación: 201-F INDIVIDUAL

Deseo recibir información de ofertas y promociones

Registrar huésped

Nacionalidad:

Tipo documento: DNI N. Documento: 73141097

Nombres: JOSE FERNANDO Apellidos: VILCA MENESSES

Teléfono: Correo electrónico: iomarigor@gmail.com

Guardar

Fuente: Elaboración propia

Figura 160 Componente GuestRegister renderizado

Deseo recibir información de ofertas y promociones

Registrar huésped

Nacionalidad:

Tipo documento: DNI N. Documento: 73141097

Nombres: JOSE FERNANDO Apellidos: VILCA MENESSES

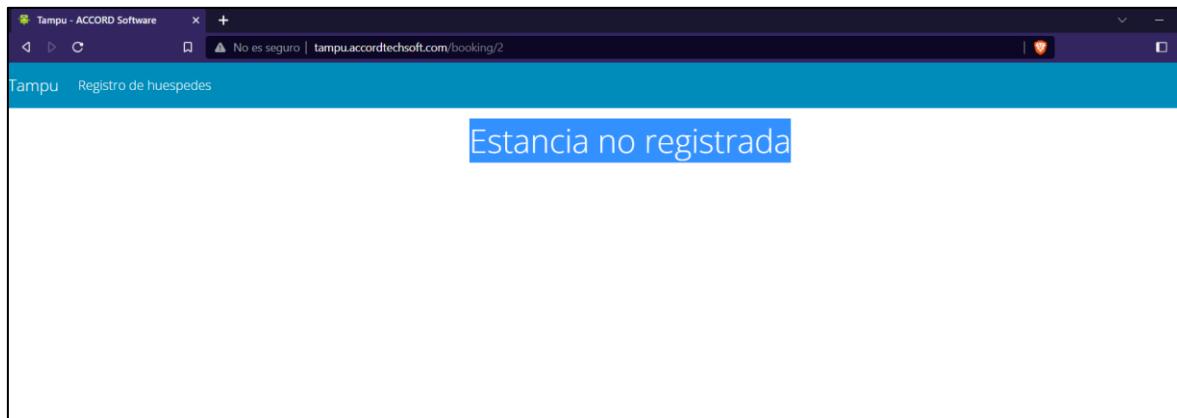
Teléfono: Correo electrónico: iomarigor@gmail.com

