



INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE CÓMPUTO

ESCOM

Trabajo terminal

“Aplicación móvil de comercio para servicios turísticos en zona arqueológica de Teotihuacán”

2020-B004

Presenta:

Aarón Antonio García González

Directores

Dra. Jazmín Ivette Jiménez Galán M. Ariel López Rojas



Diciembre, 2021



**INSTITUTO POLITÉCNICO NACIONAL
ESCUELA SUPERIOR DE CÓMPUTO
SUBDIRECCIÓN ACADÉMICA**



No. de TT: 2020-B004

Fecha de presentación de TT-II

Documento técnico

**“Aplicación móvil de comercio para servicios turísticos en zona arqueológica
de Teotihuacán”**

Presenta
Aarón Antonio García González

Directores

Dra. Jazmín Ivette Jiménez Galán M. Ariel López Rojas

RESUMEN

Palabras clave: Aplicación móvil, Código QR, Pagos electrónicos, Turismo.

aarongarcia.ipn.escom@gmail.com

Carta responsiva

Advertencia

“Este documento contiene información desarrollada por la Escuela Superior de Cómputo del Instituto Politécnico Nacional, a partir de datos y documentos con derecho de propiedad y por lo tanto, su uso quedará restringido a las aplicaciones que explícitamente se convengan.”

La aplicación no convenida exime a la escuela su responsabilidad técnica y da lugar a las consecuencias legales que para tal efecto se determinen.

Información adicional sobre este reporte técnico podrá obtenerse en:
La Subdirección Académica de la Escuela Superior de Cómputo del Instituto
Politécnico Nacional, situada en Av. Juan de Dios Bátiz s/n Teléfono:
57296000, extensión 52000.

Índice

Índice de tablas	10
Índice de figuras	11
1. Introducción.....	15
1.1. Resumen	15
1.2. Descripción del documento	15
1.3. Antecedentes.....	16
1.4. Justificación	18
1.5. Planteamiento del problema	19
1.6. Objetivos.....	20
1.6.1. Objetivo general	20
1.6.2. Objetivos particulares	20
1.7. Estado del arte	20
2. Marco conceptual	23
2.1. Ubicación geográfica y normatividad.....	23
2.1.1. Pueblos Mágicos.....	23
2.1.2. Pueblos con Encanto.....	24
2.1.3. Municipios vecinos.....	24
2.1.4. Aviso de privacidad.....	24
2.2. Pagos electrónicos	25
2.2.1. Ecommerce	25
2.2.2. Pasarela de pago	25
2.2.2.1. Razones principales para utilizar una pasarela de pago	25
2.2.2.2. Inconvenientes de las pasarelas de pago.....	26
2.2.3. Contra cargos.....	26
2.2.4. PayPal	26
2.2.4.1. Comisiones en PayPal	26
2.3. Tecnología	27
2.3.1. Arquitectura cliente servidor	27
2.3.2. Dispositivo móvil	27
2.3.3. Sistema operativo	27
2.3.4. IOS.....	27
2.3.5. Android.....	28
2.3.6. Javascript	29
2.3.7. Librerías JavaScript	29
2.3.8. CSS	30
2.3.9. HTML.....	30
2.3.10. Framework JavaScript	30
2.3.11. Aplicación móvil nativa.....	30
2.3.12. Aplicación móvil hibrida	30
2.3.13. React native	31
2.3.14. Google firebase.....	32

2.3.14.1.	Firebase Analytics	32
2.3.14.2.	Firebase Cloud Messaging	32
2.3.14.3.	Firebase Auth.....	32
2.3.14.4.	Realtime Database	33
2.3.14.5.	Firebase Storage	33
2.3.14.6.	Firebase Cloud Firestore.....	33
2.3.15.	Base de datos	34
2.3.16.	Base de datos relacionales	34
2.3.17.	Base de datos no sql	34
2.3.18.	JSON.....	34
2.3.19.	Internacionalización.....	35
2.3.20.	I18next.....	35
2.3.21.	Geolocalización	35
2.3.22.	API.....	36
2.3.23.	Entorno de desarrollo integrado	36
2.4.	Metodología de trabajo.....	37
2.4.1.	Metodología ágil.....	37
2.4.2.	Programación extrema XP.....	37
2.4.3.	Diagrama Modelo y Notación de Procesos de Negocio (BPMN)	37
3.	ANÁLISIS DEL SISTEMA.....	38
3.1.	Alcance del sistema	38
3.1.1.	Físicos y operativos	38
3.1.2.	Pagos electrónicos	38
3.1.3.	Turista.....	38
3.1.4.	Vendedor	38
3.1.5.	Administrador	39
3.2.	Metodología.....	39
3.3.	Módulos de desarrollo	40
3.4.	Elección de pasarela de pago.....	42
3.5.	Factibilidad	43
3.5.1.	Factibilidad técnica.....	43
3.5.1.1.	Hardware	43
3.5.1.2.	Sistema operativo	44
3.5.1.3.	Lenguaje de programación principal	44
3.5.1.4.	Framework de backend.....	46
3.5.2.	Factibilidad operativa	47
3.5.3.	Factibilidad de mercado.....	49
3.5.4.	Factibilidad económica.....	53
3.5.4.1.	Costo aproximado del proyecto	53
3.5.4.2.	Plan de monetización.....	54
3.5.5.	Conclusiones del análisis de factibilidad.....	57
3.6.	Usuarios del sistema	57
3.6.1.	Turista.....	57
3.6.2.	Vendedor	57
3.6.3.	Administrador.....	58

3.7.	Modelo de dominio.....	58
3.8.	Reglas de negocio	60
3.9.	Requerimientos	62
3.9.1.	Requerimientos no funcionales	62
3.9.2.	Requerimientos funcionales	63
3.9.3.	Requerimientos técnicos.....	65
3.9.3.1.	Requerimientos mínimos de software	65
3.9.3.2.	Requerimientos mínimos de hardware	65
4.	Diseño del sistema.....	66
4.1.	Generalidades	66
4.1.1.	Arquitectura del sistema	66
4.1.2.	Modelado de la información.....	67
4.1.3.	Diagrama de proceso general	67
4.1.4.	Casos de uso general	68
4.2.	Submódulo de aplicación móvil	68
4.2.1.	Modelo de navegación.....	68
4.2.2.	Inicio de sesión y registro de usuarios.....	70
4.2.2.1.	Inicio sesión.....	70
4.2.2.2.	Selección de tipo de registro de usuario	70
4.2.2.3.	Registro de usuario turista	71
4.2.2.4.	Términos y condiciones para registro de usuario	72
4.2.2.5.	Registro de usuario vendedor	72
4.2.3.	Navegación de usuario turista.....	73
4.2.3.1.	Buscar servicio	73
4.2.3.2.	Ver servicio.....	74
4.2.3.3.	Comprar servicio de hotel	74
4.2.3.4.	Comprar servicio de consumo	75
4.2.3.5.	Comprar otro servicio	75
4.2.3.6.	Mi viaje.....	76
4.2.3.7.	Comprobante de pago de servicio	76
4.2.3.8.	Pagar servicio presencial	77
4.2.3.9.	Pagar servicio – detalle.....	77
4.2.3.10.	Mi perfil.....	78
4.2.4.	Navegación de usuario vendedor.....	79
4.2.4.1.	Lista de servicios	79
4.2.4.2.	Selección de tipo de nuevo servicio	80
4.2.4.3.	Añadir servicio de hotel.....	81
4.2.4.4.	Añadir servicio de consumo	82
4.2.4.5.	Añadir otro servicio	83
4.2.4.6.	Ver servicio previamente registrado.....	84
4.2.4.7.	Lista de ventas	85
4.2.4.8.	Ver venta.....	85
4.2.4.9.	Escanear comprobante de pago realizado.....	86
4.2.4.10.	Generar orden de pago.....	87
4.2.4.11.	Código QR de orden de pago	88
4.2.4.12.	Mi perfil.....	89

4.3.	Submódulo de lógica de negocio.....	90
4.3.1.	Aplicación web de administrador	90
4.3.2.	Diagramas de negocio y secuenciación	95
4.3.2.1.	Inicio de sesión y registro de usuarios.....	95
4.3.2.2.	Usuario turista	96
4.3.2.3.	Usuario vendedor.....	99
4.4.	Submódulo de idioma.....	103
4.5.	Submódulo de generación de código QR	104
4.6.	Submódulo de pagos.....	106
5.	Desarrollo	109
5.1.	Submódulo de aplicación móvil	109
5.1.1.	Creación del proyecto	109
5.1.2.	Navegación	111
5.1.2.1.	Rutas no autenticadas	111
5.1.2.2.	Rutas autenticadas	111
5.1.2.3.	Selector de ruta de navegación	113
5.1.2.4.	Pila de navegación	114
5.1.3.	Plantilla de código para una pantalla genérica	115
5.1.4.	Componentes	116
5.1.4.1.	Selector de tiempo	116
5.1.4.2.	Selector de fecha.....	118
5.1.4.3.	Selector de nacionalidad.....	120
5.1.4.4.	Calendario.....	121
5.1.4.5.	Mapa de geolocalización	122
5.1.4.6.	Carousel de imágenes	123
5.1.4.7.	Loading	125
5.1.4.8.	Modal.....	127
5.1.5.	Librerías, paquetes y dependencias utilizadas	127
5.2.	Submódulo de lógica de negocio.....	129
5.2.1.	Inicio de sesión	129
5.2.2.	Selección de tipo de registro de usuario	132
5.2.3.	Registro de usuarios	134
5.2.4.	Obtención de usuario	137
5.2.5.	Actualización de información de usuario	137
5.2.6.	Selección de tipo de registro de nuevo servicio	138
5.2.7.	Registro de servicios.....	140
5.2.8.	Consulta de usuarios	141
5.2.9.	Consulta de servicios previamente registrados por el turista	142
5.2.10.	Consulta de servicios previamente registrados por el vendedor.....	147
5.2.11.	Comprobante de pago	151
5.2.12.	Puntuaciones y comentarios	155
5.2.13.	Ver ventas	159
5.2.14.	Ver perfil de usuario	163
5.2.15.	Aplicación de administrador.....	167
5.2.15.1.	Inicio de sesión	167
5.2.15.2.	Obtención de vendedores.....	169

5.2.15.3.	Solicitudes pendientes	170
5.2.15.4.	Solicitudes aprobadas	172
5.2.15.5.	Solicitudes rechazadas	174
5.3.	Submódulo de idioma.....	176
5.3.1.	Configuración	176
5.3.2.	Cambio de idioma desde login	177
5.3.3.	Cambio de idioma desde perfil.....	178
5.4.	Submódulo de generación de código QR	181
5.4.1.	Generación de comprobante de pago – Turista	181
5.4.2.	Lectura de comprobante de pago – Vendedor.....	185
5.4.3.	Generación de orden de pago – Vendedor.....	188
5.4.4.	Lectura de orden de pago – Turista	191
5.5.	Submódulo de pagos.....	193
5.5.1.	Obtención de credenciales para cuenta de developer	193
5.5.2.	Lógica del proceso.....	195
6.	Pruebas	199
6.1.	Persistencia de datos	199
6.2.	Happie path registro de usuario	202
6.3.	Happie path de creación de servicio	205
6.4.	Happie path de compra de servicio.....	208
6.5.	Happie path de comprobante de pago.....	210
6.6.	Happie path de generación y escaneo de orden de pago	213
Trabajo a futuro	214	
Impacto ambiental.....	215	
Conclusiones.....	216	
Referencias	217	
Apéndice A. Aviso de privacidad en encuestas aplicadas	220	
Apéndice B. Aviso de privacidad en aplicación móvil.....	221	
Apéndice C. Teoría - Tamaño de la muestra	223	
Apéndice D. Encuesta a turistas.....	225	
Apéndice E. Encuesta a vendedores.....	236	

Índice de tablas

Tabla 1. Aplicaciones similares	20
Tabla 2. Historial de versiones de IOS	28
Tabla 3. Historial de versiones de Android.....	29
Tabla 4. Comparativa de pasarelas de pago.....	42
Tabla 5. Recursos de Hardware para desarrollo del proyecto	43
Tabla 6. Comparativo de lenguaje de programación principal	44
Tabla 7. Comparativo de framework back-end a utilizar	46
Tabla 8. Histórico de turismo en Zona arqueológica de Teotihuacán 2016 - 2020	52
Tabla 9. Promedio de turismo histórico previo a pandemia	52
Tabla 10. Promedio de turismo histórico durante pandemia	52
Tabla 11. Costos operativos.....	53
Tabla 12. Costos de software y hardware	53
Tabla 13. Costos de desarrollo	54
Tabla 14. Entidad Turista	58
Tabla 15. Entidad Vendedor	58
Tabla 16. Entidad Dirección.....	59
Tabla 17. Entidad Servicio	59
Tabla 18. Entidad Venta.....	59
Tabla 19. Entidad Horario	59
Tabla 20. Reglas de negocio	60
Tabla 21. Requerimientos no funcionales	62
Tabla 22. Requerimientos funcionales	63
Tabla 23. Escenarios de lectura y generación de códigos QR	181
Tabla 24. Tabla de probabilidad de distribución	224

Índice de figuras

Figura 1. Zona Arqueológica de Teotihuacán	16
Figura 2. Placa de reconocimiento de Teotihuacán ante UNESCO	17
Figura 2. División política del Estado de México	23
Figura 3. Proceso en transacción utilizando pasarela de pago	25
Figura 4. I18next Aprende una vez, traduce en todas partes	35
Figura 5. Ejemplo de arquitectura de integración con API.....	36
Figura 6. Metodología XP	40
Figura 7. Proceso de monetización	55
Figura 8. Diagrama a bloques del sistema.....	66
Figura 9. Diagrama de base de datos NoSQL	67
Figura 10. Diagrama BPMN de proceso general.....	67
Figura 11. Diagrama general de casos de uso	68
Figura 12. Modelo de navegación de aplicación móvil	69
Figura 13. Mockup Inicio de sesión	70
Figura 14. Mockup Selección de Registro de usuario.....	70
Figura 15. Mockup Registro de usuario Turista.....	71
Figura 16. MockUp Términos y condiciones para registro de usuarios	72
Figura 17. Mockup Registro de usuario Vendedor.....	72
Figura 18. Mockup Buscar servicio	73
Figura 19. Mockup Ver servicio	74
Figura 20. Mockup Comprar servicio de hotel	74
Figura 21. Mockup Comprar servicio de consumo.....	75
Figura 22. Mockup Comprar otro servicio.....	75
Figura 23. Mockup Lista de mis viajes	76
Figura 24. Mockup Comprobante de pago para servicio	76
Figura 25. MockUp Pago de servicio presencial	77
Figura 26. MockUp Pagar servicio - detalle	77
Figura 27. MockUp Mi perfil de usuario turista	78
Figura 28. Mockup Lista de servicios	79
Figura 29. MockUp Selección de tipo de nuevo servicio.....	80
Figura 30. Mockup Añadir servicio de hotel.....	81
Figura 31. Mockup Añadir servicio de consumo	82
Figura 32. Mockup Añadir otro servicio	83
Figura 33. Mockup Ver servicio previamente registrado	84
Figura 34. Mockup Lista de ventas	85
Figura 35. Mockup Ver venta	85
Figura 36. Mockup Escanear comprobante de pago realizado	86
Figura 37. Mockup Generar orden de pago para servicio de hotel	87
Figura 38. Mockup Orden de pago presencial	88
Figura 39. Mockup Mi perfil vendedor	89
Figura 40. MockUp - Administrador, Iniciar sesión	90
Figura 41. MockUp - Administrador, Selección de operación.....	91
Figura 42. MockUp - Administrador, Solicitudes pendientes.....	92
Figura 43. MockUp - Administrador, Solicitudes aprobadas.....	93
Figura 44. MockUp - Administrador, Solicitudes rechazadas.....	94

Figura 45. Diagrama BPMN - Iniciar sesión	95
Figura 46. Diagrama BPMN - Registro de usuario	95
Figura 47. Diagrama BPMN - Turista, listar servicios	96
Figura 48. Diagrama BPMN - Turista, Ver servicio	96
Figura 49. Diagrama BPMN - Turista, Listar compras.....	97
Figura 50. Diagrama BPMN - Turista, Comprar servicio.....	97
Figura 51. Diagrama BPMN - Turista, Ver perfil	98
Figura 52. Diagrama BPMN - Turista, Actualizar servicio	98
Figura 53. Diagrama BPMN - Vendedor, Listar servicios.....	99
Figura 54. Diagrama BPMN - Vendedor, Ver servicio previamente registrado	99
Figura 55. Diagrama BPMN - Vendedor, Registrar nuevo servicio.....	100
Figura 56. Diagrama BPMN - Vendedor, Actualizar servicio	100
Figura 57. Diagrama BPMN - Vendedor, Ver perfil	101
Figura 58. Diagrama BPMN - Vendedor, Actualizar perfil	101
Figura 59. Diagrama BPMN - Vendedor, Eliminar servicio	102
Figura 60. Diagrama BPMN - Turista, Cambio de idioma	103
Figura 61. Diagrama BPMN - Turista, Escanear orden de pago	104
Figura 62. Diagrama BPMN - Turista, Generar comprobante de pago	105
Figura 63. Diagrama BPMN - Vendedor, escanea comprobante de pago	105
Figura 64. Diagrama BPMN - Vendedor, genera orden de pago.....	105
Figura 65. Funcionamiento de Paypal	106
Figura 66. Diagrama BPMN - Turista, realiza pago en Paypal	106
Figura 67. Pantalla de pago en PayPal.....	108
Figura 68. Estructura de un proyecto en react native.....	109
Figura 69. Componente Selector de tiempo en IOS y Android	118
Figura 70. Componente Selector de fecha en IOS y Android	119
Figura 71. Componente Selector de nacionalidad en IOS y And.....	120
Figura 72. Componente Calendario.....	122
Figura 73. Componente Mapa	123
Figura 74. Componente Carousel en IOS y Android	125
Figura 75. Componente Loading en IOS y Android	126
Figura 76. Pantalla de inicio de sesión en IOS y Android	131
Figura 77. Pantalla de selección de registro de tipo de usuario en IOS y Android	133
Figura 78. Pantalla de registro de usuario turista y vendedor en IOS y Android.....	136
Figura 79. Pantalla de selección de tipo de registro de servicio en IOS y Android	140
Figura 80. Pantallas de lista de servicios y ver servicio a detalle – TURISTA en IOS y Android ..	146
Figura 81. Pantallas de lista de servicios y ver servicio a detalle - VENDEDOR en IOS y Android	150
Figura 82. Pantallas de lista de compras y comprobante de pago en IOS y Android.....	154
Figura 83. Tipos de estatus de compra/venta versión textual.....	155
Figura 84. Tipos de estatus de compra/venta versión estandarizada	155
Figura 85. Pantallas de lista y creación de reviews en IOS y Android	158
Figura 86. Pantallas de lista de ventas y ver venta a detalle en IOS y Android.....	163
Figura 87. Pantallas de perfil de usuario vendedor y turista en IOS y Android	166
Figura 88. Estructura de componentes del proyecto web administrador	167
Figura 89. Pantalla de login de administardor	168
Figura 90. Pantalla de solicitudes pendientes del administardor	171

Figura 91. Pantalla de solicitudes aprobadas del administrador	173
Figura 92. Pantalla de solicitudes rechazadas del administrador	175
Figura 93. Pantalla de comprobante de pago - turista en IOS y Android.....	184
Figura 94. Pantalla de escaneo de comprobante de pago - vendedor en IOS y Android	187
Figura 95. Pantalla de generación de orden de pago - vendedor en IOS y Android	190
Figura 96. Pantallas de Lectura de Orden de pago - turista en IOS y Android	192
Figura 97 Crear App en paypal	193
Figura 99. App paypal creada correctamente	194
Figura 100. Claves y accesos paypal develop	194
Figura 101. Redirección y login en Paypal	197
Figura 102. Pago en Paypal.....	198
Figura 103. Persistencia de datos, usuarios	199
Figura 104. Persistencia de datos, roles	199
Figura 105. Persistencia de datos, servicios	200
Figura 106. Persistencia de datos, reviews	200
Figura 107. Persistencia de datos, transacciones.....	201
Figura 108. Pruebas - Registro de usuario parte 1	202
Figura 109. Pruebas - Registro de usuario parte 2	203
Figura 110. Pruebas - Registro de usuario parte 3	203
Figura 111. Pruebas - Registro de usuario parte 4	204
Figura 112. Pruebas - Creación de servicio parte 1.....	205
Figura 113. Pruebas - Creación de servicio parte 2	206
Figura 114. Pruebas - Creación de servicio parte 3	207
Figura 115. Pruebas - Compra de servicio parte 1	208
Figura 116. Pruebas - Compra de servicio parte 2	209
Figura 117. Pruebas - Validación con fallo por escaneo de otro servicio	210
Figura 118. Pruebas - Validación con fallo por doble escaneo de comprobante de pago.....	211
Figura 119. Pruebas - Validación exitosa de escaneo de comprobante de pago.....	212
Figura 120. Pruebas - Generar y escanear orden de pago	213
Figura 121. Encuesta a turistas - Pregunta #1	225
Figura 122. Encuesta a turistas - Pregunta #2	225
Figura 123. Encuesta a turistas - Pregunta #4	226
Figura 124. Encuesta a turistas - Pregunta #3	226
Figura 125. Encuesta a turistas - Pregunta #5	227
Figura 126. Encuesta a turistas - Pregunta #6	227
Figura 127. Encuesta a turistas - Pregunta #7	228
Figura 128. Encuesta a turistas - Pregunta #8	228
Figura 129. Encuesta a turistas - Pregunta #9	228
Figura 130. Encuesta a turistas - Pregunta #10.....	228
Figura 131. Encuesta a turistas - Pregunta #11	229
Figura 132. Encuesta a turistas - Pregunta #12	229
Figura 133. Encuesta a turistas - Pregunta #13	230
Figura 134. Encuesta a turistas - Pregunta #14	230
Figura 135. Encuesta a turistas - Pregunta #16	231
Figura 136. Encuesta a turistas - Pregunta #15	231
Figura 137. Encuesta a turistas - Pregunta #17	232

Figura 138. Encuesta a turistas - Pregunta #18	232
Figura 139. Encuesta a turistas - Pregunta #19	233
Figura 140. Encuesta a turistas - Pregunta #20	234
Figura 141. Encuesta a turistas - Pregunta #21	234
Figura 142. Encuesta a turistas - Pregunta #22	235
Figura 143. Encuesta a turistas - Pregunta #23	235
Figura 144. Encuesta a vendedores - Pregunta #01	236
Figura 145. Encuesta a vendedores - Pregunta #02	236
Figura 146. Encuesta a vendedores - Pregunta #03	237
Figura 147. Encuesta a vendedores - Pregunta #04	237
Figura 148. Encuesta a vendedores - Pregunta #05	237
Figura 149. Encuesta a vendedores - Pregunta #06	237
Figura 150. Encuesta a vendedores - Pregunta #07	237
Figura 151. Encuesta a vendedores - Pregunta #08	237
Figura 152. Encuesta a vendedores - Pregunta #09	237
Figura 153. Encuesta a vendedores - Pregunta #10	237
Figura 154. Encuesta a vendedores - Pregunta #11	237
Figura 155. Encuesta a vendedores - Pregunta #12	237
Figura 156. Encuesta a vendedores - Pregunta #13	237
Figura 157. Encuesta a vendedores - Pregunta #14	237
Figura 158. Encuesta a vendedores - Pregunta #15	237
Figura 159. Encuesta a vendedores - Pregunta #16	237
Figura 160. Encuesta a vendedores - Pregunta #17	237
Figura 161. Encuesta a vendedores - Pregunta #18	237
Figura 162. Encuesta a vendedores - Pregunta #19	237
Figura 163. Encuesta a vendedores - Pregunta #20	237
Figura 164. Encuesta a vendedores - Pregunta #21	237
Figura 165. Encuesta a vendedores - Pregunta #22	237
Figura 166. Encuesta a vendedores - Pregunta #23	237
Figura 167. Encuesta a vendedores - Pregunta #24	237
Figura 168. Encuesta a vendedores - Pregunta #25	237

1. Introducción

1.1. Resumen

El sistema descrito y diseñado a continuación tiene como objetivo la integración de los servicios turísticos (Hospedaje, restaurante, recreación, transporte, etc.) ofertados en la zona arqueológica de Teotihuacán, mediante una aplicación móvil nativa (IOS y Android), la cual facilita el proceso de compra y venta tanto para vendedores como para turistas nacionales e internacionales, mediante adecuación de lenguaje en español e inglés y pagos electrónicos.

1.2. Descripción del documento

Este documento es de carácter técnico, desarrollado para el trabajo terminal 2020-B004 “Aplicación móvil de comercio para servicios turísticos en zona arqueológica de Teotihuacán” durante el periodo comprendido entre febrero y diciembre de 2021.

Primeramente, se encuentra una sección introductoria, la cual hace el planteamiento del problema, la definición de la propuesta de solución, primeras aproximaciones del alcance del proyecto, así como los previos y actuales en el área.

En segundo lugar, se muestra el “Marco conceptual”, donde se detalla de manera general, el ámbito bajo que es desarrollado el sistema, es decir, una serie de conceptos, terminología, y detalles técnicos, que son de interés para la correcta comprensión de las diversas temáticas que abarca el Trabajo Terminal.

Después se incluye el apartado de “Análisis del sistema”, en el que se evalúa la factibilidad, herramientas, tecnologías y requerimientos de desarrollo, especificaciones técnicas básicas necesarias que un usuario final del sistema deberá de cumplir para el correcto funcionamiento del sistema, el alcance del proyecto, requerimientos y metodología de desarrollo.

En el apartado de “Diseño del sistema”, se presenta el proceso general del sistema, describiendo el flujo que siguen las actividades que se llevan a cabo, modelado de la información y almacenamiento, el diseño de cada modulo de desarrollo (Aplicación móvil, Idioma, Lógica de negocio, Generación de código QR y Lógica de pago) se realiza mediante casos de uso, donde se muestran los diagramas UML correspondientes.

Al final del diseño, se encuentra el apartado de “Desarrollo y pruebas”, mismo donde se explica de forma detallada y por cada módulo, la implementación del diseño del sistema que fue descrito en el apartado anterior “Diseño del sistema”, igualmente se da una descripción de las pruebas que fueron realizadas por cada uno de los módulos, las pruebas de integración, así como de operación del sistema, mostrando sus respectivos resultados.

Casi para terminar, se presenta la conclusión del ejercicio, destacando los puntos más relevantes que fueron expuestos a lo largo del documento, los desafíos y oportunidades que se detectaron una vez que el sistema fue desarrollado.

Finalmente se plantea el área de oportunidad y trabajo a futuro, tocando nuevas ideas y mejoras que podrían enriquecer este proyecto, para así continuar con el desarrollo, tanto que pudiese ser escalado a un nivel comercial en la vida real.

1.3. Antecedentes

Teotihuacán, “Lugar donde fueron hechos los dioses; ciudad de los dioses”, es el nombre que se da a la que fue una de las mayores ciudades de Mesoamérica durante la época prehispánica. El topónimo es de origen náhuatl y fue empleado por los mexicas, aun hoy día se desconoce el nombre que le daban sus habitantes.

La Ciudad Prehispánica de Teotihuacán fue uno de los centros urbanos más grandes del mundo antiguo, que llegó a concentrar una población mayor a los 100,000 habitantes en su momento de máximo esplendor. Situada en un valle rico en recursos naturales, Teotihuacán fue la sede del poder de una de las sociedades mesoamericanas más influyentes en los ámbitos político, económico, comercial, religioso y cultural, cuyos rasgos marcaron permanentemente a los pueblos del altiplano mexicano, traspasando el tiempo y llegando hasta nosotros con la misma fuerza y grandeza con que sus constructores la planearon.

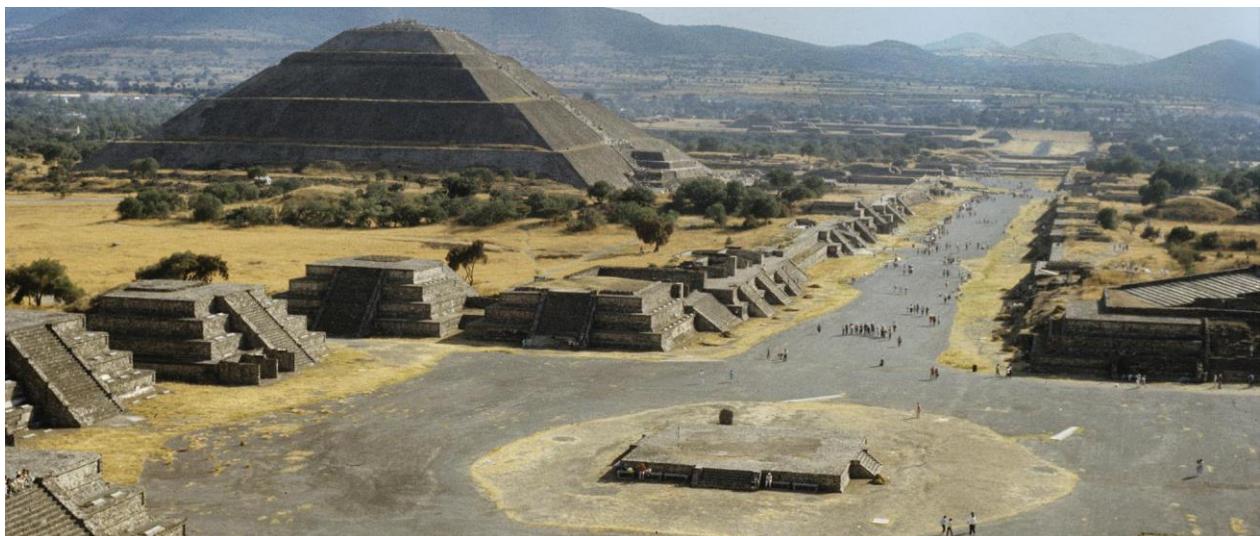


Figura 1. Zona Arqueológica de Teotihuacán

La evidencia arqueológica descubierta en el Valle de Teotihuacán revela que durante el período Clásico se desarrolló una de las sociedades urbanas más complejas de toda Mesoamérica, así como que dicha sociedad estuvo altamente estratificada, ampliamente especializada y conformada por diversos grupos étnicos.

Para los pueblos que precedieron a Teotihuacán, este sitio tuvo un significado preponderantemente sagrado. Varias fuentes históricas señalan que los aztecas y sus gobernantes llegaron a estas ruinas para orar y celebrar ritos. Posteriormente Teotihuacán fue punto de referencia desde el comienzo de la ocupación española; y en la actualidad es reconocida como uno de los testimonios más sobresalientes del urbanismo antiguo y el desarrollo estatal, por lo que es objeto de interés para investigadores de México y el mundo, que a través de distintas disciplinas científicas continúan explorando su complejidad.

El área abierta a la visita pública tiene una extensión de 264 hectáreas, donde se concentran los principales complejos de edificios monumentales, como La Ciudadela y el Templo de la Serpiente Emplumada, la Calzada de los Muertos y los conjuntos residenciales que la flanquean, las Pirámides del Sol y la Luna, el Palacio de Quetzalpálotl y 4 conjuntos departamentales con

importantes ejemplos de pintura mural, como son Tetitla, Atetelco, Tepantitla y La Ventilla, además de otros 2 conjuntos de corte habitacional denominados Yayahuala y Zacuala.

La entrada a la zona puede realizarse por 5 puertas, distribuidas e interconectadas por un camino periférico empedrado que circunda el área monumental, además hay caminos vecinales que la comunican con los conjuntos habitacionales mencionados. [1]

El Valle de Teotihuacán es una región natural localizada en el noreste del Estado de México (centro del sur de la República Mexicana) conformado por los municipios de San Juan Teotihuacán, San Martín de las Pirámides, Acolman, Otumba, Axapusco y Nopaltepec. Los atractivos y recursos turísticos con los que cuenta el Valle de Teotihuacán son muy diversos en cuanto a naturaleza, historia, recreación, salud y arqueología que pueden ser aprovechados para satisfacer las necesidades de los turistas y obtener un beneficio dentro de los municipios, ya que estos municipios “Pueblos mágicos” cuentan con dichos atractivos. [2]

El reconocimiento del sitio como patrimonio cultural es universal, pues desde 1987 forma parte de la lista de Patrimonio Mundial de la UNESCO.



Figura 2. Placa de reconocimiento de Teotihuacán ante UNESCO

Los vestigios arqueológicos de la antigua ciudad son visitados cada año por miles de personas, haciendo del sitio uno de los mayores polos de atracción turística del país.

El turismo es un sector económico importante para México, mismo que desempeña un papel destacado en el turismo a nivel mundial. El sector representa el 8.5% del PIB, el 5.8% del empleo remunerado de tiempo completo (en el sector formal) y el 77.2% de las exportaciones de servicios. Este sector contribuye de manera positiva a la balanza de pagos nacional y genera un valor superior a la media para la economía. La cifra récord de 32.1 millones de turistas internacionales aportaron 246.1 mil millones de pesos (\$15.5 mil millones USD) a la economía en 2015, con un aumento de

las personas y de los flujos monetarios hacia el país superando el crecimiento de muchas economías avanzadas y emergentes en los últimos años. Esta situación siguió a un periodo prolongado de crecimiento más modesto. El turismo nacional también es de gran relevancia, ya que aporta 88 de cada 100 pesos mexicanos consumidos por turistas en el país, apoya al empleo y el desarrollo en regiones que no atraen a visitantes internacionales. [3]

El Valle de Teotihuacán cuenta con una gran riqueza en sus atractivos y destinos turísticos los cuales son y pueden ser potencialmente aprovechados, distribuyendo la captación de turistas de los más de 2.5 millones de personas que visitan el centro religioso de Teotihuacán anualmente, siendo esta la mayor captación de turistas dentro del Valle; los municipios vecinos cuentan con la distinción de pueblos mágicos y pueblos con encanto, teniendo la inversión por parte del gobierno para ser aprovechados los recursos. [2]

Viajar a la ciudad prehispánica en donde “los hombres se convierten en dioses”, es una experiencia inolvidable, lugar donde se erigen dos pirámides de gran tamaño a las cuales se puede subir para percibir la ciudad de Teotihuacán como lo hacían los antepasados. Sin embargo, no es lo único que se puede hacer en este maravilloso lugar, hay muchas otras atracciones que conocer para tener una gran experiencia de verdad mística y poder decir que conoces Teotihuacán de principio a fin, como lo es paseo en globo, sumergirse en la gastronomía típica de la región, alojarse en un acogedor hotel, deportes extremos, convivir con los bellos ejemplares del safari de la región, pasear en bici, disfrutar de la fiesta y experiencia nocturna, acampar, maravillarse por los múltiples museos, refrescarse en los balnearios, llenarse de energía cada equinoccio, empaparse de cultura con los carnavales musicales y culturales, revitalizarse con sesiones de spa y temazcal, entre muchos otros atractivos.

En los últimos años se han realizado algunas investigaciones donde se proponen estrategias y recomendaciones para detonar el turismo de esta región [2] [4] [5], llegando a la conclusión de que para mejorar la situación, la receta es "mejor integración de producto", es decir hacer más atractivo el destino turístico, garantizando la seguridad del turista y mejorar la imagen del país, consolidar a México como el 5º país más visitado del mundo y el 10º con más divisas turísticas, incrementar la competitividad de los destinos turísticos, mejorar el nivel de vida de la población en las localidades y desarrollo turístico sustentable.

1.4. Justificación

Según el secretario de turismo Miguel Torruco que dio a conocer los resultados de la actividad turística durante el año 2019 al respecto destacó dos principales indicadores: turistas y divisas. [6] "El éxito no se mide en número de visitantes, sino en divisas captadas", México puede presumir de ser el séptimo país del mundo en visitantes, pero solo es el 16 en captación de divisas y el 40 en gasto per cápita del turista.

De acuerdo con las estadísticas de DATATUR, Secretaría de Turismo en 2019 [7], la ciudad prehispánica de Teotihuacán recibió 3,459,528 visitantes en 2019, de los cuales 2,779,408 eran turistas nacionales y 680,120 extranjeros. Los turistas que visitan Teotihuacán son viajeros de todas partes del mundo, la mayoría de los que visita la ciudad de los dioses, únicamente recorren la zona arqueológica y de museos, dejando fuera el resto de la región y los atractivos que se ofertan, esto gran parte debido a la falta de promoción, de vías de transporte, de idioma y moneda, de ser un destino mas atractivo podría generarse una mayor afluencia y derrama económica, que beneficiaría directamente a la zona y al país.

El gobierno federal se ha propuesto 5 objetivos de política pública para consolidar al turismo como motor de crecimiento económico: 1. Garantizar la seguridad en los destinos turísticos y mejorar la imagen del país, 2. Alcanzar el 5º lugar en el ranking de los países más visitados y el 10º con mayor captación de divisas, 3. Incrementar la competitividad de nuestros destinos turísticos, 4. Mejorar el nivel de vida de la población en las localidades turísticas y 5. Desarrollo turístico sustentable. [4] Los gobiernos federal, estatal y local se han esforzado para conseguir que las personas que arriban a la zona arqueológica prolonguen su estancia, implementando una serie de servicios y atracciones que generen mayor derrama económica.

Desde la década de 2010, varios empresarios y prestadores de servicios trabajan en la consolidación de los atractivos turísticos del valle de Teotihuacán, como la señalización de los distintos puntos de interés turístico o el mejoramiento de la iluminación y de las vialidades, unificación en estética de los edificios de las calles principales de los municipios cercanos (Pueblos mágicos). La Comisión para el Desarrollo Turístico del Valle de Teotihuacán (COVATE) participa además en la organización de ferias, festivales culturales y expo-venta de artesanías; se han impartido, junto con arqueólogos y especialistas del INAH, cursos de capacitación a guías de turistas y personal que labora en este sector. [8]

En palabras del secretario de Turismo, Miguel Torruco Marqués, “en los próximos años, el 90% de los viajes internacionales se decidirán a través de un dispositivo móvil, desde buscar inspiración para viajar, hacer reservaciones y encontrar nuevos productos turísticos, hasta recuerdos que compartir y experiencias que repetir”. También mencionó que el mundo de internet cuenta actualmente con 4,300 millones de usuarios, es decir, el 57% de la población mundial.

Este trabajo terminal consiste en la creación de una aplicación móvil que concentre los servicios que son ofrecidos dentro del valle de Teotihuacán en una misma plataforma, que permita hacer las transacciones monetarias (se considera la plataforma de PayPal por la sencillez y seguridad que ofrece a los usuarios al realizar pagos electrónicos) desde la palma de la mano incentivando el turismo y comercio, haciendo uso de adecuación del lenguaje y pagos electrónicos acordes al mundo globalizado.

La implementación de este proyecto a una escala de producción es una excelente estrategia de negocio para potencializar el turismo y las ganancias económicas, ya que reduce la incertidumbre para los viajeros nacionales e internacionales (angloparlantes), de tal manera que se pone a disposición una concentración de productos y servicios de la región en una misma aplicación móvil, por ende, la derrama económica beneficiara directamente a los pobladores de la zona.

1.5. Planteamiento del problema

La necesidad que atenderá la aplicación radica en dos polos complementarios: dar visibilidad a los proveedores de servicios de manera formal que están establecidos en la zona de Teotihuacán que no tienen gran visibilidad, difusión, cercanía o reputación, y por otro lado mejorar la experiencia de viaje de turistas nacionales e internacionales, reduciendo la incertidumbre de viaje y aumentando el consumo de servicios, e indirectamente y con el tiempo, hacer de Teotihuacán un destino turístico más atractivo.

1.6. Objetivos

1.6.1. Objetivo general

Desarrollar una aplicación móvil, que permita al turista realizar calificaciones, compras y pagos de servicios turísticos que ofrece la zona arqueológica de Teotihuacán con proveedores de servicios previamente registrados.

1.6.2. Objetivos particulares

- Pagar/cobrar los servicios mediante la plataforma PayPal, para dar confianza y rapidez al turista y vendedor.
- Validar manualmente los usuarios vendedores mediante clave RFC para dar seguridad al turista y autenticidad de vendedores.
- Permitir al usuario final elegir el idioma de su preferencia (inglés, español) en el que la aplicación móvil funcionará, de tal manera que el usuario tenga mayor confianza, preferencia y comodidad al usar la aplicación.
- Procesar las transacciones y operaciones solicitadas por el usuario por medio de un servidor externo al dispositivo móvil, con la finalidad de que aplicación móvil sea eficiente en cuanto a los recursos del dispositivo del usuario final.
- Documentar el trabajo realizado.

1.7. Estado del arte

Al ser un nicho de mercado tan prometedor, hay algunas empresas e iniciativas dedicadas al comercio de servicios turísticos, algunas están motivadas por lo que hemos dicho anteriormente, "mejor integración de producto" y apoyo gubernamental.

Tabla 1. Aplicaciones similares

Nombre	Descripción y características	Precio
Sistema de difusión cultural interactivo con enfoque Regional Mexicano. [9]	<ul style="list-style-type: none">- Sistema web y móvil.- Trabajo terminal ESCOM sin continuación.- Enfocado a consulta e información.- Cobertura en Teotihuacán.- Tienda online, noticias y calendario de eventos.	No aplica, es propiedad IPN
VisitMexico. [10]	<ul style="list-style-type: none">- Plataforma WEB en funcionamiento.- Cobertura nacional.- Iniciativa pública en coordinación con privada.- Busca incluir a pequeñas y medianas empresas.- Rutas establecidas de viaje.- No es una aplicación de comercio o reservaciones, es el medio de contacto entre la oferta y demanda.- Recuperación de datos demográficos.- Busca crear tarjeta para pagos electrónicos.	Gratis.
Despegar. [11]	<ul style="list-style-type: none">- Servicio internacional.- Cubre transporte, hotelería y viaje en globo.- Sistema web y móvil en funcionamiento.- Iniciativa privada.	Costo variable de acuerdo con la compra.
Expedia. [12]	<ul style="list-style-type: none">- Sistema web y móvil funcionando.- Cubre hotel y vuelos.- Iniciativa privada.	Costo variable de

	<ul style="list-style-type: none"> - Servicio internacional. 	acuerdo con la compra.
Tours & Travel Teotihuacán. [13]	<ul style="list-style-type: none"> - Página WEB estático en funcionamiento. - Registro de SECTUR. - Contacto vía correo electrónico y número telefónico. - Servicio en valle de México y enfoque en Teotihuacán. - Asesoría y elaboración personalizada de viaje persona a persona. - Horarios establecidos de operación. - Guías turísticos multilingües. - Ecoturismo y viaje en globo. - Viajes educativos. - Fuente de consulta y contacto - Iniciativa privada. 	Costo variable de acuerdo con la compra.
Explora Teotihuacán. [14]	<ul style="list-style-type: none"> - Página WEB estática funcionando. - Transacción fuera de la aplicación, y por personas intermediarias. - Contacto vía correo electrónico y número telefónico. - Horarios locales establecidos de operación. - Cubre transporte, vuelo en globo y tours guiados. - Iniciativa privada. - Fuente de consulta a otros servicios como experiencia nocturna, hospedaje en los 2 hoteles más representativos. 	Costo variable de acuerdo con la compra.
Sky Balloons. [15]	<ul style="list-style-type: none"> - Página WEB funcionando. - Compra en línea. - Paquetes de vuelo en globo. - Servicio de buffet. - Servicio de hotel y restaurante. - Detalles de experiencia (Flores, Brindis, Kit de cumpleaños, Manta de feliz cumpleaños, propuestas de relaciones de pareja, etc.) 	Costo variable de acuerdo con la compra.
Agentes de viaje independientes.	<ul style="list-style-type: none"> - Persona o grupo de personas que organizan viajes. - Utilizan redes sociales para difusión y comunicación. - Trato directo e intermediario. - Alcance a nivel zona y por referencias personales. 	Costo variable de acuerdo con la compra.

Cabe recalcar que las opciones: “VisitMexico” y “Sky Balloons” de la Tabla 1. Aplicaciones similares, son las iniciativas que más tienen similitud con la aplicación propuesta en este documento. Con respecto con “VisitMexico”, la plataforma es únicamente de consulta y esta disponible en la WEB, no aprovecha el nicho de mercado de Teotihuacán ya que su objetivo es a nivel nacional, y la aplicación propuesta es un medio de comercio electrónico, centrada en Teotihuacán y además es una aplicación móvil, que es mas cómoda al viajar comparado con una página web estática.

Por otro lado, “Sky Balloons”, esta centrada en la zona de Teotihuacán, tiene algunos convenidos con algunos de los lugares más representativos (restaurantes, hoteles y recreativos), es una página

web que realiza todos los cobros desde su interfaz, la aplicación propuesta da la posibilidad de hacer los pagos hasta el momento en que se consume el servicio, reduciendo así incertidumbre para el turista o bien pagar todo al crear un viaje, es una aplicación móvil, cualquier oferente puede registrarse en la aplicación y no depender de una empresa que acepte la colaboración, como es el caso de “Sky Balloons”.

Con el resto de las opciones mencionadas que se encuentran disponibles, buscan únicamente la difusión de sus servicios, pero sus plataformas no permiten realizar transacciones o bien a través de personas intermediarias y en horarios locales de operación de dichos intermediarios, algunas otras únicamente ofertan transporte y alojamiento.