Guardar

Registrar huésped

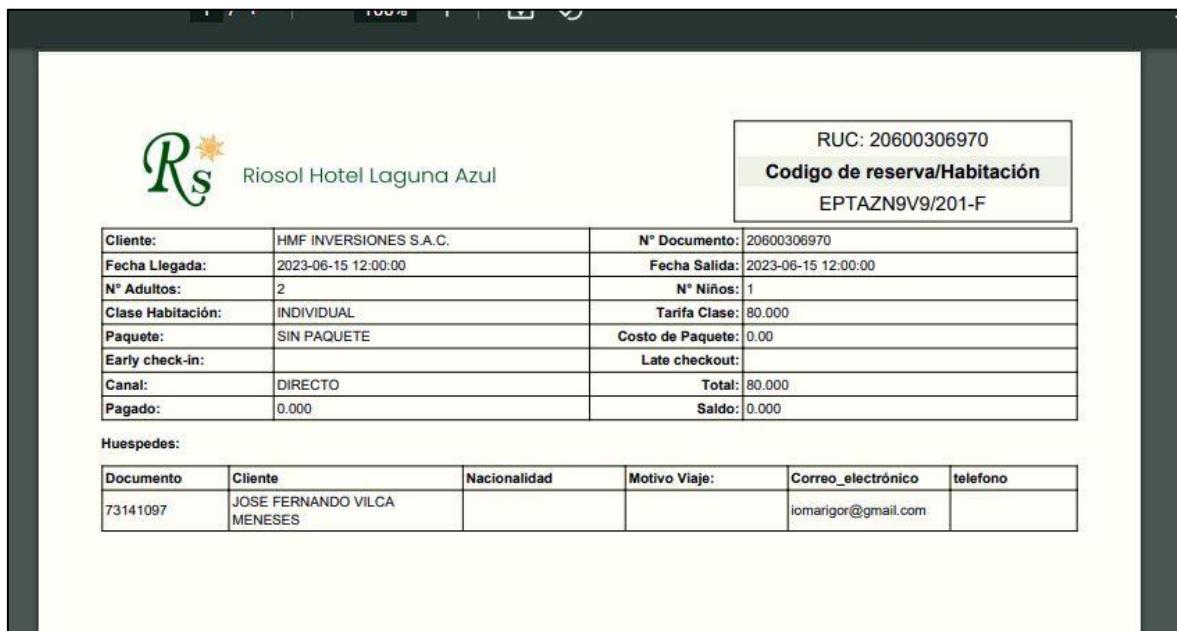
Fuente: Elaboración propia

Figura 161 Vista cuando no se encuentra la reserva solicitada



Fuente: Elaboración propia

Figura 162 Comprobante que se genera luego de terminar el registro de huéspedes



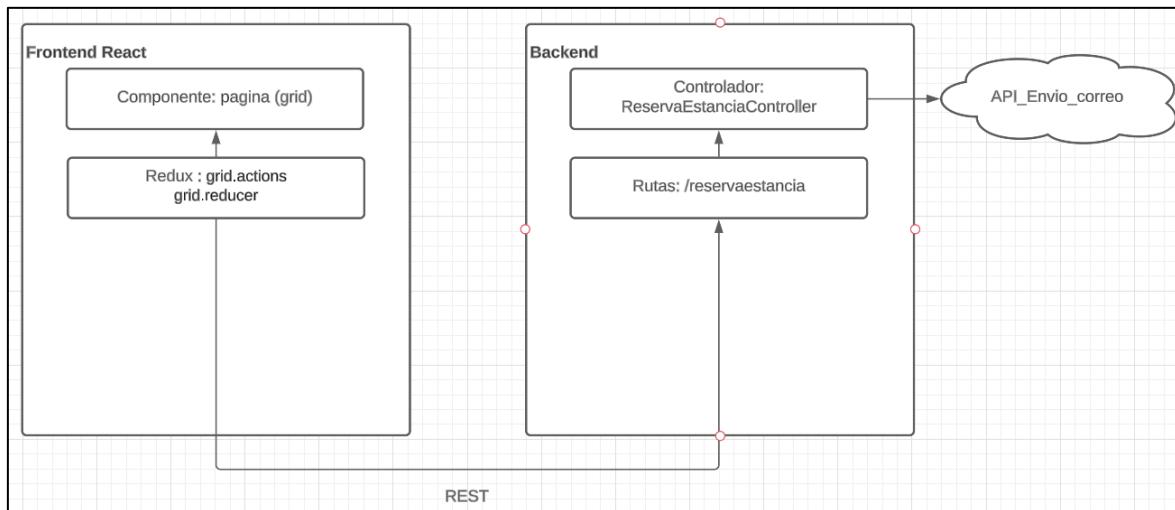
Fuente: Elaboración propia

3.7. Implementar envío de correo electrónico del enlace de registro de información de reserva (Booking)

Implementar el servicio smtp para el envío de correo electrónico con la finalidad de enviar el link donde podre registrar los huéspedes que restan. En el correo se enviará los datos de la empresa una pequeña explicación y la url de la vista pública para el registro de los huéspedes.

En la figura 163, se muestra de flujo de trabajo para el envío de correo.

Figura 163 Flujo de trabajo para el envío de correo



Fuente: Elaboración propia

3.7.1. Tareas

En la figura 164 se muestra el objetivo y las tareas para implementar un endpoint que permita el envío de correo.

Figura 164 Objetivo: creación de endpoint para envío de correo

0. Implementar envío de correo electrónico
T1 Implementar servidor smtp
T2 Crear endpoint para el envio .El correo debe mostrar la imagen del hotel, una breve descripción de contacto y el link de registro de huéspedes
T3. Cuando el estado de la habitación esté en reserva agregar el botón enviar el correo .Crear modal con un input donde se cargará automáticamente el correo del usuario contacto pero se podrá cambiar el correo para enviar a otro

Fuente: Elaboración propia

3.7.2. Ejecución de tareas

Las tareas se dividieron en backend y frontend.

3.7.2.1. Backend

3.7.2.1.1. T1 Implementar servidor smtp y T2 crear endpoint para envío de correo

Este objetivo se los parcialmente ya que se intentó implementar el servidor smtp del proveedor de hosting, desde servidor local funcionaba sin problema, los correos llegaban a Gmail y Hotmail, pero al probar desplegar el backend en el servidor de producción solo llegaban los correos a Hotmail, se realizó la consulta y no se obtuvo respuesta, se dedujo que el proveedor tenía bloqueado los puertos para que los correos se envíen a gmail, así que se para el envío de correo se utilizó “smtp2go” que es otro proveedor de servicio smtp, se utilizó en api de smtp2go para el envío de correo.

Figura 165 Endpoints de envío de correo

```
//Envío de correo de registro
Route::post('/sendreserva', 'ReservaEstanciaController@get_send_reserva');
Route::post('/sendreserva_smtp2go_api', 'ReservaEstanciaController@get_send_reserva_smtp2go_api');
```

Fuente: Elaboración propia

En la figura 165 se muestra los endpoint que se construyeron para el envío del correo uno usando el servidor smtp del proveedor y otro con el servicio de smtp2go.

En la figura 166 se muestra el fragmento de código para el envío de correo electrónico con el servidor smtp del proveedor.

Figura 166 Fragmento de código envío correo

```
public function get_send_reserva(Request $request){
    try {
        $data= array(
            "to"=>$request->input('to'),
            "urlSend"=>$request->input('urlSend'),
            "nameUser"=>$request->input('nameUser'),
            "detalle"=>$request->input('detalle'),
            "subject"=>$request->input('subject')
        );
        Mail::to($data['to'])->queue(new RegistroHuespedMail($data));
        $json = array(
            "status"=>200,
            "mensaje"=>"Correo enviado al ".$data['to'],
            "detalles"=>$data
        );
    }
}
```

Fuente: Elaboración propia

En la figura 167 se muestra el fragmento de código de la función para el envío de correo electrónico utilizando el api de smtp2go.

Figura 167 Fragmento de código del envío de correo usando api de smtp2go

```
public function get_send_reserva_smtp2go_api(Request $request){
    try {
        $data= array(
            "to"=>$request->input('to'),
            "urlSend"=>$request->input('urlSend'),
            "nameUser"=>$request->input('nameUser'),
            "detalle"=>$request->input('detalle'),
            "subject"=>$request->input('subject')
        );
        $message = view('emails.EmailReserva', [ 'data' => $data])->render();
        $data=array(
            "api_key"=>env('KEY_SMTP2GO'),
            "to"=>[$request->input('to')],
            "sender"=>env('MAIL_FROM_ADDRESS'),
            "subject"=>$request->input('subject'),
            "html_body"=>$message
        );
        //return json_encode($data);
        $client = new Client([
    
```

Fuente: Elaboración propia

3.7.2.2. Frontend

3.7.2.2.1. T3 Cuando el estado de la habitación este en reserva agregar el botón enviar el correo

Se agrego el botón de envío de encomienda que esta enlazado con el modal para ingresar el correo como se muestra en la figura 168.

Figura 168 Fragmento de código que se agregó a la vista grid

```
{parseInt(eventItem.booking.tipo_reserva_estancia) === 2 && (
  <Button
    className="rounded-0 mx-1 mb-1"
    variant="outline-info"
    size="sm"
    onClick={() => {
      this.setState({
        bookRoom: eventItem,
        mailModal: true,
      });
    }}
    data-toggle="tooltip"
    data-placement="right"
    title="Enviar mail"
  >
    <i className="far fa-envelope"></i>
  </Button>
)}
```

Fuente: Elaboración propia

En la figura 169 se muestra el fragmento de código para el consumo del endpoint de envío de encomiendas.

Figura 169 Fragmento de código para el consumo del api para el envío del correo

```
export function sendMailRegisterEstancia(form) {
  toast.info("Enviando mail...", {
    toastId: "sendMailRegisterEstancia",
    autoClose: 20000,
  });
  const request = axios.post(
    `${process.env.REACT_APP_API_URL}/api/sendreserva_smtp2go_api`,
    form
  );

  return (dispatch) =>
    request
      .then((response) => {
        console.log(response.data);
        if (parseInt(response.data.status) === 404) { ... }
      })

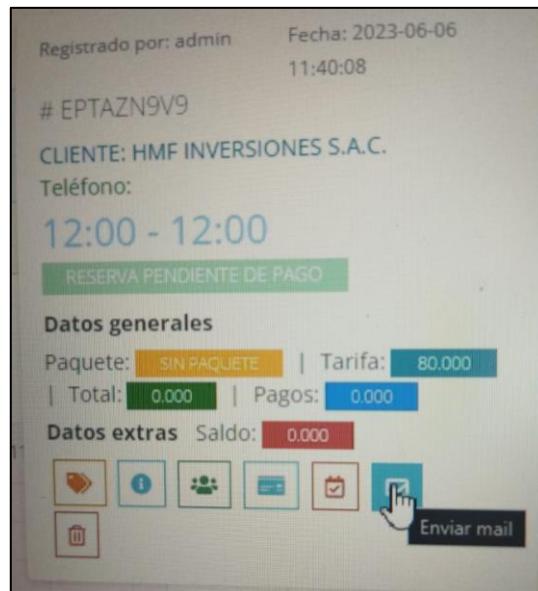
      .catch((error) => {
        toast.error("Error al enviar el correo");
        dispatch(loaderSendMail(false));
      })
      .finally(() => {
        toast.update("sendMailRegisterEstancia", {
          render: response.data.mensaje,
          type: parseInt(response.data.status) == 200 ? "success" : "error",
          isLoading: false,
          autoClose: 2000,
        });
        return dispatch(loaderSendMail(false));
      });
}
```

Fuente: Elaboración propia

3.7.3. Resultados

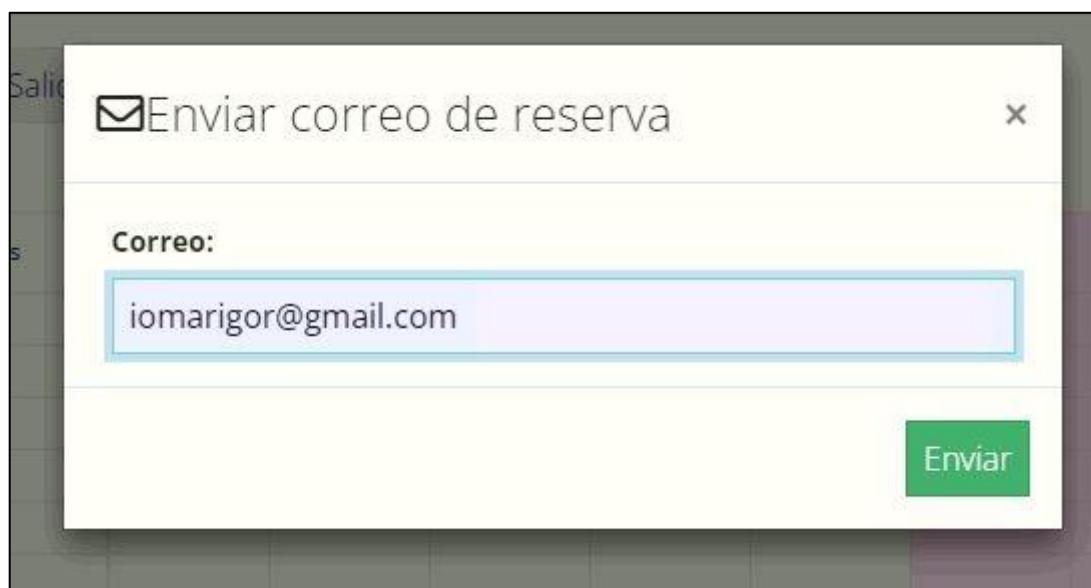
Se concluyó con las tareas y se logró el objetivo como resultado, se construyeron la api para el envío de correo y las vistas como se muestra en las figuras 170, 171 y 172.

Figura 170 Modal de reserva registrada



Fuente: Elaboración propia

Figura 171 Modal de envío de encomienda



Fuente: Elaboración propia

Figura 172 Correo electrónico que llega al correo



Fuente: Elaboración propia

3.8. Desplegar y depurar proyectos en servidor de producción (Booking y Trade)

Se realizo distintas actividades con la finalidad de desplegar los servicios en subdominios de la empresa en el hosting de GoDaddy atreves del cpanel, configurar la base de datos, subir los archivos del frontend y el backend.

3.8.1. Despliegue

3.8.1.1. Tareas

En la figura 173 se muestra el objetivo y las tareas para desplegar los proyectos de Trade y Booking.

Figura 173 Objetivo: Desplegar proyectos

O. Desplegar los proyectos Trade y Tampu
T1. Desplegar proyector de Trade en el subdominio de lamastours
T2. Desplegar el proyecto de Booking en el subdominio de Tampu

Fuente: Elaboración propia

3.8.1.2. Ejecución de tareas

3.8.1.2.1. T1 Desplegar proyectos de Trade en el subdominio “lamastours” y Booking en el de “Tampu”

El proceso de despliegue de ambos proyectos en la misma así que solo me mostrará el despliegue de uno de ellos.