2. Marco conceptual

En esta sección se detalla de manera general, el ámbito bajo que es desarrollado el sistema, es decir, una serie de conceptos, terminología, y detalles técnicos, que son de interés para la correcta comprensión de las diversas temáticas que abarca el Trabajo Terminal.

2.1. Ubicación geográfica y normatividad

El Estado de México esta conformado por 125 municipios, de los cuales actualmente cuenta con 10 Pueblos Mágicos y 22 Pueblos con Encanto.

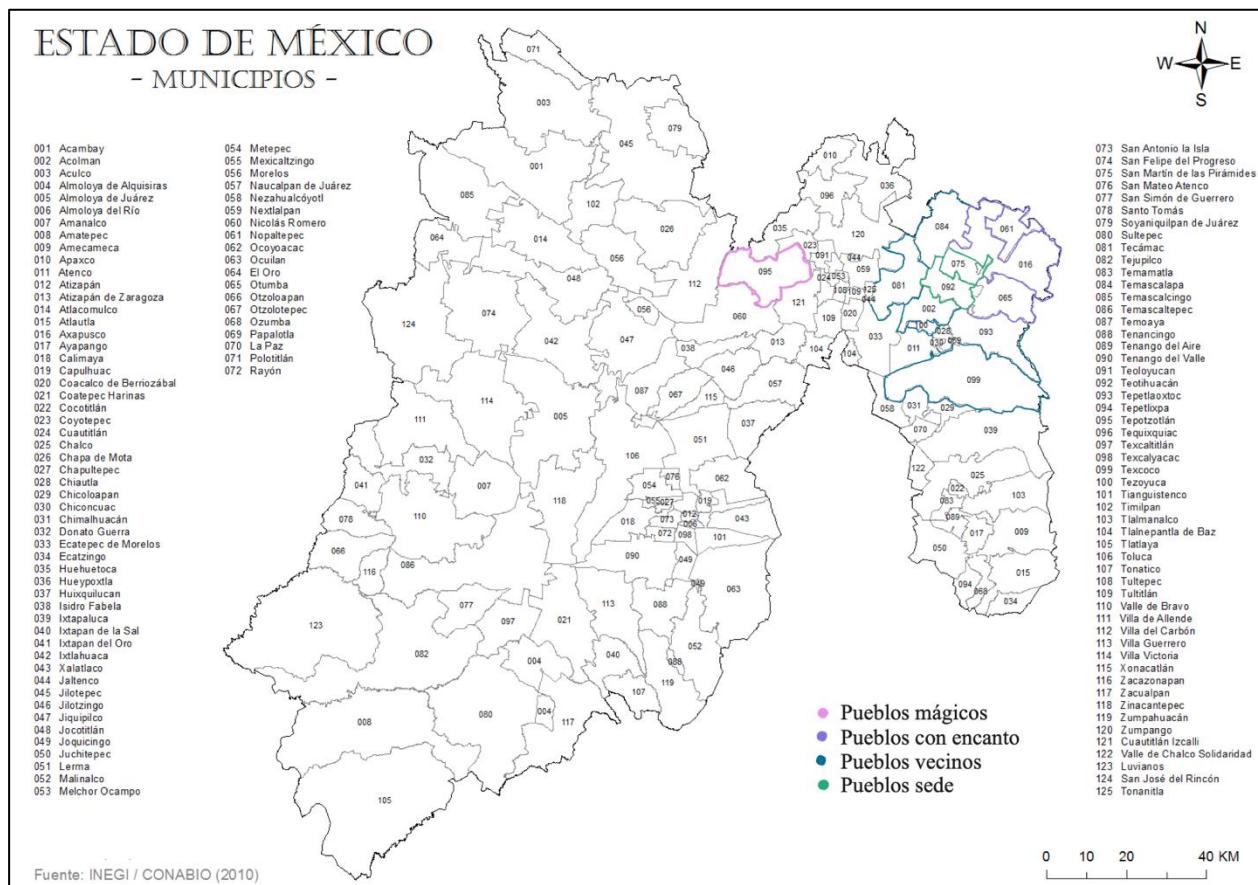


Figura 3. División política del Estado de México

2.1.1. Pueblos Mágicos

Los Pueblos Mágicos, incluso después de haber sido absorbidos por la mancha urbana, siguen conservando su valor y herencia cultural a través de un patrimonio irremplazable, que cumple con los requisitos de permanencia como lo establecen los Lineamientos Generales para la incorporación y permanencia al Programa Pueblos Mágicos a nivel nacional, publicado en el Diario Oficial de la Federación del 26 de septiembre de 2014 [16]. Lista de pueblos mágicos en el estado de México:

1. Malinalco
2. Aculco
3. Ixtapan de la Sal
4. Valle de Bravo
5. San Martín de las Pirámides*
6. El Oro
7. Teotihuacán*
8. Metepec
9. Villa del Carbón
10. Tepotzotlán*

Los pueblos mágicos que están cercanos o son sede de la zona de Teotihuacán se indican con *.

2.1.2. Pueblos con Encanto

Los Pueblos con Encanto, de acuerdo con los Lineamientos de declaración correspondientes publicados en el Periódico Oficial "Gaceta del Gobierno" del 7 de octubre de 2014 [17], son los municipios cuyos habitantes han sabido cuidar la riqueza cultural, historia, autenticidad y carácter propios del lugar, con el propósito de convertir al turismo en una opción para su desarrollo, a través de convenios de coordinación individualizados para cada proyecto.

"Pueblos con encanto" es un programa que da valor a pequeñas localidades con vocación turística en el Estado de México y que, a futuro pudieran considerarse aspirantes a lograr el nombramiento de Pueblo Mágico. Lista de pueblos con encanto:

- | | |
|---------------------|-------------------------------|
| 1. Acolman* | 12. Ozumba |
| 2. Amanalco | 13. Papalotla |
| 3. Amecameca | 14. Sultepec |
| 4. Ayapango | 15. Temascalcingo |
| 5. Axapusco* | 16. Temascaltepec de González |
| 6. Coatepec Harinas | 17. Tenango del Valle |
| 7. Donato Guerra | 18. Tepetlixpa |
| 8. Jilotepec | 19. Tlalmanalco |
| 9. Lerma | 20. Tonatico |
| 10. Nopaltepec* | 21. Zacualpan |
| 11. Otumba* | 22. Zinacantepec |

Los pueblos con encanto que están cercanos de la zona de Teotihuacán se indican con *.

2.1.3. Municipios vecinos

Los municipios vecinos de la zona te Teotihuacán se consideran todos aquellos que tienen medios de transporte desde la misma zona al municipio o con colindancia. Lista de municipios vecinos:

- | | |
|--------------------------------|-----------------|
| 1. San Martin de las pirámides | 7. Acolman |
| 2. Teotihuacán | 8. Tecámac |
| 3. Temazcalapa | 9. Texcoco |
| 4. Axapusco | 10. Tezoyuca |
| 5. Otumba | 11. Chiautla |
| 6. Tepetlaoxtoc | 12. Chinconcuac |

2.1.4. Aviso de privacidad

Documento a disposición del titular de los datos personales de forma física, electrónica o en cualquier formato generado por el responsable, con el objeto de informarle los propósitos del tratamiento de estos a partir del momento en el cual se recaben sus datos personales, para la realización de algún trámite o servicio ante la Secretaría de Economía (SE). [18]

A través de este documento, se le comunica al titular la información que se recaba respecto de su persona y la finalidad de su obtención, así como la posibilidad de ejercer los derechos de acceso, rectificación, cancelación y oposición y la forma de ejercerlos.

2.2. Pagos electrónicos

2.2.1. Ecommerce

El e-commerce o comercio electrónico es un método de compraventa de bienes, productos o servicios valiéndose de internet como medio, es decir, comerciar de manera online. [19]

2.2.2. Pasarela de pago

Es un término íntimamente relacionado al ecommerce y es que se trata de la forma en que los usuarios pueden pagar por productos o servicios en las tiendas online. Son servicios de terceros que usamos como intermediarios para procesar los pagos con tarjetas de débito o crédito y en efectivos en tiendas de conveniencia como Oxxo.

Usando este tipo de servicios, nos ahorramos lo problemático que sería para nosotros gestionar la seguridad y los permisos cuando el usuario ingresa sus datos bancarios, por ejemplo. De esta forma, todo se lo delegamos a la plataforma que nos ofrece el servicio, ahorrándonos todos esos procesos para concentrarnos en lo que verdaderamente agrega valor a nuestro negocio, además que ofrece seguridad a nuestros clientes ya que estos servicios cada vez son más conocidos y se sienten seguros al ver que sus datos no corren riesgo usando una plataforma conocida y robusta. [20]

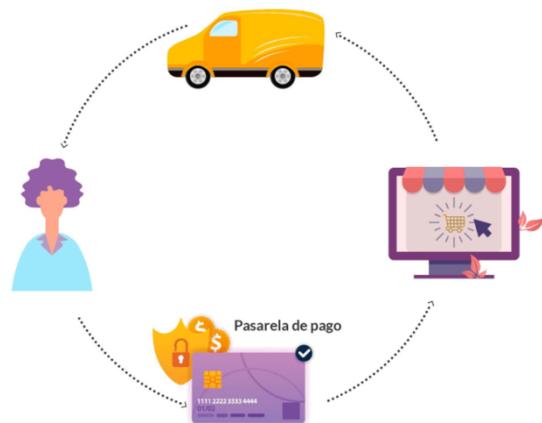


Figura 4. Proceso en transacción utilizando pasarela de pago

2.2.2.1. Razones principales para utilizar una pasarela de pago

⇒ Seguridad para el cliente

Al no poder interceptar los datos desde la tienda, el cliente se siente más seguro ya que ve que está dejando sus datos en un sistema seguro y robusto.

⇒ Experiencia de usuario

Todo es automatizado y el sistema le verifica al cliente su pago para que se siente seguro y cómodo de la compra que está realizando.

⇒ Imagen de la ecommerce

Ya no se tiene que dar el número de una tarjeta ni el nombre, ahora con una referencia todo se verá más profesional lo cual hace que la tienda en línea proyecte una mejor imagen.

2.2.2. Inconvenientes de las pasarelas de pago

Aunque no se les consideraría inconvenientes como tal, si representan algo a considerar al momento de implementar una pasarela en una tienda en línea.

⇒ Comisión por venta

Al ser un servicio de terceros, este tercero debe cobrar por su servicio, las comisiones de todas las pasarelas de pago promedian el 3% de comisión.

⇒ Disponibilidad del dinero

El dinero no estará disponible inmediatamente después de que el cliente realice el pago, antes de eso la pasarela lo procesa y posteriormente lo libera, regularmente dos días después o un poco más.

⇒ Dependencia del servicio

Finalmente se trata de un servicio de otra empresa, y si está empresa por algún motivo no tiene disponible su servicio, el vendedor no podrá cobrar.

⇒ Alta en el SAT

Algunos de los documentos que se solicitan para poder otorgar la pasarela es un RFC por lo tanto, se necesitará estar dado de alta en Hacienda.

2.2.3. Contra cargos

Los contra cargos suceden cuando una persona dueña de una tarjeta, no reconoce ante su institución bancaria un cargo, esto hace que el banco deba revertir la transacción para devolver el dinero al cliente y este dinero debe salir de algún lado, así es, del bolsillo de la pasarela de pago, el problema viene cuando el usuario genera un contra cargo de una compra que si recibió, esto ha llevado a las pasarelas de pago a ser cada vez menos flexibles y revisar muy cuidadosamente el solicitante del servicio.

2.2.4. PayPal

PayPal es un método de pago en línea que permite asociar las tarjetas de crédito o débito a una cuenta, para que, de esta manera, se puedan realizar pagos y transferencias online con tan solo iniciar sesión con un correo electrónico y contraseña, sin tener que compartir información financiera con el destinatario, el cual puede ser una persona o una empresa, sin importar si tiene o no una cuenta PayPal. Dentro de los múltiples beneficios que te ofrece PayPal se mencionan:

⇒ Acepta pagos con tarjeta de débito y crédito.

⇒ Acepta pago a meses.

⇒ No acepta pagos en efectivo.

⇒ Comisión del 3.95% + \$4 MXN.

⇒ No tienen oficina en México.

⇒ Disponibilidad de pago de 7 días.

⇒ Tiene plugin para WordPress con Woocommerce.

2.2.4.1. Comisiones en PayPal

Algunas transacciones en PayPal si cobran comisión, algunas otras no lo hacen.

No hay comisión, por ejemplo:

- ⇒ Abrir una cuenta en PayPal
- ⇒ Hacer una transferencia de dinero de la cuenta bancaria a la misma cuenta PayPal
- ⇒ Comprar artículos o servicios con la cuenta PayPal
- ⇒ Cancelar tu cuenta PayPal

Si hay comisiones, por ejemplo:

- ⇒ Recibir pagos, ya sea por productos o servicios
- ⇒ Recibir pagos de familiares y amigos realizados con una tarjeta de crédito/débito
- ⇒ Recibir pagos de personas que viven en otros países.

2.3. Tecnología

2.3.1. Arquitectura cliente servidor

Se llama arquitectura cliente-servidor a la forma de interactuar entre dos dispositivos, esto estableciendo que uno debe ser el cliente y otro el servidor, en dichos roles uno consume el contenido que tiene o genera el otro. Esta arquitectura se usa para que una infinidad de dispositivos puedan acceder a diversos contenidos sin la necesidad de que se tenga instalado en todos los clientes, igualmente cuando los recursos se requieren administrar de forma particular por lo que se concentran en el dispositivo servidor (que puede ser centralizado o distribuido), el cliente envía solicitudes y el servidor de procésalas.

2.3.2. Dispositivo móvil

Los dispositivos móviles son equipos de cómputo completos construidos en un solo circuito impreso, su diseño está centrado en sistemas de un procesador o multiprocesador con memoria RAM, puertos de E/S y todas las características necesarias para hacer una computadora funcional en un circuito impreso reducido. Los teléfonos inteligentes, tabletas electrónicas y otras computadoras de bolsillo, son ejemplo de dispositivos móviles.

2.3.3. Sistema operativo

Se puede ver desde dos puntos de vista: máquina extendida y gestor de recursos.

- ⇒ Es una máquina extendida:

Desde este punto de vista, el trabajo del sistema operativo es el de proveer a los usuarios de una máquina extendida o máquina virtual que abstrae todos los detalles del hardware y que es más conveniente de usar que la máquina actual.

- ⇒ Es un gestor de recursos:

Desde este punto de vista, el trabajo del sistema operativo es la gestión eficiente de las diferentes partes del sistema (memoria, disco duro, archivos, impresoras, red, etc.). El SO mantiene un registro de quien esta usando que recurso, otorga las requisiciones de recursos, cuenta su uso y es el mediador en conflictos de requisiciones de diferentes programas y usuarios. [19]

2.3.4. IOS

iOS es un sistema operativo móvil de la multinacional Apple Inc. Originalmente desarrollado para el iPhone (iPhone OS), después se ha usado en dispositivos como el iPod touch y el iPad. Apple no permite la instalación de iOS en hardware de terceros.

Los iOS posee una interfaz fluida, sencilla y elegante, sin mucha posibilidad de personalizar pero que ofrece al usuario una de las experiencias más cómodas del mercado. Esto se debe a que iOS está diseñado para sacar el máximo provecho al Hardware que coloca en sus dispositivos el cual siempre se ha diferenciado considerablemente de los demás fabricantes.

La última versión del sistema operativo es iOS 14 (lanzada en septiembre de 2020) que sustituye a iOS 13.1 con el objetivo principal de mejorar la experiencia del usuario, integrando cambios relevantes, entre ellos el nuevo Modo Oscuro o la posibilidad de utilizar memorias externas con la app Archivos del sistema. De acuerdo con Apple, los modelos de iPhone compatibles con iOS 14, son:

iPhone 12 mini	iPhone X
iPhone 12	iPhone SE (2da generación)
iPhone 12 Pro	iPhone 8
iPhone 12 Pro Max	iPhone 8 Plus
iPhone 11	iPhone 7
iPhone 11 Pro	iPhone 7 Plus
iPhone 11 Pro Max	iPhone 6s
iPhone XR	iPhone 6s Plus
iPhone XS	iPhone SE (1era generación)
iPhone XS Max	

Tabla 2. Historial de versiones de IOS

Numero de versión	Fecha de lanzamiento	Status
3.1.3	2 de febrero de 2010	Versión Descontinuada
4.2.1	21 de noviembre de 2010	
5.1.1	7 de mayo de 2012	
6.1.6	23 de febrero de 2014	
7.1.2	29 de junio de 2014	
9.3.6	22 de julio de 2019	
10.3.4	22 de julio de 2019	
12.5	14 de diciembre de 2020	
13.7	1 de septiembre de 2020	
14.3	14 de diciembre de 2020	Última versión

2.3.5. Android

Android es un sistema operativo móvil basado en núcleo Linux y otros softwares de código abierto. Fue diseñado para dispositivos móviles con pantalla táctil, como teléfonos inteligentes, tabletas, relojes inteligentes (Wear OS), automóviles (Android Auto) y televisores (Android TV).

Inicialmente fue desarrollado por Android Inc., que adquirió Google en 2005. Android fue presentado en 2007 junto con la fundación del Open Handset Alliance (un consorcio de compañías de hardware, software y telecomunicaciones) para avanzar en los estándares abiertos de los dispositivos móviles. El código fuente principal de Android se conoce como Android Open Source Project (AOSP), que se licencia principalmente bajo la Licencia Apache. Android es el sistema operativo móvil más utilizado del mundo, con una cuota de mercado superior al 90 % al año 2018, muy por encima de iOS.

Las versiones de Android recibían hasta la versión 9, en inglés, el nombre de diferentes postres o dulces. En cada versión el postre o dulce elegido empieza por una letra distinta, conforme a un orden alfabético:

Tabla 3. Historial de versiones de Android

Nombre de código	Número de versión	Fecha de lanzamiento	Status
Apple Pie	1.0	23 de septiembre de 2008	Versión Descontinuada
Banana Bread	1.1	9 de febrero de 2009	
Cupcake	1.5	25 de abril de 2009	
Donut	1.6	15 de septiembre de 2009	
Eclair	2.0 – 2.1	26 de octubre de 2009	
Froyo	2.2 – 2.2.3	20 de mayo de 2010	
Gingerbread	2.3 – 2.3.7	6 de diciembre de 2010	
Honeycomb	3.0 – 3.2.6	22 de febrero de 2011	
Ice Cream Sandwich	4.0 – 4.0.5	18 de octubre de 2011	
Jelly Bean	4.1 – 4.3.1	9 de julio de 2012	
KitKat	4.4 – 4.4.4	31 de octubre de 2013	
Lollipop	5.0 – 5.1.1	12 de noviembre de 2014	
Marshmallow	6.0 – 6.0.1	5 de octubre de 2015	
Nougat	7.0 – 7.1.2	15 de junio de 2016	
Oreo	8.0 – 8.1	21 de agosto de 2017	Versión antigua pero vigente
Pie	9.0	6 de agosto de 2018	
10	10.0	3 de septiembre de 2019	
11	11.0	8 de septiembre de 2020	Última versión
12	12.0	agosto de 2021	Versión preliminar

2.3.6. Javascript

JavaScript (JS) es un lenguaje de programación ligero, interpretado, o compilado (just-in-time) con funciones de primera clase. Si bien es más conocido como un lenguaje de scripting (secuencias de comandos) para páginas web, y es usado en muchos entornos fuera del navegador, tal como Node.js, Apache CouchDB y Adobe Acrobat. JavaScript es un lenguaje de programación basada en prototipos, multiparadigma, de un solo hilo, dinámico, con soporte para programación orientada a objetos, imperativa y declarativa (por ejemplo, programación funcional). [20]

La peculiaridad de este lenguaje es que los propios navegadores se encargan de leer el código y llevar a cabo las órdenes que le indica, por lo que no necesita compilación. De este modo, facilita el desarrollo de la parte de la funcionalidad frontend en las aplicaciones web modernas y sitios web. Se utiliza para hacer que una página muestre actualizaciones de contenido más complejas como, por ejemplo, animaciones o interacciones con mapas. JavaScript hace que consigamos una mejor experiencia de navegación para el usuario sin perjudicar la velocidad de carga de la web.

2.3.7. Librerías JavaScript

En general, las librerías JavaScript son un código reutilizable que a menudo tiene un caso de uso principal. Las librerías proporcionan muchas funcionalidades estándar para que los desarrolladores no tengan que preocuparse por muchas funciones. Así, pueden usar estas para crear páginas web

fácilmente utilizando componentes de la interfaz de usuario, utilidades de lenguaje, funciones matemáticas y más. Una librería consta de varias funciones, objetos y métodos, según el idioma. Además, las puedes incluir en un proyecto sin depender de una estructura en particular. Es decir, libertad de usar una, dos o tantas librerías JavaScript como se requiera. Por ejemplo, algunas de las librerías JavaScript más útiles: jQuery, Moment, anime, Ramda, D3, Chart, MathJS, Hammer, React, React native, Redux, Glimmer, etc). [21]

2.3.8. CSS

Hojas de Estilo en Cascada (del inglés Cascading Style Sheets) o CSS es el lenguaje de estilos utilizado para describir la presentación de documentos HTML o XML (incluyendo varios lenguajes basados en XML como SVG, MathML o XHTML). CSS describe como debe ser renderizado el elemento estructurado en la pantalla, en papel, en el habla o en otros medios.

CSS es uno de los lenguajes base de la Open Web y posee una especificación estandarizada por parte del W3C. Desde CSS3, el alcance de las especificaciones se incrementó de forma significativa y el progreso de los diferentes módulos de CSS comenzó a mostrar varias diferencias, lo que hizo más efectivo desarrollar y publicar recomendaciones separadas por módulos. [22]

2.3.9. HTML

HTML (Lenguaje de Marcas de Hipertexto, del inglés HyperText Markup Language) es el componente más básico de la Web. Define el significado y la estructura del contenido web. Además de HTML, generalmente se utilizan otras tecnologías para describir la apariencia/presentación de una página web (CSS) o la funcionalidad/comportamiento (JavaScript). [23]

2.3.10.Framework JavaScript

Los frameworks de JavaScript son una parte esencial del desarrollo web front-end moderno, los cuales proveen a los desarrolladores herramientas probadas y testeadas para la creación de aplicaciones web interactivas y escalables. Muchas empresas modernas utilizan frameworks como parte estándar de sus herramientas, por lo que muchos trabajos de desarrollo front-end en la actualidad requieren experiencia en frameworks.

Cada framework destacado de JavaScript tiene un enfoque diferente para actualizar el DOM, manejar los eventos del navegador, y brindarte una experiencia de desarrollo satisfactoria. [24]

2.3.11.Aplicación móvil nativa

Las Apps nativas son aquellas aplicaciones que están desarrolladas para un equipo o plataforma determinada. Es decir, funciona en el equipo sin necesidad de ningún programa externo ya que se ha desarrollado en el lenguaje de programación específico de cada equipo. El término de App Nativa está habitualmente asociado a los dispositivos móviles y por tanto hay Apps Nativas para cada sistema operativo como iOS o Android.

2.3.12.Aplicación móvil híbrida

Las aplicaciones híbridas son aplicaciones móviles diseñadas en un lenguaje de programación web ya sea HTML5, CSS o JavaScript, junto con un framework que permite adaptar la vista web a cualquier vista de un dispositivo móvil. En otras palabras, no son más que una aplicación construida para ser utilizada o implementada en distintos sistemas operativos móviles, tales como, iOS, Android o Windows Phone, evitándonos la tarea de crear una aplicación para cada sistema operativo. De esta manera, una aplicación híbrida puede ser adaptada a múltiples plataformas

móviles sin crear nuevos códigos, pero ajustándose a algunos cambios operacionales para cada uno de ellos.

2.3.13.React native

React Native es un framework JavaScript para crear aplicaciones reales nativas para iOS y Android, basado en la librería de JavaScript React para la creación de componentes visuales, cambiando el propósito de estos para, en lugar de ser ejecutados en navegador, correr directamente sobre las plataformas móviles nativas, en este caso iOS y Android. Es decir, en lugar de desarrollar una aplicación web híbrida o en HTML5, lo que se obtiene al final como resultado es una aplicación real nativa, indistinguible de la que podría desarrollarse con código en Swift o Java.

React Native usa el mismo paradigma fundamental de construcción de bloques de UI (componentes visuales con los que interacciona el usuario) que las aplicaciones nativas reales de Android e iOS, pero gestiona la interacción entre los mismos utilizando las capacidades de JavaScript y React. [25]

Con esta idea de construcción de aplicaciones React Native nos proporciona las siguientes funcionalidades:

⇒ Compatibilidad Cross-Platform

La mayoría de las APIs de React Native lo son de por sí, lo cual ayuda a los propios desarrolladores a crear aplicaciones que puede ser ejecutados tanto en iOS como Android simultáneamente con el mismo código base.

⇒ Funcionalidad nativa

Las aplicaciones creadas mediante React Native funcionan de la misma manera que una aplicación nativa real creada para cada uno de los sistemas usando su lenguaje nativo propio. La unión de React Native junto con JavaScript permite la ejecución de aplicaciones más complejas de manera suave, mejorando incluso el rendimiento de las apps nativas y sin el uso de un WebView.

⇒ Actualizaciones instantáneas (para desarrollo y/o test)

Con la extensión de JavaScript, los desarrolladores tienen la flexibilidad de subir los cambios contenidos en la actualización directamente al dispositivo del usuario sin tener que pasar por las tiendas de aplicaciones propias de cada sistema y sus tediosos ciclos de procesos obligatorios previos. Hay que aclarar que este uno es exclusivo de versiones de desarrollo o para test, es ilegal, y puede llegar a conllevar castigos que llegan hasta la retirada definitiva de la aplicación si se realizan cambios directos sobre código con aplicaciones ya publicados y en producción. La tienda de Apple lleva un control muy exhaustivo sobre este tipo de prácticas.

⇒ Sencilla curva de aprendizaje

React Native es extremadamente fácil de leer y sencillo de aprender ya que se basa en los conceptos fundamentales del lenguaje JavaScript, siendo especialmente intuitivo tanto para los ya expertos en dicho lenguaje o incluso para las personas sin experiencia en él, ya que nos provee de un rango muy amplio de componentes, incluyendo ejemplo como los maps y filters típicos que se han usado siempre.

⇒ Experiencia positiva para el desarrollador

Si bien la curva de aprendizaje hemos dicho que es sencilla, también el propio lenguaje nos motiva y ayuda a la hora de la evolución según aumentamos nuestro conocimiento y dominio de este. Nos

ofrece varias características importantes como, por ejemplo, el Hot reloading que nos refresca la app en el momento en que guardamos cambios, y nos ofrece una gran ventaja para el desarrollo y testing de nuevas versiones, como hemos comentado arriba. O el uso del flexbox layout engine gracias al cual nos permite abstraernos de muchos de los tediosos detalles de la generación de cada uno de los layouts correspondientes a iOS y Android. Así como el uso del debugger de las herramientas de desarrollados del navegador Google Chrome, facilitando de sobre manera la tarea de depuración de código.

2.3.14. Google firebase

Firebase es una plataforma para el desarrollo de aplicaciones web y aplicaciones móviles desarrollada por Google en 2014.

Firebase permite la creación de mejores aplicaciones, minimizando el tiempo de optimización y desarrollo mediante diferentes funciones, entre las que destacan la detección de errores y el testeo, lo cual supone poder dar un salto de calidad a la aplicación. Poder almacenar toda la información en la nube y configurarla de manera distribuida, son las características más destacadas de Firebase.

Es una plataforma ubicada en la nube, integrada con Google Cloud Platform, que usa un conjunto de herramientas para la creación y sincronización de proyectos que serán dotados de alta calidad, haciendo posible el crecimiento del número de usuarios y dando resultado también a la obtención de una mayor monetización.

Los desarrolladores tienen una serie de ventajas al usar esta plataforma:

- ⇒ Sincronizar fácilmente los datos de sus proyectos sin tener que administrar conexiones o escribir lógica de sincronización compleja.
- ⇒ Usa un conjunto de herramientas multiplataforma: se integra fácilmente para plataformas web como en aplicaciones móviles. Es compatible con grandes plataformas, como IOS, Android, aplicaciones web, Unity y C++.
- ⇒ Usa la infraestructura de Google y escala automáticamente para cualquier tipo de aplicación, desde las más pequeñas hasta las más potentes.
- ⇒ Crea proyectos sin necesidad de un servidor: Las herramientas se incluyen en los SDK para los dispositivos móviles y web, por lo que no es necesario la creación de un servidor para el proyecto.

2.3.14.1. Firebase Analytics

Firebase Analytics es una aplicación gratuita que proporciona una visión profunda sobre el uso de la aplicación por parte de los usuarios.

2.3.14.2. Firebase Cloud Messaging

Antiguamente conocido como Google Cloud Messaging (GCM), Firebase Cloud Messaging (FCM) es una plataforma para mensajes y notificaciones para Android, iOS, y aplicaciones web que actualmente puede ser usada de forma gratuita.

2.3.14.3. Firebase Auth

Firebase Auth es un servicio que puede autenticar los usuarios utilizando únicamente código del lado del cliente. Incluye la autenticación mediante proveedores de inicio de sesión como Facebook, GitHub, Twitter, Google, Yahoo y Microsoft; así como los métodos clásicos de inicio de sesión

mediante correo electrónico y contraseña. Además, incluye un sistema de administración del usuario por el cual los desarrolladores pueden habilitar la autenticación de usuarios con email y contraseña que se almacenarán en Firebase.

Este servicio busca facilitar la creación de sistemas de autenticación, a la vez que mejora la incorporación, acceso y seguridad para los usuarios. Gracias a esto, el cliente no tiene que preocuparse por desarrollar métodos de autenticación clásicos, ya que Firebase le aporta de manera sencilla, eficaz y segura métodos para gestionar sus usuarios.

También aporta muchas funcionalidades extra, como la recuperación y verificación de cuentas, tanto por correo electrónico como por SMS, y cuotas de registro para los usuarios, todo esto gestionado mediante los servidores de la plataforma.

2.3.14.4. Realtime Database

Firebase proporciona una base de datos en tiempo real, back-end y organizada en forma de árbol JSON. El servicio proporciona a los desarrolladores de aplicaciones una API que permite que la información de las aplicaciones sea sincronizada y almacenada en la nube de Firebase. La compañía habilita integración con aplicaciones Android, iOS, JavaScript, Java, Objective-C, Swift y Node.js. La base de datos es también accesible a través de una REST API e integración para varios sistemas de Javascript como AngularJS, React, Ember.js y Backbone.js. La REST API utiliza el protocolo SSE (del inglés Server-Sent Events), el cual es una API para crear conexiones de HTTP para recibir notificaciones push de un servidor.

La sincronización en tiempo real de esta base de datos permite que los usuarios accedan a la información de sus datos desde cualquier dispositivo en tiempo real, compartiendo una instancia de Realtime Database, y cada vez que un usuario realice una modificación en esta, se almacena dicha información en la nube y se notifica simultáneamente al resto de dispositivos.

Una funcionalidad interesante de esta base de datos es que, si un usuario realiza cambios y pierde a la vez su conexión a Internet, el SDK de la plataforma usa una caché local en el dispositivo donde guarda estos cambios; y una vez que vuelve a tener conexión, automáticamente se sincronizan los datos locales.

2.3.14.5. Firebase Storage

Firebase Storage proporciona cargas y descargas seguras de archivos para aplicaciones Firebase, sin importar la calidad de la red. El desarrollador lo puede utilizar para almacenar imágenes, audio, vídeo, o cualquier otro contenido generado por el usuario. Firebase Storage se basa en el almacenamiento de Google Cloud Storage.

2.3.14.6. Firebase Cloud Firestore

Cloud Firestore es un servicio de almacenamiento de datos derivado de Google Cloud Platform, adaptado a la plataforma de Firebase. Al igual que Realtime Database, es una base de datos NoSQL, aunque presenta diversas diferencias. Se organiza en forma de documentos agrupados en colecciones, y en ellos se pueden incluir tanto campos de diversos tipos (cadenas de texto, números, puntos geográficos, referencias a la propia base de datos, arrays, booleanos, marcas de tiempo, e incluso objetos propios) como otras sub-colecciones. Entre sus limitaciones más destacadas encontramos la de no soportar las búsquedas de texto tipo "LIKE", eso es, buscar por sub-cadenas

del texto almacenado, y la de no poder filtrar las búsquedas con condiciones que impliquen más de un campo, si no es por búsquedas por el texto exacto.

2.3.15.Base de datos

Una base de datos es una colección organizada de información estructurada, o datos, típicamente almacenados electrónicamente en un sistema de computadora. Una base de datos es usualmente controlada por un sistema de gestión de base de datos (DBMS). En conjunto, los datos y el DBMS, junto con las aplicaciones que están asociados con ellos, se conocen como un sistema de base de datos, que a menudo se reducen a solo base de datos.

Los datos dentro de los tipos más comunes de bases de datos en funcionamiento hoy en día se modelan típicamente en filas y columnas en una serie de tablas para que el procesamiento y la consulta de datos sean eficientes. Luego se puede acceder, administrar, modificar, actualizar, controlar y organizar fácilmente los datos. La mayoría de las bases de datos utilizan lenguaje de consulta estructurado (SQL) para escribir y consultar datos. [26]

2.3.16.Base de datos relacionales

Las bases de datos relacionales se popularizaron en los años ochenta. Los elementos de una base de datos relacional se organizan como un conjunto de tablas con columnas y filas. La tecnología de base de datos relacional proporciona la manera más eficiente y flexible de acceder a información estructurada.

SQL es un lenguaje de dominio específico utilizado en programación usado por casi todas las bases de datos relacionales para consultar, manipular y definir datos, y para proporcionar control de acceso.

2.3.17.Base de datos no sql

Una NoSQL, o una base de datos no relacional, permite que los datos no estructurados y semiestructurados se almacenen y manipulen, a diferencia de una base de datos relacional, que define cómo deben componerse todos los datos insertados en la base de datos. Las bases de datos NoSQL se hicieron populares a medida que las aplicaciones web se hacían más comunes y complejas.

2.3.18.JSON

JSON es un formato de datos basado en texto que sigue la sintaxis de objeto de JavaScript.

Aunque es muy parecido a la sintaxis de objeto literal de JavaScript, puede ser utilizado independientemente de JavaScript, y muchos ambientes de programación poseen la capacidad de leer y generar JSON.

JSON está constituido por dos estructuras, una colección de pares de nombre/valor. En varios lenguajes esto es conocido como un objeto, registro, estructura, diccionario, tabla hash, lista de claves o un arreglo asociativo. Una lista ordenada de valores. En la mayoría de los lenguajes, esto se implementa como arreglos, vectores, listas o secuencias. Estas son estructuras universales; virtualmente todos los lenguajes de programación las soportan de una forma u otra. Es razonable que un formato de intercambio de datos que es independiente del lenguaje de programación se base en estas estructuras, a continuación, se muestra un ejemplo de un objeto JSON:

```
{  
  "project": {
```

```

        "name" : " Aplicación móvil de comercio para servicios turísticos ...",
        "id" : "b004"
    }
}

```

2.3.19. Internacionalización

La internacionalización es el proceso de diseño y desarrollo de un producto, una aplicación o el contenido de un documento mediante el cual se prepara un elemento o producto para permitir su adaptación con destino a audiencias de diferentes culturas, regiones o idiomas. En el caso de los programas informáticos implica prepararlo para poder ser traducido a varios idiomas, utilizar monedas diferentes o usar distintos formatos de fecha, por citar unos ejemplos. En algunos casos puede implicar pantallas o procesos de negocio diferentes. Así se puede decir que un programa informático está internacionalizado cuando permite su adaptación a diferentes regiones.

La palabra internacionalización a veces se escribe "i18n", donde 18 es la cantidad de letras entre la i y la n.

2.3.20. I18next

I18next es un framework escrito en y para JavaScript. Pero es mucho más que eso, i18next va más allá de proporcionar las características estándar de i18n como (plurales, contexto, interpolación, formato). Proporciona una solución completa para localizar el producto desde la web a dispositivos móviles y de escritorio. Trabaja bajo la filosofía “Aprende una vez, traduce en todas partes”. [27]

La comunidad i18next creó integraciones para marcos frontend como React, AngularJS, Vue.js y muchos más. También puede usar i18next con Node.js, PHP, iOS, Android y otras plataformas.



Figura 5. I18next Aprende una vez, traduce en todas partes

2.3.21. Geolocalización

La geolocalización es la capacidad para obtener la ubicación geográfica real de un objeto, como un radar, un teléfono móvil o un ordenador conectado a Internet. La geolocalización puede referirse a la consulta de la ubicación, o bien para la consulta real de la ubicación. El término geolocalización está estrechamente relacionado con el uso de sistemas de posicionamiento, pero puede distinguirse de estos por un mayor énfasis en la determinación de una posición significativa (por ejemplo, una dirección de una calle) y no solo por un conjunto de coordenadas geográficas. Este proceso es generalmente empleado por los sistemas de información geográfica, un conjunto organizado de

hardware y software, más datos geográficos, que se encuentra diseñado especialmente para capturar, almacenar, manipular y analizar en todas sus posibles formas la información geográfica referenciada.

2.3.22.API

El término API es una abreviatura de Application Programming Interfaces, que en español significa interfaz de programación de aplicaciones. Se trata de un conjunto de definiciones y protocolos que se utiliza para desarrollar e integrar el software de las aplicaciones, permitiendo la comunicación entre dos aplicaciones de software a través de un conjunto de reglas, sin necesidad de saber cómo están implementados. Esto simplifica el desarrollo de las aplicaciones y permite ahorrar tiempo y dinero. Las API le otorgan flexibilidad; simplifican el diseño, la administración y el uso de las aplicaciones, y proporcionan oportunidades de innovación, lo cual es ideal al momento de diseñar herramientas y productos nuevos (o de gestionar los actuales). [28]

Las API son un medio simplificado para conectar la propia infraestructura a través del desarrollo de aplicaciones nativas de la nube, pero también permiten compartir datos con clientes y otros usuarios externos. Las API públicas representan un valor comercial único porque simplifican y amplían la forma en que se conecta con los colaboradores y, además, pueden rentabilizar datos (un ejemplo conocido es la API de Google Maps).

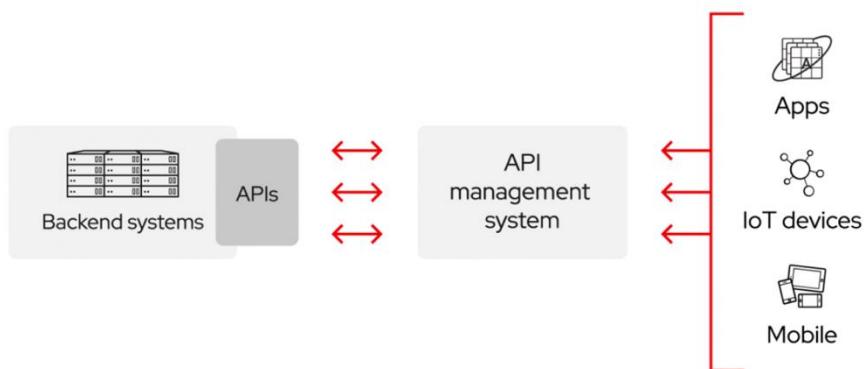


Figura 6. Ejemplo de arquitectura de integración con API

2.3.23.Entorno de desarrollo integrado

Un IDE es un entorno de desarrollo integrado que te permite maximizar la productividad como programador.

En pocas palabras, un IDE es un editor de texto con potentes funciones de gestión de proyectos que incluyen herramientas como: automatización de la compilación, un depurador y un intérprete. Algunos IDEs también incluyen navegadores de clases, objetos y diagramas de jerarquía de clases para cuando haces programación orientada a objetos, los IDE pueden ser aplicaciones por si solas o pueden ser parte de aplicaciones existentes.

2.4. Metodología de trabajo

2.4.1. Metodología ágil

Son el resultado de un nuevo enfoque que se basa en la pronta entrega de software incremental, proveniente de un desarrollo iterativo durante todo el ciclo de vida del software. Son ideadas en 1990, oponiéndose a las metodologías de desarrollo tradicional. Sus propulsores argumentaban que para obtener un software eficiente se debe buscar una flexibilización en las restricciones, en la burocratización del trabajo y de las comunicaciones.

En el año 2001, un grupo de desarrolladores, expertos y escritores sobre el tema, entre los que encontramos a Kent Beck, Alistair Cockburn, Martin Fowler y Jim Highsmith, firmaron el “Manifiesto para el desarrollo ágil de software” [31]. En él se describe lo citado a continuación:

“Estamos descubriendo formas mejores de desarrollar software tanto por nuestra propia experiencia como ayudando a terceros. A través de este trabajo hemos aprendido a valorar:

- ⇒ Individuos e interacciones sobre procesos y herramientas.
- ⇒ Software funcionando sobre documentación extensiva.
- ⇒ Colaboración con el cliente sobre negociación contractual.
- ⇒ Respuesta ante el cambio sobre seguir un plan.

Esto es, aunque valoramos los elementos de la derecha, valoramos más los de la izquierda”. Significa que, en las metodologías ágiles predominan los individuos, las entregas funcionales, la colaboración y la respuesta al cambio.

2.4.2. Programación extrema XP

Extreme Programming es una metodología de desarrollo de software ligera planteada por Kent Beck en el año 1999. XP unifica prácticas conocidas desde los inicios del desarrollo de software, que reunidas buscan satisfacer al cliente con software de desarrollo sencillo, facilitar la respuesta a los cambios que pueden experimentar las necesidades del cliente en el tiempo, y maximizar la productividad del grupo de trabajo.

XP se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico.

La principal suposición que se realiza en XP es la posibilidad de disminuir la mítica curva exponencial del costo del cambio a lo largo del proyecto, lo suficiente para que el diseño evolutivo funcione. Esto se consigue gracias a las tecnologías disponibles para ayudar en el desarrollo de software y a la aplicación disciplinada de las siguientes prácticas. Las características esenciales de XP se consideran; historias de usuario (requerimientos), roles, proceso y prácticas. [32]

2.4.3. Diagrama Modelo y Notación de Procesos de Negocio (BPMN)

Business Process Model and Notation (BPMN), es una notación gráfica estandarizada que permite el modelado de procesos de negocio, en un formato de flujo de trabajo. El objetivo de un BPMN es ofrecer una representación visual de un procedimiento de negocio específico para que todos los interesados puedan comprenderlo.

3. Análisis del sistema

En esta sección se define el alcance del proyecto en el que se evalúa la factibilidad (técnica, operativa, económica y de mercado), herramientas, tecnologías y requerimientos de desarrollo, especificaciones técnicas básicas necesarias para un usuario final, que deberá de cumplir para el correcto funcionamiento del sistema, requerimientos y metodología de desarrollo.

3.1. Alcance del sistema

3.1.1. Físicos y operativos

A modo de programación de la aplicación móvil, se desarrolla para teléfonos inteligentes IOS y Android, no se incluyen tabletas electrónicas u otros dispositivos con estos sistemas operativos.

De igual manera la aplicación móvil se trabajará bajo Expo go, la cual es una herramienta de desarrollo para crear experiencias con gestos y gráficos interactivos, usando JavaScript y React Native, de manera que no se realizaran los despliegues en las tiendas oficiales de cada sistema operativo móvil, pero es posible visualizar la aplicación como si lo estuviese en un ambiente productivo o real.

Dada la naturaleza social, económica y cultural de la zona arqueológica de Teotihuacán, los vendedores en su mayoría son personas locales al país o región, por lo que la traducción o adaptación de lenguaje no se considera para la aplicación móvil con rol de usuario vendedor, de la misma forma para la aplicación web del administrador tampoco es considerada, únicamente para los usuarios de rol turista, misma que podrán hacer un cambio entre inglés a español y viceversa.

3.1.2. Pagos electrónicos

Se contempla un único método de pago (PayPal) en esta etapa pilotito, donde únicamente se realizará la transferencia monetaria cliente vendedor, sin cobros intermedios o comisiones por parte de este proyecto. Más adelante se explica la elección de la pasarela de pago.

3.1.3. Turista

La calificación de servicios únicamente abarcara puntuación por estrellas, es decir no se contempla en esta idea piloto, la captura de reseñas o comentarios.

Para el caso de la compra de servicios de realizara de manera individual, es decir no incluye carrito de compras.

3.1.4. Vendedor

En la vida real hay empresarios y emprendedores que tienen más de un negocio o servicio ofertado al público, que tienen colaboradores remunerados, donde hay un jefe, donde solo este último puede hacer ciertas acciones y un colaborador (llámese cajero, mesero, host, etc.) tiene asignadas algunas otras responsabilidades, y algunas veces sucede que no están en el mismo lugar geográficamente hablando. En la realización de este ejercicio, únicamente se da la posibilidad a los vendedores de ofrecer uno o más servicios si y solo si, se encuentran en el mismo lugar geográfico, de igual manera solo hay una persona o dispositivo autorizado para realizar las transacciones de rol de vendedor por cada empresa que se registra, es decir no hay niveles jerárquicos o más roles de usuario para vendedores, por ende, únicamente una persona debe de acceder de manera simultanea a cada cuenta de usuario.

Solo los vendedores que cuentan con clave RFC licita ante hacienda será posible que oferten sus servicios en la aplicación, por lo que ahora no se considera el comercio informal dentro de las reglas de operación.

3.1.5. Administrador

La validación de vendedores consistirá únicamente y de manera manual, en aceptar o rechazar un vendedor, para esto se dirigirá a la institución o sitios correspondientes donde capturará el RFC del vendedor y determinará la acción correspondiente a capturar en el sistema.