En la figura 174 se muestra la creación de las bases de datos en el servidor de producción, por razones obvias no se mostrarán captura de la conexión a la base de datos mediante HeidiSQL así que las imágenes de las figuras 175 y 176 son referenciales, para la conexión a la base de datos se ponen las credenciales y se procede con la importación de las bases de datos.

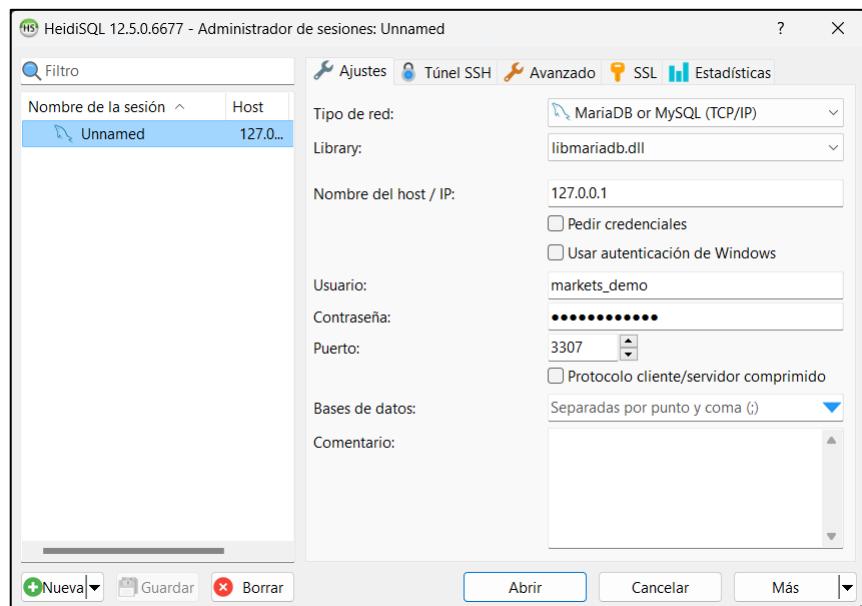
Figura 174 Creación de las DB

Base de datos	Tamaño	Usuarios con privilegios
bipos_trade	10,25 MB	[trash]
chazarp	34,71 MB	[trash]
i8561723_wp1	72,21 MB	[trash]
lamas_tours	23,57 MB	[trash]
tampu_booker	28,12 MB	[trash]

Fuente: Elaboración propia

En la figura 175 se muestra la conexión remota a la base de producción, se ingresan las credenciales para acceder a esta.

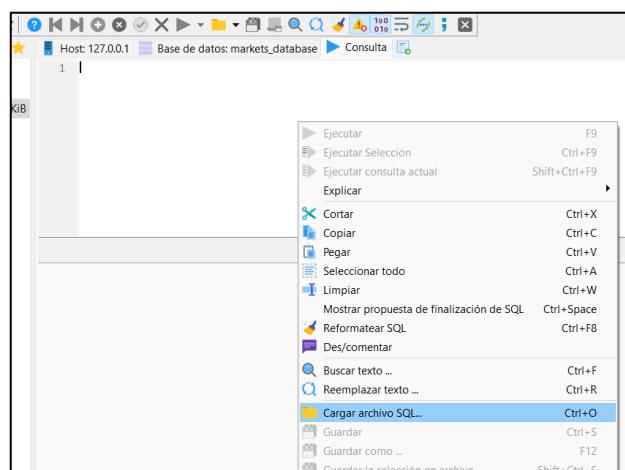
Figura 175 Conexión a la base de datos de manera remota



Fuente: Elaboración propia

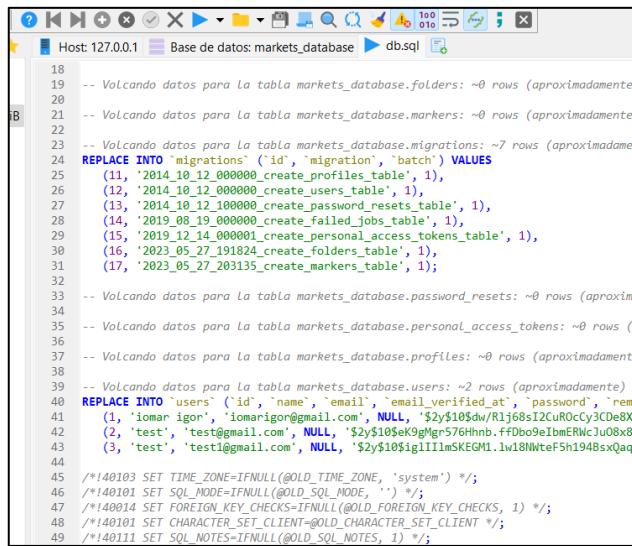
En las figuras 176 y 177 muestra como importar las bases de datos.