3.2. Metodología

Para el desarrollo del sistema se realizará bajo la metodología ágil XP, se optó por este marco de trabajo considerando que el proyecto es desarrollado bajo la responsabilidad de una persona, mismo que todos los roles serán cubiertos de igual manera, el cliente son, la persona que propone la idea y los profesores asignados al proyecto que proporcionan retroalimentación con base a los artefactos que se esperan a lo largo del desarrollo, por lo que tal y como lo sugiere XP, se aplica en equipos de desarrollo pequeños, en el proyecto se busca que se desarrolle bajo el énfasis de implementación de la aplicación móvil, por lo que lo más importante en agilidad es el producto funcional y aunado con XP, es lo que tiene mayor prioridad. También al ser una idea piloto con sus respectivas limitantes y posibles cambios con base a retroalimentaciones constantes, se busca afrontar los cambios de manera simple e incremental en cada una de las iteraciones sin perder la calidad.

XP es especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico, más adelante se realiza el estudio de factibilidad técnica y operativa, dadas las tecnologías a utilizar, curva de aprendizaje y el recurso temporal, donde encontramos que el nivel de riesgo técnico es medio, además se pueden dar cambios en el proceso de desarrollo o se da la posibilidad de hacer limitaciones al proyecto, XP es adecuado para este proyecto.

Más adelante listaremos los requisitos funcionales (equivalente a historias de usuario tradicionales en XP, aumentando el nivel de profundidad en la especificación), a su vez, se distribuye en 5 módulos de desarrollo, es decir, habrá 5 iteraciones, cada una de estas iteraciones comienza con su análisis, diseño, codificación y pruebas.

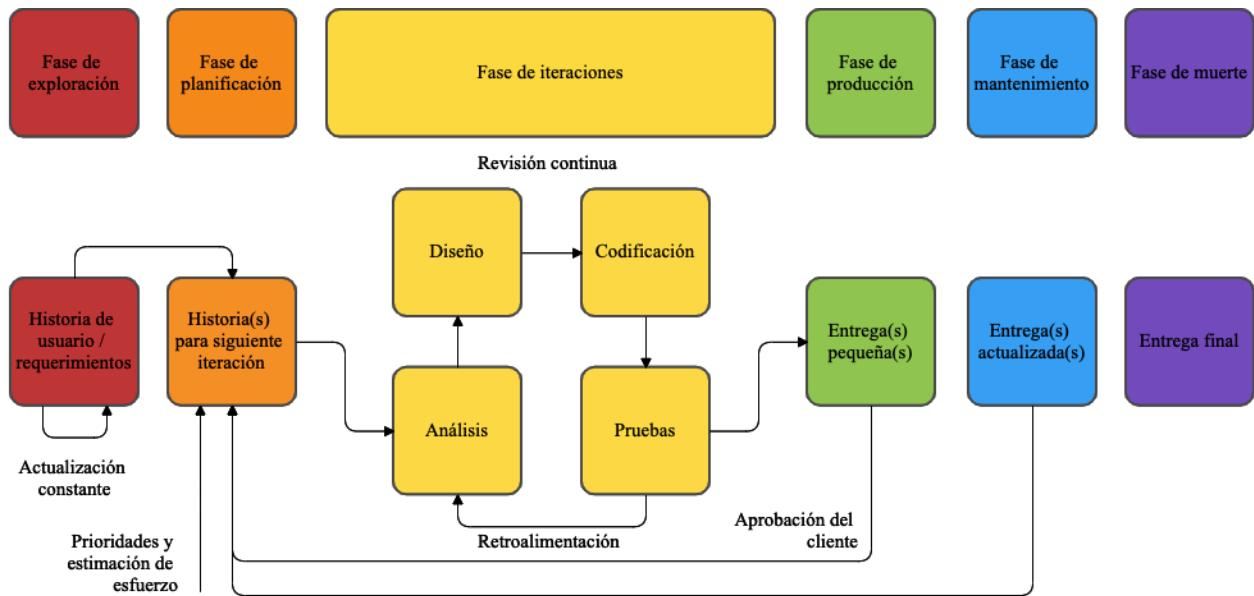


Figura 7. Metodología XP

3.3. Módulos de desarrollo

Ahora que ya se analizó la metodología de desarrollo que se utilizará y los recursos que el proyecto contempla, se segmentará en 5 módulos de desarrollo que serán implementados en el orden que a continuación se mencionan.

Aplicación móvil

En este primer módulo se realizará la maquetación de la aplicación móvil sin funcionalidad, se busca desarrollar el inicio de sesión, registro de usuarios, barra de navegación, formularios de registro de usuarios según el tipo de usuario, registro de servicios, pantalla de pagos, entre otros.

El objetivo de este módulo es obtener un panorama general visible de lo que se espera al final del desarrollo, preparar la navegación de los módulos siguientes, familiarizarse con el lenguaje de programación de frontend, estandarizar diseño, comportamientos de la aplicación móvil y de haber retroalimentación o cambios, serán reflejados en los siguientes módulos.

Lógica de negocio

En este modulo se desarrolla la lógica de negocio de los dos módulos anteriores, se ofrece el inicio de sesión y registro de usuarios, las transacciones en base de datos se realizan sin lógica de pagos, direccionamiento de avisos de privacidad, creación de servicios por parte del vendedor, compra por parte del turista, etc.

Idioma

El módulo de idioma esta basado en los dos módulos anteriores, se desarrollará la lógica de cambio de idioma de inglés a español y viceversa, definiendo los archivos correspondientes para el cambio estático de idioma de la aplicación, en el caso de datos almacenados en el servidor, realizar la copia o tabla de traducción correspondiente para las que así sea necesario y su visualización traducida.

Generación de códigos QR

En el módulo de generación de códigos QR se monta bajo la lógica de negocio tanto del usuario turista como vendedor, ya que ambos son capaces de generar este tipo de códigos por cada transacción, almacenarlos y actualizar vigencias de cada código QR que se crea con base a cada venta, la lectura por parte de la aplicación móvil de estos mismos códigos y su procesamiento.

Lógica de pagos

En el módulo final se desarrolla la lógica de pagos, que comienza cuando el turista compra un servicio, redirección de la aplicación que funge como pasarela de pago y descuento de la comisión correspondiente, avisos de estado de transacciones, y la conexión con el código QR correspondiente.

3.4. Elección de pasarela de pago

A continuación, se realiza la comparación de las pasarelas de pago más comunes y utilizadas en México respecto a las comisiones y disponibilidad del dinero.

Tabla 4. Comparativa de pasarelas de pago

Pasarela de pagos	Pagos con tarjeta	Pagos en efectivo	Disponibilidad del dinero en días
 conekta	2.9% + \$2.50 MXN	3.9%	2
 Openpay a BBVA COMPANY	2.9% + \$2.50 MXN	2.9% + \$2.50 MXN	7
 pagofacil.net	3.5%	2.9% + \$3 MXN	3
 2checkout	3.5% + \$0.35 USD	NA	7
 PayU	3.5% + \$4 MXN	3.5% + \$4 MXN	3
 stripe	3.6% + \$3 MXN	Modo beta	7
 mercado pago	3.49% + \$4 MXN	3.49% + \$4 MXN	0
 PayPal	3.95% + \$4 MXN	NA	7

De los cuales, Stripe, Mercado pago y PayPal son los más robustos., Desde que llegó Stripe oficialmente a México, son una gran opción como pasarela de pago con un excelente soporte técnico y servicio, es una de las mejoras opciones en este momento, lamentablemente su difusión y aceptación aun no es lo suficientemente madura, por otro lado, mercado pago destaca en que el tiempo de revisión es muy corto y el soporte técnico y atención es de primera y algo que cabe destacar, es que es posible bajar sus comisiones aumentando el plazo de pago, y finalmente, PayPal, es un viejo amigo con el que muchos comenzamos hace muchos años, sigue siendo una excelente opción, la ventaja es que muchos usuarios en el mundo confían, incluso prefieren pagar por medio de PayPal y su seguridad cumple con los mayores estándares en el ámbito, debido a su alcance a nivel turista y vendedor, la disponibilidad del dinero y la documentación existente para integración con aplicaciones móviles, elegiremos esta pasarela de pago para este proyecto.

3.5. Factibilidad

En el análisis de factibilidad revisaremos la disponibilidad de los recursos necesarios para llevar a cabo los objetivos del proyecto bajo el alcance señalado, la factibilidad se apoya en cuatro aspectos básicos, técnico, operativo, económico y de mercado.

Al final de este análisis esperamos concluir que el desarrollo de este proyecto bajo los objetivos y razón de ser del mismo sea viable y satisfactorio, de igual manera considerar posibles contratiempos, problemas o conflictos, con el propósito de evitarlos, planificarlos y considerarlos con respecto a los recursos limitados que se tienen.

3.5.1. Factibilidad técnica

Se evaluarán los recursos necesarios como herramientas, conocimientos, habilidades, experiencia, etc., para llevar a cabo las actividades y procesos que requiere el proyecto. Se recolecta la información necesaria sobre estos recursos para analizar si los recursos técnicos actuales son suficientes o deben complementarse.

Para la implementación del sistema se cuentan con las herramientas necesarias de software, puesto que existen, están disponibles y accesibles para su uso, cabe mencionar que todas las herramientas son completas en funcionalidades, soporte, documentación, etc.

Con respecto a los conocimientos necesarios para el desarrollo hay algunos aspectos a considerar, el proceso operativo, documental, de negocio y comunicación se considera que se tienen los conocimientos necesarios que fueron adquiridos a lo largo de la educación en el instituto, por otra parte los conocimientos técnicos de lenguaje de programación principal son en un 50% nuevos para el desarrollador, se comenzó desde una etapa temprana del proyecto con la capacitación externa a la institución para dominar dicho punto, y finalmente en el proceso de implementación es muy posible se requieran conocimientos complementarios, por lo que el riesgo, es nivel medio-bajo, dado el recurso humano de una persona en el proyecto se facilita el no empatar conocimientos con más personas, entonces el riesgo es aun menor y es aceptable.

De acuerdo con los requisitos del sistema se evalúan sus componentes bajo dos enfoques: hardware y software.

3.5.1.1. Hardware

Se requiere 1 equipo de computo para la documentación, desarrollo y pruebas de la aplicación móvil, para la ejecución y pruebas de la aplicación móvil en IOS es necesario un dispositivo móvil iPhone o bien una computadora MAC OS para la respectiva simulación, para el caso del SO Android no es indispensable tener un dispositivo con este SO, se pueden hacer las simulaciones correspondientes desde MAC OS, Linux o Windows.

Una vez mostradas las posibilidades que la tecnología nos ofrece, se detallan los recursos con los que cuenta el proyecto:

Tabla 5. Recursos de Hardware para desarrollo del proyecto

Dispositivo	Características	Comentarios
MacBook pro	⇒ 13 pulgadas en diagonal ⇒ 8 GB RAM	Es suficiente y bastante sobrado para los

	<ul style="list-style-type: none"> ⇒ 128 GB SSD ⇒ Procesador 2.7 GHz Dual-Core Intel Core i5 ⇒ Gráficos Intel Iris Graphics 6100 1536 MB ⇒ Sistema operativo MAC OS Big Sur 	requerimientos del proyecto, da la posibilidad de simular todos los modelos de iPhone actuales y dispositivos Android.
iPhone 7	<ul style="list-style-type: none"> ⇒ 4.7 pulgadas en diagonal ⇒ 32 GB Almacenamiento ⇒ Chip A10 Fusion ⇒ Cámara de 12 MP ⇒ Ubicación GPS asistido, GLONASS, Galileo y QZSS ⇒ Sistema operativo IOS 13 	Es suficiente para la ejecución de la aplicación móvil, dado que la aplicación no demandará grandes recursos de memoria y almacenamiento.

3.5.1.2. Sistema operativo

El sistema operativo base sobre el que se desarrolla el sistema es, macOs Big Sur versión 11.2.1, dicha versión se descargo y actualizo algunos meses posteriores a su lanzamiento con el objetivo de evitar posibles detalles de compatibilidad, a lo largo de todo este proyecto no se actualizara a nuevas versiones que la compañía lance en 2021, este sistema operativo se integra perfectamente con otros software de aplicación como lo son, el navegador web, el IDE, software de pruebas, el simulador propio de la empresa y la instalación de paquetes por medio de la terminal de comandos.

Para el caso de la ejecución de la aplicación móvil en el entorno real, el modelo de iPhone antes mencionado en el apartado de hardware, incluye el sistema operativo IOS 14.4 correspondiente al año 2020, esta versión de sistema operativo trabaja en armonía con el hardware del dispositivo, como lo son la cámara y el GPS, de igual manera con otras aplicaciones como lo son Expo go y PayPal.

En el verano de 2021, la empresa Apple lanzará una nueva versión del sistema operativo IOS, se actualizará el dispositivo en caso de estar en posibilidad de recibir la actualización y soporte. De igual manera para el sistema operativo Android, la versión actual es la 11.0, misma que es la versión con la que se harán las pruebas correspondientes a lo largo del desarrollo de la aplicación.

3.5.1.3. Lenguaje de programación principal

Se busca desarrollar una aplicación móvil para plataformas IOS y Android, vamos a analizar el framework de programación que mejor se adapta a los recursos y objetivos de la aplicación.

Tabla 6. Comparativo de lenguaje de programación principal

Framework	Características	Ventajas	Desventajas
Swift	<ul style="list-style-type: none"> ⇒ Desarrollo de aplicaciones IOS 	<ul style="list-style-type: none"> ⇒ Swift es un lenguaje expresivo y limpio que tiene una gramática y sintaxis simplificadas. ⇒ Menor competencia para un desarrollador que lo aprende. 	<ul style="list-style-type: none"> ⇒ Se requiere un dispositivo Mac OS para desarrollar ⇒ No soporta anotaciones @ ⇒ Encontrar desarrolladores de Swift puede ser un

		<ul style="list-style-type: none"> ⇒ Gestión de memoria automática 	<ul style="list-style-type: none"> problema, hay pocos en comparación con otras comunidades. ⇒ Curva de aprendizaje nivel medio ⇒ Rendimiento nativo ⇒ Muy estable
Kotlin	<ul style="list-style-type: none"> ⇒ Desarrollo de aplicaciones Android 	<ul style="list-style-type: none"> ⇒ No se requiere un dispositivo específico para desarrollar ⇒ Soporte de intelliJ y google ⇒ Compatible con java 	<ul style="list-style-type: none"> ⇒ Tamaño de APK grandes ⇒ Recursos limitados ⇒ Más lento que java ⇒ Todavía en la fase de evolución
React Native	<ul style="list-style-type: none"> ⇒ Desarrollo de aplicaciones IOS, Android y aplicaciones web ⇒ Desarrollado por Facebook en 2015 ⇒ Trabaja sobre componentes 	<ul style="list-style-type: none"> ⇒ No se requiere un dispositivo específico para desarrollar ⇒ Maduro, usado en producción por grandes organizaciones ⇒ Soporte por una comunidad muy grande ⇒ Buena aceptación en el mercado ⇒ Basado en lenguaje JavaScript ⇒ Mayor velocidad que flutter ⇒ Curva de aprendizaje fácil si se conoce JavaScript ⇒ Bueno para equipos pequeños y MVP 	<ul style="list-style-type: none"> ⇒ Se requiere aplicación de terceros para ejecutar la aplicación en pruebas ⇒ Cambios constantes
Flutter	<ul style="list-style-type: none"> ⇒ Desarrollo de aplicaciones IOS y Android ⇒ Desarrollado por google en 2018 ⇒ Trabaja sobre Widgets 	<ul style="list-style-type: none"> ⇒ No se requiere un dispositivo específico para desarrollar ⇒ Basado en lenguaje Dart 	<ul style="list-style-type: none"> ⇒ No tan maduro, pero creciendo ⇒ Menor cantidad de bibliotecas de componentes ⇒ Curva ligeramente inclinada, es necesario aprender Dart antes y lenguaje C ⇒ Flutter necesita más tiempo de desarrollo que React Native

El framework de programación que será utilizado en el presente proyecto será React Native, debido a que el desarrollo de aplicaciones móviles y web es más sencillo. Además, JavaScript nos da la posibilidad de compilar una sola vez el proyecto y poder ejecutarlo en cualquier plataforma sin tener que recomilarlo de nuevo. Asimismo, React Native nos brinda seguridad, portabilidad y fiabilidad, la comunidad y documentación que tiene es muy grande, lleva 2 años de ventaja sobre flutter que es el segundo marco de trabajo más viable para desarrollo multiplataforma.

De igual forma el lenguaje de programación detrás de React Native es JavaScript, este lenguaje es dominado por el desarrollador que encarga de la aplicación móvil, por lo que la curva de aprendizaje será aún más reducida, y para implementación web igual da la posibilidad, mientras que flutter no lo hace.

3.5.1.4. Framework de backend

Tabla 7. Comparativo de framework back-end a utilizar

Framework	Ventajas	Desventajas
Spring boot	<ul style="list-style-type: none"> ⇒ Lenguaje Java ⇒ Fácil configuración ⇒ Open source ⇒ Estandarizado 	<ul style="list-style-type: none"> ⇒ La aplicación se adapta al sistema
Django	<ul style="list-style-type: none"> ⇒ Desarrollo rápido ⇒ Open source ⇒ Fácil de aprender ⇒ MVT ⇒ Lenguaje Python 	<ul style="list-style-type: none"> ⇒ Templates no tan robustos ⇒ Se reinicia el servidor al recargar ⇒ ORM no tan robusto
Firebase	<ul style="list-style-type: none"> ⇒ Conexión con los lenguajes de programación más usados ⇒ Fácil configuración ⇒ Desarrollo por google ⇒ Plataforma en la nube ⇒ Disminuye el tiempo de programación backend, el acceso a datos y seguridad ya están implementados. ⇒ Infraestructura analítica, autenticación, monetización, hosting, reportes de fallos ya implementado. ⇒ Gran soporte por la comunidad y también por google de manera gratuita ⇒ El desarrollador se enfoca en la lógica y la visualización de datos. ⇒ El desarrollador no se preocupa por el hardware, lo hace google 	<ul style="list-style-type: none"> ⇒ Gratuito para desarrollo (Limitado a 100 conexiones y 1 GB de almacenamiento), coste de producción acorde al consumo unitario ⇒ Para aplicaciones gigantes como Amazon o Facebook, la base de datos no es compleja o robusta
Node	<ul style="list-style-type: none"> ⇒ JavaScript 	Limitado a una

	<ul style="list-style-type: none"> ⇒ High-performance ⇒ Open source ⇒ Asíncrono 	CPU
--	--	-----

Con base en la tabla anterior, el framework para desarrollar o implementar la parte back end (almacenamiento y manipulación de datos) será google Firebase, el cual provee todos los recursos que requiere la aplicación, como es el almacenamiento de archivos (imágenes), la manipulación de base de datos mediante archivos JSON, la facilidad con que se integra con el framework de programación principal que es React Native es tan simple como importarlo y hacer uso de las funcionalidades que Firebase provee, además que la seguridad se desliga y se robustece con google, por ejemplo los mecanismos de autenticación ya están resueltos, por lo que nos enfocaremos a la lógica de negocio y la programación visual de la aplicación.

3.5.2. Factibilidad operativa

En esta etapa de análisis se refiere a todos aquellos recursos donde interviene algún tipo de actividad (procesos), se depende de los recursos humanos que participan durante la operación del proyecto. Durante esta etapa se identifican todas aquellas actividades que son necesarias para lograr el objetivo y se evalúa y determina todo lo necesario para llevarla a cabo.

El listado de actividades a realizar en el desarrollo de este proyecto es el siguiente:

- ⇒ Comunicación con directores y sinodales asignados al proyecto
La comunicación efectiva es vital para atender correcciones y aplicar mejoras que se puedan aplicar en tiempo y forma, afortunadamente las tecnologías de la información hacen sencillo este proceso.
- ⇒ Análisis, diseño y documentación total del proyecto
La distribución de tareas acorde a la metodología XP a utilizar, se realizó de manera estratégica, ya que primero se codifica la aplicación sin funcionalidad, con esto se garantiza que el desarrollador se familiarice con React Native, al comenzar a madurar el proyecto, se es más sencillo implementar y codificar los últimos 2 módulos referentes al pago, así mismo la documentación constante es una buena técnica para cubrir los requisitos académicos que solicita el proyecto y evitar desfases significativos de avance documental y realista, dada la metodología XP a utilizar que no se enfoca demasiado en la documentación técnica del sistema y más en programación, se documentará solo lo esencial, por lo que no deberá de generar grandes requerimientos de tiempo, exceptuando los primeros grandes apartados de este reporte técnico.
- ⇒ Implementación y pruebas de la aplicación móvil
Los primeros dos módulos darán una pauta general visual de lo que se espera obtener como resultado final al término del desarrollo de la aplicación, estos dos serán la guía para los 3 últimos módulos de desarrollo, a los cuales se les dará lógica conforme madure la aplicación.
Se sabe que, al implementar la aplicación, el desarrollador tiene que probar y verificar que funciona correctamente, por lo que la etapa de pruebas será rápida, esto debido con que se realizarán pruebas por cada modulo, entonces la integración entre cada uno de estos se espera que sea limpia y con bajo acoplamiento y alta cohesión entre los mismos.

Para el caso de la lógica de pagos es donde se podrían encontrar ciertas dificultades ya que es un tema nuevo para el desarrollador, de. Igual manera hay una basta cantidad de documentación.

⇒ Presentación de resultados

Se realizarán dos presentaciones a lo largo del proyecto, presentación de trabajo terminal 1 y presentación de trabajo terminal 2, las cuales mostrarán los avances de cada periodo, la primera presentación abarcara hasta el módulo #2, el módulo #3 se desarrolla en el periodo comprendido entre la primera presentación y comienzo del segundo semestre académico o trabajo terminal 2, por lo que la segunda presentación abarcara los módulos restantes.

Operativamente el proyecto es viable, debido a que se inicio de manera temprana con la capacitación en tecnologías desconocidas, documentación en tiempo y forma, la distribución de módulos y tiempos de trabajo es estratégica, las tecnologías a utilizar tienen una curva de aprendizaje bastante baja debido a los conocimientos previos que se tienen, a la armonía y cohesión con que se integran dichas tecnologías entre si, y se hace énfasis en el desarrollo.

3.5.3. Factibilidad de mercado

En la factibilidad de mercado se analizarán algunos factores relacionados con el mercado para determinar si el proyecto podría ser exitoso, se basa en las encuestas de turista y vendedor que fueron aplicadas (Ver Encuesta a turistas y Encuesta a vendedores) y algunos datos numéricos abiertos por el gobierno de México.

Tamaño de la muestra

Es aquel número determinado de encuestas que componen la muestra extraída de una población, necesarios para que los datos obtenidos sean representativos de la población, calculamos el tamaño de las muestras necesarias para la aplicación de encuestas a turistas y vendedores. (Ver)

Para encuestas a turistas, dado que la población es mayor a 100,000 habitantes, emplearemos la siguiente expresión:

$$n = \frac{k^2 P Q}{e^2}$$

$$k = 95.5\% = 2$$

$$p = 0.7$$

$$q = 0.3$$

$$e = 10\%$$

Entonces:

$$n = \frac{2^2 * 0.7 * 0.3}{0.1^2} = 84$$

Para encuestas a vendedores, dado que la población es menor a 100,000 habitantes, emplearemos la siguiente expresión:

$$n = \frac{k^2 P Q N}{0.1^2(N - 1) + k^2 P Q}$$

$$N = 100$$

$$k = 80\% = 1.28$$

$$p = 0.7$$

$$q = 0.3$$

$$e = 15\%$$

Entonces,

$$n = \frac{1.28^2 * 0.7 * 0.3 * 100}{0.15^2(100 - 1) + 1.28^2 * 0.7 * 0.3} = 13.37$$

Por lo que el tamaño de muestras es de 84 y 13 para encuestas a turistas y vendedores respectivamente.

De acuerdo con la distribución por rango de edad (Ver Figura 124. Encuesta a turistas - Pregunta #3), el 50% de los turistas encuestados tiene entre 18 y 22 años, el 20.4% tiene entre 23 y 25 años, el 9.2% tiene entre 26 y 30 años, 10.2% entre 31 y 40 años y 8.2% entre 41 y 60 años, es decir, entre 18 y 60 años se concentra el 98% de la muestra, de igual manera en el corto plazo, el 48% de la muestra tiene entre 23 y 60 años, este segmento de población es la que en general tiene el poder adquisitivo propio, salud e interés para realizar viajes como lo es en la zona de Teotihuacán, a nivel nacional representa alrededor del 52% de la población total, por lo que el cliente objetivo es aquel turista entre 22 y 60 años de edad.

El 88.8% de los encuestados dijo haber visitado Teotihuacán (Ver Figura 126. Encuesta a turistas - Pregunta #6), donde el 51% lo ha hecho entre 2 y 5 veces, el 14.3% más de 10 veces, el 13.3% una vez y el 9.2% entre 5 y 10 veces, por lo que es apreciable reconocer que la zona es un lugar para viajar más de una vez, así mismo, dentro de la zona de Teotihuacán existen pueblos vecinos denominados con encanto o mágicos, donde solo el 65.3% conoce o ha escuchado hablar de estos lugares (Ver Figura 127. Encuesta a turistas - Pregunta #7), tal y como se planteo en los objetivos del proyecto, se busca dar visibilidad a estos lugares cercanos que se puede observar que aproximadamente el 40% de los turistas nacionales no está enterado de dicha oferta.

Con respecto a lo anterior, se ve reflejado en el conocimiento de la oferta de servicios (Ver Figura 128. Encuesta a turistas - Pregunta #8), ya que por cuestiones de espacio, económicas, culturales, sociales y políticas, los servicios no están geográficamente en una única ubicación, por ende hay servicios que el turista convencional no percibe, por ejemplo; la renta de vehículos (Bicicletas, motocicletas y automóviles) solo el 28.6% de los encuestados dice saber de su existencia, 12.2% sabe de la existencia del Zoológico, el 27.6% sabe de la presencia de bares, antros, clubes nocturnos o actividades de acampado, 13.3% sobre balnearios y spa, tirolesa o deportes extremos únicamente el 11.2%, 16.3% reconoce que la actividad Gotcha es ofertado, inesperadamente solo el 5.1% sabe de la realización de festivales musicales y el 1% del espectáculo de luces por la noche en la zona arqueológica, es evidente el desconocimiento de estas grandes atracciones que son poco conocidas, cuando se hace la pregunta relacionada en contraste a que servicios el turista ha adquirido, la distribución es aun más precaria, aproximadamente los servicios mencionados pero con disminución del 70% en consumo (Ver Figura 129. Encuesta a turistas - Pregunta #9), por otro lado, se les pregunta en la encuesta de turistas sobre los servicios ofertados en la zona, si les gustaría conocer o volver a disfrutar en Teotihuacán alguno de los servicios mencionados anteriormente, 4 de cada 10 personas dijo que si (Ver Figura 130. Encuesta a turistas - Pregunta #10).

Del lado del vendedor, indicaron que sus clientes se enteran de sus servicios principalmente por redes sociales, recomendaciones de otras personas que ya consumieron el servicio o familiares del negocio, tienen alguna pagina web o bien, están registrados en sitios de viajes (Ver Figura 146. Encuesta a vendedores - Pregunta #03).

El 60.2% de los encuestados dice utilizar transporte privado para llegar a las pirámides, el 33.7% transporte público y el 66.7% de los vendedores afirma que existe algún medio de transporte público para llegar a la locación de sus servicios (Ver Figura 157. Encuesta a vendedores - Pregunta #14), tal y como lo dice Torruco [6], la integración de servicios, seguridad y coordinación es la estrategia para seguir, en la zona no hay servicio de Uber, el medio de transporte desde la ciudad de México más eficiente es por autobús, esta es un área de oportunidad a considerar posterior a la idea piloto que se estudia ahora.

Cuando se le propone al turista una aplicación como la que incluye este proyecto, sobre si se interesa o si la pudiera usar, el 76.5% responde que si y el 20.4% dice “tal vez”, entonces por parte de los turistas se espera buena aceptación.

El 66.7% de los vendedores encuestados dijo que estaría interesada en una aplicación como la que se propone en este proyecto, respecto al 33.3% que indicó como respuesta “tal vez”, de igual manera el 100% contesto que si le interesa tener más clientes que sean turistas internacionales, dado que en promedio únicamente 2 de cada 10 turistas son extranjeros (Ver Figura 168. Encuesta a vendedores - Pregunta #25).

Con respecto a los pagos, el 40.8% prefiere reservar o pagar por el servicio, horas, días o semanas antes, el 33.7% se inclina por pagar cuando ingresa al establecimiento y el 25.5% prefiere pagar cuando termina de disfrutar el servicio, entonces observamos que los 3 tiempos de pago tienen preferencias con pesos bastante significativos, por lo que se debe de tratar de incluirlos para continuar con la aceptación esperada. Según los vendedores encuestados, el 66.7% cobra por el servicio, horas, días o semanas antes de entregar el servicio y el 33.3% lo hace cuando el turista se va (Ver Figura 162. Encuesta a vendedores - Pregunta #19), además de todos dar la posibilidad de agendar con tiempo previo.

El 35.7% paga más con dinero en efectivo, el 27.6% lo hace con tarjeta bancaria en físico por medio de una terminal, el 20.4% desde transferencias bancarias con la aplicación móvil de su banco y el 11.2% desde la pasarela de pago mas común, PayPal. Solo el 68.4% de los encuestados confían en los pagos electrónicos, por lo que un aspecto a considerar sin duda alguna es el cuidado de los datos personales y monetarios de los usuarios, informarles sobre las políticas de privacidad para generar confianza, etc.

En el 2020 comenzó en México la pandemia COVID 19, desafortunadamente para el sector del turismo no es una oportunidad, en la encuesta de turistas, el 50% dijo que no visitaría la zona mientras la pandemia es parte de la vida diaria, el 25.5% dijo que tal vez lo haría y el 24.5 dijo que, si realizaría una visita, por parte de los vendedores, el 100% de ellos han adecuado sus establecimientos para hacer frente a la pandemia (Ver Figura 137. Encuesta a turistas - Pregunta #17).

Todos los pronósticos indican que el momento en que la pandemia este controlada o deje de representar un peligro para la vida, el turismo se disparará como nunca en la historia se ha visto, mismo que el 98% de los encuestados dijo que saldría de vacaciones o realizar alguna actividad turística cuando ese momento llegue (Ver Figura 143. Encuesta a turistas - Pregunta #23).

La gran mayoría de los vendedores que contestaron la encuesta dice ofrecer servicio bilingüe (Ver Figura 147. Encuesta a vendedores - Pregunta #04), esto en la práctica no siempre es así, además de por la propia naturaleza de un turista extranjero y posiblemente adaptado a la tecnología, preferiría utilizar su dispositivo para tener menor grado de incertidumbre.

En la siguiente tabla se observa el flujo turístico en la zona arqueológica de Teotihuacán en el periodo comprendido del 2016 al 2020, la cual hasta el 2019 los resultados del ejercicio turístico fueron realmente buenos, en el 2020 por causas de la pandemia esos números bajaron, aun así, se espera que lentamente se recupere la actividad, por lo que el mercado es amplio.

Tabla 8. Histórico de turismo en Zona arqueológica de Teotihuacán 2016 - 2020

Año	Periodo	Nacional		Extranjero		Total periodo (Turistas)	Total anual (Turistas)
2016	Enero-Junio	1,613,523	85%	279,433	15%	1,892,956	5,745,085
	Julio-Diciembre	3,216,397	83%	635,732	17%	3,852,129	
2017	Enero-Junio	1,666,207	85%	288,379	15%	1,954,586	6,139,603
	Julio-Diciembre	3,513,346	84%	671,671	16%	4,185,017	
2018	Enero-Junio	1,588,021	84%	293,453	16%	1,881,474	5,948,672
	Julio-Diciembre	3,400,732	84%	666,466	16%	4,067,198	
2019	Enero-Junio	1,459,367	82%	314,022	18%	1,773,389	3,459,528
	Julio-Diciembre	1,320,041	78%	366,098	22%	1,686,139	
2020	Enero-Junio	405,004	70%	169,480	30%	574,484	1,148,968
	Julio-Diciembre	405,004	70%	169,480	30%	574,484	

Con base a la tabla anterior, podemos obtener algunos promedios:

Tabla 9. Promedio de turismo histórico previo a pandemia

Tipo turismo	Enero-Junio (Turistas)	Julio-Diciembre (Turistas)	Total (Turistas)
Nacional	1,581,780	3,376,825	4,958,605
internacional	293,822	657,956	951,778
			5,910,383

Tabla 10. Promedio de turismo histórico durante pandemia

Tipo turismo	Enero-Junio (Turistas)	Julio-Diciembre (Turistas)	Total (Turistas)
Nacional	405,004	862,522	1,267,527
internacional	169,480	267,789	437,269
			1,704,796

3.5.4. Factibilidad económica

En el estudio de la factibilidad económica se suele evaluar si, desde un punto de vista económico y financiero, un proyecto puede llevarse a cabo, mantenerse en marcha y generar valor. Por tanto, permite conocer si existen o se pueden conseguir los recursos económicos y financieros necesarios para llevar a cabo un negocio.

Dados los alcances de este proyecto y algunas limitantes (tiempo, acceso a información real en el contexto del proyecto, pandemia covid 19, objetivos de la realización del trabajo terminal, etc.), únicamente se calculará el costo aproximado para el desarrollo del proyecto y una propuesta de plan de ventas con el objetivo de recuperación de inversión y mantenimiento del sistema.

3.5.4.1. Costo aproximado del proyecto

Para realizar este cálculo se detalla a continuación los costos en tres tópicos, costos operativos, costos de software, hardware y costos de personal, de cada tópico se consideran 10 meses de ocupación, mismo que corresponde con los períodos escolares del calendario académico.

En los costos operativos se consideran los costos de renta para servicios y alojamiento.

Tabla 11. Costos operativos

Producto / Servicio	Precio unitario por mes [Pesos]	Subtotal (Precio unitario * 10 meses) [Pesos]
Internet	\$ 480	\$ 4,800
Luz	\$ 80	\$ 800
Escritorio (Inversión inicial)	\$ 2000	\$ 2,000
Silla (Inversión inicial)	\$ 1500	\$ 1,500
Papelería	\$ 50	\$ 500
Renta de habitación (Sin considerar gastos personales y amenidades)	\$ 1800	\$ 18,000
Total		\$ 27,600

En los costos de software y hardware se consideran las licencias y compra de equipo de cómputo. De manera específica para el equipo de cómputo se busca la posibilidad de renta de equipo por mes, dado que son 10 meses no es eficiente, por lo que se pretende realizar una inversión inicial.

Tabla 12. Costos de software y hardware

Producto / Servicio	Precio unitario por mes [Pesos]	Subtotal (Precio unitario * 10 meses) [Pesos]
Macbook Pro 13 Retina 2015 Core I5, 128gb Ssd, 8gb Ram (Inversión inicial)	\$ 15,0000	\$ 15,000
React Native	\$ 0	\$ 0
Cacoo	\$ 0	\$ 0
Microsoft 365 Empresa Básico	\$ 97	\$ 970
Google firebase Plan spark	\$ 0	\$ 0
Google localización API	\$ 0	\$ 0
Google traductor API	\$ 0	\$ 0

Total	\$ 15,970
-------	-----------

En los costos de personal, únicamente se incluye una persona a cargo del desarrollo del proyecto, por lo que asignaremos salarios promedio para obtener un costo aproximado de dicho gasto.

Tabla 13. Costos de desarrollo

Rol	Precio unitario por mes [Pesos]	Subtotal (Precio unitario * 10 meses) [Pesos]
Desarrollador móvil tiempo completo en México [31]	\$ 22,000	\$ 220,000
Total		\$ 220,000

Con base a las tablas anteriores, el aproximado total del proyecto es de \$ 263,570 pesos comprendidos en 10 meses de duración que tiene el proyecto en el año 2021.

3.5.4.2. Plan de monetización

Con base a la encuesta a turistas, la pregunta “¿Cuál de las siguientes es más cómoda para ti?” (Ver Figura 137. Encuesta a turistas - Pregunta #17), que refiere al plan de pago de la aplicación, el 66.3% de los encuestados indicó preferir que la aplicación sea gratuita e incluya publicidad, mientras que el 33.7% podría pagar por el uso de la aplicación.

Por otro lado, la encuesta a vendedores, la pregunta “Si ofrecieras tus servicios al utilizar una aplicación móvil ¿Qué plan de negocio es más rentable para tu empresa?” (Ver Figura 165. Encuesta a vendedores - Pregunta #22), que refiere al plan de pago de la aplicación, donde el 66.7 % de los proveedores de servicios prefieren ellos no pagar a la aplicación, pero que se generen comisiones adicionales para el turista por cada venta, y el 33.3% restante de los proveedores de servicios prefiere pagar cierta cantidad establecida por cada venta.

Con el objetivo de una posible recuperación de inversión y con base a lo anterior, se propone que los vendedores oferten sus servicios a su precio habitual, luego la aplicación móvil calcula una comisión (que incluye la comisión de la pasarela de pago y la comisión de la aplicación), entonces se suma el costo del servicio y la comisión, eso es lo que los turistas tendrían que pagar, la aplicación móvil al recibir el pago, descuenta la comisión y hace los pagos correspondientes por transacción (Ver Figura 7. Proceso de monetización).

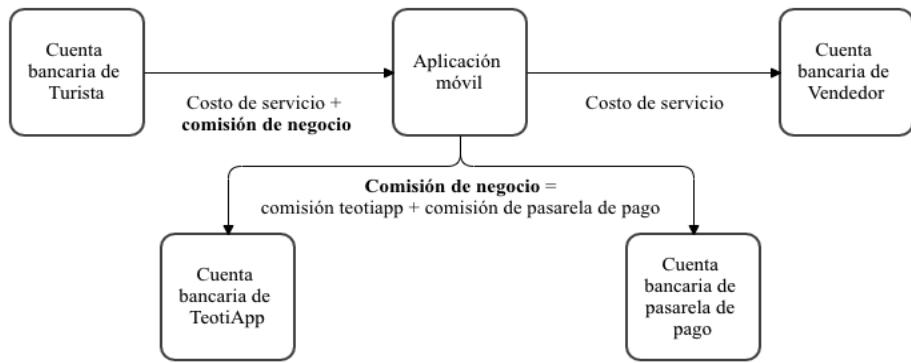


Figura 8. Proceso de monetización

Con base en la factibilidad de mercado descrita anteriormente, y de acuerdo con el Perfil del turista en CDMX [34], se pueden resumir de manera generalizada los siguientes puntos:

- El 48% de la muestra tiene entre 23 y 60 años, es decir tiene cierto poder adquisitivo.
- El 98% de los encuestados dijo que saldría de vacaciones o realizar alguna actividad turística.
- El 76.5% de los turistas encuestados responde que sí y el 20.4% dice “tal vez” asistiría a Teotihuacán.
- El 66.7% de los vendedores encuestados dijo que estaría interesada en una aplicación similar.
- En una normalidad sin pandemia, al año ingresan 4,958,605 turistas nacionales a Teotihuacán, representa el 83.85% del total.
- En una normalidad sin pandemia, al año ingresan 951,778 Turistas Internacionales a Teotihuacán, representa el 16.15% del total.
- El gasto promedio diario de los turistas nacionales es de \$788 MXN.
- El gasto promedio diario de los viajeros foráneos es de \$2,058 MXN.
- Comisión PayPal de 3.95% + \$4 MXN.
- Comisión TeotiApp, ejemplo de 1.0%
- Monto total de inversión del proyecto de \$ 263,570 MXN.

Entonces, de acuerdo con el plan de negocios propuesto, el retorno de la inversión se estima:

$$X = \text{Tamaño del mercado de turistas nacionales al año} = 4,958,605 \text{ Turistas Nacionales anuales} \\ * 66.7\% \text{ interés por la aplicación} = 3,307,389 \text{ Turistas Nacionales anuales.}$$

$$Y = \text{Tamaño del mercado de turistas internacionales al año} = 951,778 \text{ Turistas Internacionales anuales} \\ * 66.7\% \text{ interés por la aplicación} = 634,836 \text{ Turistas Internacionales anuales.}$$

$$Z = \text{Ganancia de acuerdo con el Gasto promedio por día de turista nacional con comisiones PayPal y teotiApp} = \text{Gasto promedio por día de turista nacional} * \text{Comisión TeotiApp} = \$788 \text{ MXN} * 1.0\% = \$ 7.88 \text{ MXN}$$

$W = \text{Ganancia de acuerdo con el Gasto promedio por día de turista Internacional con comisiones PayPal y teotiApp} = \text{Gasto promedio por día de turista Internacional * Comisión}$
 $\text{TeotiApp} = \$2,058 \text{ MXN} * 1.0\% = \$ 20.58 \text{ MXN}$

Hagamos el cálculo de 1 año, entonces

$$X*Z + Y*W = (3,307,389 \text{ Turistas Nacionales anuales} * \$ 7.88 \text{ MXN}) = \$26,062,225.34 \text{ MXN} + \\ (634,836 \text{ Turistas Internacionales anuales} * \$ 20.58 \text{ MXN}) = \$ 13,064,924.88 \text{ MXN} \\ = \$ 39,127,150.22 \text{ MXN anuales}$$

Este número es excesivamente positivo, vamos a suponer que del 66.7% de las personas que dicen tener interés por una aplicación como esta, por ejemplo, únicamente el 10% es quien, si utiliza la aplicación y hace compras dentro de la misma, entonces tenemos un 6.67% del total de los encuestados como mercado, realizaremos nuevamente los cálculos anteriores con esta nueva cantidad (Z, W permanecen constantes):

$$X = \text{Tamaño del mercado de turistas nacionales al año} = 4,958,605 \text{ Turistas Nacionales anuales} \\ * 6.67\% \text{ interés por la aplicación} = 330,738 \text{ Turistas Nacionales anuales.}$$

$$Y = \text{Tamaño del mercado de turistas internacionales al año} = 951,778 \text{ Turistas Internacionales anuales} \\ * 6.67\% \text{ interés por la aplicación} = 63,483 \text{ Turistas Internacionales anuales.}$$

Hagamos el cálculo de 1 año, entonces

$$X*Z + Y*W = (330,738 \text{ Turistas Nacionales anuales} * \$ 7.88 \text{ MXN}) = \$2,606,222.53 \text{ MXN} + \\ (63,483 \text{ Turistas Internacionales anuales} * \$ 20.58 \text{ MXN}) = \$ 1,306,492.48 \text{ MXN} \\ = \$ 3,912,715.022 \text{ MXN anuales}$$

Por lo que tanto en un escenario ideal y pesimista, al término de un año se ve sobrado el retorno de la inversión, cabe destacar que hay infinidad de factores que podrían alterar este resultado y variables no tomadas en cuenta, este es un resultado superficial a modo de muestra.

3.5.5. Conclusiones del análisis de factibilidad

Ahora que se ha analizado el proyecto bajo 4 puntos (técnico, operativo, financiero y de mercado), se dice que el proyecto es viable con los recursos que se tienen, los objetivos de este, ya que técnicamente no hay complejidad ni dificultades considerables, operativamente hablando requiere un esfuerzo bastante prolongado, pero de seguir al pie de la letra los tiempos, metodología, planeación y delimitación del proyecto, se puede terminar en tiempo y forma, financieramente es viable con recuperación económica al mediano plazo con alguna forma de negocio (puede ser o no la propuesta) y en el mercado parece que pudiese ser un proyecto exitoso que satisface los planteamientos y necesidades de turistas y vendedores, este proyecto en la vida real esta al igual que el mundo entero, inmerso en la incertidumbre de la pandemia.

3.6. Usuarios del sistema

En este apartado se definen todos los usuarios bajo el contexto en el que se desarrolla la aplicación y los requerimientos para cada uno.

3.6.1. Turista

Un turista es aquella persona que se traslada y que se encuentra realizando un viaje entre dos lugares geográficamente distintos, otras regiones o países diferentes del propio y fuera de su entorno habitual, hacia con la finalidad de pasar allí momentos de ocio, conocer otras culturas, visitar lugares específicos que están ausentes en la región de residencia habitual, etc.

Puede entenderse que el rol del turista es el consumidor que puede registrarse en la aplicación, ver y comprar servicios disponibles que se muestran en la aplicación, relativos a la estadía en el valle de Teotihuacán.

Perfil

- ⇒ Conocimientos de idioma español o inglés.
- ⇒ Agilidad en uso básico de teléfonos móviles.
- ⇒ Estar registrado en PayPal.
- ⇒ Interés por viajar a Teotihuacán.

3.6.2. Vendedor

Un prestador de servicios turísticos es una persona física o moral que ofrece, proporciona, o contrata con el turista, la prestación de los servicios a que se refiere esta Ley. [29]

Puede entenderse que el rol del vendedor es el prestador de servicios turísticos, que puede registrarse en la aplicación, ver y publicar sus servicios disponibles que se mostraran al turista en la aplicación, relativos a la estadía en el valle de Teotihuacán.

Perfil

- ⇒ Facilidad de comunicación y servicio al cliente.
- ⇒ Estar registrado en S.A.T.
- ⇒ Agilidad en uso básico de teléfonos móviles.
- ⇒ Estar registrado en PayPal.