Figura 176 Importación de base de datos



Fuente: Elaboración propia

Figura 177 Ejecución de las consultas para la creación de las bases de datos

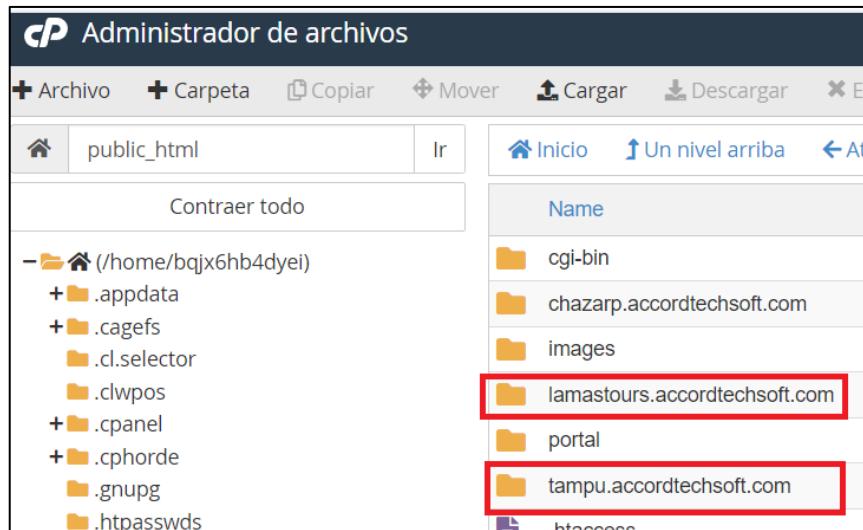


```
Host: 127.0.0.1 Base de datos: markets_database db.sql
18 -- Volcando datos para la tabla markets_database.folders: ~0 rows (aproximadamente)
19
20 -- Volcando datos para la tabla markets_database.markers: ~0 rows (aproximadamente)
21
22 -- Volcando datos para la tabla markets_database.migrations: ~7 rows (aproximadamente)
23
24 REPLACE INTO `migrations` (`id`, `migration`, `batch`) VALUES
25     ('11', '2014_10_12_000001_create_profiles_table', 1),
26     ('12', '2014_10_12_000002_create_users_table', 1),
27     ('13', '2014_10_12_100001_create_password_resets_table', 1),
28     ('14', '2019_08_19_000001_create_failed_jobs_table', 1),
29     ('15', '2019_12_14_000001_create_personal_access_tokens_table', 1),
30     ('16', '2023_05_27_191824_create_folders_table', 1),
31     ('17', '2023_05_27_203135_create_markers_table', 1);
32
33 -- Volcando datos para la tabla markets_database.password_resets: ~0 rows (aproximadamente)
34
35 -- Volcando datos para la tabla markets_database.personal_access_tokens: ~0 rows (aproximadamente)
36
37 -- Volcando datos para la tabla markets_database.profiles: ~0 rows (aproximadamente)
38
39 -- Volcando datos para la tabla markets_database.users: ~2 rows (aproximadamente)
40 REPLACE INTO `users` (`id`, `name`, `email`, `email_verified_at`, `password`, `remember_token`)
41     ('1', 'iomar igor', 'iomarigorg@gmail.com', NULL, '$2y$10$dw/Rlj68s1ZCuR0CcY3CDe8X0',
42     ('2', 'test', 'test@gmail.com', NULL, '$2y$10$K9ghgr576hnb.fffDbo9eIbmRWcJu08x8q',
43     ('3', 'test', 'test@gmail.com', NULL, '$2y$10$ig1l1ImSKEGM1.lw18lNteF5h194BsxAq
44
45 /*!40103 SET TIME_ZONE=IFNULL(@OLD_TIME_ZONE, 'system') */;
46 /*!40101 SET SQL_MODE=IFNULL(@OLD_SQL_MODE, '') */;
47 /*!40014 SET FOREIGN_KEY_CHECKS=IFNULL(@OLD_FOREIGN_KEY_CHECKS, 1) */;
48 /*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
49 /*!40111 SET SQL_NOTES=IFNULL(@OLD_SQL_NOTES, 1) */;
```

Fuente: Elaboración propia

En la figura 178 se muestra el directorio de subdominios para ambos proyectos, en la figura 179 se muestra como se subieron los archivos y ya tendríamos los proyectos desplegados.