3.6.3. Administrador

El administrador del sistema tiene básicamente un objetivo o tarea a realizar, se encarga de validar las claves RFC de los vendedores ante la pagina WEB del SAT, esto con el propósito de evitar fraudes y dar mayor confianza a los turistas.

Perfil

- ⇒ Facilidad de comunicación y servicio al cliente.
- ⇒ Conocimiento y experiencia básica de S.A.T.
- ⇒ Agilidad en uso básico de computadora.
- ⇒ Habilidades blandas: Responsabilidad y Honestidad.

3.7. Modelo de dominio

El modelo de dominio se crea con el fin de representar el vocabulario y los conceptos clave del dominio del problema. El modelo de dominio también identifica las relaciones entre todas las entidades comprendidas en el ámbito del dominio del problema, y comúnmente identifica sus atributos.

Tabla 14. Entidad Turista

Nombre	Tipo	Descripción
Nombre	Palabra corta	Nombre o nombres del usuario.
Apellido	Palabra corta	Primer apellido del usuario.
Correo electrónico	Correo, identificador	Correo electrónico paypal del usuario, identificador dentro del sistema.
Fecha de nacimiento	Fecha	Fecha de nacimiento del usuario.
Género	Género	Género del usuario
Nacionalidad	Nacionalidad	Nacionalidad del usuario
Contraseña	Contraseña	Contraseña para acceder a esta aplicación.
Estado de pago	Booleano	Booleano de correo electrónico paypal verificado.

Tabla 15. Entidad Vendedor

Nombre	Tipo	Descripción
Nombre	Palabra corta	Nombre de la empresa.
Correo electrónico	Correo, identificador	Correo electrónico paypal de la empresa.
Contraseña	Contraseña	Contraseña para acceder a esta aplicación.
Estado de pago	Booleano	Booleano de correo electrónico paypal verificado.
RFC	Clave	Clave de persona física o moral en México para realizar cualquier actividad económica lícita por la que esté obligada a pagar impuestos.
Dirección	Multivalor	Dirección física donde se encuentra el establecimiento o servicio.
Teléfono	Teléfono	Número telefónico de contacto.

Tabla 16. Entidad Dirección

Nombre	Tipo	Descripción
Estado	Entidad	Estado donde se ofrece el servicio
Municipio	Palabra	Municipio dentro del estado donde se ofrece el servicio.
Colonia	Colonia	Colonia dentro del municipio donde se ofrece el servicio.
Calle	Palabra corta	Calle dentro de la colonia donde se ofrece el servicio.
CP	Código postal	Código postal al cual pertenece el lugar donde se ofrece el servicio.

Tabla 17. Entidad Servicio

Nombre	Tipo	Descripción
Identificador	Número	Numero de servicio utilizado para identificarlo dentro del sistema.
Nombre	Frase corta	Nombre del servicio ofertado.
Descripción	Descripción	Descripción del servicio ofertado.
Horario	Multivalor	El servicio se ofrece en un horario.
Dirección	Multivalor	El servicio reside en una dirección.

Tabla 18. Entidad Venta

Nombre	Tipo	Descripción
Vendedor	Identificador	Identificador del vendedor.
Turista	Identificador	Identificador del turista.
Detalles	Texto	Descripción detallada de la compra realizada.
Monto	Número	Monto total de la venta.
Hora	Hora	Hora en que se realiza la venta.
Fecha	Fecha	Fecha en que se realiza la venta.

Tabla 19. Entidad Horario

Nombre	Tipo	Descripción
Semana hora inicio	Hora	Hora en que comienza el servicio entre semana (Lunes, Martes, Miércoles, Jueves y Viernes)
Semana hora fin	Hora	Hora en que termina el servicio entre semana (Lunes, Martes, Miércoles, Jueves y Viernes)
Fin semana hora inicio	Hora	Hora en que comienza el servicio en fin de semana (Sábado y Domingo)
Fin semana hora fin	Hora	Hora en que termina el servicio en fin de semana (Sábado y Domingo)

3.8. Reglas de negocio

Las reglas de negocio sirven para definir o restringir acciones que serán implementadas en funcionalidad. En la presente sección, se muestran las reglas de negocio del sistema.

Tabla 20. Reglas de negocio

Id	Descripción
RN-01	El usuario de tipo turista y vendedor, deberán estar registrados e iniciar sesión en la aplicación para poder usarla.
RN-02	Al registrar un usuario, la contraseña registrada debe de tener mínimo 6 caracteres alfanuméricos, no se aceptan acentos, comas ni diéresis.
RN-03	Las contraseñas de usuario para el acceso se deberán de almacenar de manera cifrada.
RN-04	El usuario de tipo turista y vendedor, deberá tener una cuenta activa en PayPal para poder realizar el pago de servicios.
RN-05	Un vendedor será el único usuario que podrá realizar venta y cobro de servicios, es decir no hay otro tipo de usuario en jerarquía con el vendedor.
RN-06	El turista puede pagar cualquier servicio ofertado en la aplicación (a excepción de restaurantes, solo se hacen reservaciones), haciendo uso de su dispositivo móvil desde cualquier parte del mundo.
RN-07	El turista puede pagar cualquier servicio ofertado en la aplicación cuando un vendedor le solicite el pago correspondiente, desde el lugar donde se oferta el servicio.
RN-08	El vendedor puede solicitar pagos presenciales a los turistas por el pago de servicios correspondientes que no han sido pagados, desde el lugar donde se oferta el servicio.
RN-09	Un turista puede reservar y pagar más de un servicio a la vez, previo al estar en la ubicación del servicio.
RN-10	Un turista puede pagar solo un servicio de manera simultanea desde el lugar donde se oferta el servicio al momento en que un vendedor le solicita el pago correspondiente.
RN-11	Un vendedor de servicios puede proveer más de 1 servicio, solo si los servicios se encuentran en el mismo domicilio, de lo contrario únicamente podrá proveer 1 servicio.
RN-12	Un vendedor de servicios deberá tener un registro federal de contribuyentes, es decir, clave de persona física o moral lícita.
RN-13	Todos los servicios de restaurante únicamente pueden cobrar de manera remota el costo de reservación, es decir los costos de consumo solo se pueden percibir en el establecimiento físico mediante la aplicación, mismo que el vendedor solicitará al turista.
RN-14	Un vendedor puede solicitar al turista la comprobación de su compra para poder tener acceso al servicio.
RN-15	Deberá existir 1 usuario de tipo administrador para todo el sistema, el cual deberá iniciar sesión en una página web de uso exclusivo para realizar sus tareas.
RN-16	El usuario vendedor no podrá vender sus servicios hasta que el administrador apruebe su clave RFC.
RN-17	Un usuario turista podrá visualizar todos los servicios que estén disponibles para su compra.
RN-18	Un usuario turista podrá elegir el horario en que disfrutará del servicio, de acuerdo con la oferta y disponibilidad del mismo servicio indicado por el usuario vendedor.

RN-19	Un usuario vendedor deberá definir el aforo de su servicio que ofertará para turistas direccionados a partir de compras mediante la aplicación, con el objetivo de brindar la posibilidad de compra desde la aplicación y de manera presencial, es decir no limitar al vendedor.
RN-20	Las reservaciones en hoteles se podrán hacer por rangos mayores o iguales a 1 día, siempre y cuando sean días consecutivos.
RN-21	Las reservaciones en hoteles se podrán hacer hasta con 2 semanas (14 días naturales) de anticipación previos al check-in y check-out.
RN-22	El horario de check-in en los hoteles es a partir de las 14:00 horas. La hora de check-out es a las 12:00 horas.
RN-23	Un usuario de tipo vendedor únicamente deberá de acceder a la aplicación móvil con sus credenciales, el compartir sus credenciales con un tercero y de manera simultanea realizar operaciones dentro de la aplicación móvil, no se considera dentro del alcance del proyecto.
RN-24	El usuario turista podrá visualizar la información de la aplicación móvil en ingles o español, según sea su preferencia.
RN-25	<p>Se deberá informar previo al registro de usuario el aviso de privacidad que conlleva el registro ante la aplicación móvil, el cual deberá contener la siguiente información mínima establecida por el capitulo 2, articulo 16 de la ley federal de protección de datos personales en posesión de particulares.:</p> <ul style="list-style-type: none"> ⇒ La identidad y domicilio del responsable que los recaba ⇒ Las finalidades del tratamiento de datos ⇒ Las opciones y medios que el responsable ofrezca a los titulares para limitar el uso o divulgación de los datos ⇒ Los medios para ejercer los derechos de acceso, rectificación, cancelación u oposición, de conformidad con lo dispuesto en esta Ley ⇒ En su caso, las transferencias de datos que se efectúen ⇒ El procedimiento y medio por el cual el responsable comunicará a los titulares de cambios al aviso de privacidad
RN-26	Solo se registrará usuarios siempre y cuando den el consentimiento del tratamiento sus datos.
RN-27	El estatus de validación de RFC de 1 usuario vendedor, deberá ser dictaminado a lo más en 3 días hábiles posterior al registro del prestador de servicios en la aplicación móvil.
RN-28	Los comentarios que podrá visualizar el usuario turista por cada servicio serán 5, los cuales se mostraran de manera aleatoria y cambiarian cada vez que se recargue la aplicación móvil.
RN-29	Cada turista podrá hacer a lo más 1 evaluación completa (puntuación y comentario) por servicio sin importar el numero de veces que adquiera el mismo.
RN-30	Solo los turistas que compren algún servicio podrán evaluar dicho servicio.
RN-31	Cada comentario podrá ser editado por el usuario propietario que realizó dicho comentario, incluyendo calificación numérica y reseña.

3.9. Requerimientos

3.9.1. Requerimientos no funcionales

Los requisitos no funcionales o atributos de calidad especificarán los criterios que pueden usarse para juzgar la operación del sistema en lugar de sus comportamientos específicos.

Tabla 21. Requerimientos no funcionales

Id	Nombre	Descripción
RNF-01	Usabilidad	La aplicación móvil cuenta con un diseño ergonómico y de fácil comprensión de tal forma que un usuario pueda utilizarla sin complicaciones, y a su vez con un tiempo de aprendizaje de a lo más dos horas. La navegación entre páginas deberá de ser realizada mediante palabras o iconografía.
RNF-02	Uniformidad	La aplicación móvil mantiene el mismo diseño, estilo, paleta de colores, tamaño y tipo de fuente en todas sus pantallas.
RNF-03	Eficiencia	El servidor debe ser capaz de operar adecuadamente y responder a las peticiones de la aplicación móvil en a lo más 4 segundos, cuando las condiciones de red y equipo sean las necesarias.
RNF-04	Escalabilidad	La arquitectura modular bajo la que es construido el sistema permite que este sea escalable, de bajo acoplamiento y pueda expandirse hacia más funcionalidades.
RNF-05	Rapidez	Cuando se envíe una petición al servidor por parte de la aplicación móvil para realizar un pago, este debe responder y mostrar la información en un tiempo máximo de 40 segundos.
RNF-06	Concurrencia	El servidor y aplicación deben ser capaces de atender a todos los nodos que estos soporten sin fallos de causas internas del sistema.
RNF-07	Seguridad	Las contraseñas de acceso a la aplicación deberán de ser almacenadas en la base de datos del sistema de manera cifrada por un algoritmo de tipo hash en 256 bits.
RNF-08	Sencillez	Para todas las funciones de la aplicación móvil, existirá solo una forma de acceder a cada función, es decir sin redundancia de navegación.
RNF-09	Transparencia	La aplicación móvil debe proporcionar mensajes de estatus después de cada ejecución de los procesos generales, que sean informativos y orientados a usuario final.
RNF-10	Arquitectura	El sistema se desarrolla bajo la arquitectura cliente servidor, donde el cliente es la aplicación móvil.
RNF-11	Idioma	La aplicación debe manejar fuentes del alfabeto en inglés y español.
RNF-12	Portabilidad en desarrollo	El desarrollo de la aplicación debe ser portable en plataformas GNU/Linux, Mac Os y Windows, con máquinas que presentan arquitecturas de 64 bits.
RNF-13	Portabilidad en aplicación móvil	La aplicación móvil debe ser portable en plataformas IOS y Android, únicamente en teléfonos, es decir, no se incluyen tabletas electrónicas.

RNF-14	Portabilidad en servidor.	El servidor remoto debe al menos ser accesible desde los navegadores Chrome, safari y Firefox.
--------	---------------------------	--

3.9.2. Requerimientos funcionales

Con los requisitos funcionales definiremos las capacidades que deberá tener el sistema, especificando los comportamientos específicos de las actividades y servicios que el sistema proveerá, en pocas palabras, las funcionalidades que el sistema debe realizar.

Tabla 22. Requerimientos funcionales

Id	Nombre	Descripción	Referencia
RF-01	Inicio de sesión	El sistema permite al usuario ingresar a la aplicación móvil, debe capturar su correo electrónico y contraseña, una vez autenticado se desplegarán las funcionalidades dependiendo del rol; turista o vendedor.	RN-01,
RF-02	Registro de usuario	El sistema permite registrar usuarios de tipo turista y vendedor, realiza la distinción para solicitar datos según el tipo de usuario.	RNF-07, RN-02, RN-03
RF-03	Registro de servicio de restaurante	El sistema le permite al usuario de tipo vendedor registrar un restaurante, la aplicación deberá de solicitar el nombre, descripción, ubicación, horarios, costo de reservación, disponibilidad actual y al menos 3 fotografías del establecimiento y/o servicio que ofrece.	
RF-04	Registro de otros servicios	El sistema le permite al usuario de tipo vendedor registrar servicios de hotelería, zoológicos, spa, temazcal, ciclismo, motociclismo, escalamiento, museo, antro, bar, globo aerostático, gotcha, tirolesa, etc., a excepción de restaurantes. [Ver RF-03] La aplicación deberá de solicitar el nombre, descripción, ubicación, horarios, costo de servicio individual, disponibilidad actual y al menos 3 fotografías del establecimiento y/o servicio que ofrece.	
RF-05	Visualización de servicios	El sistema le permite al turista visualizar todos los servicios disponibles en la aplicación móvil.	
RF-06	Actualización de servicios	El sistema le permite al vendedor actualizar sus servicios previamente registrados en la aplicación móvil, incluida la oferta.	
RF-07	Actualización de disponibilidad de servicios en tiempo real	El sistema actualiza automáticamente la disponibilidad de servicios según la oferta proporcionada por el vendedor.	
RF-08	Cambio de idioma	El sistema le permite al usuario turista cambiar el idioma de la aplicación móvil: inglés y español.	

RF-09	Registro de servicio	El sistema le permite al vendedor registrar servicios turísticos bajo la misma dirección que proporciona al registrarse como usuario vendedor.	RN-11, RN-12
RF-10	Reservación de restaurante	El sistema le permite al turista reservar uno o más lugares en un restaurante, dependiendo de la oferta y disponibilidad de este.	RN-13
RF-11	Compra individual adelantado de servicio	El sistema le permite al turista comprar 1 servicio desde la aplicación móvil, previo al consumo del servicio, a excepción para servicio de restaurante. [Ver RF-10]	
RF-12	Calificación de servicios	El sistema le permite al usuario turista calificar servicios que ha comprado, mediante la estrategia de valoración por estrellas. [30]	
RF-13	Visualización de calificación de servicios.	El sistema calcula automáticamente el promedio de calificaciones por servicio y lo muestra como parte de la información de cada servicio.	
RF-14	Comprobación de compra previa para el turista.	El sistema genera un código QR por compra, el turista podrá visualizar el código QR desde la aplicación móvil para mostrarlo al vendedor cuando este lo solicita para comprobar y otorgar el acceso al servicio en cuestión.	RN-06
RF-15	Comprobación de compra previa para el vendedor.	El sistema le permite al vendedor mediante la aplicación móvil, el escaneo del código QR de comprobación de compra de servicio, que el turista le proporciona al vendedor.	RN-10, RN-14
RF-16	Cobro presencial de servicio	El sistema le permite al vendedor mediante la aplicación móvil la generación personalizada de un código QR correspondiente al cobro de algún servicio, mismo que podrá visualizar en la pantalla de su dispositivo para que el turista lo escanee y pague.	RN-08
RF-17	Pago presencial de servicio	El sistema le permite al turista mediante la aplicación móvil la lectura del código QR que el vendedor le proporciona para realizar el pago correspondiente de algún servicio.	RN-07
RF-18	Registro de administrador	El sistema tendrá 1 usuario administrador para el mismo, se da de alta desde la base de datos en firebase.	
RF-19	Despliegue de búsquedas de servicios, ventas y compras.	<p>Considerar los siguientes casos:</p> <ul style="list-style-type: none"> ⇒ La aplicación de usuario vendedor lista sus servicios previamente registrados o lista las ventas realizadas. ⇒ La aplicación de usuario turista lista los servicios con o sin filtro de búsqueda o lista las compras realizadas. <p>Se despliegan de 7 en 7, cada que se haga scroll en el final de la pantalla, la aplicación buscara y</p>	RF-05

		desplegará los siguientes 7 ítems para mostrar, esto se repite hasta que no haya más ítems.	
--	--	---	--

3.9.3. Requerimientos técnicos

Para utilizar la aplicación móvil es necesario que los usuarios cuenten con ciertos requisitos en sus teléfonos móviles, dichos requisitos se enunciarán a continuación.

3.9.3.1. Requerimientos mínimos de software

- ⇒ IOS
IOS versión 14 o superior.
- ⇒ Android
Android Oreo (versión 8.0) o superior.

3.9.3.2. Requerimientos mínimos de hardware

Dispositivos con cámara de resolución de al menos 5 Mega Pixeles y con autoenfoque, para dispositivos Apple, deberán ser compatibles con la versión de sistema operativo IOS 14 (Ver modelos de iPhone compatibles con IOS 14).

4. Diseño del sistema

4.1. Generalidades

4.1.1. Arquitectura del sistema

El sistema completo se basa en la arquitectura cliente servidor, podemos identificar 2 clientes, la aplicación móvil para turistas, vendedores y la aplicación web básica de administrador. En la aplicación móvil, se realiza la lógica de negocio principal, para el caso del servidor se extraerán datos de localización, servicio de traducción, almacenamiento de datos e imágenes según soliciten los clientes.

React Native funcionará como lenguaje de frontend y backend para el desarrollo de los dos tipos de clientes que consumirán servicios de firebase y APIs de google como servidor.

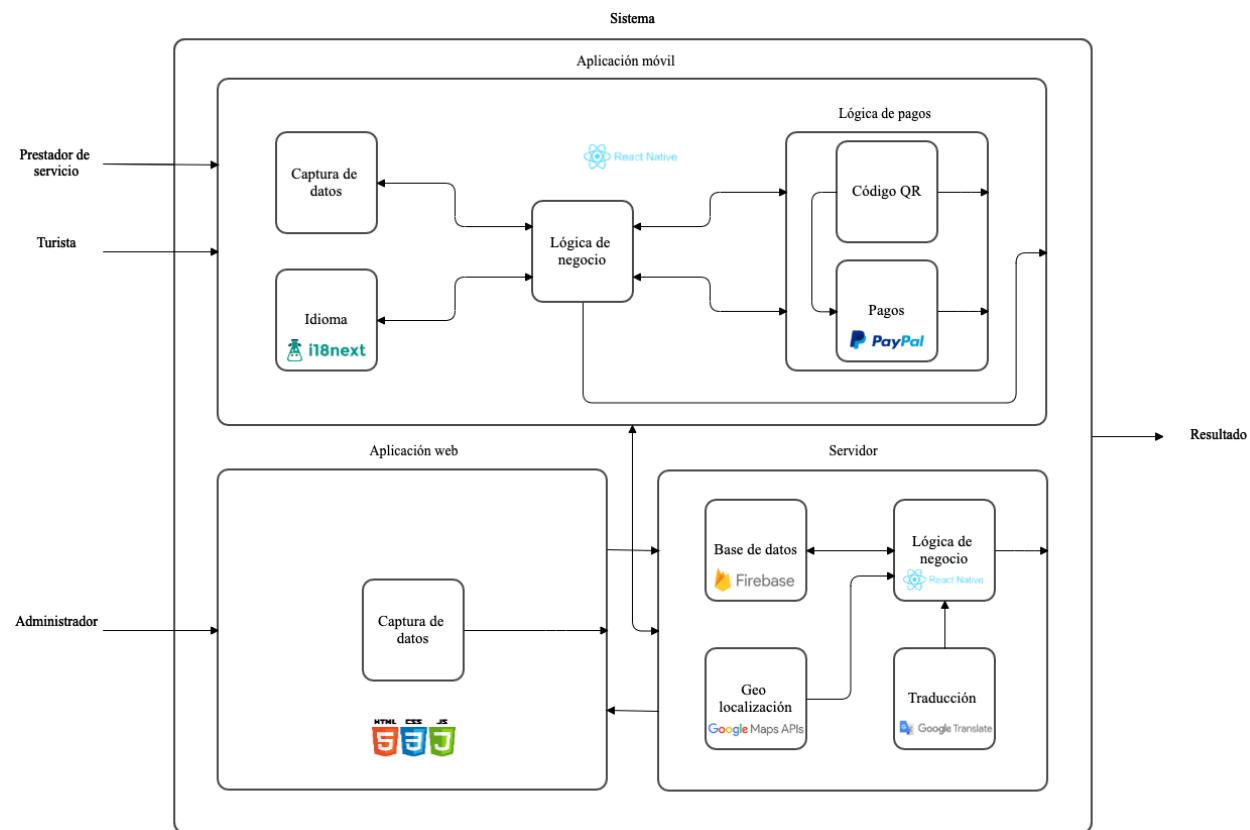


Figura 9. Diagrama a bloques del sistema

4.1.2. Modelado de la información

Base de datos NoSQL, la cual se trabaja desde firebase.

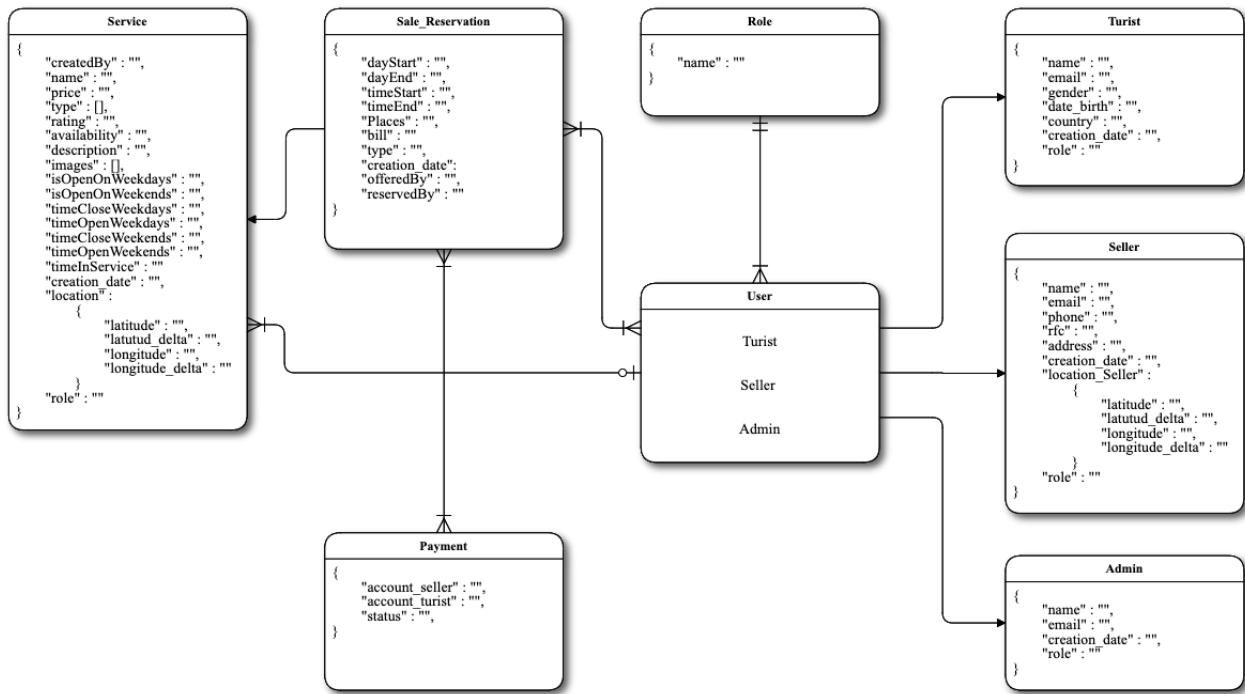


Figura 10. Diagrama de base de datos NoSQL

4.1.3. Diagrama de proceso general

En el siguiente diagrama BPMN, se representa el conjunto de actividades que permite describir las funcionalidades principales del sistema, considerando que un vendedor ha sido validado por el administrador, posteriormente el vendedor registra al menos un servicio, a partir de ahí se realiza el proceso de compra de servicios.

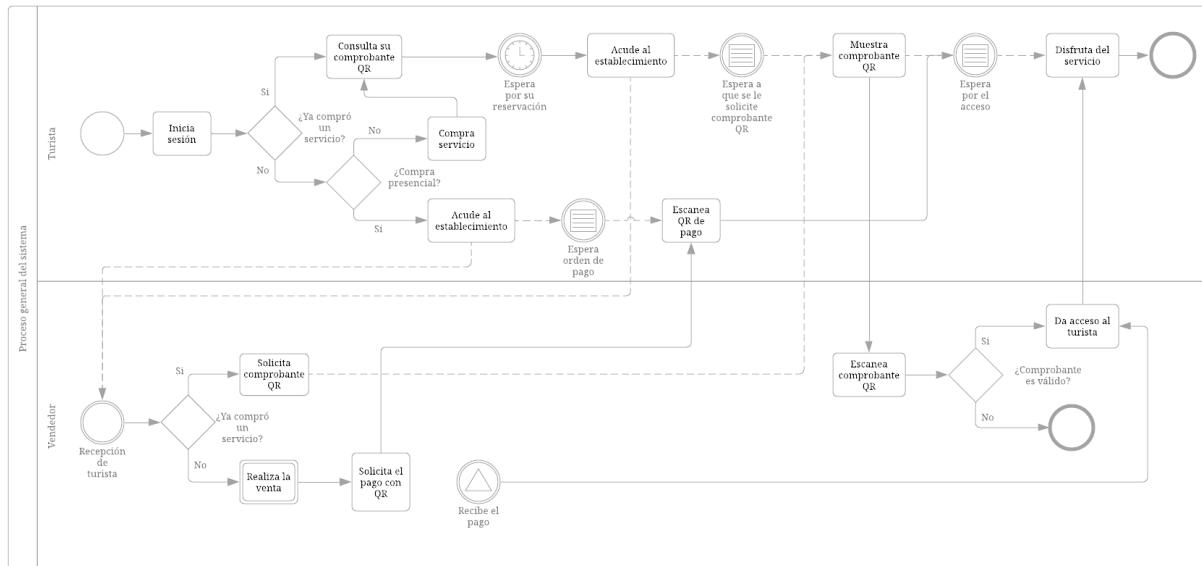


Figura 11. Diagrama BPMN de proceso general

4.1.4. Casos de uso general

Se muestra el diagrama de casos de uso general, es decir, representa los casos de uso o funcionalidades de todo el sistema.

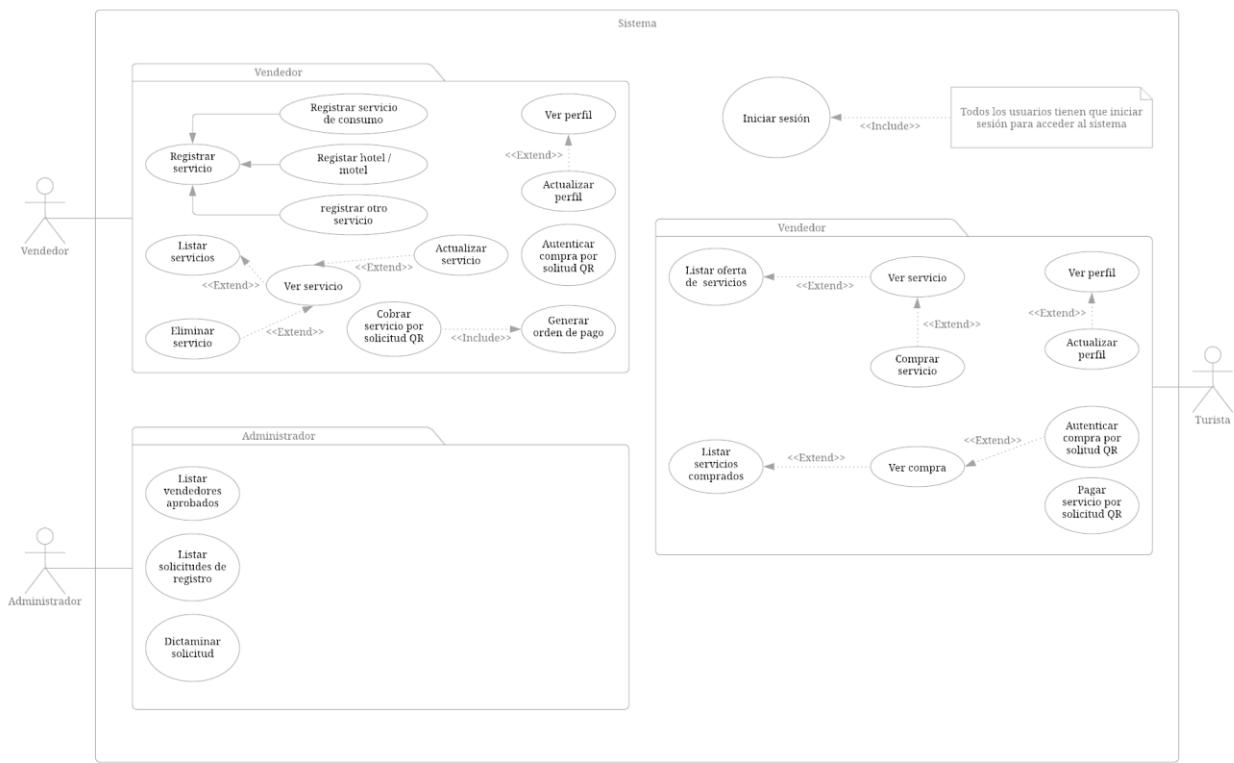


Figura 12. Diagrama general de casos de uso

4.2. Submódulo de aplicación móvil

En esta sección se diseña el modelo de navegación de la aplicación móvil, incluye las pantallas o interfaces de contempla el desarrollo.

4.2.1. Modelo de navegación

El mapa de navegación es, básicamente, un gráfico o esquema en forma de árbol que representa la estructura o arquitectura general de un sistema. Es el primer paso para distribuir, organizar y jerarquizar el contenido que se verá en la pantalla de la aplicación móvil.

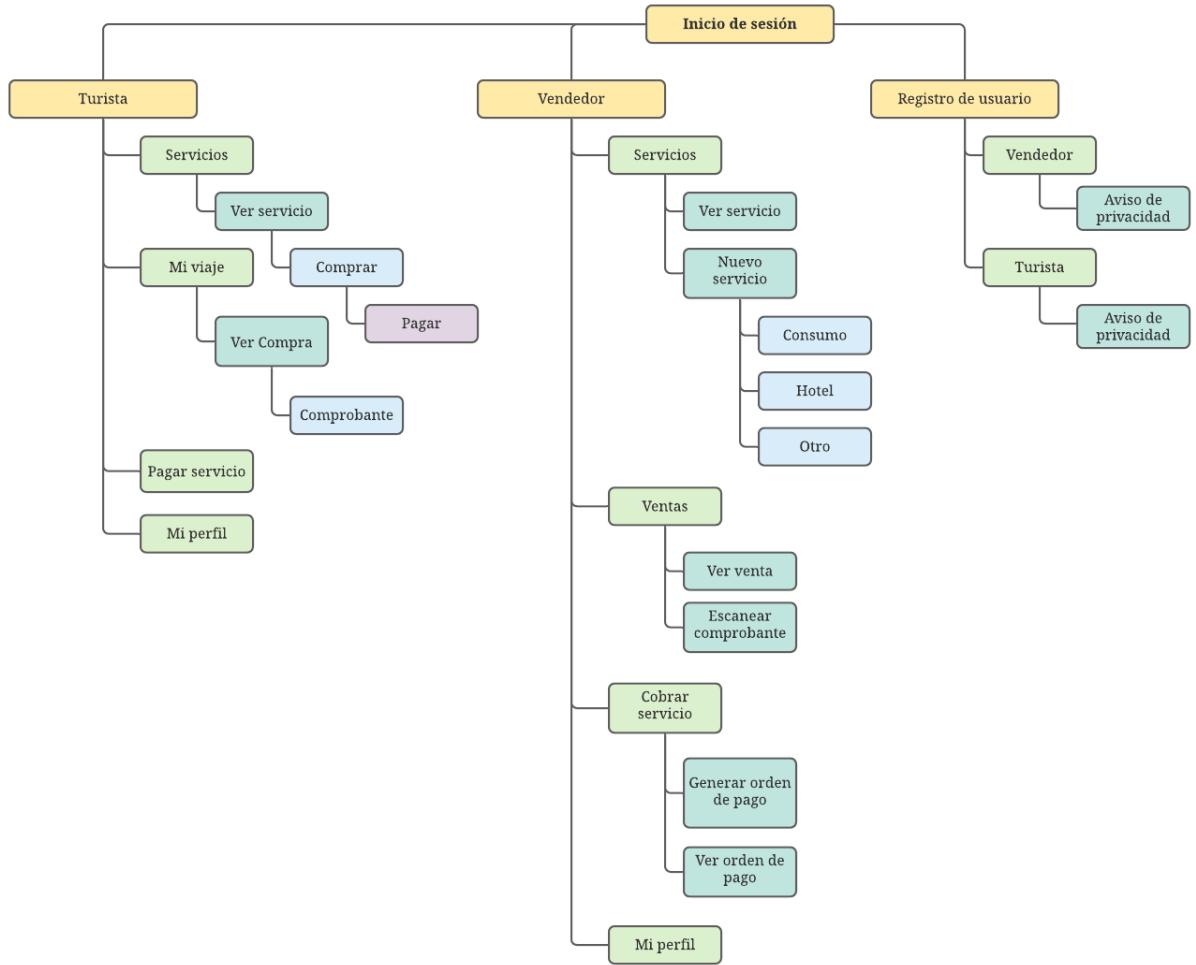


Figura 13. Modelo de navegación de aplicación móvil

La barra de navegación para el usuario turista esta constituida por: “Servicios”, “Mi viaje”, “Pagar servicio” y “Mi perfil”, para el caso del usuario vendedor esta constituida por: “Servicios”, “Ventas”, “Cobrar servicio” y “Mi perfil”.

4.2.2. Inicio de sesión y registro de usuarios

4.2.2.1. Inicio sesión



Figura 14. Mockup
Inicio de sesión

Entradas

- Correo electrónico
- Contraseña

Salidas

- Ninguna

Acciones

- **Iniciar:** Verifica los datos ingresados por el usuario y de ser correctos se muestra la pantalla de búsqueda de servicios (Ver **Error! Reference source not found.**) o para el vendedor la pantalla de búsqueda de servicios (Ver Figura 29. Mockup Lista de servicios).
- **Registrarse:** Se muestra la pantalla de selección de registro (Ver Figura 15. Mockup Selección de Registro de usuario).

Mensajes

- Datos de usuario no son correctos.
- Todos los datos solicitados son obligatorios.

Observaciones

Primera pantalla que se muestra al abrir la aplicación móvil.

4.2.2.2. Selección de tipo de registro de usuario

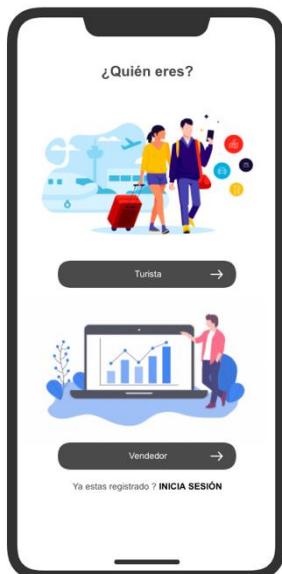


Figura 15. Mockup
Selección de Registro de
usuario

Entradas

Selección de tipo de usuario a registrar.

Salidas

- Ninguna

Acciones

- **Turista:** Se muestra la pantalla de registro de usuarios turistas (Ver Figura 16. Mockup Registro de usuario Turista).
- **Vendedor:** Se muestra la pantalla de registro de usuarios vendedores (Ver Figura 18. Mockup Registro de usuario Vendedor).
- **Inicia sesión:** Se muestra la pantalla de inicio de sesión (Ver Figura 14. Mockup Inicio de sesión)

Mensajes

- Ninguno

Observaciones

Ninguna

4.2.2.3. Registro de usuario turista



Figura 16. Mockup Registro de usuario Turista

Entradas

Nombre completo, correo electrónico, fecha de nacimiento, género, nacionalidad, contraseña.

Salidas

- Ninguna

Acciones

- **Registrar:** Valida que todos los campos se hayan llenado, correo electrónico con estructura correcta, haber seleccionado el género y nacionalidad, que las contraseñas coincidan y verificar que se aceptaron los términos y condiciones, de cumplir, se registrar el usuario en el sistema, se registrar el usuario en el sistema y de todo salir correctamente se muestra la aplicación móvil de vendedor (Ver **Error! Reference source not found.**).
- **Inicia sesión:** Se muestra la pantalla de inicio de sesión (Ver Figura 14. Mockup Inicio de sesión)
- **Términos y condiciones:** Se muestra la pantalla de términos y condiciones (Ver Figura 17. MockUp Términos y condiciones para registro de usuarios).

Mensajes

- Mientras se registrar el usuario, se muestra el mensaje de “Registrando usuario”

Observaciones

Ninguna

4.2.2.4. Términos y condiciones para registro de usuario



Figura 17. MockUp Términos y condiciones para registro de usuarios

Entradas

- Ninguna

Salidas

- Ninguna **Error! Reference source not found.Error! Reference source not found.**

Acciones

- Regresar: Regresar a la pantalla de registro de usuario turista (Ver Figura 16. Mockup Registro de usuario Turista).

Mensajes

- Ninguno

Observaciones

Se muestran los términos y condiciones para el registro de usuarios (Ver Aviso de privacidad en aplicación móvil).

4.2.2.5. Registro de usuario vendedor



Figura 18. Mockup Registro de usuario Vendedor

Entradas

- Nombre de la empresa, correo electrónico, RFC, dirección, teléfono, contraseña.

Salidas

- Ninguna

Acciones

- **Registrar:** Valida que todos los campos se hayan llenado, correo electrónico con estructura correcta, haber seleccionado el género y nacionalidad, que las contraseñas coincidan y verificar que se aceptaron los términos y condiciones, de cumplir, se registrar el usuario en el sistema y de todo salir correctamente se muestra la aplicación móvil de vendedor (Ver Figura 29. Mockup Lista de servicios).
- **Inicia sesión:** Se muestra la pantalla de inicio de sesión (Ver Figura 14. Mockup Inicio de sesión)
- **Términos y condiciones:** Se muestra la pantalla de términos y condiciones (Ver Figura 17. MockUp Términos y condiciones para registro de usuarios).

Mensajes

- Mientras se registrar el usuario, se muestra el mensaje de “Registrando usuario”

Observaciones

La selección de la dirección se realiza mediante un mapa.

4.2.3. Navegación de usuario turista

4.2.3.1. Buscar servicio

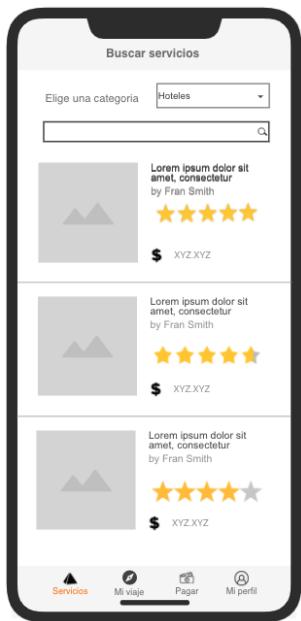


Figura 19. Mockup Buscar servicio

Entradas

- Nombre de servicio, selección de tipo de servicio.

Salidas

- Servicios coincidentes con la búsqueda.

Acciones

- **Buscar:** Busca los servicios que coinciden con la búsqueda, mostrara los primeros 10 servicios encontrados.
- **Seleccionar:** Presionar sobre cualquier servicio para consultar a detalle el servicio, redirige a la pantalla de un servicio en especial (Ver Figura 20. Mockup Ver servicio).

Mensajes

- Ninguno

Observaciones

Se muestran los primeros 10 servicios coincidentes para la búsqueda, al deslizar la pantalla hacia abajo se muestran de 10 en 10 servicios, con el objetivo de no saturar el procesamiento y rendimiento de la aplicación.

4.2.3.2. Ver servicio

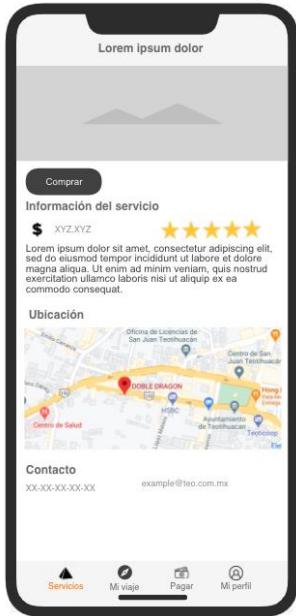


Figura 20. Mockup Ver servicio

Entradas

- Ninguna

Salidas

- Ninguna

Acciones

- **Comprar:** Muestra la pantalla de formulario de especificaciones de compra acorde al servicio en cuestión
 - a. Restaurante o bar (Ver Figura 21. Mockup Comprar servicio de hotel)
 - b. Hotel (Ver Figura 22. Mockup Comprar servicio de consumo)
 - c. Otro servicio (Ver Figura 23. Mockup Comprar otro servicio)
- **Regresar:** Regresar a la pantalla de búsqueda de servicios (Ver **Error! Reference source not found.**).

Mensajes

- Ninguno

Observaciones

Esta pantalla muestra los detalles del servicio seleccionado, muestra el precio unitario de costo de servicio o reservación, nombre, descripción, fotografías, mapa con ubicación geográfica, número telefónico, correo electrónico, calificación con base a estrellas.

4.2.3.3. Comprar servicio de hotel



Figura 21. Mockup Comprar servicio de hotel

Entradas

- Selección de día de check-in, check-out y número de huéspedes.

Salidas

- Ninguna

Acciones

- **Pagar:** Verifica que se hayan seleccionado todos los campos del formulario y su disponibilidad, se ser valido, muestra una alerta de confirmación para continuar.
- **Regresar:** Regresar a la pantalla de ver servicio (Ver Figura 20. Mockup Ver servicio).

Mensajes

- Mensaje de confirmación.

Observaciones

Deberá primero llenarse el día de check-in y check-out, posterior a ello el número de huéspedes, la disponibilidad se mostrará de acuerdo a la oferta que se tenga.

4.2.3.4. Comprar servicio de consumo



Figura 22. Mockup
Comprar servicio de consumo

Entradas

- Día, hora y cantidad de lugares unitarios para reservación.

Salidas

- Ninguna

Acciones

- **Pagar:** Verifica que se hayan seleccionado todos los campos del formulario y su disponibilidad, se ser valido, muestra una alerta de confirmación para continuar.
- **Regresar:** Regresar a la pantalla de ver servicio (Ver Figura 20. Mockup Ver servicio).

Mensajes

- Mensaje de confirmación.

Observaciones

Deberá primero llenarse el día y hora que se desea reservar, posterior a ello el número de personas que disfrutarán del servicio, la disponibilidad se mostrará de acuerdo con la oferta que se tenga en el campo cantidad.

4.2.3.5. Comprar otro servicio

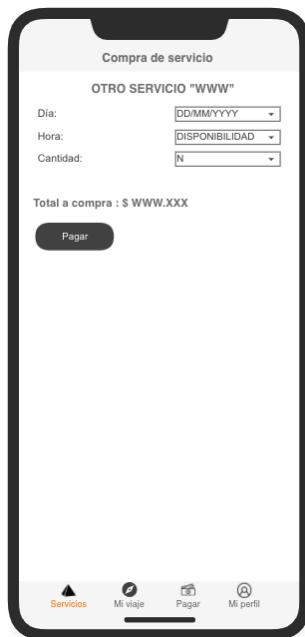


Figura 23. Mockup
Comprar otro servicio

Entradas

- Día, hora y cantidad de lugares unitarios para reservación.

Salidas

- Ninguna

Acciones

- **Pagar:** Verifica que se hayan seleccionado todos los campos del formulario y su disponibilidad, se ser valido, muestra una alerta de confirmación para continuar.
- **Regresar:** Regresar a la pantalla de ver servicio (Ver Figura 20. Mockup Ver servicio).

Mensajes

- Mensaje de confirmación.

Observaciones

Deberá primero llenarse el día y hora que se desea disfrutar del servicio, posterior a ello el número de personas que disfrutarán del servicio, la disponibilidad se mostrará de acuerdo con la oferta que se tenga en el campo cantidad.

4.2.3.6. Mi viaje



Figura 24. Mockup Lista de mis viajes

Entradas

- Ninguna

Salidas

- Código QR comprobante

Acciones

- **Seleccionar:** Presionar sobre cualquier servicio para consultar a detalle el servicio, redirige a la pantalla de un servicio en especial (Ver Figura 20. Mockup Ver servicio).
- **Comprobante:** Se muestra una pantalla, la cual muestra el código QR que es un comprobante de pago para el vendedor cuando escanee el mismo. (Ver Figura 25. Mockup Comprobante de pago para servicio).

Mensajes

- Ninguno

Observaciones

Se muestran los servicios comprados previamente, se verá el estatus de la compra, posibilidad de acceder al comprobante del servicio y visualizar información de la compra por cada servicio.

4.2.3.7. Comprobante de pago de servicio



Figura 25. Mockup Comprobante de pago para servicio

Entradas

- Ninguna

Salidas

- Ninguna

Acciones

- **Regresar:** Regresar a la pantalla de lista de servicios comprados (Ver Figura 24. Mockup Lista de mis viajes).

Mensajes

- Ninguno

Observaciones

Código QR que representa los detalles de compra realizada, esta pantalla debe ser escaneada por el vendedor cuando lo solicite, las leyendas de instrucciones se adecuan al idioma seleccionado.

4.2.3.8. Pagar servicio presencial



Figura 26. MockUp Pago de servicio presencial

Entradas

- Ninguna

Salidas

- Orden de pago a confirmar

Acciones

- **Escanear QR:** Abre el escáner de código QR por medio de la cámara del dispositivo, lee y detecta el código, posterior a ello redirecciona a la pantalla de confirmación de pago (Ver Figura 27. MockUp Pagar servicio - detalle).

Mensajes

- Mensaje mientras se procesa el QR con la leyenda "Procesando QR".

Observaciones

Si es la primera vez que se utiliza la aplicación, entonces se solicita permiso para utilizar la cámara del dispositivo, mismo que se deberá de autorizar para continuar con el proceso.

4.2.3.9. Pagar servicio – detalle

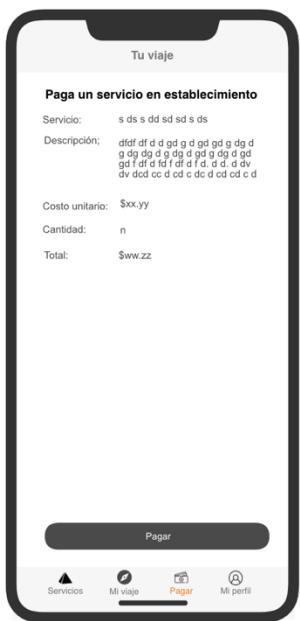


Figura 27. MockUp Pagar servicio - detalle

Entradas

- Ninguna

Salidas

- Comprobante en código QR

Acciones

- **Pagar:** Redirección a pasarela de pago para realizar el pago del servicio, la cual incluye los datos necesarios para la aplicación de cobro.

Mensajes

- Ninguno

Observaciones

Ninguna

4.2.3.10. Mi perfil

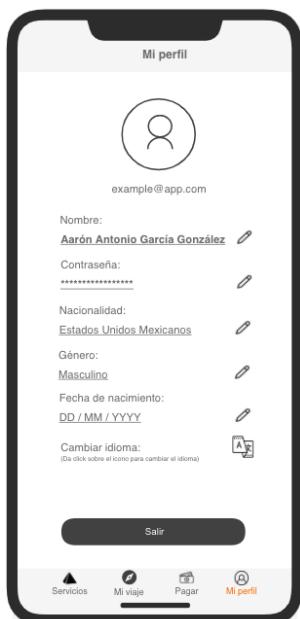


Figura 28. MockUp Mi perfil de usuario turista

Entradas

- Nombre completo, contraseña, género, nacionalidad, fecha de nacimiento.

Salidas

- Ninguna

Acciones

- **Actualizar nombre:** Se muestra una ventana modal, donde se permite actualizar el nombre completo del usuario.
- **Actualizar contraseña:** Se muestra una ventana modal, donde se permite actualizar la contraseña de acceso del usuario.
- **Actualizar Nacionalidad:** Se muestra una ventana modal, donde se permite actualizar la nacionalidad del usuario por medio de una lista de selección.
- **Actualizar Género:** Se muestra una ventana modal, donde se permite actualizar el género del usuario por medio de una lista de selección.
- **Actualizar Fecha de nacimiento:** Se muestra una ventana modal, donde se permite actualizar la fecha de nacimiento del usuario por medio de una lista de selección de fecha.
- **Cambiar idioma:** Cambia el idioma de la aplicación, de inglés a español y de español a inglés.
- **Añadir/Actualizar fotografía:** Se muestra una ventana de selección de fotografía de la galería multimedia del dispositivo móvil, la foto selecciona, actualiza la existente en caso de haber.

Mensajes

- Mensaje mientras se actualizan los datos con la leyenda "Guardando cambios".

Observaciones

Si es la primera vez que se utiliza la aplicación, entonces se solicita permiso para acceder a la galería multimedia del dispositivo, mismo que se deberá de autorizar para continuar con el proceso.

4.2.4. Navegación de usuario vendedor

4.2.4.1. Lista de servicios



Figura 29. Mockup Lista de servicios

Entradas

- Ninguna

Salidas

- Ninguna

Acciones

- **Nuevo servicio:** Se muestra la pantalla de selección de tipo de nuevo servicio (Ver Figura 30. MockUp Selección de tipo de nuevo servicio) si y solo si la verificación del RFC es activa, de lo contrario, si se encuentra pendiente muestra el Mensaje A y si es Rechazado muestra el Mensaje B.
- **Seleccionar:** Presionar sobre cualquier servicio (Servicios previamente registrados por el usuario vendedor actual), se muestra la pantalla con la información del servicio (Ver Figura 36. Mockup Ver venta).

Mensajes

- **Mensaje A:** Tu RFC se encuentra en proceso de validación, este proceso toma entre 1 y 3 días hábiles. Intente de nuevo más tarde.
- **Mensaje B:** Se determinó que el RFC XXYYZZWWAABB no es válido para el sistema, utiliza otra RFC valida o bien comunícate al correo@mail.com para aclaraciones.

Observaciones

Primera pantalla que se muestra para el usuario vendedor autenticado en el sistema.

Todos los servicios mostrados en dicha lista de servicios se encuentran en la misma ubicación geográfica.

4.2.4.2. Selección de tipo de nuevo servicio



Figura 30. MockUp
Selección de tipo de nuevo servicio

Entradas
Selección de tipo de servicio a crear.
Salidas
- Ninguna
Acciones
<ul style="list-style-type: none">- Hospedaje: Se muestra la pantalla de formulario de registro de hotel o motel (Ver Error! Reference source not found.).- Consumo: Se muestra la pantalla de formulario de registro de servicio de consumo, como lo son: restaurante, bar, antro, club, es decir solo se permite hacer reservación (Ver Error! Reference source not found.).- Otro: Se muestra la pantalla de formulario de registro de servicios que no son hospedaje ni consumo, por ejemplo: zoológico, spa, etc. (Ver Error! Reference source not found.).- Regresar: Regresar a la pantalla de lista de servicios comprados (Ver Figura 29. Mockup Lista de servicios).
Mensajes
<ul style="list-style-type: none">- Ninguno
Observaciones
Ninguna

4.2.4.3. Añadir servicio de hotel



Figura 31. Mockup Añadir servicio de hotel

Entradas

Nombre de hotel, costo unitario de reservación por noche, disponibilidad simultanea por día, descripción, fotografías.

Salidas

- Ninguna

Acciones

- **Añadir imagen:** Al presionar en el ícono de imagen, se despliega el selector multimedia para elegir 1 imagen, la cual al seleccionarla se muestra a la derecha del ícono de imagen de manera miniatura, es posible realizar el mismo proceso hasta para 5 imágenes en total.
- **Descartar imagen:** Al presionar sobre cualquier imagen cargada, se puede eliminar o descartar la imagen seleccionada, el número de imágenes disponibles a subir se actualiza al valor original.
- **Añadir servicio:** Verifica que todos los datos del formulario estén completos, registra los datos en el servidor, al terminar este proceso muestra la pantalla donde esta la lista de servicios, donde aparece el servicio previamente creado (Ver Figura 29. Mockup Lista de servicios).
- **Regresar:** Regresar a la pantalla de selección de registro de nuevo servicio (Ver Figura 30. MockUp Selección de tipo de nuevo servicio).

Mensajes

- Mensaje de mientras se guarda la información con la leyenda de “Registrando servicio”.
- Mensaje de confirmación para descartar imagen cargada.

Observaciones

Para poder registrar un servicio, el RFC del usuario vendedor debe de haber sido validado con éxito por el Administrador (Para consultar status de validación, Ver Figura 40. Mockup Mi perfil vendedor).

4.2.4.4. Añadir servicio de consumo



Figura 32. Mockup Añadir servicio de consumo

Entradas

Multiselectador de tipo de servicio, costo de reservación unitaria, tiempo promedio de servicio, disponibilidad simultanea total, selección de apertura (fines de semana, entre semana), descripción, selección de hora de apertura y cierre en fines de semana (en caso de haber seleccionado apertura en fines de semana), selección de hora de apertura y cierre entre semana (en caso de haber seleccionado apertura entre semana), fotografías.

Salidas

- Ninguna

Acciones

- **Añadir imagen:** Al presionar en el ícono de imagen, se despliega el selector multimedia para elegir 1 imagen, la cual al seleccionarla se muestra a la derecha del ícono de imagen de manera miniatura, es posible realizar el mismo proceso hasta para 5 imágenes en total.
- **Descartar imagen:** Al presionar sobre cualquier imagen cargada, se puede eliminar o descartar la imagen seleccionada, el número de imágenes disponibles a subir se actualiza al valor original.
- **Añadir servicio:** Verifica que todos los datos del formulario estén completos, registra los datos en el servidor, al terminar este proceso muestra la pantalla donde esta la lista de servicios, donde aparece el servicio previamente creado (Ver Figura 29. Mockup Lista de servicios).
- **Regresar:** Regresar a la pantalla de selección de registro de nuevo servicio (Ver Figura 30. MockUp Selección de tipo de nuevo servicio).

Mensajes

- Mensaje de mientras se guarda la información con la leyenda de “Registrando servicio”.
- Mensaje de confirmación para descartar imagen cargada.

Observaciones

Para poder registrar un servicio, el RFC del usuario vendedor debe haber sido validado con éxito por el Administrador (Para consultar estatus de validación, Ver Figura 40. Mockup Mi perfil vendedor).

4.2.4.5. Añadir otro servicio



Figura 33. Mockup Añadir otro servicio

Entradas

Multi selector de tipo de servicio, costo de consumo unitario de servicio, tiempo promedio de servicio, disponibilidad simultanea total, selección de apertura (fines de semana, entre semana), descripción, selección de hora de apertura y cierre en fines de semana (en caso de haber seleccionado apertura en fines de semana), selección de hora de apertura y cierre entre semana (en caso de haber seleccionado apertura entre semana), fotografías.

Salidas

- Ninguna

Acciones

- **Añadir imagen:** Al presionar en el icono de imagen, se despliega el selector multimedia para elegir 1 imagen, la cual al seleccionarla se muestra a la derecha del icono de imagen de manera miniatura, es posible realizar el mismo proceso hasta para 5 imágenes en total.
- **Descartar imagen:** Al presionar sobre cualquier imagen cargada, se puede eliminar o descartar la imagen seleccionada, el número de imágenes disponibles a subir se actualiza al valor original.
- **Añadir servicio:** Verifica que todos los datos del formulario estén completos, registra los datos en el servidor, al terminar este proceso muestra la pantalla donde esta la lista de servicios, donde aparece el servicio previamente creado (Ver Figura 29. Mockup Lista de servicios).
- **Regresar:** Regresar a la pantalla de selección de registro de nuevo servicio (Ver Figura 30. MockUp Selección de tipo de nuevo servicio).

Mensajes

- Mensaje de mientras se guarda la información con la leyenda de “Registrando servicio”.
- Mensaje de confirmación para descartar imagen cargada.

Observaciones

Para poder registrar un servicio, el RFC del usuario vendedor debe de haber sido validado con éxito por el Administrador (Para consultar status de validación, Ver Figura 40. Mockup Mi perfil vendedor).

4.2.4.6. Ver servicio previamente registrado



Figura 34. Mockup Ver servicio previamente registrado

Entradas

Opcionales y excluyentes entre si:

Para todos los servicios, Nombre del servicio, multi selección de tipo de servicio, descripción del servicio, disponibilidad unitaria, costo unitario.

Para los servicios de consumo y otros servicios, tiempo promedio de cliente en consumo, días de apertura; entre semana (adicional seleccionar hora de apertura y hora de cierre) y fines de semana (adicional seleccionar hora de apertura y hora de cierre).

Salidas

- Ninguna

Acciones

- **Editar información:** Al presionar sobre cualquier elemento de entrada, se despliega una ventana modal para editar la información y actualizar la misma, se muestra el mensaje A mientras se lleva a cabo el proceso.
- **Eliminar servicio:** Verifica si hay alguna compra inconclusa, de no existir, permite eliminar el servicio de manera definitiva, incluyendo los recursos de tipo imagen registrado, se muestra el mensaje B mientras se realiza el proceso. Al eliminar correctamente un servicio se muestra la pantalla de lista de servicios (Ver Figura 29. Mockup Lista de servicios) de manera actualizada, es decir, sin el servicio eliminado y se muestra el mensaje C.
- **Eliminar imagen:** Se presiona sobre una imagen y se muestra el mensaje D, a modo de confirmación de dicho proceso.
- **Agregar imagen:** Al presionar en el ícono de imagen, se despliega el selector multimedia para elegir 1 imagen, es posible realizar el mismo proceso hasta para 5 imágenes en total.

Mensajes

- **Mensaje A:** Mensaje de confirmación con la leyenda “Actualizando información”, la actualizar un elemento.
- **Mensaje B:** Mensaje de confirmación con la leyenda “¿Estás seguro de eliminar servicio?”, al eliminar el servicio.
- **Mensaje C:** Mensaje de estado con la leyenda “Se eliminó correctamente el servicio”, al finalizar un servicio correctamente.
- **Mensaje D:** Mensaje de confirmación con la leyenda “¿Estás seguro de eliminar esta imagen?”, al eliminar una imagen.

Observaciones

Ninguna

4.2.4.7. Lista de ventas



Figura 35. Mockup Lista de ventas

Entradas

Selección de fecha de venta.

Salidas

- Ninguna

Acciones

- **Escanear comprobante:** Muestra la pantalla de escanear comprobante de pago (Ver Figura 37. Mockup Escanear comprobante de pago realizado).
- **Seleccionar venta:** Muestra los detalles de una venta en particular (Ver Figura 33. Mockup Añadir otro servicio).

Mensajes

- Ninguno

Observaciones

Se visualiza en forma de lista cada una de las ventas realizadas acorde con la fecha de venta seleccionada, en la cual cada ítem de venta muestra el nombre de servicio vendido, el rango de fecha reservado, la hora de llegada, el número de servicios y precio total, y finalmente el estatus de la venta (Ya registró el ingreso al establecimiento o no).

4.2.4.8. Ver venta



Figura 36. Mockup Ver venta

Entradas

- Ninguna

Salidas

- Ninguna

Acciones

- **Regresar:** Regresar a la pantalla de lista de ventas (Ver Figura 35. Mockup Lista de ventas).

Mensajes

- Ninguno

Observaciones

Pantalla que muestra los detalles de la compra realizada, la cual es direccionada de acuerdo con la venta seleccionada, viene de la pantalla de lista de ventas (Ver Figura 35. Mockup Lista de ventas).

4.2.4.9. Escanear comprobante de pago realizado



Figura 37. Mockup Escanear comprobante de pago realizado

Entradas

- Ninguna

Salidas

- Ninguna

Acciones

- **Escanear QR:** Al presionar el botón se abre la cámara del dispositivo móvil, donde se escanea el código QR proporcionado por el turista, de ser correcto el QR (existir en base de datos) se muestra a detalle los detalles de compra (Ver Figura 36. Mockup Ver venta) y el vendedor procede a dar acceso al turista.

Mensajes

- Mensaje mientras se procesa el código QR con la leyenda de “Procesando QR, espera por favor”.
- Mensaje de error con la leyenda “El proceso de escaneo de QR tuvo un error, intenta más tarde”, cuando ocurra cualquier error que no permita continuar con el proceso de escaneo – acceso.

Observaciones

Si es la primera vez que se utiliza la aplicación, entonces se solicita permiso para utilizar la cámara del dispositivo, mismo que se deberá de autorizar para continuar con el proceso.

4.2.4.10. Generar orden de pago



Figura 38. Mockup
Generar orden de pago
para servicio de hotel

Entradas

Selección de tipo de servicio a cobrar (Servicios registrados por el vendedor),

- Para hospedaje
Día de check-in, día de check-out, número de huéspedes.
- Para consumo
Día, hora de reservación, número de reservaciones.
- Para otros
Día, hora de goce de servicio, numero total de compras.

Salidas

- Orden de pago (QR)

Acciones

- **Generar orden de pago:** Verifica que todos los campos solicitados en el formulario estén llenos y sean correctos, disponibilidad acorde a la demanda solicitada y de ser correcto todo se genera una orden de pago que despliega un código QR (que deberá escanear el turista, ver Figura 39. Mockup Orden de pago presencial), cuando se realice el pago, se muestra el mensaje A, se agrega la venta, a la lista de ventas (Ver Figura 35. Mockup Lista de ventas).

Mensajes

- Mensaje mientras se genera la orden de pago con la leyenda “Generando orden de pago, espere por favor”.
- **Mensaje A:** “Transacción realizada correctamente, el turista puede disfrutar del servicio ofertado”.

Observaciones

Ninguna

4.2.4.11. Código QR de orden de pago



Figura 39. Mockup Orden de pago presencial

Entradas
- Ninguna
Salidas
- Ninguna
Acciones
- Ninguna
Mensajes
- Ninguno
Observaciones
El usuario vendedor muestra esta pantalla al usuario turista, el usuario turista escanea el código QR, al finalizar el pago, en la pantalla actual de usuario vendedor que generó la orden de pago, se muestra una alerta donde indica que la transacción se completo correctamente.

4.2.4.12. Mi perfil

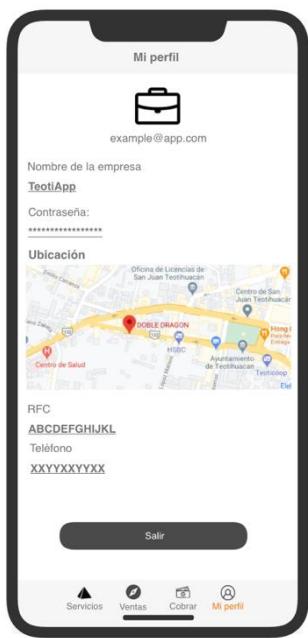


Figura 40. Mockup Mi perfil vendedor

Entradas

- Nombre, contraseña, ubicación, teléfono.

Salidas

- Ninguna

Acciones

- **Actualizar nombre:** Se muestra una ventana modal, donde se permite actualizar el nombre completo de la empresa.
- **Actualizar contraseña:** Se muestra una ventana modal, donde se permite actualizar la contraseña de acceso del usuario.
- **Actualizar ubicación geográfica:** Se muestra una ventana modal, donde se permite actualizar la ubicación geográfica de la empresa por medio de un selector de mapa.
- **Actualizar numero telefónico:** Se muestra una ventana modal, donde se permite actualizar el numero telefónico del usuario.
- **Actualizar Fecha de nacimiento:** Se muestra una ventana modal, donde se permite actualizar la fecha de nacimiento del usuario por medio de una lista de selección de fecha.
- **Añadir/Actualizar fotografía:** Se muestra una ventana de selección de fotografía de la galería multimedia del dispositivo móvil, la foto selecciona, actualiza la existente en caso de haber.

Mensajes

- Mensaje mientras se actualizan los datos con la leyenda "Guardando cambios".

Observaciones

Si es la primera vez que se utiliza la aplicación, entonces se solicita permiso para acceder a la galería multimedia del dispositivo, mismo que se deberá de autorizar para continuar con el proceso.

4.3. Submódulo de lógica de negocio

4.3.1. Aplicación web de administrador

El administrador será cargado directamente en base de datos, mencionado lo anterior, solo se permite iniciar sesión y no un registro como tal, de igual manera, listar solicitudes a validar, listar RFC's aprobados y rechazados.

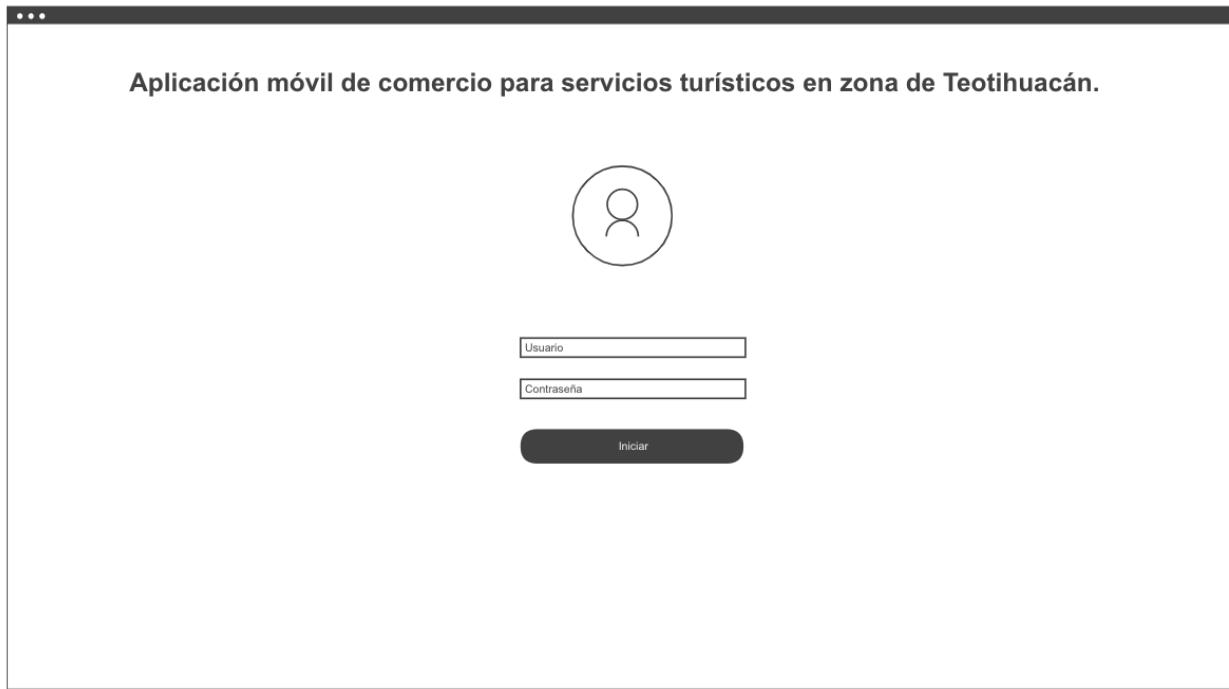


Figura 41. MockUp - Administrador, Iniciar sesión

Entradas
- Identificador de usuario y contraseña proporcionadas por el administrador del sistema (Puede ser o no la misma persona que usara este subsistema web).
Salidas
- Ninguna
Acciones
- Iniciar : Verifica los datos ingresados por el usuario y de ser correctos se muestra la pantalla de Selección de operación (Ver Figura 42. MockUp - Administrador, Selección de operación), en caso de existir algún error se muestra el mensaje MSG-01.
Mensajes
- MSG-01: Datos incorrectos, intente nuevamente
Observaciones
Ninguna

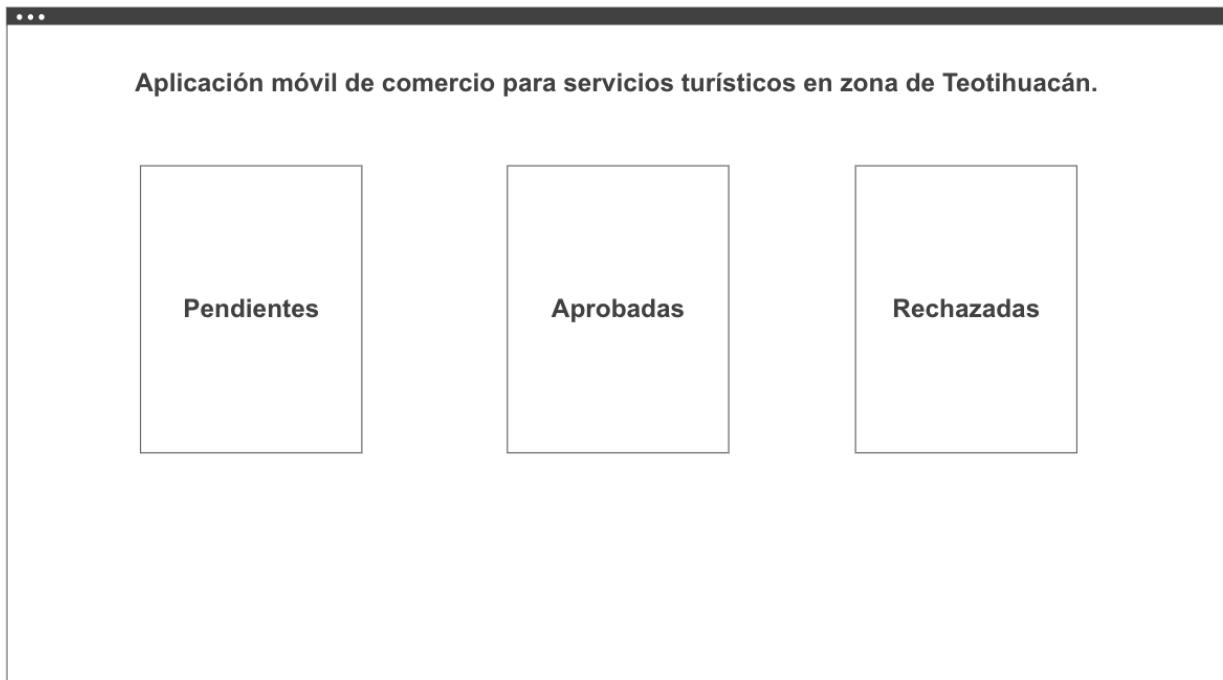


Figura 42. MockUp - Administrador, Selección de operación

Entradas

- Selección de acción (Pendientes, Aprobadas y Rechazadas).

Salidas

- Ninguna

Acciones

- **Pendientes**: para solicitudes no dictaminadas, al dar clic se muestra la pantalla que lista las solicitudes no dictaminadas (Ver Figura 43. MockUp - Administrador, Solicitudes pendientes).
- **Aprobadas**: para solicitudes dictaminas como aprobadas, al dar clic se muestra la pantalla que lista las solicitudes dictaminadas como aprobadas anteriormente (Ver Figura 44. MockUp - Administrador, Solicitudes aprobadas).
- **Rechazadas**: para solicitudes dictaminadas como rechazadas, al dar clic se muestra la pantalla que lista las solicitudes dictaminadas como rechazadas anteriormente (Ver Figura 45. MockUp - Administrador, Solicitudes rechazadas).

Mensajes

- Ninguna

Observaciones

Ninguna

Aplicación móvil de comercio para servicios turísticos en zona de Teotihuacán.		
Pendientes		
RFC	Ubicación	Acción
GNRNNNDGN7653	Av 16 de Septiembre Sur 62, San Martín Centro, 55850 San Martín Centro, Méx., Mexico	✓ ✗
BTNENCNGRN	Calle Bernardino De Sahagún 445, Otumba Centro, 55880 Otumba de Gómez Fariás, Méx., Mexico	✓ ✗
JENTVUANTN67	Tuxpan 18, San Martín Centro, 55850 San Martín Centro, Méx., Mexico	✓ ✗

Figura 43. MockUp - Administrador, Solicitudes pendientes

Entradas
- Selección de acción (Aprobar, rechazar)
Salidas
- Ninguna
Acciones
<ul style="list-style-type: none"> - Aprobar: Aprueba la clave RFC del vendedor que registro dicha RFC, dándole permiso al usuario vendedor de registrar servicios, esta solicitud podrá ser consultada en la Pantalla de Solicitudes aprobadas (Ver Figura 44. MockUp - Administrador, <u>Solicitudes aprobadas</u>), misma donde se podrá revocar el dictamen a rechazada. - Rechazar: Rechaza la clave RFC del vendedor que registro dicha RFC, esta solicitud podrá ser consultada en la Pantalla de Solicitudes rechazadas (Ver Figura 45. MockUp - Administrador, <u>Solicitudes rechazadas</u>), misma donde se podrá revocar el dictamen a aprobada.
Mensajes
- Ninguno
Observaciones
El usuario administrador tendrá que validar la clave RFC en cuestión ante la página del SAT (https://portalsat.plataforma.sat.gob.mx/ConsultaRFC/) de ser una RFC valida podrá aprobar la solicitud, de lo contrario, rechazarla.

Aplicación móvil de comercio para servicios turísticos en zona de Teotihuacán.

Aprobadas

RFC	Ubicación	Acción
GNRNNNDGN7653	Av 16 de Septiembre Sur 62, San Martín Centro, 55850 San Martín Centro, Méx., Mexico	X
BTNENCNGRN	Calle Bernardino De Sahagún 445, Otumba Centro, 55880 Otumba de Gómez Fariás, Méx., Mexico	X
JENTVUANTN67	Tuxpan 18, San Martín Centro, 55850 San Martín Centro, Méx., Mexico	X

Figura 44. MockUp - Administrador, Solicitudes aprobadas

Entradas
- Ninguna
Salidas
- Ninguna
Acciones
- Rechazar: Rechaza la clave RFC del vendedor que registró dicha RFC, por lo que se revocará el permiso de registrar nuevos servicios al vendedor en cuestión, esta solicitud podrá ser consultada en la Pantalla de Solicituds rechazadas (Ver Figura 45. MockUp - Administrador, Solicituds rechazadas), misma donde se podrá revocar el dictamen a aprobada de nuevo.
Mensajes
- Ninguno
Observaciones
Ninguna

Aplicación móvil de comercio para servicios turísticos en zona de Teotihuacán.		
Rechazadas		
RFC	Ubicación	Acción
GNRNNNDGN7653	Av 16 de Septiembre Sur 62, San Martín Centro, 55850 San Martín Centro, Méx., Mexico	✓
BTNENCNGRN	Calle Bernardino De Sahagún 445, Otumba Centro, 55880 Otumba de Gómez Farías, Méx., Mexico	✓
JENTVUANTN67	Tuxpan 18, San Martín Centro, 55850 San Martín Centro, Méx., Mexico	✓

Figura 45. MockUp - Administrador, Solicitudes rechazadas

Entradas
- Ninguna
Salidas
- Ninguna
Acciones
- Aprobar: Aprueba la clave RFC del vendedor que registro dicha RFC, dándole permiso al usuario vendedor de registrar servicios, esta solicitud podrá ser consultada en la Pantalla de Solicitudes aprobadas (Ver Figura 44. MockUp - Administrador, Solicitudes aprobadas), misma donde se podrá revocar el dictamen a rechazada.
Mensajes
- TODO
Observaciones
Ninguna

4.3.2. Diagramas de negocio y secuenciación

A continuación, se muestran diagramas BPMN de cada una de las funcionalidades expresadas de el diagrama de funcionalidades del sistema (Ver Figura 11. Diagrama general de casos de uso).