Figura 178 Directorio de subdominios



Fuente: Elaboración propia

Figura 179 Subida de archivos

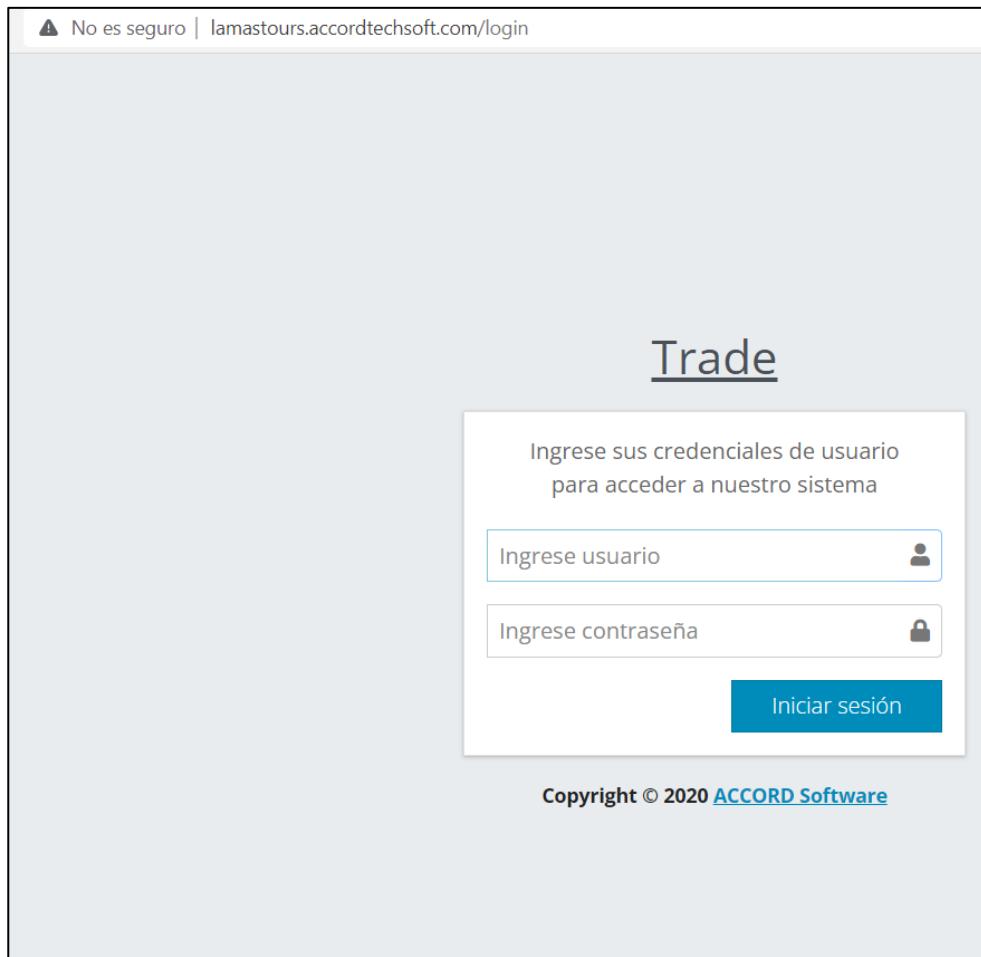


Fuente: Elaboración propia

3.8.1.3. Resultado

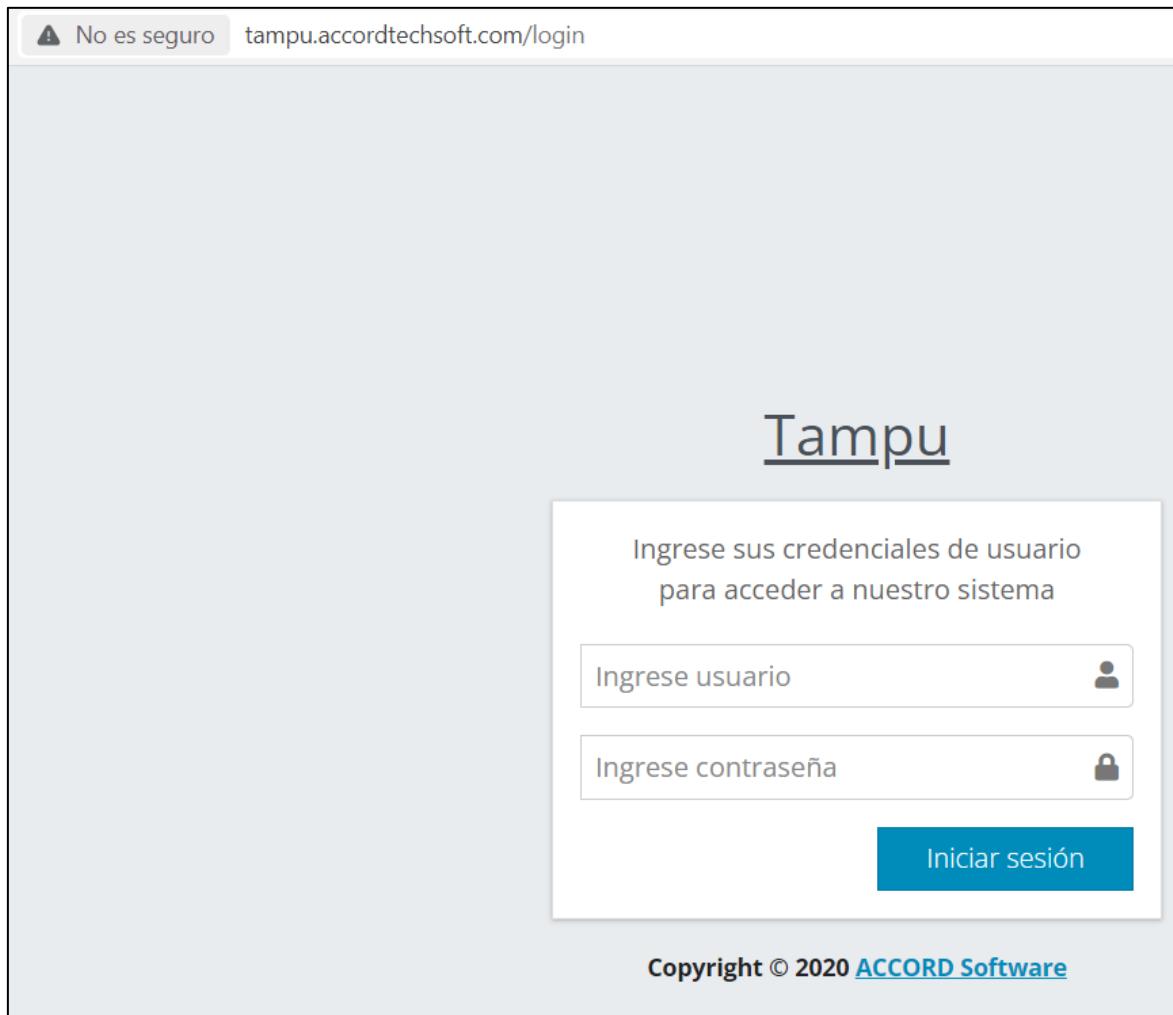
Se concluyeron las tareas y se logró el objetivo de desplegar los proyectos, en las figuras 180 y 181 se muestran el inicio de sesión de los proyectos desplegados.

Figura 180 Proyecto Trade desplegado



Fuente: Elaboración propia

Figura 181 Proyecto de Booking (Tampu) desplegado



Fuente: Elaboración propia

3.8.2. Depuración

Luego de desplegar los proyectos, se realizaron pruebas de caso de uso y se descubrieron muchos errores causados por no formatear los tipos de datos de texto a enteros y decimales, el servidor al estar desplegado en un entorno Linux nos retorna los datos en texto, en la figura 182 y figura 183 se muestran los casos de usos aplicados a cada sistema.