4.3.2.1. Inicio de sesión y registro de usuarios

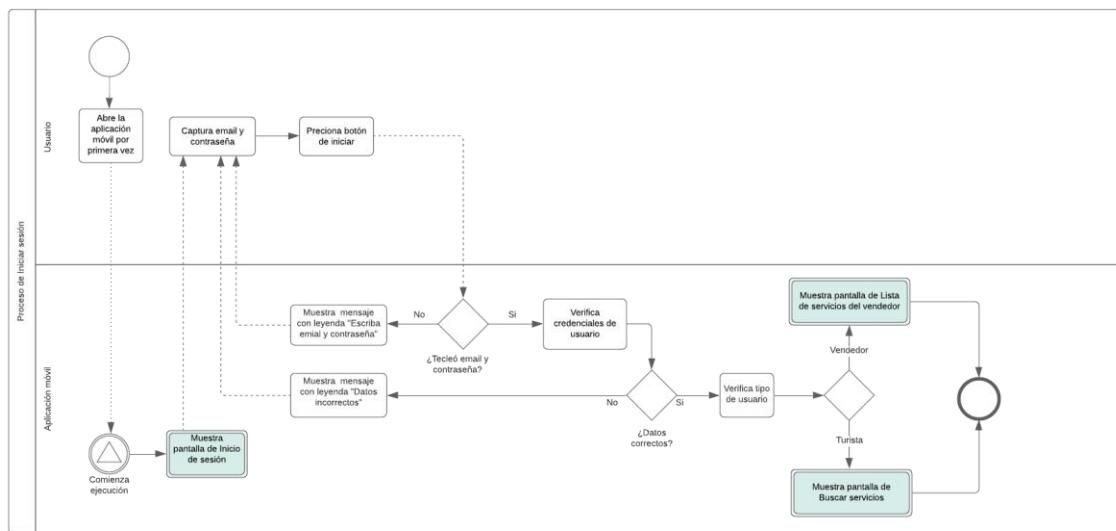


Figura 46. Diagrama BPMN - Iniciar sesión

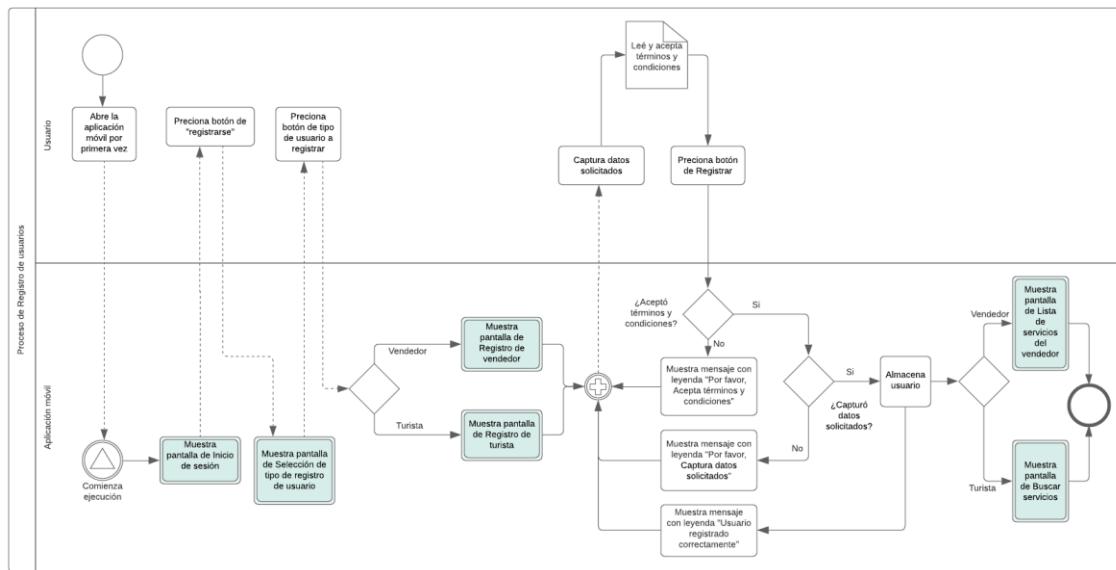


Figura 47. Diagrama BPMN - Registro de usuario

4.3.2.2. Usuario turista

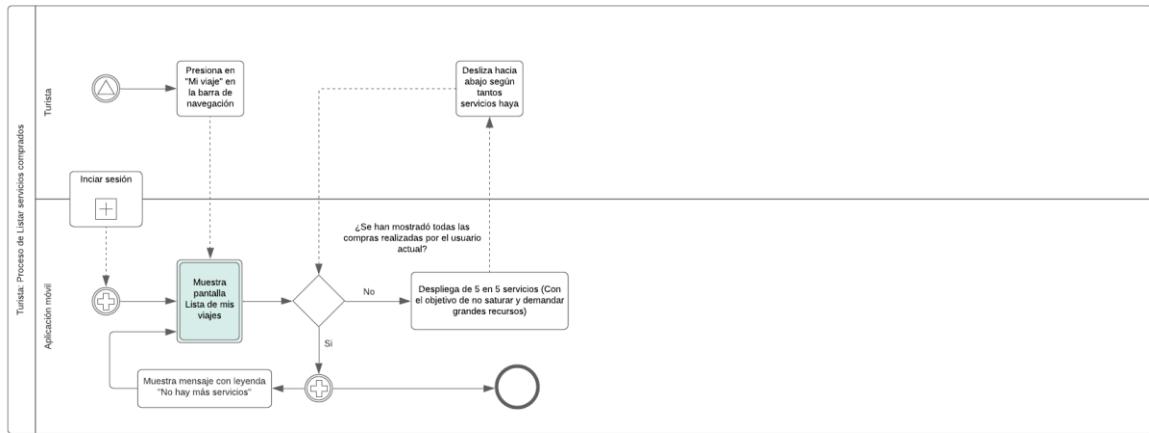


Figura 48. Diagrama BPMN - Turista, listar servicios

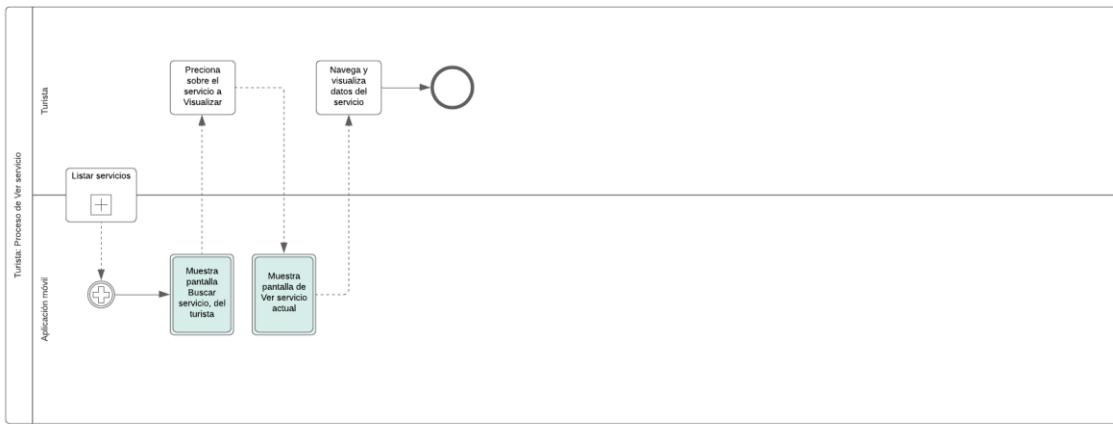


Figura 49. Diagrama BPMN - Turista, Ver servicio

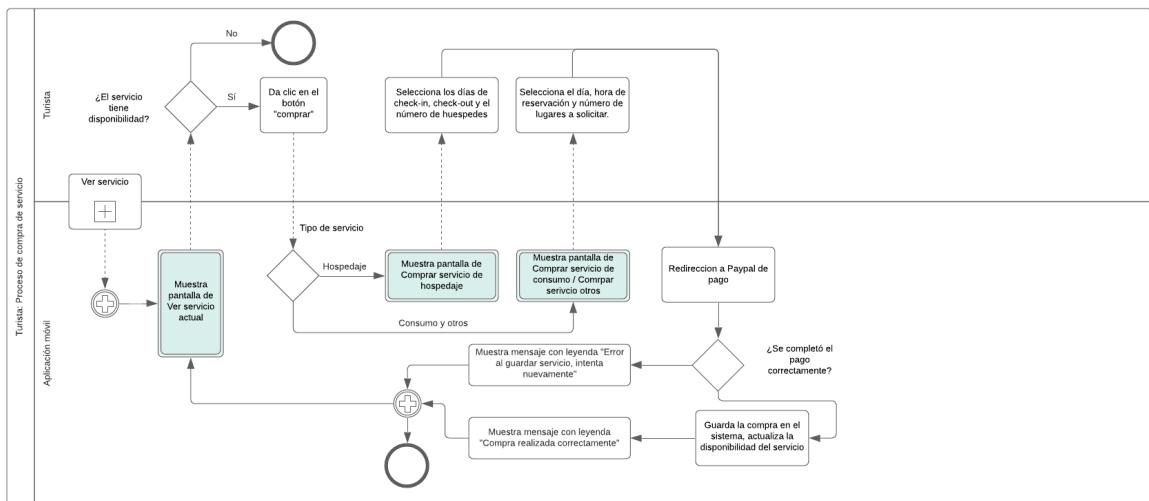


Figura 51. Diagrama BPMN - Turista, Comprar servicio

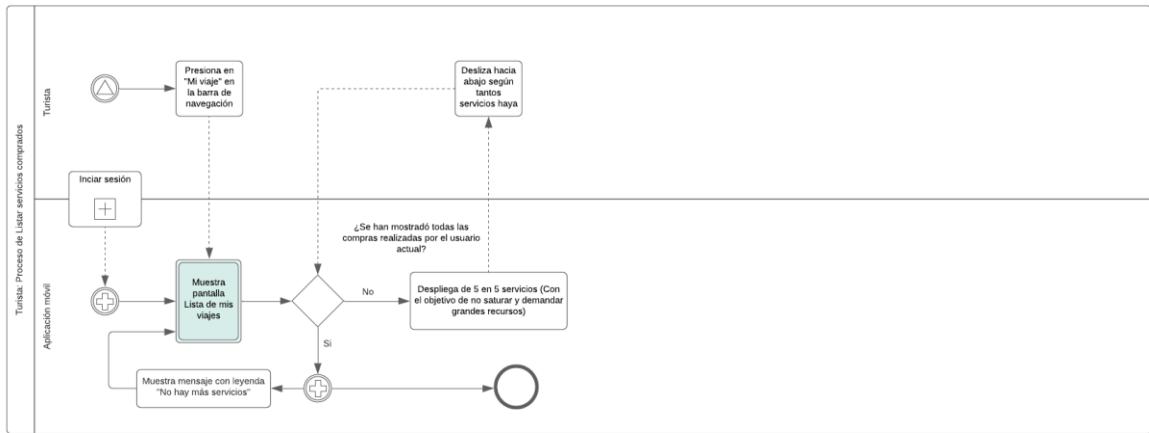


Figura 50. Diagrama BPMN - Turista, Listar compras

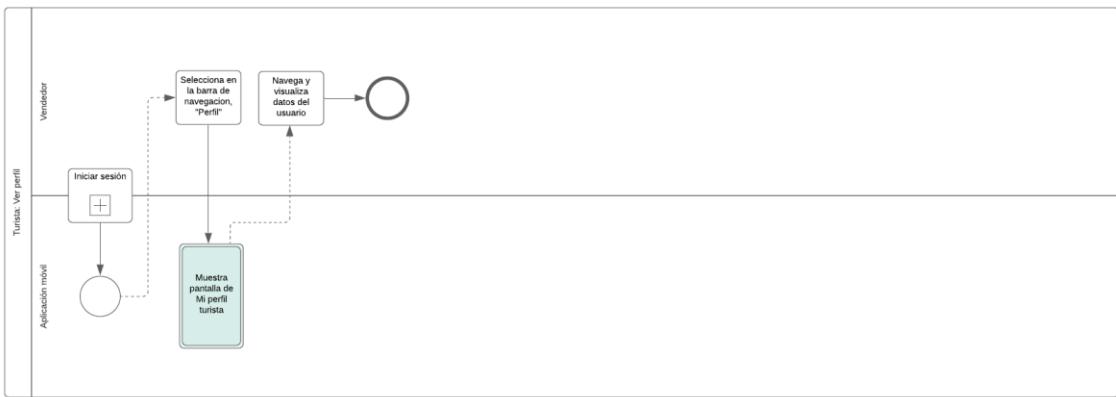


Figura 52. Diagrama BPMN - Turista, Ver perfil

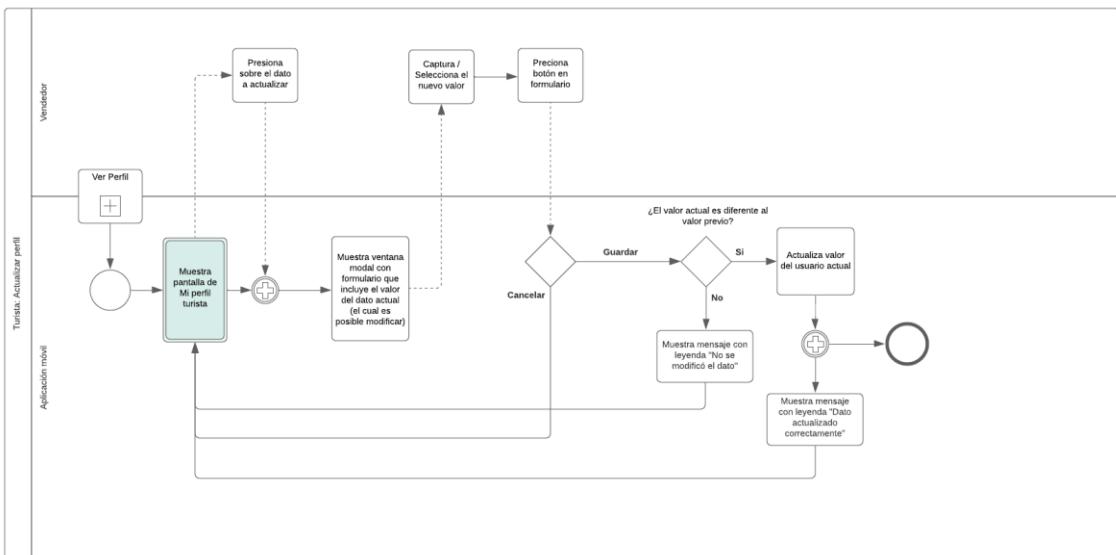


Figura 53. Diagrama BPMN - Turista, Actualizar servicio

4.3.2.3. Usuario vendedor

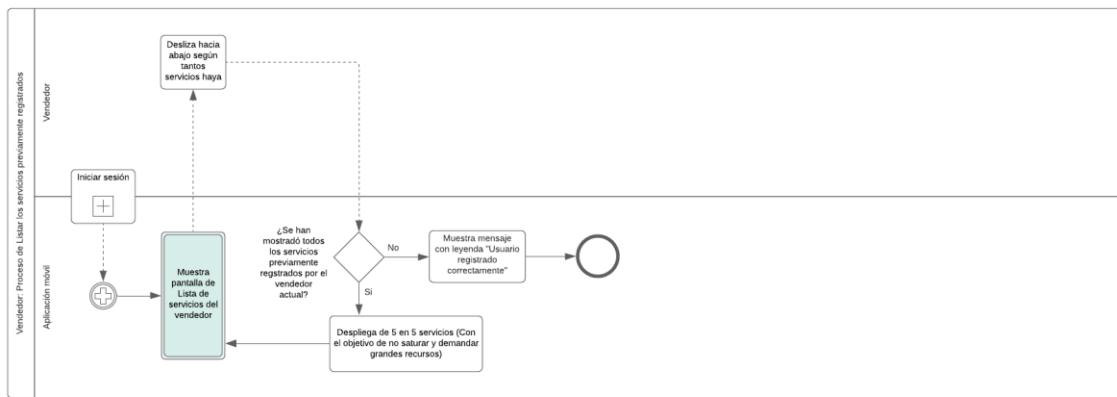


Figura 54. Diagrama BPMN - Vendedor, Listar servicios

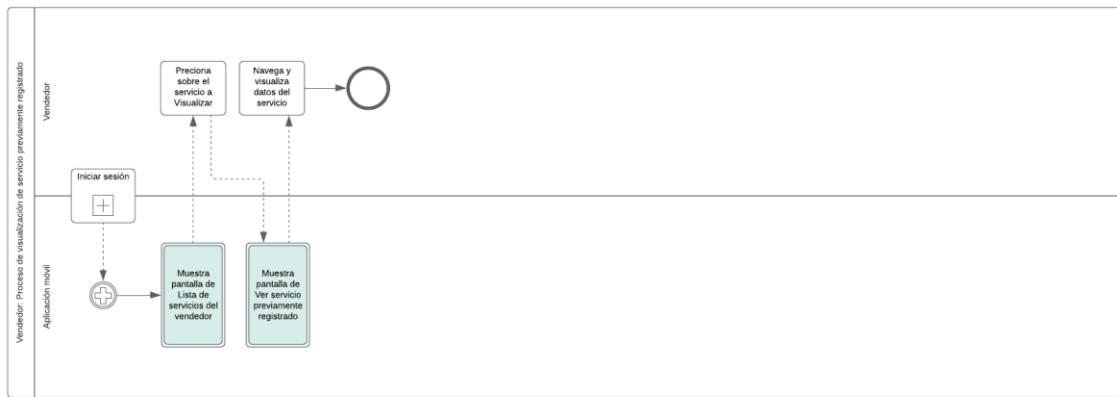


Figura 55. Diagrama BPMN - Vendedor, Ver servicio previamente registrado

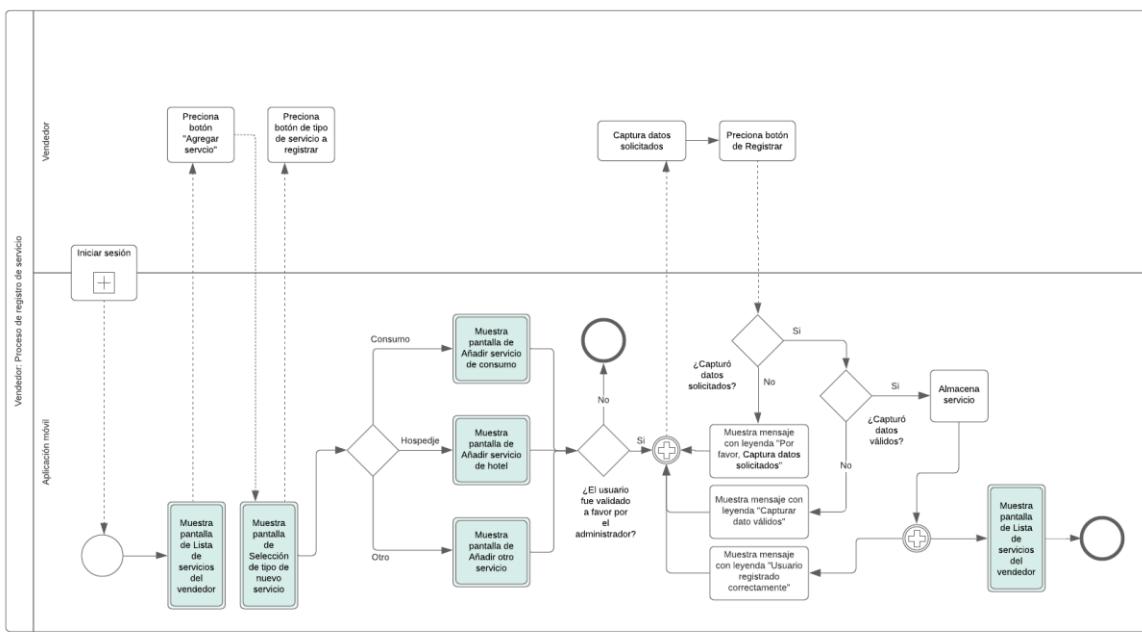


Figura 56. Diagrama BPMN - Vendedor, Registrar nuevo servicio

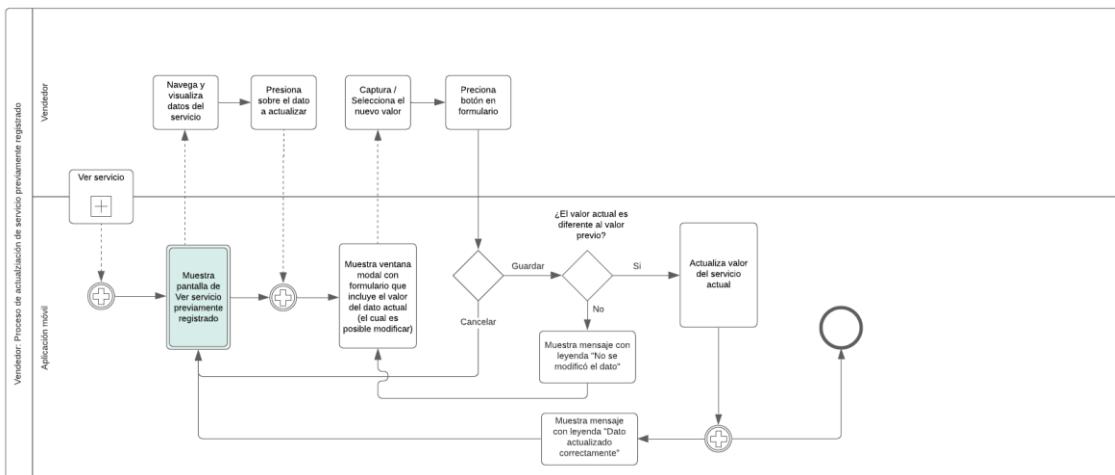


Figura 57. Diagrama BPMN - Vendedor, Actualizar servicio

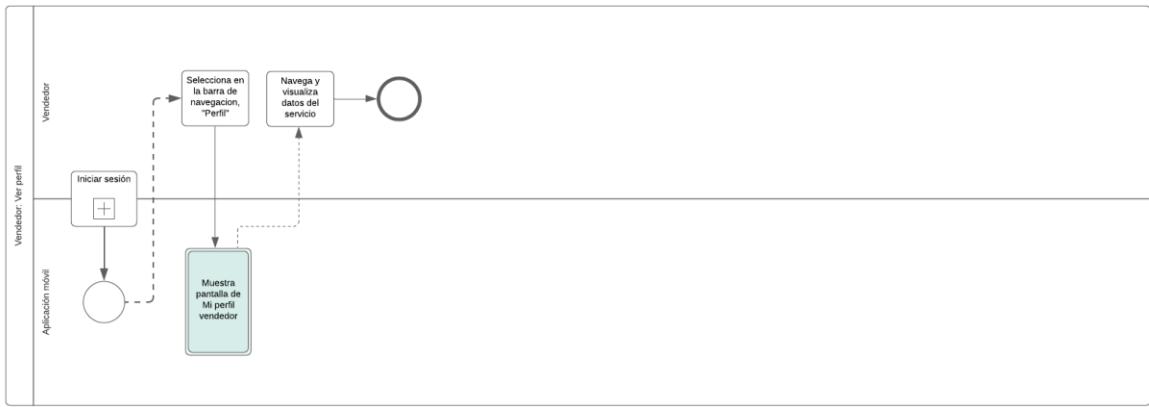


Figura 58. Diagrama BPMN - Vendedor, Ver perfil

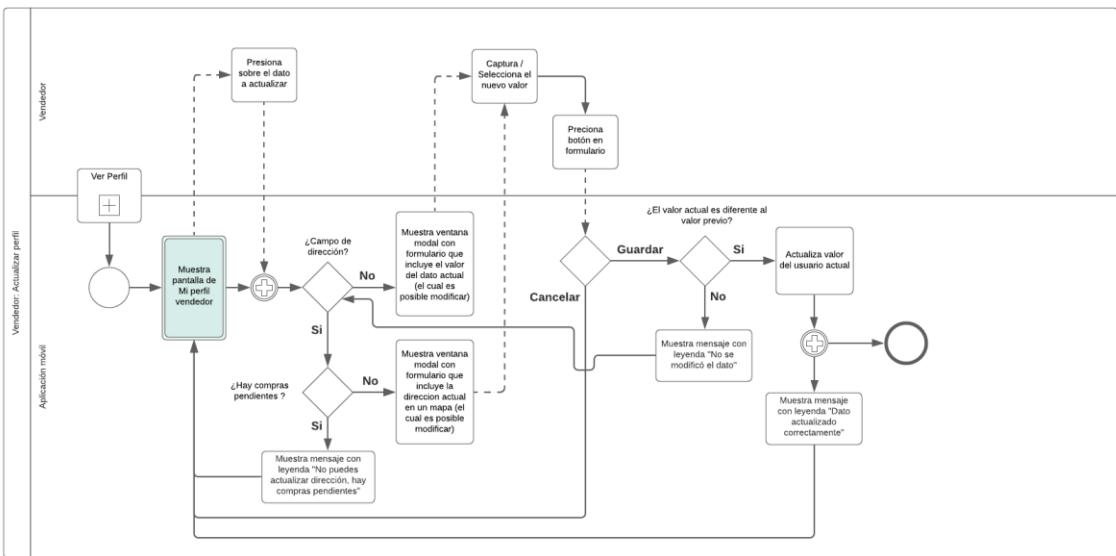


Figura 59. Diagrama BPMN - Vendedor, Actualizar perfil

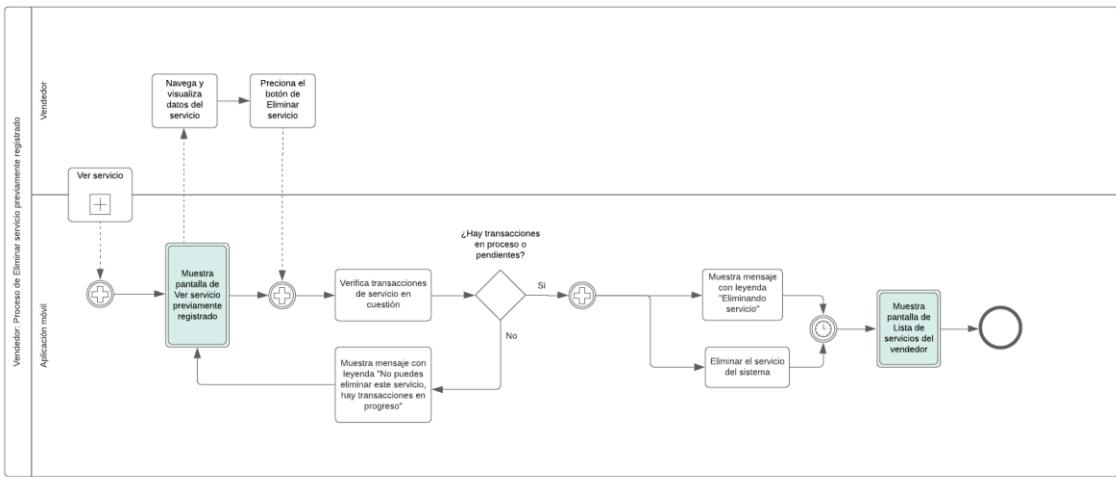


Figura 60. Diagrama BPMN - Vendedor, Eliminar servicio

4.4. Submódulo de idioma

El proyecto esta integrado por 3 tipos de usuario; turista, vendedor y administrador, dada la naturaleza y requerimientos por cada tipo de usuario únicamente el rol turista tiene la necesidad de traducción del lenguaje, por ello se consideran dos momentos donde se puede adecuar el idioma: en el inicio de sesión y cuando el usuario turista requiera cambiar el idioma ya estando dentro de su sesión.

- Inicio de sesión

La aplicación móvil comienza con la pantalla de inicio de sesión (Ver Figura 13. Mockup Inicio de sesión), donde puede iniciar sesión o registrarse, cualquiera de estas opciones se puede llevar a cabo por el usuario turista o vendedor, entonces en este punto podría darse la necesidad de adecuación al lenguaje del turista, para realizar alguno de los requerimientos que se proveen en este punto de la aplicación, es por ello por lo que aquí se proporcionará la traducción completa de la aplicación.

La manera en que se propone llevar a cabo este proceso es agregar un botón en la pantalla de inicio (Ver Figura 13. Mockup Inicio de sesión) en la parte inferior posterior a las leyendas y contenidos propios de esta pantalla, el cual al ser presionado cambiará el idioma de toda la aplicación al idioma correspondiente, es decir de inglés a español y viceversa.

- Cambio desde la aplicación de turista

La aplicación del usuario turista incluye una barra horizontal de navegación, la cual contiene los apartados: “Servicios”, “Mi viaje”, “Pagos” y “Mi perfil”, dentro de este último se encuentran los datos del turista que registró en la aplicación (Ver Figura 27. MockUp Mi perfil de usuario turista), se encuentra un ítem “lenguaje”, el cual al ser presionado cambiará el idioma de toda la aplicación al idioma correspondiente, es decir de inglés a español y viceversa.

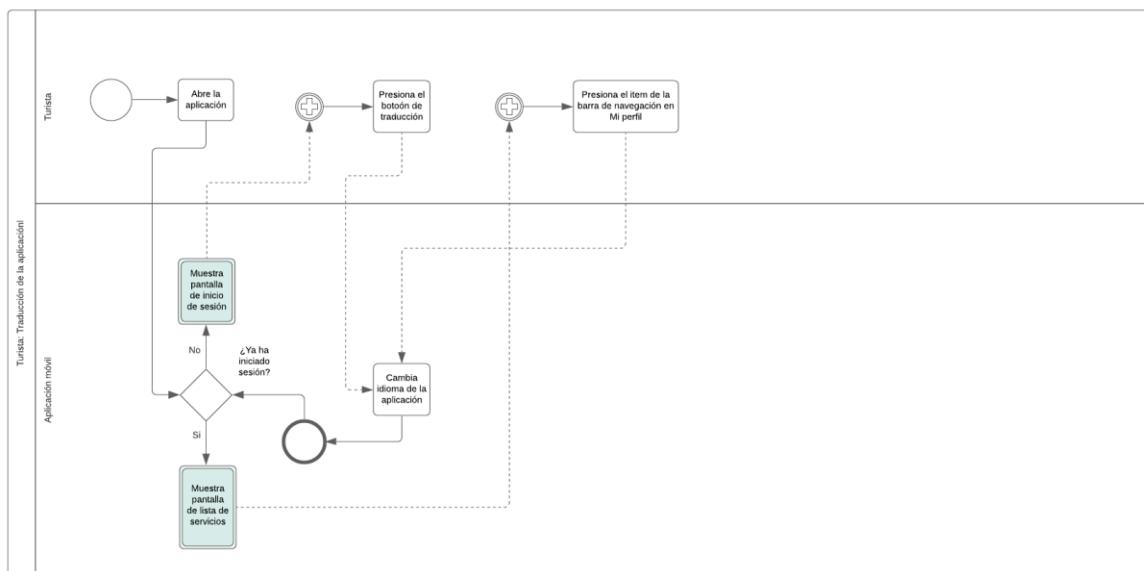


Figura 61. Diagrama BPMN - Turista, Cambio de idioma

4.5. Submódulo de generación de código QR

El sistema de aplicación móvil este compuesto por 2 usuarios, turista y vendedor, entre estos usuarios se hace el “hand shake” en el comercio de servicios turísticos ofertados en la zona arqueológica de Teotihuacán, dentro de este proceso encontramos la siguiente bifurcación de escenarios:

- El turista ya ha comprado y pagado un servicio por medio de la aplicación móvil, se presenta físicamente al establecimiento donde se oferta su servicio en tiempo y forma, solicita acceso al vendedor, el vendedor deberá autenticar la veracidad de la transacción, aquí es donde por medio de un código QR que el turista tiene en el apartado de su compra, mismo que mostrara al vendedor, luego entonces el vendedor correspondiente escanea dicho código QR, de ser valido se le permite el acceso.
- El turista no ha comprado ni pagado por un servicio determinado, acude al establecimiento físico donde se oferta el servicio correspondiente, por lo que solicita al usuario vendedor que le venda el acceso al mismo, el vendedor por medio de su aplicación móvil en usuario vendedor podrá generar una orden de pago de acuerdo con las necesidades indicadas por el turistas, al completar el llenado de todos los campos requeridos en este proceso, la aplicación móvil generara un código QR, mismo que podrá escanearlo el usuario turista y proceder con el pago.

A continuación, se muestran diagramas BPMN de cada una de las funcionalidades expresadas de el diagrama de funcionalidades del sistema (Ver Figura 11. Diagrama general de casos de uso).

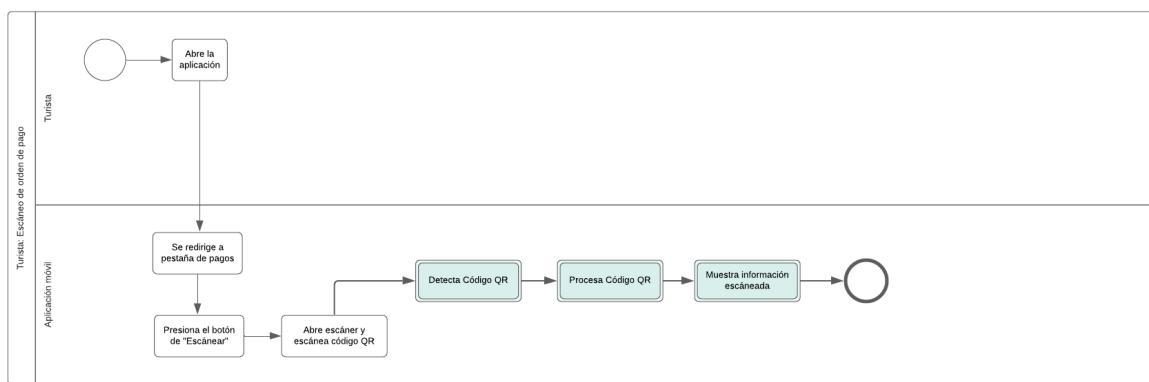


Figura 62. Diagrama BPMN - Turista, Escanear orden de pago

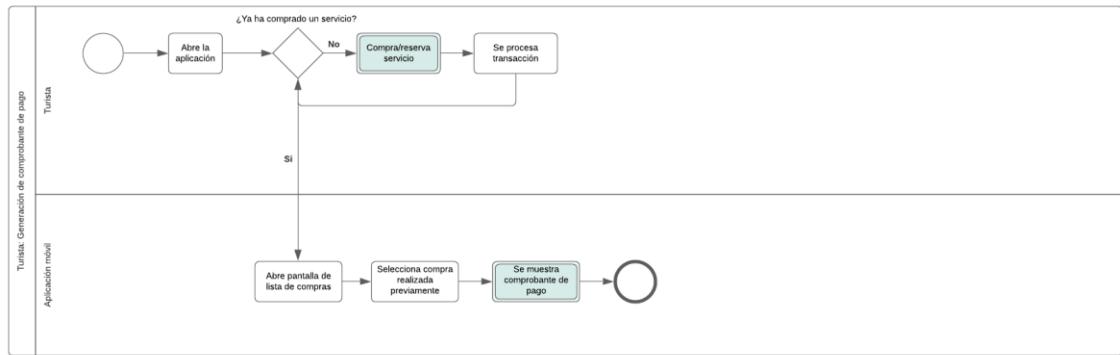


Figura 63. Diagrama BPMN - Turista, Generar comprobante de pago

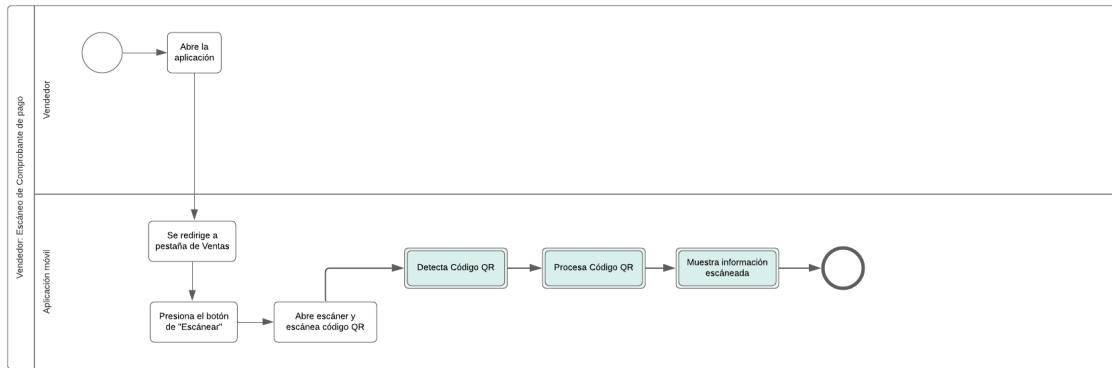


Figura 64. Diagrama BPMN - Vendedor, escanea comprobante de pago

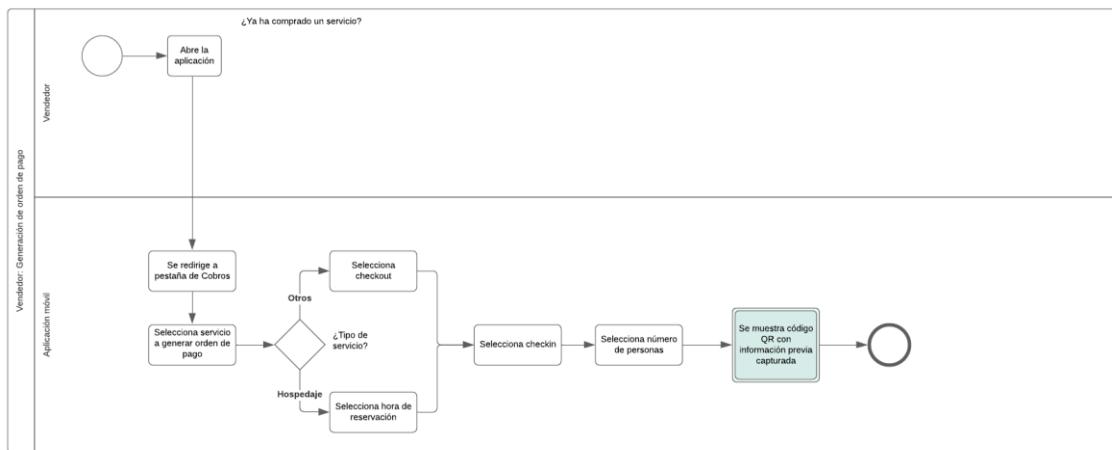


Figura 65. Diagrama BPMN - Vendedor, genera orden de pago

4.6. Submódulo de pagos

Este modulo es la evolución del modulo anterior (Generación y lectura de códigos QR), de manera breve se retoma lo anterior; hay 1 escenario para cada tipo de usuario que puede leer y generar un código QR, mismos que uno de cada uno de estos se extiende la funcionalidad para continuar con el proceso de pago, estos son a) Turista compra servicio por si mismo o bien, b) el turista paga el servicio a partir de una orden de pago que le genera el vendedor correspondiente.

Lo que se busca diseñar es un componente de pagos reutilizable para estas dos acciones mencionadas anteriormente, bajo la premisa de programar una vez y utilizar N veces el producto funcional de tal manera que se haga escalable para futuras versiones y lanzamientos.

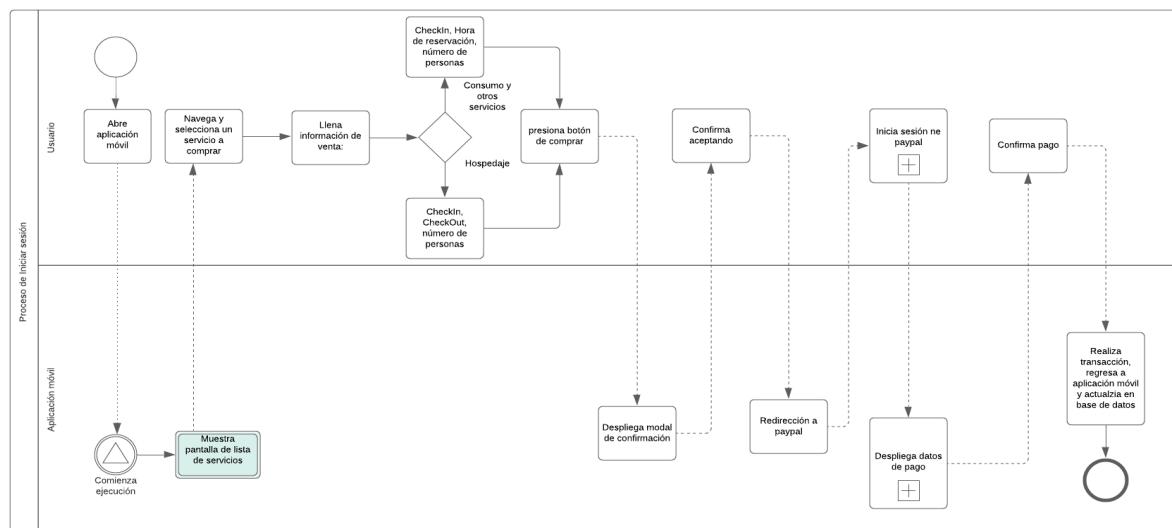


Figura 67. Diagrama BPMN - Turista, realiza pago en Paypal

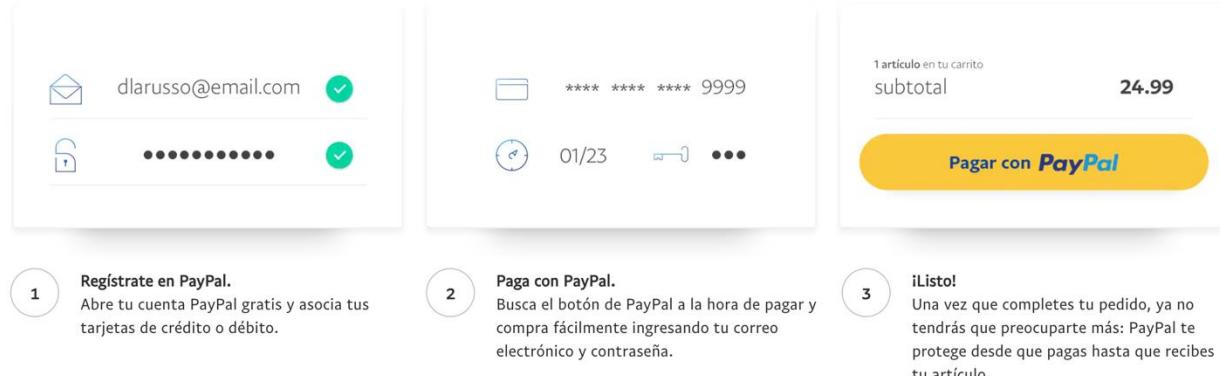


Figura 66. Funcionamiento de Paypal

El proceso de integración de Paypal con react native se puede detallar de la siguiente manera:

Si no ha creado una cuenta de paypal sandbox, asegúrese de hacerlo dirigiéndose a:
<https://developer.paypal.com/developer/accounts/>

Crearemos dos cuentas sandbox:

- Cuenta de comerciante de Paypal : se utiliza para aceptar pagos del usuario final.
- Cuenta personal de Paypal : se utiliza para realizar el pago a la cuenta del comerciante.

A continuación, necesitamos crear una aplicación API REST en paypal que se utilizará para configurar paypal en nuestro backend. Hay que asegurarse de seleccionar la cuenta de comerciante de PayPal que creó con anterioridad al crear la aplicación REST.

Instalar las bibliotecas de backend

```
npm install react-native-paypal --save or $ yarn add react-native-paypal
```

Uso de react native paypal

Primero se necesita obtener un token válido de nuestro servidor.

Luego se puede ejecutar el siguiente código, por ejemplo, reaccionando al presionar un botón.

```
importar { requestOneTimePayment , requestBillingAgreement } de 'react-native-paypal' ;  
  
// Para pagos  
  
únicos const {  
nonce , payerId ,  
email ,  
firstName ,  
lastName ,  
teléfono  
} = Await requestOneTimePayment (  
símbolo ,  
{  
cantidad : '5' , // requerida  
// cualquier PayPal apoyó la moneda (ver aquí:  
https://developer.paypal.com/docs/integration/direct/rest/currency-codes/#paypal-account-payments  
moneda : 'GBP' ,  
// cualquier configuración regional compatible con PayPal (consulte aquí:  
https://braintree.github.io/braintree\_ios/Classes/BTPayPalRequest.html#/c:objc\(cs\)BTPayPalRequest\(py \) localeCode  
localeCode : 'en_GB' ,  
shippingAddressRequired : false ,  
userAction : 'commit' , // mostrar 'Pagar ahora' en la página de revisión  
de PayPal  
// uno de 'autorizar' , 'venta' , 'pedido'. por defecto es 'autorizar'. vea  
los detalles aquí: https://developer.paypal.com/docs/api/payments/v1/#payment-create-request-body  
intent : 'autorizar' ,  
}
```

```

) ;

// Para guardar una cuenta de PayPal, consulte:
https://developers.braintreepayments.com/guides/paypal/vault
const  {
    nonce ,
    payerId ,
    email ,
    firstName ,
    lastName ,
    teléfono
}  =  Await  requestBillingAgreement (
símbolo ,
{
    billingAgreementDescription : 'Su descripción del acuerdo' , // requerida
    // cualquier PayPal apoyó la moneda (ver aquí:
    https://developer.paypal.com/docs/integration/direct/rest/currency-codes / #
paypal-account-payments)
    moneda : 'GBP' ,
    // cualquier configuración regional compatible con PayPal (consulte aquí:
    https://braintree.github.io/braintree_ios/Classes/BTPayPalRequest.html#/c:objc
(cs)BTPayPalRequest (py) localeCode)
    localeCode : 'en_GB' ,
}
) ;

```

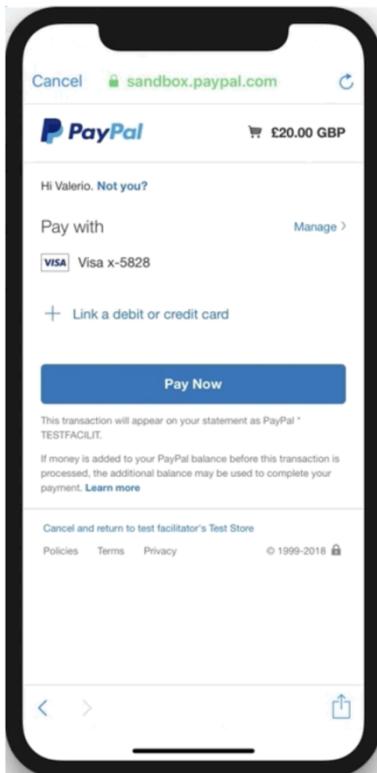


Figura 68. Pantalla de pago en PayPal

5. Desarrollo

5.1. Submódulo de aplicación móvil

5.1.1. Creación del proyecto

Para iniciar a el proyecto se necesita instalar la herramienta, ejecutar:

```
$ yarn global add create-react-native-app
```

Esto instalará de forma global el comando create-react-native-app

Para crear el proyecto solo se ejecuta:

```
$ create-react-native-app proyecto
```

```
$ cd proyecto
```

La estructura del proyecto, se tendrá una carpeta con los archivos para construir tu aplicación:



Figura 69. Estructura de un proyecto en react native

- App.js, contiene el código de la app, compartimos un solo archivo para Android y IOS
- App.test.js, contiene test del componente en App.js
- Package.js, es el manifiesto del proyecto

Para ejecutar el proyecto, permite correr el proyecto en entorno de desarrollo, permitiendo conectar desde nuestros dispositivos y ver los cambios usando hot reloading.

```
yarn start
```

Cuando se ejecuta yarn start en la consola, va a arrojar un código QR. Para poder acceder desde un dispositivo se deberá descargar la de Expo, dentro de la app escanear el código y se podrá usar la app.

Después de ejecutar la aplicación, se quedará el promp a la espera de acciones, por ejemplo, los siguientes caracteres son importantes para ejecutar en web, IOS y Android.

- › Press **a** | open Android
- › Press **i** | open iOS simulator
- › Press **w** | open web

En la carpeta raíz, se crea la carpeta src, donde se crearán los directorios, los cuales son la base de todo el desarrollo de la aplicación móvil.

- Components

Funciones y componentes reutilizables de interfaz gráfica reutilizables, es decir su uso es en más de una vez, por ejemplo: Carousel de imágenes, Mapa de geolocalización, Modales, Loadings, Restauración de contraseña, etc.

- Language

Carpeta con archivos JSON de idioma, los cuales incluyen todos los textos que la aplicación usa, se tienen objetos los cuales son pantallas, donde como atributo se indica en clave – valor, el idioma y el texto en respectivo idioma.

- Catalogs

Carpeta que incluye archivos JSON que generalmente incluyen bastantes datos, incluye catálogos, avisos de privacidad, entre otros.

- navigation

Este directorio es de vital importancia para el funcionamiento de la aplicación, dado que se ofertan dos tipos de usuario: turista y vendedor, esta carpeta incluye archivos que definen la estructura de la barra de navegación para tipo de usuario, y todos los vínculos a las pantallas que incluye la aplicación, las rutas que son válidas o alcanzables previo y posterior al inicio de sesión.

- screens

Ya que se ha definido la estructura de navegación de la aplicación móvil, esta carpeta almacena todos los archivos que se muestran en la navegación, es decir pantallas, donde cada una de estas puede embeber componentes creados previamente y hacer la redirección a otras pantallas.

- utils

Archivos JS con funcionalidades específicas, como lo son el acceso a base de datos, datos de configuración y de acceso a firebase, obtención de usuarios, funciones para validaciones, etc.

5.1.2. Navegación

5.1.2.1. Rutas no autenticadas

La aplicación móvil permite el inicio de sesión, registro de usuarios, restauración de contraseña, selección del tipo de usuario a crear, por lo que se crea un archivo .js, este archivo se incluyen las pantallas que no requieren autenticar usuarios, es decir, cualquier usuario las puede ver, por ello que se manda a llamar desde el App.js, entonces será la segunda función por defecto que se ejecutara en el proyecto.

Es necesario instalar las librerías de navegación nativa y pila de navegación:

```
yarn add @react-navigation/native  
yarn add @react-navigation/stack
```

Posterior a ello se crea el archivo UnauthenticatedRoutes.js, en el cual se importan las librerías instaladas previamente (por defecto siempre se importa la librería de React), se importan las pantallas que incluye esta navegación (se muestran más adelante), se exporta por defecto la función llamada igual que el archivo, dentro de esta función se expresa cual será la pantalla por defecto a mostrar, la cual es la pantalla de login, y posterior a ello se indican cuales son las demás que conforman la navegación no autenticada.

5.1.2.2. Rutas autenticadas

En rutas autenticadas se tienen dos archivos .js: turistas y vendedores, en donde se especifica el número de ítems que tendrá la barra de navegación para cada tipo de usuarios (Ver Figura 12. Modelo de navegación de aplicación móvil).

Rutas autenticas - turista

Se importaran las pilas de navegación para la aplicación turista (más adelante se muestra un ejemplo de este tipo de archivo, se define la función TabBar(), la cual regresa la barra de navegación diseñada previamente, es por ello que se incluyen 4 elementos de tipo <Tab.Screen />.

```
const TabBar = () => {  
    const { t, i18n } = useTranslation()  
  
    return (  
        <Tab.Navigator  
            initialRouteName = "services"  
            tabBarOptions = {[  
                inactiveTintColor: "#646464",  
                activeTintColor: "#EAA11B"  
            ]}  
  
            screenOptions = {({route})=>({  
                tabBarIcon: ({color}) => showIcon(route, color)  
            })}  
        >  
        <Tab.Screen  
            component = {ServiceStack}  
            name = "services"  
            options = {({title: t('login.tab_bar.services')})}  
        />  
  
        <Tab.Screen  
            component = {MyTripStack}  
            name = "mytrip"  
        />  
    )  
}
```

```

        options = {{title: t('login.tab_bar.my_trip')}}}
    />

    <Tab.Screen
        component = {PaymentStack}
        name = "payment"
        options = {{title: t('login.tab_bar.payments')}}}
    />

    <Tab.Screen
        component = {ProfileStack}
        name = "profile"
        options = {{title: t('login.tab_bar.profile')}}}
    />

</Tab.Navigator>
)
}

```

Se está trabajando una barra de navegación vertical mediante el componente nativo <Tab.Navigator>, a cada ítem se le agrega un ícono representativo desde material design icons, para ello se usa a función showIcon(route, color), la cual recibe como argumentos el nombre del stack de navegación correspondiente y el color con el cual se ilumina el ítem seleccionado.

```

function showIcon(route, color) {
    let iconName;

    switch(route.name) {
        case "services":
            iconName = "magnify";
            break;
        case "mytrip":
            iconName = "cart-arrow-down";
            break;
        case "payment":
            iconName = "credit-card-outline";
            break;
        case "profile":
            iconName = "account-circle-outline";
            break;
        default:
            break;
    }

    return <Icon
        type="material-community"
        name = {iconName}
        size={24}
        color = {color}
    />
}

```

Finalmente se exporta la función por defecto, de igual nombre al archivo contenedor.

```
export default function AuthenticatedRoutesTurist() {
    return(
        <NavigationContainer>
            <TabBar/>
        </NavigationContainer>
    )
}
```

De igual manera se realiza para el caso de la aplicación para vendedores, únicamente modificando las pilas de navegación por las de vendedor y nombres de iconos de designicons.

5.1.2.3. Selector de ruta de navegación

Ya se ha definido la navegación para rutas autenticadas y no autenticadas, la aplicación después de autenticarse es capaz de detectar el tipo de usuario que ingresa a la aplicación y mostrarle las pantallas correspondientes dadas por la navegación programada previamente:

```
export default function SwitchNavigator (){
    const [loading, setLoading] = useState(true);
    const [user, setUser] = useState({});

    useEffect(() =>{
        (
            async () => {
                setUser(await getUserData())
            }
        )()
    }

    setTimeout(() =>{
        setLoading(false);
    }, 1000)
}, [])

if(loading){
    return <Loading isVisible = {loading} text = "Loading configuration"
/>
} else {
    if(user.role == "seller"){
        return <AuthenticatedRoutesSeller/>
    }else{
        return <AuthenticatedRoutesTurist/>
    }
}
}
```

Más adelante revisaremos conexión a firebase, en este punto es suficiente conocer que se hace una consulta del usuario actual en la aplicación, donde se pone énfasis en el atributo tipo de usuario, es por ello por lo que se detecta el tipo de rol que tiene el usuario.

5.1.2.4. Pila de navegación

Las pilas de navegación son archivos .js, los cuales se incluyen las referencias a cada pantalla por cada ítem, por ejemplo, el ítem “servicios” de la barra de navegación de la aplicación para turista permite listar los servicios coincidentes con una búsqueda, de igual manera permite ver los detalles de un servicio en particular además permite comprar el servicio, es decir, se agregan X elementos a la pila de navegación, a continuación, se muestra la pila de navegación para el ítem Servicio de la aplicación turista.

```
const Stack = createStackNavigator();

export default function MyTrypStack() {
  const { t, i18n } = useTranslation()

  return(
    <Stack.Navigator>
      <Stack.Screen
        component = {MyTrip}
        name = "mytrip"
        options = {{
          title: t('mytrip.stack.mytrip_list')
        }}
      />

      <Stack.Screen
        component = {Receipt}
        name = "receipt"
        options = {{
          title: t('mytrip.stack.receipt'),
          headerStyle: {backgroundColor : "#EAA11B"},
          headerTintColor: "#fff"
        }}
      />

      <Stack.Screen
        component = {ShoppingCar}
        name = "shoppingcar"
        options = {{
          title: t('mytrip.stack.shopping_car'),
          headerStyle: {backgroundColor : "#EAA11B"},
          headerTintColor: "fff"
        }}
      />
    </Stack.Navigator>
  )
}
```

El atributo “options” dentro de cada <Stack.Screen /> nos permite definir un encabezado en cada pantalla, este atributo es posible cambiarlo desde la pantalla en la que se hace referencia.

5.1.3. Plantilla de código para una pantalla genérica

Cada una de las pantallas que se agregaron en las pilas de navegación y rutas no autenticadas tienen la siguiente estructura base, cada una agrega funcionalidad y elementos según sea el caso y requerimientos. En la gran mayoría de etiquetas de React es posible añadir estilos, la sintaxis de estilos es CSS tradicional, hay infinidad de librerías que permiten embeber componentes para así el desarrollador dedicarse en lo mayor posible a la lógica de operación.

```
import React, {useState, useEffect} from 'react'
import { View, Text, StyleSheet } from 'react-native'

export default function Pay() {
    return (
        <View style={styles.view_container}>
        </View>
    )
}

const styles = StyleSheet.create({
    view_container:{
        flex: 1,
        backgroundColor: '#fff',
    }
})
```

5.1.4. Componentes

5.1.4.1. Selector de tiempo

El selector de tiempo o “Date Picker selector”, se usará para seleccionar el horario de citas o reservaciones por. Cada tipo de servicio a excepción de hospedaje se usa el mismo componente, pero con cada sistema operativo lo procesa y muestra gráficamente acorde al componente nativo correspondiente, es posible utilizar componentes específicos por cada sistema operativo, esto se logra mediante validaciones incrustadas dentro del componente principal mediante estructuras de control if, else.

Para este caso se decidió realizar el componente desde cero, ya que al utilizar el componente genérico de React Native (hay infinidad de opciones libres en la web), genera inconvenientes variados de acuerdo con el sistema operativo en donde se utiliza, y no cumple con el 100% de las necesidades planteadas, por ejemplo, se requieren rangos de mitad de hora, en horarios a partir de una hora mínima y hasta una hora máxima.

Instalar las siguientes librerías:

```
yarn add moment  
yarn add react-native-calendars
```

Se crea un archivo JavaScript llamado HourSelection.js, en el cual se exporta una función por defecto con el nombre de HourSelection(), como retorno de la función se incluye el siguiente fragmento de código.

```
<ScrollView style={styles.scroll_view}>  
    <Text style={styles.txt_header}>  
        Selecciona tu hora de reservación  
    </Text>  
  
    <FlatList  
        data={half_hours}  
        renderItem={(half_hour) => (<HalfHourItem half_hour={half_hour}  
            selected={selected} setSelected={setSelected}/>) }  
        keyExtractor={(item, index) => index.toString()}  
    />  
  
    <View style={styles.viewBtn}>  
        <Button  
            title="Guardar"  
            containerStyle={styles.viewBtnContainerSave}  
            buttonStyle={styles.viewBtnSave}  
            onPress={() => OnSubmit()}  
        />  
        <Button  
            title="Cancelar"  
            containerStyle={styles.viewBtnContainerCancel}  
            buttonStyle={styles.viewBtnCancel}  
            onPress={() => setShowModal(false)}  
        />  
    </View>  
</ScrollView>
```

Se destaca <FlatList />, el cual es un componente que muestra una lista de ítems, recibe como argumentos el arreglo de tiempo (contiene cadenas con horas intermedias y completas, por ejemplo:

8:00, 8:30, 9:00, ...), también recibe el componente genérico en como se procesara cada elemento del arreglo, para esto se muestra el fragmento de código del ítem genérico.

```
function HalfHourItem(props) {
    const { half_hour, selected, setSelected } = props

    const check_selected = (self_index) => {
        if(selected == null){
            return false
        }else if (selected != self_index){
            return false
        }else {
            return true
        }
    }

    return (
        <ListItem bottomDivider
            onPress={() => {
                setSelected(half_hour.index)
            }}
        >
            <ListItem.Content>
                <ListItem.Title
style={styles.item_title}>{half_hour.item}</ListItem.Title>
                </ListItem.Content>
                <ListItem.Chevron type="material-community" size={24} name =
{check_selected(half_hour.index) ? "check-circle": "checkbox-blank-circle-
outline"} color={"#404040"} />
            </ListItem>
        )
    )
}
```

Este fragmento permite seleccionar únicamente un ítem, cada que se presiona sobre otro, se desactiva el actual y se activa el más reciente presionado, como se observa, cada ítem de hora mostrara únicamente la hora y un selector (Chevron).