Figura 182 Casos de uso Trade

Trade	
Registrar un nuevo conductor, agregar dos encomiendas a la oficina de tarapoto conductor, realizar un adelanto	validar que las encomiendas se muestren en los usuarios de la oficina de tarapoto, validar que se actualizan los estados de las encomiendas, validar el pdf de impresión, que los nombres, fecha y monto sean los correctos
Realizar un registro desde la sucursal lamas, confirmar la recepcion desde la oficina de tarapoto,	realizar la entrega de encomienda, validar el estado en la vista publica, validar el pdf de impresión, que los nombres, fecha y monto sean los correctos
Validar que los correlativos de las salidas y adelantos se reinicie el contador cada dia	
Verificar que los reportes de dashboard muestren corectamente los datos.	
Facturar un servicio de garaje y otro con (garaje y venta de pasaje), añadir una verde descripcion a la venta de pasaje,	verificar si ma impresión de la factura es correcto y que se muestre el detalle adicional

Fuente: HMF Inversiones

Figura 183 Casos de uso Booking

Booking	
realizar una reserva como persona natural con 2 adultos y 1 niño, enviar correo de reserva al correo "iomar.alegre@unas.edu.pe", validar si llega el correo a hotmail	Como usuario contacto, registrar el niño en la vista publica, actualizar pagina, registrar el segundo adulto y registrar reserva
realizar una reserva como empresa con 2 adultos, enviar correo de reserva al correo "iomar.alegre@unas.edu.pe", validar si llega el correo a hotmail	Como usuario contacto, registrar el niño en la vista publica, actualizar pagina, registrar el segundo adulto y registrar reserva
realizar una reserva con 2 adultos y 1 niño, enviar correo de reserva al correo "iomarigor@gmail.com", validar si llega el correo a gmail	
Registrar dos estancias, y dos reservas con fecha de hoy, hacer el check-out de una estancia, validar los reportes de parte diario y de salidas	

Fuente: HMF inversiones

3.8.2.1. Tareas

En la figura 184 se muestra el objetivo y las tareas para depurar los errores producidos en los servidores de producción.

Figura 184 Objetivo: Depurar errores

O. Depuración de errores en produccion
T1. Identificar y parsear los datos entero y decimales en el proyecto de trade y booking

Fuente: Elaboración propia

3.8.2.2. Ejecución de tareas

3.8.2.2.1. Identificar y formatear los datos enteros y decimales

Con la ayuda de una herramienta del ide ve visual studio code como se muestra en la figura 185, se identificó los datos realizando un filtrado de todos los archivos del proyecto que tengan los operadores “==” sirve para comparar si un dato es del mismo tipo y valor del que lo compara y “!=” sirve para comparar si un dato es de diferente tipo y valor del que lo compara.

De forma manual se fueron formateando los datos como se muestra en la figura 186.

Figura 185 Identificación de los datos usando la herramienta del ide

The screenshot shows a developer's workspace with several open tabs and panels. The main area displays a component named `BranchTable.js` from the `src\components\pages\estab...` file. The code uses conditional logic based on the value of `thisRow.estado`, which is checked against the value 1. It includes imports for `parseInt` and `PageTransitionEvent`. The code is part of a larger application structure involving components like `ClientsProviders.js`, `AddClientsProvidersModal.js`, and `Search.js`.

SEARCH

Replace

20000 results in 239 files - Open in editor

values.imagen_comprobante === null) {
tValues.porcentaje_ivg === "" ? 0 : tValues.porcen...
if (e.target.type === "checkbox") {
if (e.target.type === "text") {
if (e.target.type === "select-one") {
if (e.target.type === "number") {
if (e.target.type === "email") {
if (e.target.name === "img_logo_small") {
if (e.target.name === "img_logo_large") {
if (parseInt(thisRow.estado) === 1) {
if (params.row.estado === 1) {
&& props.establecimientoId === "") {
values.codigo_anexo.length === 0) {
tipo_establecimiento.length === 0) {
if (values.sucursal.length === 0) {
if (e.target.type === "checkbox") {
checked = checked === true ? 1 : 0;
if (e.target.type === "text") {
if (e.target.type === "select-one") {
if (e.target.type === "number") {
if (e.target.type === "email") {
if(e.target.type ==='checkbox'){
src > components > pages > establishments > tables > BranchTable.js
flex: 1,
},
field: "estado",
headerName: "Estado",
headerAlign: "center",
disableClickEventBubbling: true,
width: 110,
renderCell: (params) => {
const onClick = () => {
//cambia estado del establecimiento
const thisRow = params.row;
if (parseInt(thisRow.estado) === 1) {
thisRow.es = parseInt(function par...
} else {
@PageTransitionEvent
thisRow.estado = 1;
}
props.updateStatusBranch(thisRow);
};
if (params.row.estado === 1) {
return (
<Button
variant="success"
onClick={onClick}
size="sm"
serve -s build
Find out more about deployment here:
bit.ly/CRA-deploy
PS C:\AplicacionesWeb\tradeFront>

Fuente: Elaboración propia

Figura 186 Formateo de forma manual de los datos

The screenshot shows a code editor with several tabs at the top: 'AddClientsProvidersModal.js', 'SearchVoucherModal.js', 'ListCr...', and 'ExpensesReceipt.js'. The 'ExpensesReceipt.js' tab is active. The code is written in JavaScript and uses JSX syntax. A specific section of the code is highlighted with a yellow background, indicating the area being formatted. The code snippet is as follows:

```
        neoername: 'ACCIONES',
        headerAlign: 'center',
        width: 110,
        disableClickEventBuddling: true,
        renderCell: (params) => {
            //mostrar informacion en modal
            const onClickInfo = () => {
                const thisRow = params.row;
                setSelectReceipt(thisRow);
                showExpensesReceiptModal();
            }
            if (parseInt(params.row.estado) === 0) {
                return <></>;
            } else {
                return (
                    <>
                        <Button
                            onClick={onClickInfo}
                            className="mr-xl-1"
                            variant="primary"
                            data-toggle="tooltip"
                            title="Ver Detalles"
                        >
                            Ver
                        </Button>
                    
```

Fuente: Elaboración propia

3.8.2.3. Resultados

Se realizó el formateo de todos los componentes de ambos proyectos, y se solucionaron los errores recurrentes que se producían al momento de consultar las vistas de las páginas del proyecto desplegado.