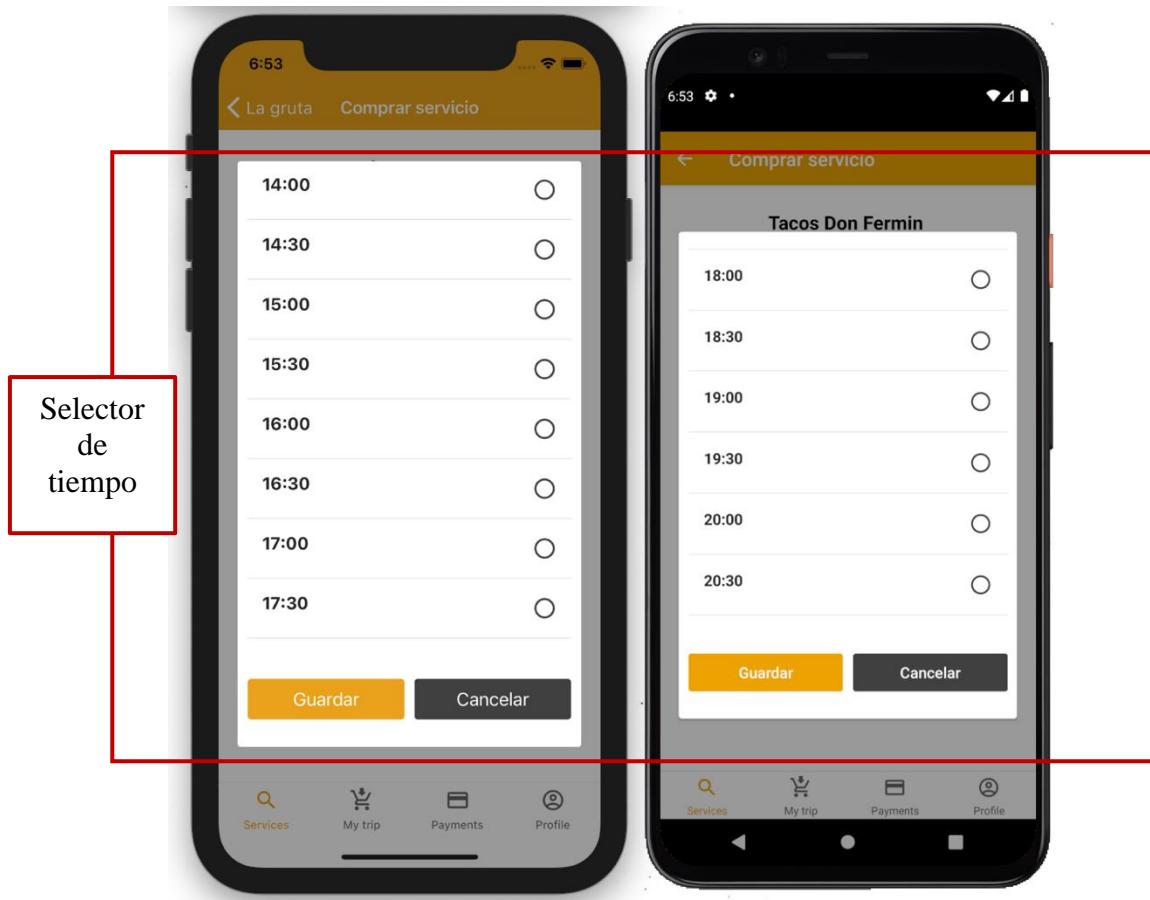


Figura 70. Componente Selector de tiempo en IOS y Android

5.1.4.2. Selector de fecha

El selector de fecha o “Date Picker selector”, se usará para seleccionar fechas de nacimiento. Cada sistema operativo lo procesa y muestra gráficamente acorde al componente nativo correspondiente, es posible utilizar componentes específicos por cada sistema operativo, esto se logra mediante validaciones incrustadas dentro del componente principal mediante estructuras de control if, else.

Para este caso se usar el componente genérico, por ello se tienen que instalar las siguientes librerías:

```
yarn add moment
yarn add react-native-modal-datetime-picker
yarn add react-native-modal-selector
```

<DateTimePickerModal /> es un componente ya creado y libre por la comunidad de React Native, funciona para fecha y para horas, después de hacer pruebas, se decidió usar únicamente la versión para fechas, ya que no genero errores y se adapta a las necesidades, recibe algunos argumentos de configuración, como lo es si esta visible o no, el modo de operación que en este caso es “date”, y los manejadores de acción en caso de confirmar o cancelar la operación.

```
<DateTimePickerModal
  isVisible={isDatePickerVisible}
  mode="date"
  onConfirm={handleConfirmDatePicker}
```

```
onCancel={hideDatePicker}  
/>
```

Ahora se muestran las funciones de confirmar y cancelar (estas funciones se retoman más adelante en la sección de lógica de negocio).

```
// Funcion para manegar el proceso despues a seleccion de fecha  
const handleConfirmDatePicker = (date) => {  
  
    setDateBirth(moment(date).format('L'));  
    hideDatePicker();  
};  
  
// Funcion bandera para deshabilitar modal picker de fecha  
const hideDatePicker = () => {  
    setDatePickerVisibility(false);  
};
```

Finalmente, en la siguiente imagen se observa gráficamente cada selector acorde al componente nativo de cada sistema operativo.

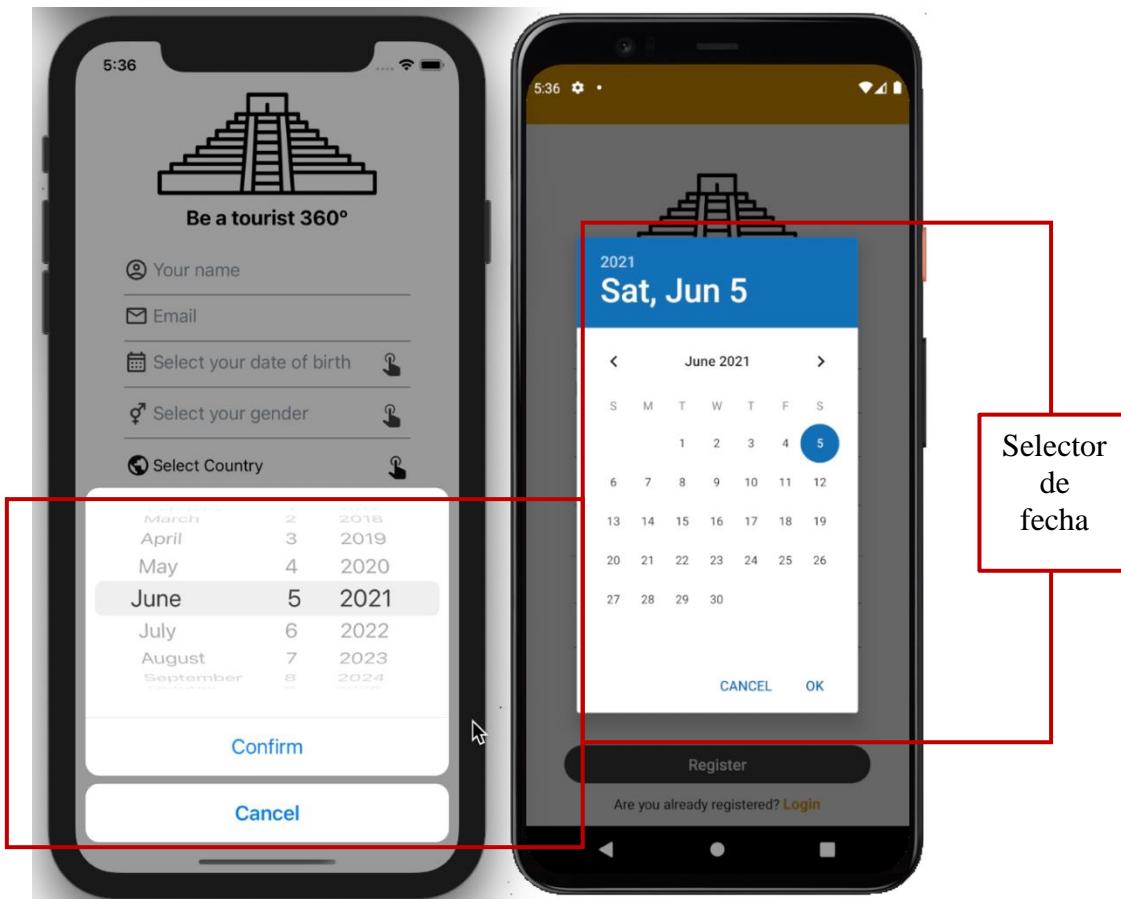


Figura 71. Componente Selector de fecha en IOS y Android

5.1.4.3. Selector de nacionalidad

El selector de nacionalidad o “Country Picker selector”, se usará este componente en el formulario de registro de usuarios turistas, en el cual el turista selecciona su nacionalidad o país de residencia.

Este componente no es oficial de react native, pero en la comunidad de este lenguaje se ha desarrollado esta librería OpenSource, por lo cual se añade al proyecto de la siguiente manera:

```
yarn add react-native-country-picker-modal
```

El selector de países proporciona un modal que permite al usuario seleccionar un país de una lista. Muestra por cada país; la bandera de manera gráfica, código de número de teléfono y el nombre del país según el idioma en que se configura por medio de las variadas propiedades de las que dispone.

```
<CountryPicker
  {...{
    countryCode, translation:'spa', withFilter, withFlag,
    withCountryNameButton, withAlphaFilter, withCallingCode,
    withEmoji, onSelect,
  } }
/>
```

Finalmente, en la siguiente imagen se observa gráficamente cada selector acorde al componente nativo de cada sistema operativo.

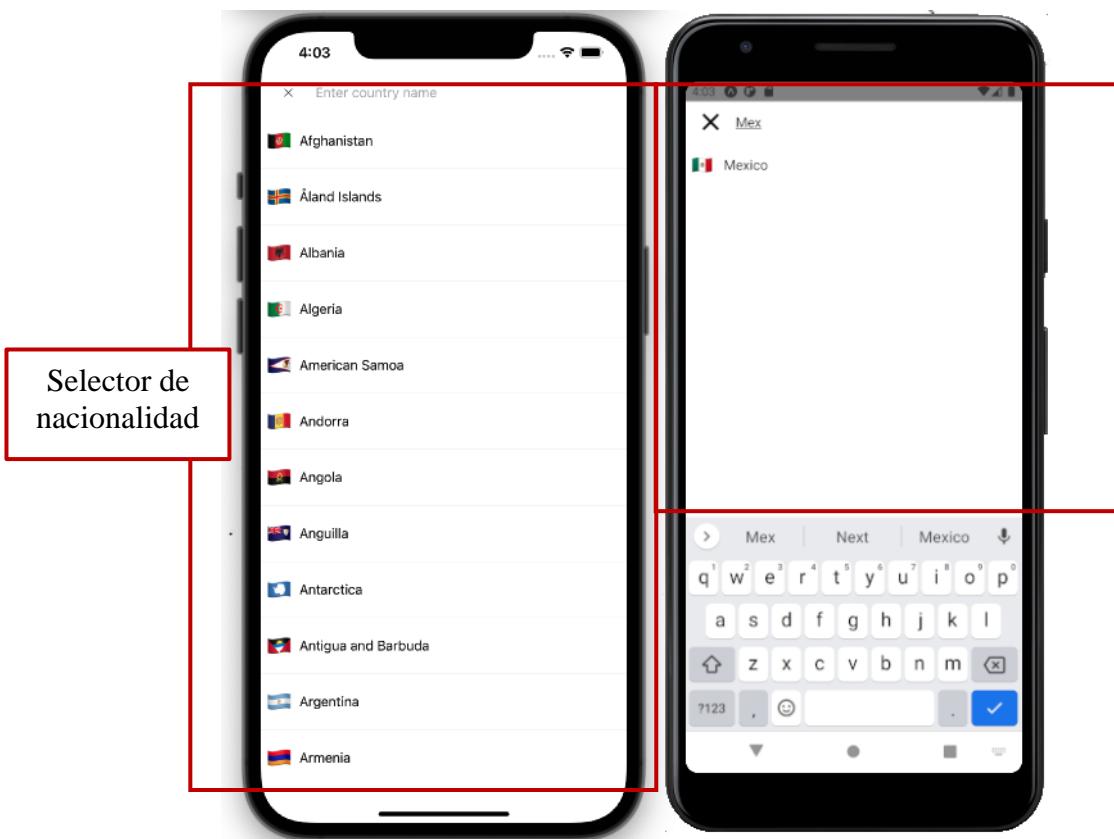


Figura 72. Componente Selector de nacionalidad en IOS y And

5.1.4.4. Calendario

El componente calendario es necesario para la selección de fechas de CheckIn, CheckOut, día de reservación, por lo que se reutiliza al menos 3 veces, se utiliza un componente ya existente que se adapta a las necesidades de la aplicación móvil, como lo es establecer fecha mínima y máxima de selección, estilos de amrcado de fechas seleccionadas, etc.

Instalar las siguientes librerías:

```
yarn add moment  
yarn add react-native-calendars
```

<Calendar /> es un componente muy versátil que se agrega dentro del retorno de la función por defecto CalendarSelection() en el archivo de mismo nombre que la función, es un componente ya desarrollado por la comunidad, requiere algunos argumentos básicos para su operación, como lo es la fecha actual (muy por defecto se refiere a la fecha de hoy), fecha mínima y máxima de selección, estilos, función que aplica al presionar el elemento del calendario, estilos en días marcados, etc.

```
<Calendar  
    current={null}  
    minDate={moment().format("YYYY-MM-DD")}  
    maxDate={moment().add(15, 'days').format("YYYY-MM-DD")}  
    style={styles.calendar}  
    onDayPress={onDayPress}  
    markedDates={  
        [selected]: {  
            selected: true,  
            disableTouchEvent: true,  
            selectedColor: '#EAA11B',  
            selectedTextColor: 'white',  
        }  
    }  
/>
```

La siguiente imagen muestra este componente funcionando en cada sistema operativo móvil, para este caso no se nota una gran diferencia en el procesamiento grafico nativo.

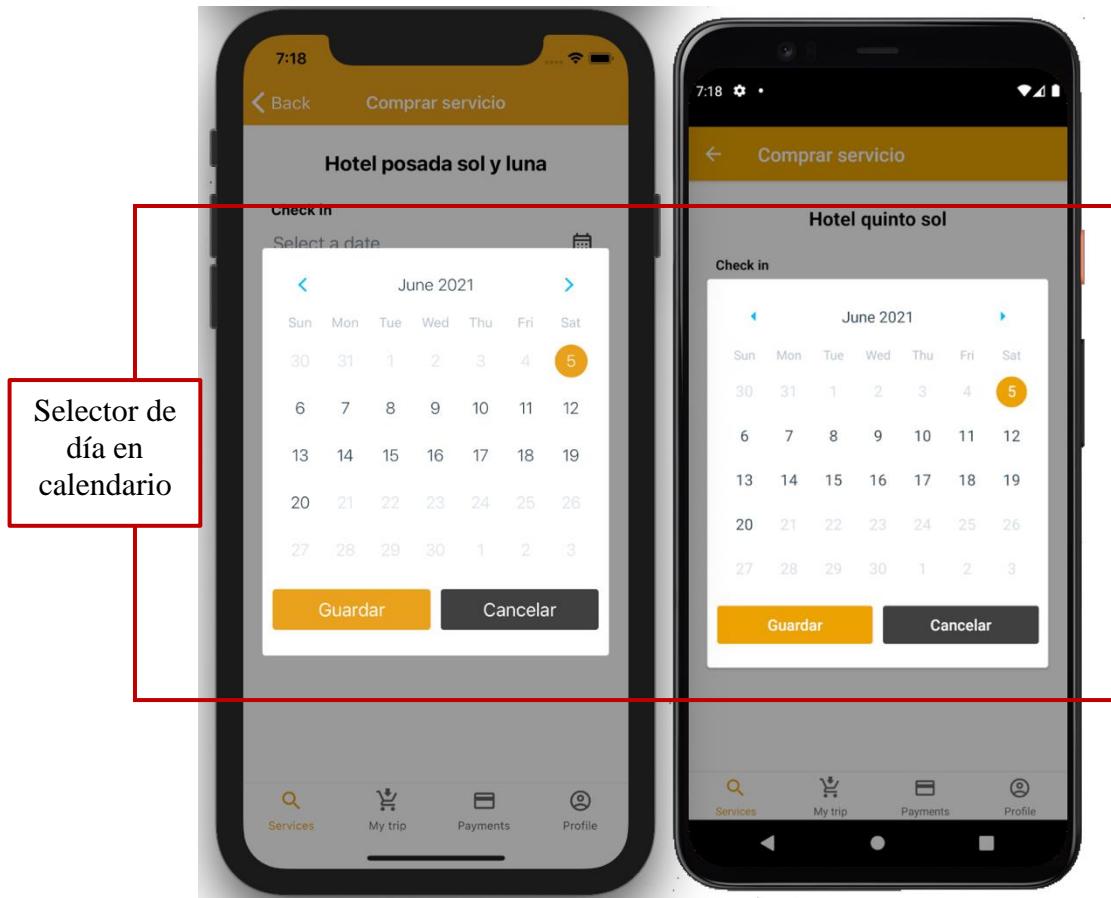


Figura 73. Componente Calendario

5.1.4.5. Mapa de geolocalización

Este componente se ocupa en 3 momentos dentro de la aplicación móvil, al registrar el usuario vendedor, consultar el perfil de usuario vendedor y consultar servicio desde el usuario turista,

Instalar las siguientes librerías:

```
yarn add react-native-maps
yarn add react-native-open-maps
```

Este componente requiere obtener coordenadas geográficas (latitud y longitud) para poner ubicar en el plano el punto de la ubicación actual, en el caso de hacer clic sobre el mapa se abre la aplicación de mapas nativa al sistema operativo (Google maps para Android y Maps para IOS), misma que puede ser la región inicial que se muestra al abrir el componente.

```
export default function Map(props) {
  const {location, height, name} = props

  const openAppMap = () => {
    openMap({
      latitude: location.latitude,
      longitude: location.longitude,
      zoom: 19,
      query: name
    });
  };
}
```

```

        return (
          <MapView
            style={{height:height, width: "100%", marginTop: 15}}
            initialRegion={location}
            onPress={openAppMap}
          >
            <MapView.Marker
              coordinate={{
                latitude: location.latitude,
                longitude: location.longitude,
              }}
            />
          )
        }
    }
  }
}

```

Se observa la siguiente imagen el componente de Mapa adecuado para edición (a la izquierda) y para consulta (A la derecha).

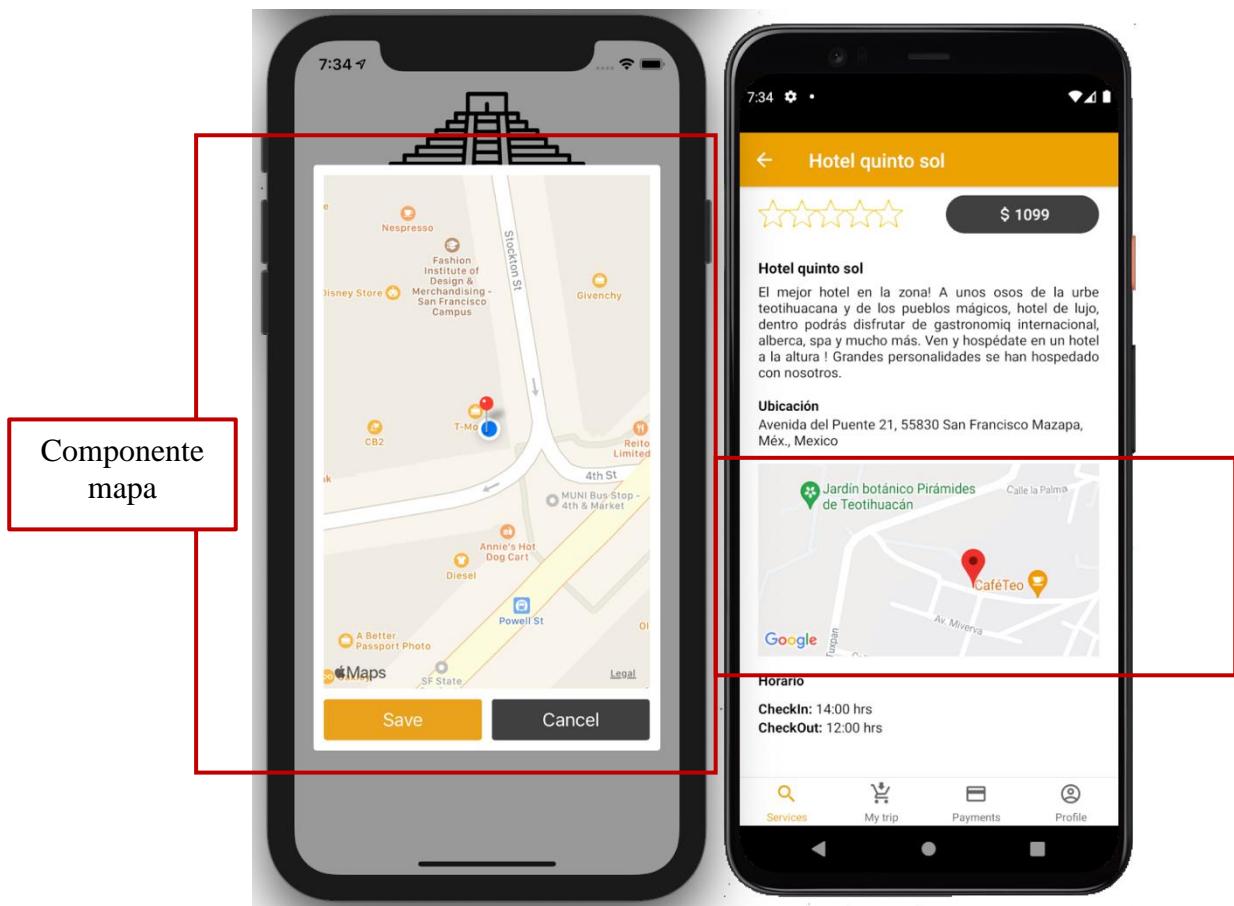


Figura 74. Componente Mapa

5.1.4.6. Carousel de imágenes

Componente Swiper / carrusel para React Native con vistas previas, diseños múltiples, imágenes de paralaje, manejo eficaz de una gran cantidad de elementos y otros más. Compatible con iOS y Android. Este componente será usado en la aplicación vendedor para añadir y editar las imágenes de cada servicio, también será embebido en la aplicación del turista para mostrar las imágenes que

el vendedor dio de alta sobre su servicio, mediante deslizamiento en la pantalla touch del dispositivo es posible visualizar todas y cada una de las imágenes.

Instalar las siguientes librerías:

```
yarn add react-native-snap-carousel  
yarn add lodash
```

Se crea un archivo JavaScript llamado Carousel.js, en el cual se exporta una función por defecto con el nombre de CarouselImages(), donde se cachan los argumentos que recibirá el componente, estos son el arreglo de imágenes a mostrar en el carousel, el alto y ancho del carousel, el identificador del documento referente a firebase de donde se obtendrán los datos, etc.

```
export default function CarouselImages(props) {  
    const {  
        arrayImages,  
        height,  
        width,  
        id_doc,  
        setReloadInfo,  
        setIsLoading,  
        toastRef  
    } = props;  
  
    const [arrayImagenes, setArrayImagenes] = useState([])  
const wildcard_image =  
"https://firebasestorage.googleapis.com/v0/b/tteotiapp.appspot.com/o/services%  
2Fupload_image.png?alt=media&token=6f3884ba-8545-469b-a9af-70b6191790f9"
```

Dentro del retorno de esta función, se anida una vista, dentro de la vista se agrega el componente <Carousel />.

```
<Carousel  
    layout={"default"}  
    data={arrayImagenes}  
    sliderWidth={width}  
    itemWidth={width}  
    renderItem={renderItem}  
/>
```

Este componente tiene una serie de propiedades, donde se le manda el arreglo de imágenes, el alto y ancho, y el ítem genérico a procesar por cada imagen del arreglo, el cual se muestra a continuación.

```
const renderItem = ({ item }) => {  
    return (  
        <TouchableOpacity onPress={() =>  
            removeImage(item, arrayImagenes, id_doc, setReloadInfo)}>  
            <View>  
                <Image style={{ width, height }} source={{ uri: item }} />  
            </View>  
        </TouchableOpacity>  
    )  
}
```

A cada ítem que mostrará el carousel será un sub componente renderItem(), el cual esta constituido y anidado por un <TouchableOpacity />, <View /> respectivamente, donde se coloca una imagen <Image /> y se le pasan los argumentos necesarios, es decir tamaño y URI de la imagen.

Ahora se muestra de manera visual la implementación de un Carousel en ambos sistemas operativos móviles:

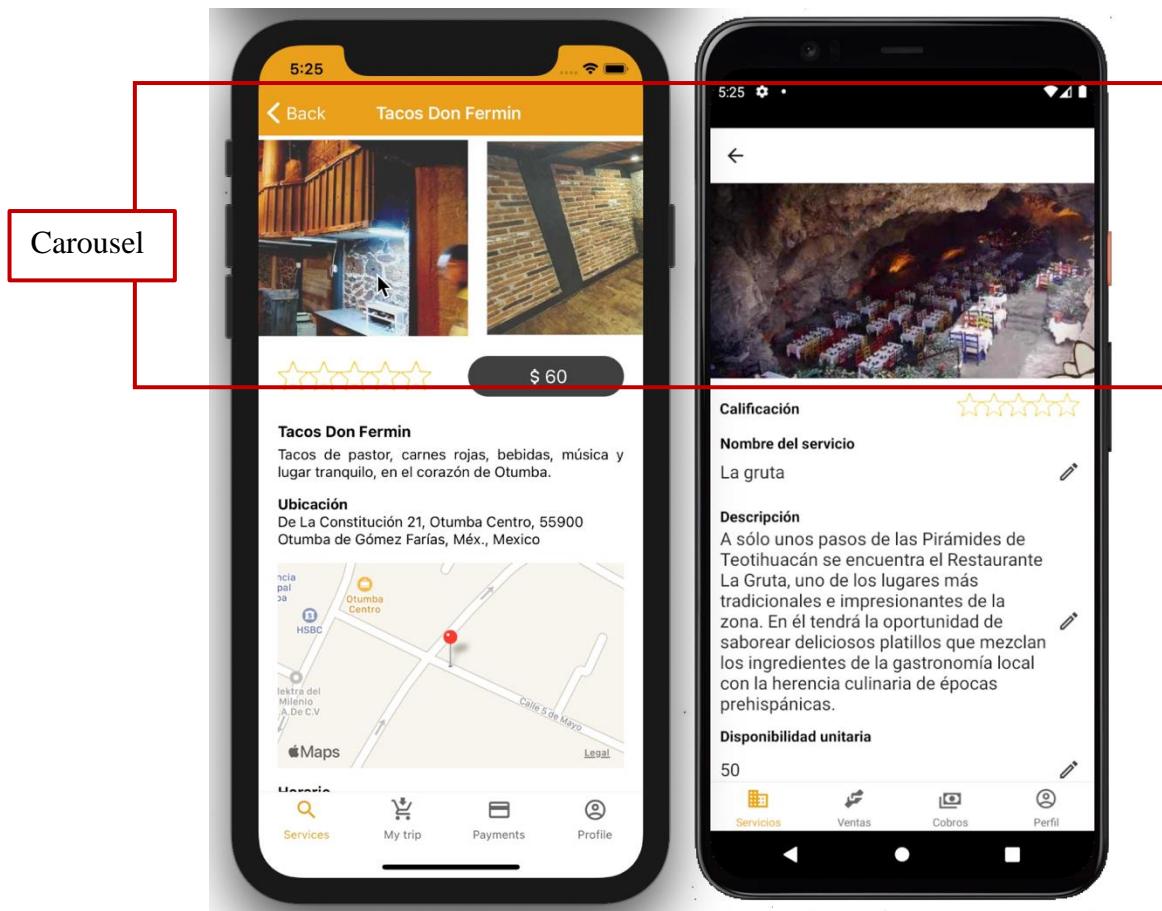


Figura 75. Componente Carousel en IOS y Android

5.1.4.7. Loading

Probablemente este sea el componente que mas se reutiliza en la aplicación y de la misma manera el más simple, ya que cada transacción que se ejecuta en firebase, se muestra un mensaje según sea el caso, por lo que, al iniciar sesión, actualizar cualquier dato o comprar servicios, se mostrara un <Loading />

Instalar la siguiente librería:

```
yarn add react-native-animated-spinkit
```

Este componente se construye sobre el comonente <Overlay />, este componente es un tipo de modal sin botones, dentro se incluye una vista, dentro de la vista se incluye un <Bouce /> el cual

es un componente de animación y además un <Text /> que pinta en pantalla el argumento texto que recibe la función por defecto.

```
export default function Loading(props) {
  const {isVisible, text} = props;

  return (
    <Overlay isVisible={isVisible} overlayStyle={styles.overlay}>
      <View style={styles.container}>
        <Bounce size={180} color="#EAA11B"/>
        {text && <Text style={styles.txt_loading}>{text}</Text>}
      </View>
    </Overlay>
  )
}
```

La siguiente figura muestra el componente <Loading /> en acción, recibe los textos “¡Loading your profile!” y “¡Loading configuration!” respectivamente.

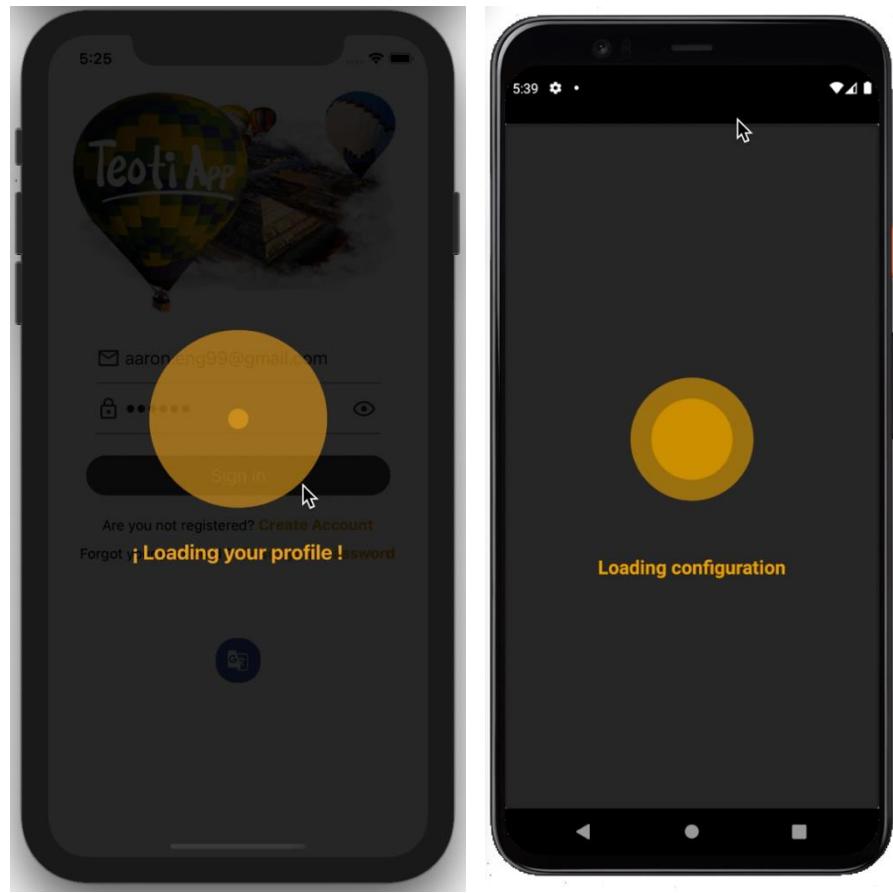


Figura 76. Componente Loading en IOS y Android

5.1.4.8. Modal

Este componente tiene la funcionalidad de contener casi cualquier otro componente, de hecho, todos los componentes anteriores se presentan mediante un <Modal />. Este componente se adapta al tamaño del contenido que recibe, es decir, tendrá el tamaño (alto y ancho) que se especifique en el componente interno, requiere de 3 argumentos, el estado completo de visibilidad y el hijo o componentes a mostrar.

```
export default function Modal(props) {
  const { isVisible, setIsVisible, children } = props;
  const closeModal = () => setIsVisible(false);

  return (
    <Overlay
      isVisible={isVisible}
      windowBackgroundColor="rgba(0,0,0,0.5)"
      overlayBackgroundColor="transparent"
      overlayStyle={styles.overlay}
      onBackdropPress={closeModal}
    >
      {children}
    </Overlay>
  );
}
```

5.1.5. Librerías, paquetes y dependencias utilizadas

```
"@react-native-community/masked-view": "0.1.10",
"@react-native-community/picker": "^1.8.1",
"@react-native-picker/picker": "~1.9.2",
"@react-navigation/bottom-tabs": "~5.2.6",
"@react-navigation/drawer": "~5.9.0",
"@react-navigation/native": "~5.1.5",
"@react-navigation/stack": "~5.2.10",
"@sorakrisc/react-native-datetimepicker": "^1.0.5",
"add": "^2.0.6",
"base-64": "^1.0.0",
"expo": "~40.0.0",
"expo-image-picker": "~9.2.0",
"expo-location": "~10.0.0",
"expo-permissions": "~10.0.0",
"expo-status-bar": "~1.0.3",
"firebase": "~7.9.0",
"lodash": "^4.17.21",
"moment": "^2.29.1",
"native-base": "^2.15.2",
"random-uuid-v4": "~0.0.9",
"react": "16.13.1",
"react-dom": "16.13.1",
"react-native": "^0.63.4",
"react-native-24h-timepicker": "^1.1.0",
"react-native-animated-spinkit": "^1.5.2",
"react-native-country-picker-modal": "^2.0.0",
"react-native-dropdown-picker": "^4.0.2",
"react-native-easy-toast": "^2.0.0",
"react-native-elements": "^3.2.0",
"react-native-geocoding": "^0.5.0",
"react-native-gesture-handler": "~1.8.0",
```

```
"react-native-keyboard-aware-scroll-view": "~0.9.1",
"react-native-maps": "~0.27.1",
"react-native-material-textfield": "^0.16.1",
"react-native-modal": "^11.7.0",
"react-native-modal-datetime-picker": "^9.1.0",
"react-native-modal-selector": "^2.0.3",
"react-native-paper": "^4.7.2",
"react-native-picker-module": "^2.0.4",
"react-native-picker-select": "^8.0.4",
"react-native-raw-bottom-sheet": "^2.2.0",
"react-native-reanimated": "~1.13.0",
"react-native-safe-area-context": "3.1.9",
"react-native-screens": "~2.15.2",
"react-native-simple-time-picker": "^1.3.0",
"react-native-web": "~0.13.12",
"react-time-picker": "^4.2.1",
"yarn": "^1.22.10"
```

5.2. Submódulo de lógica de negocio

5.2.1. Inicio de sesión

En el formulario de inicio de sesión vamos a utilizar algunos elementos como botones, campos de texto, iconos, etc. Primeramente, se importan todas las librerías necesarias, posterior a ello se hace el retorno de función que incluye todos los elementos, todos se incluyen en un elemento contenedor el cual es una vista.

El siguiente fragmento de código codifica la entrada de texto para capturar el correo electrónico del usuario a autenticarse, se declara el texto en segundo plano de ayuda, se deshabilita que la primera letra sea mayúscula, se le pasa el atributo estilo, se agrega un ícono a la izquierda el cual con base a material community indicará un ícono de buzón.

```
<Input
    placeholder = "Email"
    autoCapitalize='none'
    containerStyle = {styles.input}
    leftIcon = {{
        type: "material-community",
        name: "email-outline",
        color: "#404040",
    }}
/>
```

A la entrada de texto anterior le sigue otra más, la cual está destinada para capturar la contraseña:

```
<Input
    placeholder = "Password"
    containerStyle = {styles.input}
    rightIcon = {{
        type: "material-community",
        name: showPassword ? "eye-off-outline": "eye-outline",
        color: "#404040",
        onPress: () => setShowPassword(!showPassword)
    }}
    leftIcon = {{
        type: "material-community",
        name: "lock-outline",
        color: "#404040",
    }}
/>
```

Se continua con el botón para entrar a la aplicación móvil:

```
<Button
    title = "Sign in"
    containerStyle = {styles.btn_sign} // out
    buttonStyle = {{backgroundColor: "#404040"}}
    onPress = {() => login()}
/>
```

Después se agregan los textos de “Crear cuenta y recuperar contraseña”:

```
<Text style={styles.txt_create_account}>
    Are you not registered?
    <Text
        style={styles.txt_account}
```

```

        onPress={() => navigation.navigate("registerselection")}>
        {" " }Create Account
    </Text>
</Text>

<Text style={styles.txt_reset_password}>
    Forgot your password ?
    <Text
        style={styles.txt_account}
        onPress={() => navigation.navigate("resetpassword")}>
        {" " } Reset your password
    </Text>
</Text>

```

Y finalmente, el botón de cambio de idioma, inglés a español y español a inglés:

```

<TouchableOpacity style={styles.btn_login_social}>
    <Icon
        size={24}
        type="material-community"
        name = "google-translate"
        color = "#FFFFFF"
        background = "transparent"
    />
</TouchableOpacity>

```

El resultado de la codificación anterior es la siguiente pantalla, donde se muestra la ejecución de la pantalla de login en dispositivo IOS y Android, esta pantalla es la implementación del diseño Figura 13. Mockup Inicio de sesión.

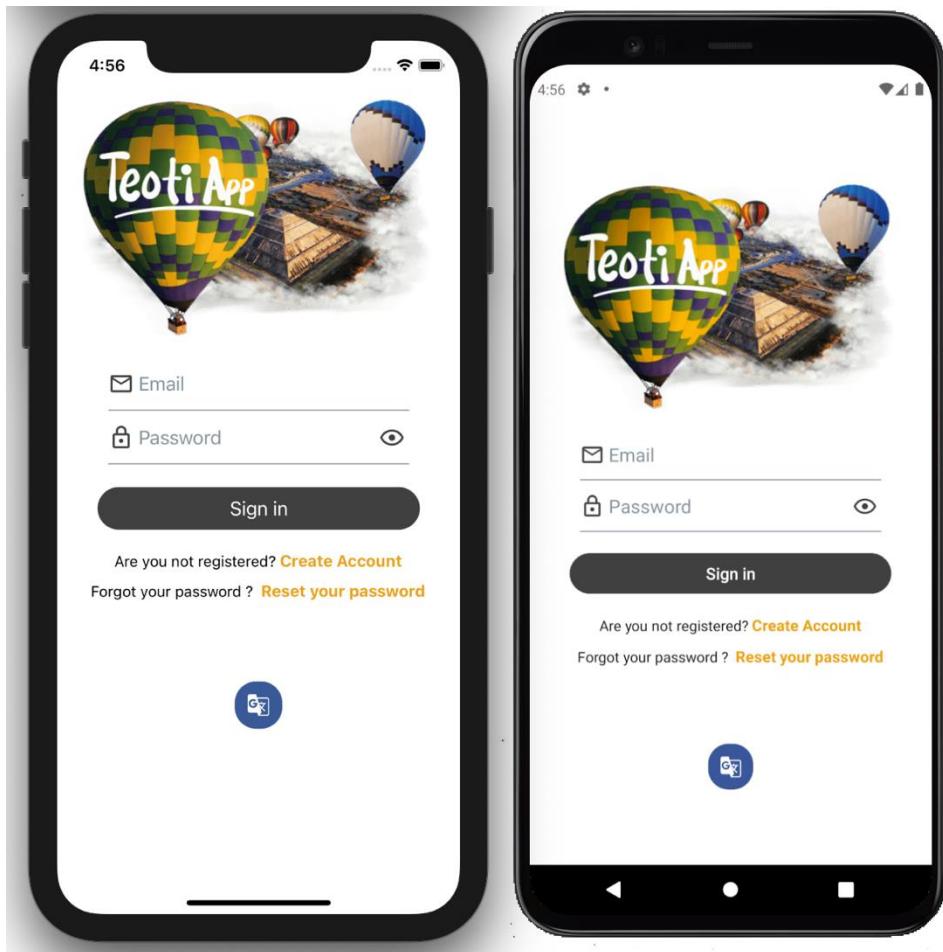


Figura 77. Pantalla de inicio de sesión en IOS y Android

Hasta este punto se entiende el funcionamiento y lógica aplicada en la codificación que se usara a lo largo de la aplicación, a partir de este punto se muestra la codificación las principales funcionalidades.

5.2.2. Selección de tipo de registro de usuario

En este punto de la aplicación puede ser accedida desde el usuario turista o vendedor, ya que no han iniciado sesión y se intenta crear una cuenta de usuario, por lo que se hace un filtro previo al formulario de creación de usuario, esto es, se le pregunta al usuario el tipo de cuenta a crear.

En el siguiente fragmento de código se observa la distribución de los componentes que se ocuparon para la implementación de esta pantalla, donde se incluye una imagen y un botón para registro de vendedor, y otro para de estos para el registro de turista:

```
export default function RegisterSelectionForm() {
  const navigation = useNavigation()
  const { t, i18n } = useTranslation()

  return (
    <View style={styles.container}>
      <Text
        style={styles.txt_header}>{t('login.title_who_are_you_lbl')}
```

El resultado de la codificación anterior es la siguiente pantalla, donde se muestra la ejecución de la pantalla de selección de tipo de registro de usuario en dispositivo IOS y Android, esta pantalla es la implementación del diseño Figura 14. Mockup Selección de Registro de usuario.

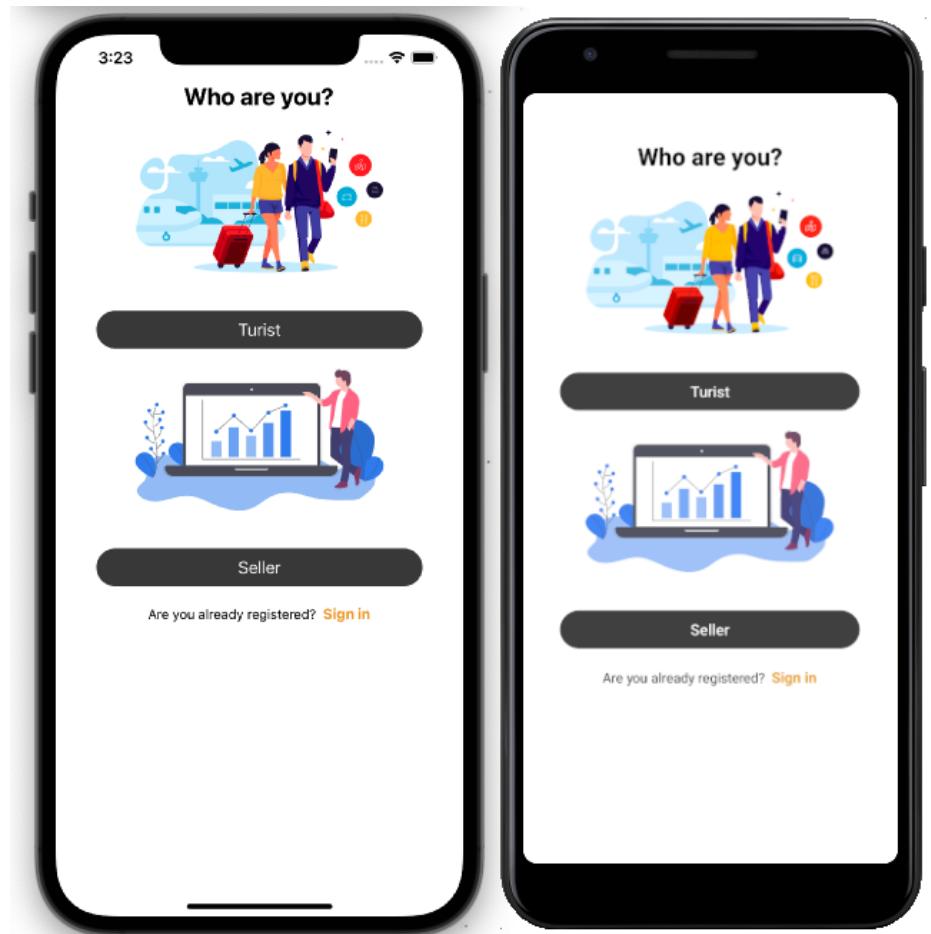


Figura 78. Pantalla de selección de registro de tipo de usuario en IOS y Android

5.2.3. Registro de usuarios

El proceso de registro de usuarios turistas y vendedores es distinto entre si, ya que los datos que se almacenan son completamente diferentes, en caso de turistas se requiere validar todos y cada uno de los campos del formulario (Ver Figura 15. Mockup Registro de usuario Turista), de todo ser correcto, se realiza el registro de usuario en firebase:

```
// Funcion para registrar usuario de tipo turista
const addTurist = async () => {
    if(isEmpty(name) || isEmpty(email) || isEmpty(date_birth) ||
isEmpty(gender) ||
    isEmpty(land) || isEmpty(password) || isEmpty(password_confirm)) { // campos vacios
        toastRef.current.show("All the fields are required")
    } else if(!validateEmail(email)) {
        toastRef.current.show("Email is not valid")
    } else if(password !== password_confirm){ // contraseñas iguales
        toastRef.current.show("Your passwords do not match")
    } else if(size(password) < 6 ){
        toastRef.current.show("Your password must be at least 6 characters long")
    } else if(!checkTerms){
        toastRef.current.show("You have to accept terms and conditions")
    }else {
        setShowLoading(true);

        const prueba = await firebase
            .auth()
            .createUserWithEmailAndPassword(email, password)
            .then(response => {
                setShowLoading(false);
                addRegister("users", response.user.uid, {
                    name,
                    email,
                    date_birth,
                    land,
                    gender,
                    creation_date : new Date(),
                    role: "turist"
                });
            })
            .catch(() => {
                setShowLoading(false);
                toastRef.current.show("Error, try with another email")
            })
    }
}
```

En caso de ser un usuario vendedor, igual validaremos cada campo del formulario (Ver Figura 17. Mockup Registro de usuario Vendedor), y de ser correcto se aplica el registro de usuario de este tipo.

```
// Funcion para registrar usuario vendedor
const addSeller = async () => {
    if(!name || !email || !rfc || !address || !password ||
!passwordConfirm){
```

```

        toastRef.current.show("All the fields are required")
    }else if(!validateEmail(email)) {
        toastRef.current.show("Email is not valid")
    }else if(password !== passwordConfirm){ // contraseñas iguales
        toastRef.current.show("Your passwords do not match")
    } else if(size(password) < 6 ){
        toastRef.current.show("Your password must be at least 6 characters
long")
    } else if(size(phone) < 10) {
        toastRef.current.show("Your phone number is not valid")
    } else if(!checkTerms){
        toastRef.current.show("You have to accept terms and conditions")
    }else{ // Se puede registrar
        setShowLoading(true);

        const prueba = await firebase
            .auth()
            .createUserWithEmailAndPassword(email, password)
            .then(response => {
                setShowLoading(false);
                addRegister("users", response.user.uid, {
                    name,
                    email,
                    phone,
                    rfc,
                    creation_date : new Date(),
                    locationSeller,
                    address,
                    role: "seller"
                });
            })
            .catch(() => {
                setShowLoading(false);
                toastRef.current.show("Error, try with another email")
            })
    }
}

```

El resultado de la codificación anterior son las siguientes pantallas, donde se muestra la ejecución de la pantalla de registro de usuario turista y vendedor en sistemas operativos IOS y Android respectivamente, esta pantalla es la implementación del diseño de la Figura 15. Mockup Registro de usuario Turista y Figura 17. Mockup Registro de usuario Vendedor.