CONCLUSIONES

En conclusión, se logró cumplir con el objetivo general de construir nuevos módulos, desplegar y depurar los proyectos Booking y Trade. Se realizaron avances significativos en ambas áreas, lo que contribuirá a mejorar la funcionalidad y eficiencia de las aplicaciones.

En cuanto a los objetivos específicos, se concluye lo siguiente:

Se logró construir con éxito los submódulos de salida, adelantos y encomiendas en el proyecto Trade, lo que permitirá gestionar de manera eficiente estas áreas.

Se logró acoplar los reportes del submódulo dashboard en el proyecto Trade, lo que proporcionará información valiosa para la toma de decisiones.

Se construyó una vista pública para la consulta de encomiendas en el proyecto Trade, lo que facilitará a los usuarios acceder a esta información de manera sencilla.

Se implementaron nuevas funcionalidades al submódulo de facturación en el proyecto Trade, lo que mejorará el proceso de facturación y contribuirá a una mayor eficiencia.

Se implementaron reportes en el submódulo grid en el proyecto Booking, lo que brindará a los usuarios información visualmente atractiva y útil.

Se construyó una vista pública para el registro de huéspedes en el proyecto Booking, lo que simplificará el proceso de registro y mejorará la experiencia del usuario.

Se implementó el envío de correos electrónicos con el enlace de registro de huéspedes de las reservas en el proyecto Booking, lo que agilizará la comunicación con los huéspedes y mejorará la eficiencia en el proceso de registro.

Se logró desplegar y depurar los proyectos Booking y Trade en el servidor de producción, lo que permitirá su acceso y uso por parte de los usuarios finales de manera estable y segura.

En general, se ha logrado un avance significativo en los objetivos establecidos, lo que contribuirá a mejorar la funcionalidad y eficiencia de los proyectos Booking y Trade. Estas mejoras brindarán beneficios tanto a los usuarios como a la organización en términos de una mayor productividad y una mejor experiencia del usuario.

RECOMENDACIONES

Se recomienda implementar pruebas unitarias para mejorar la calidad del código y así minimizar los errores. Así como también construir los componentes y asignarle una sola responsabilidad siguiendo uno de los principios SOLID.

También se recomienda incorporar otro miembro en el equipo para ir desarrollando a la par los proyectos para luego realizar las pruebas de casos de uso uno al desarrollo del otro.

También se recomienda utilizar Docker para contener los proyectos del backend, si se hubiera tenido el proyecto en un contenedor con el sistema Linux se hubieran solucionado los errores de formateo antes que los proyectos se pasen a producción.

ANEXOS

Ahed Abugabah, & Louis Sanzogni. (n.d.). *enterprise-resource-planning-erp-system-in-higher-education-a-literature-review-and-implications*.

AnyDesk - Javatpoint. (n.d.). Retrieved May 26, 2023, from
<https://www.javatpoint.com/anydesk>

Eugenia Bahit. (n.d.). *POO y MVC en PHP*. Retrieved May 20, 2023, from
<http://eugeniamahit.blogspot.com>

Filipova, O., & Vilão, R. (2018). Requirements, Commitment, and Deadlines. *Software Development From A to Z*, 47–65. https://doi.org/10.1007/978-1-4842-3945-2_3

Geekflare. (n.d.). Retrieved June 1, 2023, from <https://geekflare.com/es/backend-solutions-for-web-and-mobile-apps/>

Gerasimov, A., Heuser, P., Ketteniß, H., Letmathe, P., Michael, J., Netz, L., Rumpe, B., & Varga, S. (2020). *Generated Enterprise Information Systems: MDSE for Maintainable Co-Development of Frontend and Backend*. www.se-rwth.de/publications/

GoDaddy PE. (n.d.). Retrieved May 26, 2023, from <https://www.godaddy.com/es>

Greenbank, P. (2001). Objective setting in the micro-business. *International Journal of Entrepreneurial Behaviour & Research*, 7(3), 108–127.
<https://doi.org/10.1108/EUM0000000005531>

Gruhn, V., & Striemer, R. (n.d.). *The Essence of Software Engineering*.

IEEE COMPUTE SOCIETY. (n.d.). *Guide to the Software Engineering Body of Knowledge SWEBOK ® A Project of the IEEE Computer Society*.

MANFRED BORTENSCHLAGER, & STEVEN WILLMOTT. (n.d.). *The API Owner's Manual | PDF | Business Model | Application Programming Interface*. Retrieved May 26, 2023, from <https://es.scribd.com/document/379816448/The-API-Owner-s-manual>

PHP: ¿Qué es PHP? - Manual. (n.d.). Retrieved May 26, 2023, from https://www.php.net.translate.goog/manual/en/intro-whatis.php?_x_tr_sl=auto&_x_tr_tl=es&_x_tr_pto=wapp&_x_tr_hl=uk

Quick Start – React. (n.d.). Retrieved May 26, 2023, from <https://react.dev/learn>

RIVERA, J. (n.d.). *Management y Liderazgo en Peter Drucker.*

Roger S. Pressman. (n.d.). *Id-Ingenieria.de.software.enfoque.practico.7ed.Pressman.*

SMTP2GO Software. (n.d.). Retrieved May 26, 2023, from

<https://www.softwareadvice.com/transactional-email/smtp2go-profile/>

The PHP Framework For Web Artisans. (n.d.). Retrieved May 26, 2023, from

<https://laravel.com/docs/10.x#why-laravel>

What Is cPanel? (n.d.). Retrieved May 26, 2023, from

<https://www.hostinger.com/tutorials/what-is-cpanel>

What Is MySQL? / Oracle. (n.d.). Retrieved May 26, 2023, from

<https://www.oracle.com/mysql/what-is-mysql/>

What is REST? / Codecademy. (n.d.). Retrieved May 26, 2023, from

<https://www.codecademy.com/article/what-is-rest>