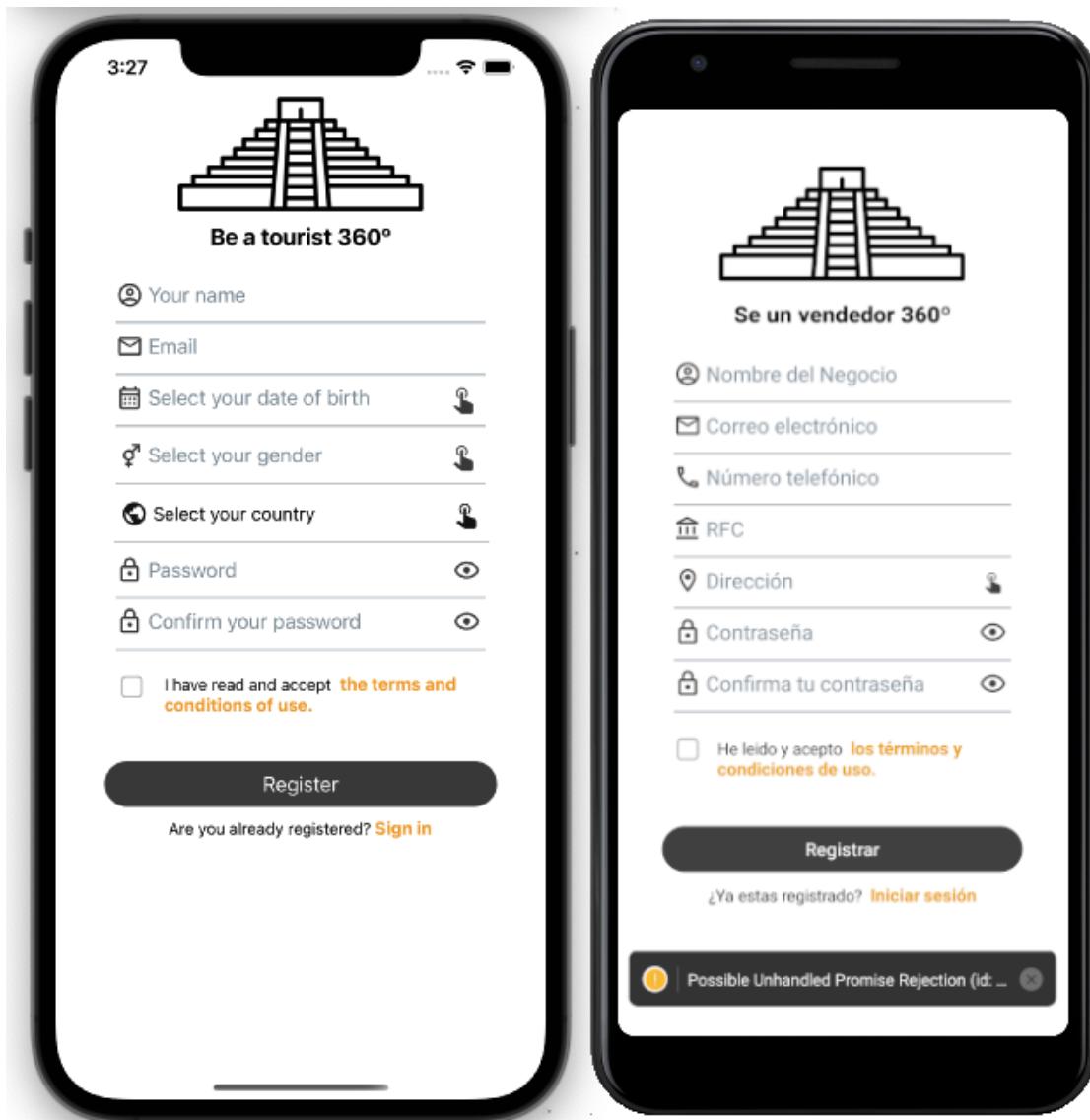


Figura 79. Pantalla de registro de usuario turista y vendedor en IOS y Android

5.2.4. Obtención de usuario

Realizar la obtención de datos usuarios es una función altamente eficiente, ya que en cada transacción que se realiza es necesario operar respecto a un usuario y sus datos, primeramente, se muestra la obtención del usuario:

```
export const getUser = () => {
  return firebase.auth().currentUser;
};
```

Posterior obtener datos de usuario basado en el correo electrónico:

```
export const getUserData = async () => {
  let user = {}

  await db
    .collection("users")
    .where("email", "==", getUser().email)
    .get()
    .then((response) => {
      response.forEach((doc) => {
        const prueba = doc;
        user = prueba.data()
      });
    })
    .catch((err) => {
      console.log("Error while trying to get user by email")
    });
  return user;
}
```

De igual forma es posible obtener el usuario dado su id:

```
export const getCurrentUserData = async () => {
  let user_data = {}

  await db
    .collection("users")
    .doc(getUser().uid)
    .get()
    .then((response) => {
      user_data = response.data()
    })
    .catch((err) => {
      console.log("Error while trying to get user by uid")
    });
  return user_data
}
```

5.2.5. Actualización de información de usuario

La implementación de esta funcionalidad se hace de manera escalable, dado que hay múltiples campos que pueden ser actualizados, entonces se crea una función la cual verifica si hay cambios y si los cambios son válidos, en caso de cumplir, se requiere la clave del atributo a modificar y el nuevo valor, entonces se realiza la actualización en firebase:

```

if(!newData){
    setError(placeholder + " no puede estar en blanco")
} else if(data === newData){
    setError(placeholder + " no se ha modificado")
} else {
    setIsLoading(true)

    firebase
        .firestore()
        .collection("users")
        .doc(id_doc)
        .update({
            [field] : newData,
        })
        .then(() => {
            console.log(placeholder + " actualizado correctamente")
            setShowModal(false)
            setReloadInfo(true)
            toastRef.current.show("Perfil actualizado correctamente");
        })
        .catch(() => {
            setError("Error mientras al actualizar " + placeholder)
        })
        .finally(() => {
            setIsLoading(false)
        })
}
}

```

5.2.6. Selección de tipo de registro de nuevo servicio

El usuario vendedor puede registrar una amplia variedad de servicios, por ejemplo: hoteles, moteles, restaurantes, antros, clubs, cafeterías, deportes extremos, transporte, etc. De manera general estos servicios se pueden clasificar en 3 grupos: Hospedaje, Consumo y Otro tipo de servicios. Explicado lo anterior, cuando un vendedor requiere registrar un servicio pasará primero por una pantalla filtro de selección de tipo de servicio.

En el siguiente fragmento de código se observa la distribución de los componentes que se ocuparon para la implementación de esta pantalla:

```

export default function SelectNewService(props) {
    const { navigation } = props

    return (
        <ScrollView style={styles.view_container}>
            <Image
                source = {require("../../../../../assets/hotel.jpeg")}
                style = {styles.img_logo}
            />

            <Button
                title = "Hospedaje"
                containerStyle = {styles.btn_sign} // out
                buttonStyle = {{backgroundColor: "#404040"}}
                onPress = {() => {
                    navigation.navigate("newservice", {type_Service: "hotel"})
                }}
            />
    )
}

```

```

        <Image
            source = {require("../../../../../assets/restaurant.jpeg") }
            style = {styles.img_logo}
        />

        <Button
            title = "Restaurant, bar and antro"
            containerStyle = {styles.btn_sign} // out
            buttonStyle = {{backgroundColor: "#404040"}}
            onPress = { () => {
                navigation.navigate("newservice", {type_Service:
"restaurant"})
            } }
        />

        <Image
            source = {require("../../../../../assets/Other-services.jpeg") }
            style = {styles.img_logo}
        />

        <Button
            title = "Other services"
            containerStyle = {styles.btn_sign} // out
            buttonStyle = {{backgroundColor: "#404040"}}
            onPress = { () => {
                navigation.navigate("newservice", {type_Service: "other"})
            } }
        />
    </ScrollView>
)
}

```

El resultado de la codificación anterior es la siguiente pantalla, donde se muestra la ejecución de la pantalla de selección de registro de tipo de servicio en sistemas operativos IOS y Android respectivamente, esta pantalla es la implementación del diseño de la Figura 29. MockUp Selección de tipo de nuevo servicio.

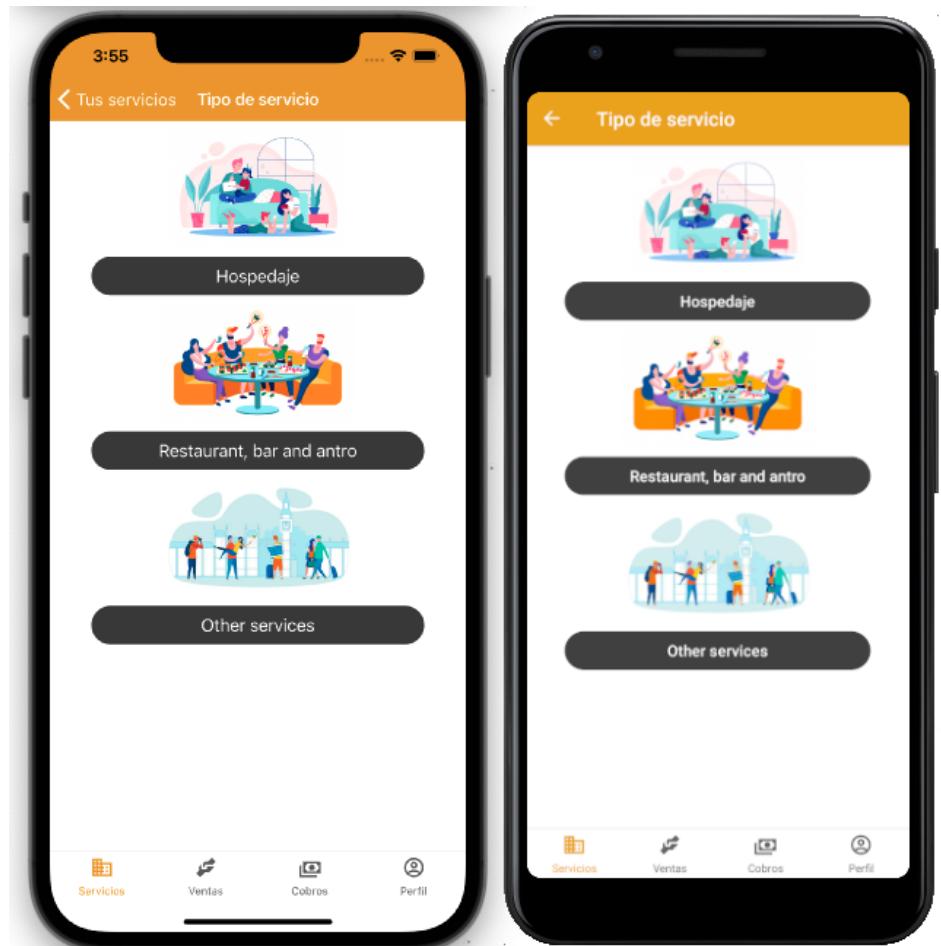


Figura 80. Pantalla de selección de tipo de registro de servicio en IOS y Android

5.2.7. Registro de servicios

Es posible registrar tres tipos de servicio: Hospedaje, consumo y otros, de igual manera que los demás registros, se tiene que validar los campos de cada formulario (Ver Figura 32. Mockup Añadir otro servicio), de todo salir correctamente se continua con el registro en firebase, donde primero se almacenan las imágenes en el storage, posterior a ello se hace una consulta para obtener las URL de cada imagen y poder ligarlas con el servicio a registrar como parte de sus atributos de dicho servicio.

```
const addService = () => {
    // Validaciones
    if(!Name || !typeOfServices || !Price || !TimeInService ||
!Availability || !Description) {
        toastRef.current.show("Todos los campos son obligatorios.")
    } else if(size(ImageSelected) === 0){
        toastRef.current.show("Por favor selecciona al menos una iamgen.")
    } else if (OpenWeekDays && !TimeOpenWeekday && !TimeCloseWeekday) {
        toastRef.current.show("Por favor compelta los tiempos
apertura/cierre de días entre semana.")
    } else if (OpenWeekend && !TimeOpenWeekend && !TimeCloseWeekend) {
        toastRef.current.show("Por favor compelta los tiempos
apertura/cierre de días de fin de semana.")
    } else {
```

```

        setIsLoading(true);
        UploadImageStorage().then((response) => {
            db.collection("services")
                .add({
                    name: Name,
                    type: typeOfServices,
                    price: Price,
                    timeInService: TimeInService,
                    availability: Availability,
                    description: Description,
                    isOpenOnWeekDays: OpenWeekDays,
                    isOpenOnWeekends: OpenWeekend,
                    timeOpenWeekdays: TimeOpenWeekday,
                    timeCloseWeekDays: TimeCloseWeekday,
                    timeOpenWeekends: TimeOpenWeekend,
                    timeCloseWeekend: TimeCloseWeekend,
                    location: UserData.locationSeller,
                    creationAt: new Date(),
                    images: response,
                    rating: 0,
                    createBy: firebase.auth().currentUser.uid
                })
                .then(() => {
                    navigation.navigate("servicelist")
                    setIsLoading(false)
                })
                .catch(() => {
                    setIsLoading(false)
                    toastRef.current.show("Error while upload service, try
later")
                })
        })
    }
}

```

5.2.8. Consulta de usuarios

La consulta de servicios que coinciden con una búsqueda es relativamente sencilla, pero en este caso es necesario instalar una librería que simplifica en gran medida la consulta, es posible realizar consultas como si fuese lenguaje SQL.

`yarn add firesql`

Después de importar la librería descargada, procedemos a realizar la consulta:

```

const query = `SELECT * FROM services WHERE name LIKE '${search}%' OR type
CONTAINS '${search}'`
const query_all = `SELECT * FROM services ORDER BY rating`

useEffect(() => {
    if(search){
        fireSQL.query(query)
            .then((response) => {
                setServices(response)
            })
    }else{
        fireSQL.query(query_all)
            .then((response) => {
                setServices(response)
            })
    }
})

```

```
        }
    , [search])
```

5.2.9. Consulta de servicios previamente registrados por el turista

El usuario turista tiene la posibilidad de listar los servicios previamente registrados y visualizar a detalle servicio por servicio, el siguiente fragmento de código muestra la implementación de listar servicios:

```
export default function Services(props) {
    const { navigation } = props
    const [user, setUser] = useState(null)
    const [search, setSearch] = useState(null)
    const [services, setServices] = useState([])
    const [isLoading, setIsLoading] = useState(false)
    const { t, i18n } = useTranslation()

    const query = `SELECT * FROM services WHERE name LIKE '${search}%' OR type
CONTAINS '${search}'`
    const query_all = `SELECT * FROM services ORDER BY rating`

    useEffect(() => {
        firebase.auth().onAuthStateChanged((userInfo) => {
            setUser(userInfo)
        })
    , [])

    useFocusEffect(
        useCallback(() => {
            if(search){
                fireSQL.query(query)
                .then((response) => {
                    setServices(response)
                })
            }else{
                fireSQL.query(query_all)
                .then((response) => {
                    setServices(response)
                })
            }
        })
    , [search])
}

return (
    <View style={styles.view_container}>
        <SearchBar
            placeholder= {t('services.search_service')}
            onChangeText={(e) => {
                setSearch(e)
            }}
            value={search}
            containerStyle={styles.seachBar_container}
            inputStyle={styles.seachBar_Inside}
            inputContainerStyle={{backgroundColor: 'white',
borderWidth:0}}
        />
```

```

        {services.length === 0 ? (
            <NotFoundServices />
        ) : (
            <FlatList
                data={services}
                renderItem={(service) => (
                    <ServiceItem
                        service={service}
                        navigation={navigation}
                    />
                )}
                keyExtractor={(item, index) => index.toString()}
            />
        ) }
    </View>
)
}

```

Donde cada ítem de la lista se codifica de la siguiente manera:

```

function ServiceItem(props) {
    const {service, navigation} = props
    const { id, images, name, price, description, availability, timeInService,
rating } = service.item
    const image_service = images[0]

    const goService = () => {
        navigation.navigate("seeservice", {id, name})
    }

    return (
        <TouchableOpacity onPress={goService}>
            <View style={styles.service_item_container}>
                <View style={styles.service_image_container}>
                    <Image
                        resizeMode="cover"
                        PlaceholderContent={<ActivityIndicator color="#123456">
/>}
                        source={
                            image_service
                            ? { uri : image_service }
                            : require("../../../../../assets/no-image.png")
                        }
                        style={styles.img_service}
                    />
                </View>

                <View style={styles.data_container}>
                    <Text style={styles.service_name}>{ name }</Text>
                    <Text>Tiempo de servicio: { timeInService }</Text>
                    <Text>Disponibilidad unitaria: { availability }</Text>
                    <Text>Precio: $ { price }</Text>
                    <Text>Rating: { rating.toFixed(2) }</Text>
                </View>
            </View>
        </TouchableOpacity>
    )
}

```

La codificación de consulta el servicio a detalle se aprecia en el siguiente fragmento de código:

```
export default function SeeService(props) {
  const {navigation, route} = props
  const {id, name} = route.params
  const [service, setService] = useState(null)
  const [rating, setRating] = useState(0)
  const { t } = useTranslation()

  useLayoutEffect(() => {
    navigation.setOptions({
      title: name === '' ? "" : name,
    });
  }, [navigation, name])

  useEffect(() => {
    db.collection("services")
      .doc(id)
      .get()
      .then((response) => {
        const data = response.data()
        data.id = response.id
        setService(data)
        setRating(data.rating)
      })
  }, [])

  if(!service) {
    return <Loading isVisible={true} text={t('services.loading')} />
  }

  return (
    <ScrollView vertical style={styles.view_body} >
      <CarouselRead
        arrayImages={service.images}
        height={200}
        width={screen_width}
      />

      <View style={styles.container_inside}>
        <View style={{flex: 1, flexDirection: "row", justifyContent: "space-between"} }>
          <Rating
            type='star'
            style={styles.rating}
            imageSize={30}
            readonly
            startingValue={parseFloat(rating)} />
          <Button
            title={service.price}
            containerStyle={styles.viewBtnBuy}
            buttonStyle={styles.viewBtnStyleBuy}
            icon={
              <Icon
                type="material-community"
                name="currency-usd"
                size={20}
                color="white"
              />
            }
          />
        </View>
      </View>
    </ScrollView>
  )
}
```

```

                />
            }
            onPress={() =>
                navigation.navigate("buyservice", {service})
            }
        />
    </View>

    <TitleService
        name={name}
        description={service.description}
        rating={rating}
    />

    <ServiceMap
        location={service.location}
        name={name}
        t={t}
    />

    <Text style={styles.schedule}>{t('services.schedule')}</Text>

    {service &&
        checker(service.type, type_service.hosting) &&
        <View style={styles.extra_info}>
            <InfoHosting />
        </View>
    }

    {service &&
        checker(service.type, type_service.consumption) &&
        <View style={styles.extra_info}>
            <InfoConsumption
                service={service}
                t={t}
            />
        </View>
    }

    {service &&
        checker(service.type, type_service.other) &&
        <View style={styles.extra_info}>
            <InfoOther
                service={service}
                t={t}
            />
        </View>
    }

    <Text style={styles.schedule}>{t('services.reviews')}</Text>
    <ListReviews
        idService={service.id}
    />

    </View>
</ScrollView>
)
}

```

El resultado de las codificaciones anteriores se muestra a continuación, donde se muestra la ejecución de las pantallas de listar servicios y ver servicio a detalle en sistemas operativos IOS y Android respectivamente, estas pantallas son la implementación del diseño de la Figura 18. Mockup Buscar servicio y Figura 19. Mockup Ver servicio.

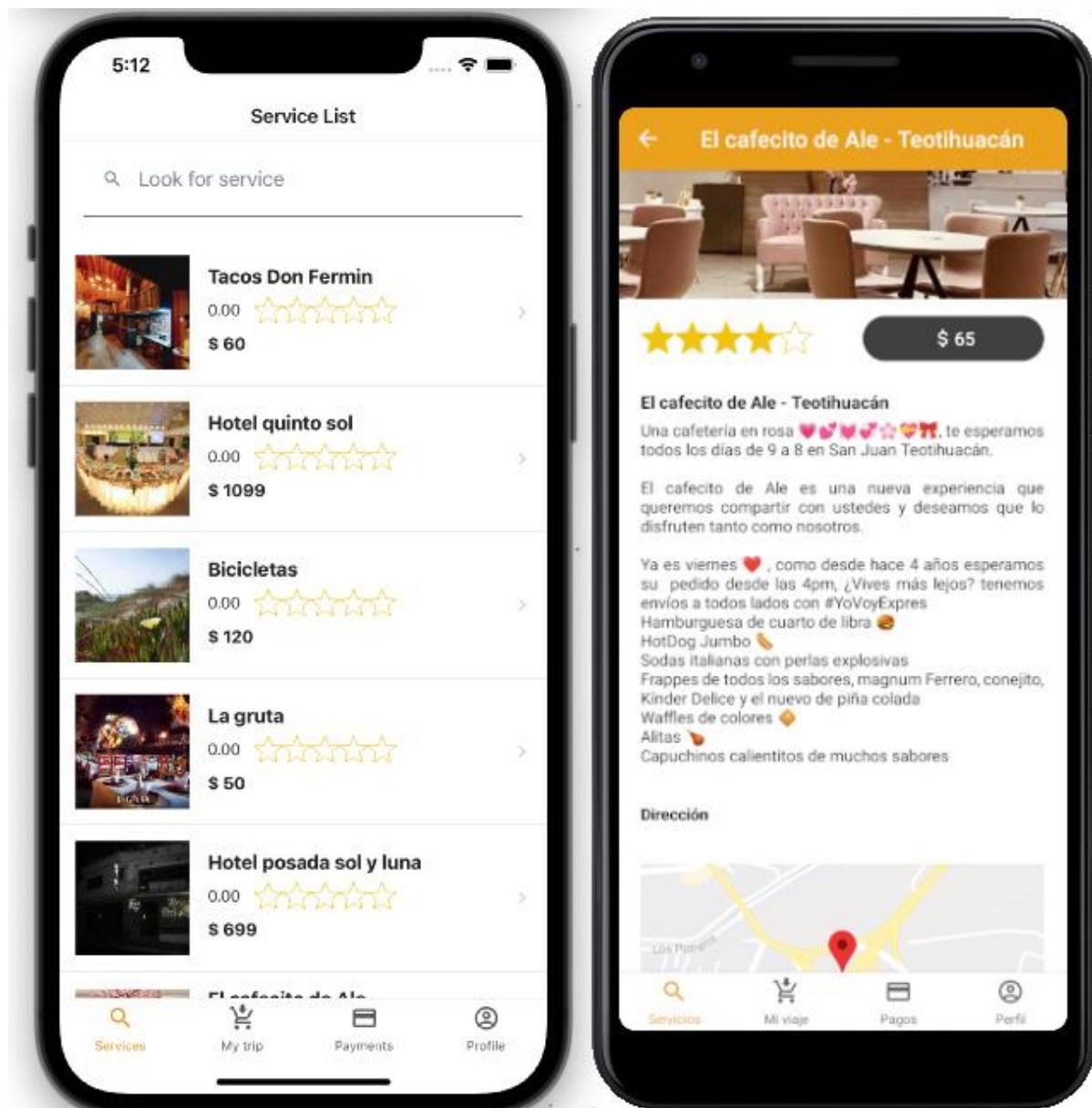


Figura 81. Pantallas de lista de servicios y ver servicio a detalle – TURISTA en IOS y Android

5.2.10.Consulta de servicios previamente registrados por el vendedor

El usuario vendedor posterior a registrar un servicio se da la posibilidad de listar los servicios previamente registrados y visualizar a detalle servicio por servicio, el siguiente fragmento de código muestra la implementación de listar servicios:

```
export default function ListServices(props) {
  const {services, handleLoadMore, isLoading} = props
  const navigation = useNavigation()

  return (
    <View style={styles.view_container}>
      {size(services) > 0 ? (
        <FlatList
          data={services}
          renderItem={({service}) =>
            <ServiceItem
              service={service}
              navigation={navigation}
            />}
            keyExtractor={(item, index) => index.toString()}
            onEndReachedThreshold={0.5}
            onEndReached={handleLoadMore}
            ListFooterComponent=<FooterList isLoading={isLoading}/>
          />
        ) : (
          <View style={styles.container_activity}>
            <Text style={{alignSelf: "center"}}>No hay servicios aún,
            registrada el primero ahora.</Text>
          </View>
        )
      )
    </View>
  )
}
```

Donde cada ítem de la lista se codifica de la siguiente manera:

```
function ServiceItem(props) {
  const {service, navigation} = props
  const { id, images, name, price, description, availability, timeInService,
  rating } = service.item
  const image_service = images[0]

  const goService = () => {
    navigation.navigate("seeservice", {id, name})
  }

  return (
    <TouchableOpacity onPress={goService}>
      <View style={styles.service_item_container}>
        <View style={styles.service_image_container}>
          <Image
            resizeMode="cover"
            PlaceholderContent=<ActivityIndicator color="#123456">
          />
          source={
            image_service
          }
        </View>
      </View>
    </TouchableOpacity>
  )
}
```

```

                ? { uri : image_service }
                : require("../../../../../../assets/no-image.png")
            }
            style={styles.img_service}
        />
    </View>

    <View style={styles.data_container}>
        <Text style={styles.service_name}>{ name }</Text>
        <Text>Tiempo de servicio: { timeInService }</Text>
        <Text>Disponibilidad unitaria: { availability }</Text>
        <Text>Precio: $ { price }</Text>
        <Text>Rating: { rating.toFixed(2) }</Text>
    </View>
</View>
</TouchableOpacity>
)
}
}

```

La codificación de consulta el servicio a detalle se aprecia en el siguiente fragmento de código:

```

export default function SeeService(props) {
    const {navigation, route} = props
    const {id, name} = route.params
    const [service, setService] = useState(null)
    const [rating, setRating] = useState(0)
    const [showModal, setShowModal] = useState(false)
    const [renderComponent, setRenderComponent] = useState(null)
    const [reloadInfo, setReloadInfo] = useState(false)
    const [isLoading, setIsLoading] = useState(false)
    const [error, setError] = useState(null)

    ...

    return (
        <View style= {styles.containe_general}>
        <ScrollView vertical style={styles.scroll_view}>
            <View >
                <CarouselImages
                    arrayImages = {service.images}
                    height = {200}
                    width = {screen_width}
                    id_doc={id}
                    setReloadInfo={setReloadInfo}
                    setIsLoading={setIsLoading}
                />
                <RatingService
                    rating={rating}
                    name={service.name}
                />

                <View style= {styles.conatiner_witout_carousel}>
                    {service &&
                        checker(service.type, type_service.hosting) &&
                        <View>
                            <BasicInfoService
                                service={service}
                                showModal={showModal}

```

```

        setShowModal={setShowModal}
        renderComponent={renderComponent}
        setRenderComponet={setRenderComponet}
        setReloadInfo={setReloadInfo}
    />
</View>
}

{service &&
  checker(service.type, type_service.consumption) &&
<View>
  <BasicInfoService
    service={service}
    showModal={showModal}
    setShowModal={setShowModal}
    renderComponent={renderComponent}
    setRenderComponet={setRenderComponet}
    setReloadInfo={setReloadInfo}
  />
  <SpecialInfoService
    service={service}
    showModal={showModal}
    setShowModal={setShowModal}
    renderComponent={renderComponent}
    setRenderComponet={setRenderComponet}
    setReloadInfo={setReloadInfo}
    setIsLoading={setIsLoading}
    isLoading={isLoading}
  />
</View>
}

{service &&
  checker(service.type, type_service.other) &&
<View>
  <BasicInfoService
    service={service}
    showModal={showModal}
    setShowModal={setShowModal}
    renderComponent={renderComponent}
    setRenderComponet={setRenderComponet}
    setReloadInfo={setReloadInfo}
  />
  <SpecialInfoService
    service={service}
    showModal={showModal}
    setShowModal={setShowModal}
    renderComponent={renderComponent}
    setRenderComponet={setRenderComponet}
    setReloadInfo={setReloadInfo}
    setIsLoading={setIsLoading}
    isLoading={isLoading}
  />
</View>
}

{ /* COMENTARIOS */
<Text style={styles.title}>Reviews</Text>
<ListReviews idService={service.id}/>

```

```

        <Button
            title="Eliminar servicio"
            containerStyle={styles.viewMapBtnContainerDelete}
            buttonStyle={styles.viewMapBtnDelete}
            onPress={() => {
                deleteService(id, service.images)
            }}
        />
    </View>
</View>
<Loading text="Actualizando ..." isVisible={isLoading} />
</View>
)
}

```

El resultado de las codificaciones anteriores se muestra a continuación, donde se muestra la ejecución de las pantallas de listar servicios y ver servicio a detalle en sistemas operativos IOS y Android respectivamente, estas pantallas son la implementación del diseño de la Figura 28. Mockup Lista de serviciosFigura 18. Mockup Buscar servicio y Figura 33. Mockup Ver servicio previamente registrado.

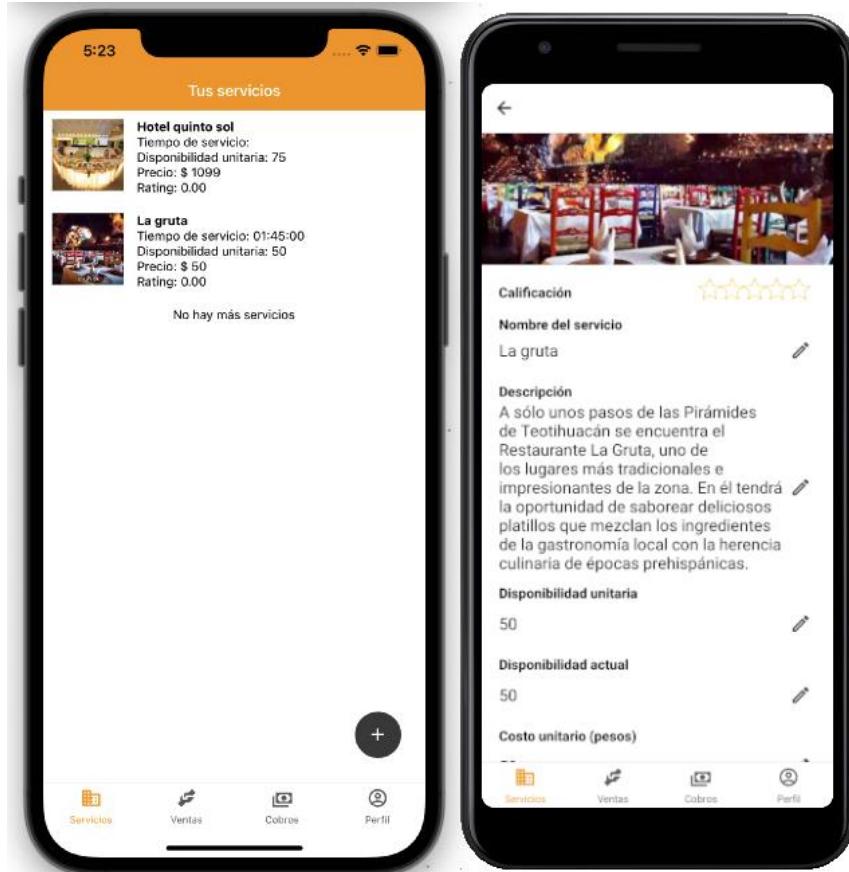


Figura 82. Pantallas de lista de servicios y ver servicio a detalle - VENDEDOR en IOS y Android

5.2.11.Comprobante de pago

El usuario turista posterior a realizar su compra, podrá listar todas las compras realizadas, así como consultar el comprobante de pago (Código QR), en el siguiente fragmento de código se implementa la funcionalidad de listar compras:

```
export default function MyTrip(props) {
    const { navigation } = props
    const [user, setUser] = useState(null)
    const [shopping, setShopping] = useState([])
    const [totalShopping, setTotalShopping] = useState(0)

    const limitShopping = 6

    const [startShopping, setstartShopping] = useState(null)
    const [isLoading, setIsLoading] = useState(false)
    const { t } = useTranslation()

    const query = `SELECT * FROM transactions WHERE turist LIKE
    '${firebase.auth().currentUser.uid}' ORDER BY creationAt DESC`
```

...

```
    return (
        <View style={styles.view_container}>
            {shopping.length === 0 ? (
                <NotFoundShopping />
            ) : (
                <FlatList
                    data={shopping}
                    renderItem={(shopping) => (
                        <ShoppingItem
                            shopping={shopping}
                            navigation={navigation}
                        />
                    )}
                    keyExtractor={(item, index) => index.toString()}
                    onEndReachedThreshold={0.00}
                    onEndReached={handleLoadMore}
                    ListFooterComponent={<View style={{height: 20}}/>}
                    ListFooterComponent={<FooterList isLoading={isLoading}/>}
                />
            )}
        </View>
    )
}
```

...

```
    return (
        <TouchableOpacity onPress={goShopping}>
            {type == "hosting" &&
                <View style={styles.sale_item_container}>
                    <View style={styles.sale_image_container}>
                        <Image
                            resizeMode="cover"
                            PlaceholderContent={<ActivityIndicator
color="#123456" />}
                            source={
```

```

                status == 1 ?
require("./../../../../assets/status_icon_payment_visit.png")
: status == 2 ?
require("./../../../../assets/status_icon_payment_no_visit.png")
:
require("./../../../../assets/status_icon_no_payment_no_visit.png")
}
style={styles.img_sale}
/>
</View>

<View style={styles.data_container}>
    <Text style={styles.sale_name}>{ name_sale }</Text>
    <Text>
        {t('mytrip.shopping_item.buy_date')}

{getDateTimeString(creationAt.toDate().toString().split(" "))}
    </Text>
    <Text>
        {t('mytrip.shopping_item.reservation_date')}
        {moment(new Date(checkin_day.seconds * 1000).toLocaleDateString("en-US")).format("YYYY-MM-DD")}
        {" - "}
        {moment(new Date(checkout_day.seconds * 1000).toLocaleDateString("en-US")).format("YYYY-MM-DD")}
    </Text>
    <Text>
        {t('mytrip.shopping_item.places_number')} {people}
    </Text>
    <Text>
        {t('mytrip.shopping_item.payment')} $ {total}
    </Text>
</View>
</View>
}

{type != "hosting" &&
<View style={styles.sale_item_container}>
    <View style={styles.sale_image_container}>
        <Image
            resizeMode="cover"
            PlaceholderContent={<ActivityIndicator
color="#123456" />}
            source={
                status == 1 ?
require("./../../../../assets/status_icon_payment_visit.png")
: status == 2 ?
require("./../../../../assets/status_icon_payment_no_visit.png")
:
require("./../../../../assets/status_icon_no_payment_no_visit.png")
}
            style={styles.img_sale}
        />
    </View>

    <View style={styles.data_container}>
        <Text style={styles.sale_name}>{ name_sale }</Text>
        <Text>

```

```

        {t('mytrip.shopping_item.buy_date')}}

{getDateTimeString(creationAt.toDate().toString().split(" "))}
    </Text>
    <Text>
        {t('mytrip.shopping_item.reservation_date')}

{getDateTimeString(shopping.item.reservation_date_time.toDate().toString().split(" "))}

    </Text>
    <Text>
        {t('mytrip.shopping_item.places_number')} {people}
    </Text>
    <Text>
        {t('mytrip.shopping_item.payment')} $ {total}
    </Text>
</View>
</View>
}
</TouchableOpacity>
)
}

```

Con el objetivo de mantener un buen rendimiento en la aplicación y no sobrecargar la misma cuando se consultan las compras, se decide mostrar X número de ítems de compra, posterior a visualizar los elementos mostrados, si se hace scroll hacia abajo, se mostrarán nuevamente X nuevas compras, en el siguiente fragmento de código se implementa esta funcionalidad:

```

const handleLoadMore = () => {
    const result_shopping = []
    //sales.length < totalSales && setisLoading(true)

    db
        .collection("transactions")
        .where("turist", "==", getUser().uid)
        .orderBy("creationAt", "desc")
        .startAfter(startShopping.data().creationAt)
        .limit(limitShopping)
        .get()
        .then( response => {
            if(response.docs.length > 0){
                setstartShopping(response.docs[response.docs.length - 1])
            }else {
                setLoading(false)
            }

            response.forEach((doc) => {
                const shopping = doc.data()
                shopping.id = doc.id
                result_shopping.push(shopping)
            })
        }
        .then( () => {
            setShopping([...shopping, ...result_shopping]))
        })
    }
}

```

El resultado de las codificaciones anteriores se muestra a continuación, donde se muestra la ejecución de las pantallas de listar compras y ver comprobante a detalle en sistemas operativos IOS y Android respectivamente, estas pantallas son la implementación del diseño de la Figura 23. Mockup Lista de mis viajes y la Figura 24. Mockup Comprobante de pago para servicio.

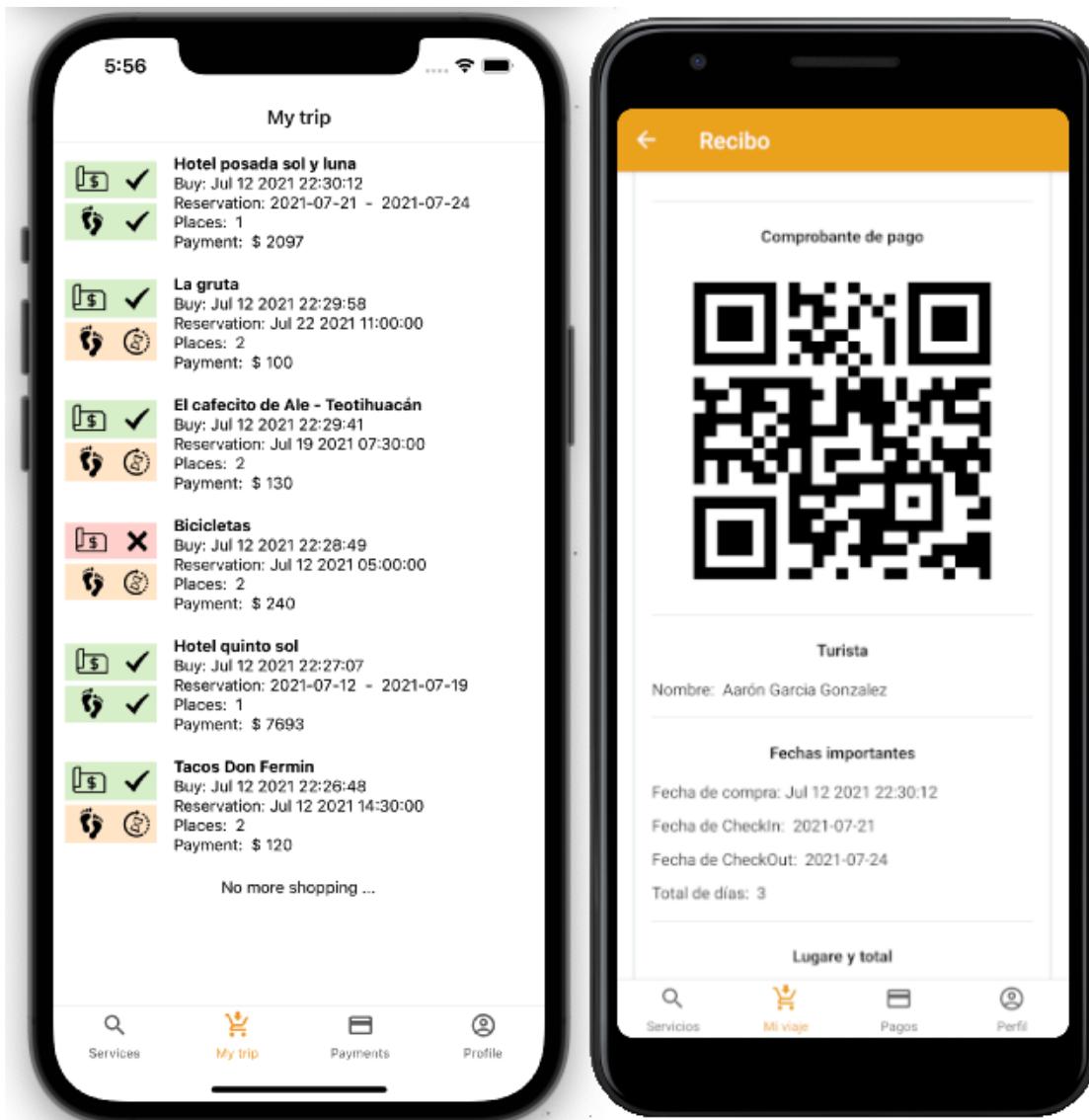


Figura 83. Pantallas de lista de compras y comprobante de pago en IOS y Android

Cuando un usuario vendedor o turista consulta un comprobante de pago se incluye el estatus de dicha compra/venta, hay 3 posibles estatus:

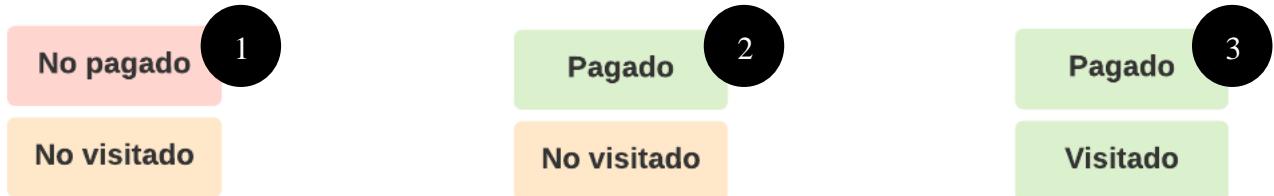


Figura 84. Tipos de estatus de compra/venta versión textual

1. El servicio no ha sido pagado ni ha sido visitado (CheckIn)
2. El servicio ya fue pagado, pero aún no ha sido visitado (CheckIn)
3. El servicio ya fue pagado y ya ha sido visitado (CheckIn)

Con el objetivo de agilizar y unificar la aplicación móvil, estos estatus son rediseñados para mostrarse de la misma manera para el usuario turista sin importar el idioma en que se encuentre:



Figura 85. Tipos de estatus de compra/venta versión estandarizada

5.2.12. Puntuaciones y comentarios

Los usuarios turistas son los únicos que pueden calificar y hacer comentario a un servicio, recordar que esto solo es posible si el turista compra el servicio, y solo puede calificar una vez por cada servicio, a continuación, se muestra el código de implementación del formulario de creación de comentarios:

```
export default function ReviewMark(props) {
  const {id_service,
    setShowModal,
    completeReview,
    setCompleteReview,
    hasBeenModifiedReview,
    setHasBeenModifiedReview
  } = props

  const [rating, setRating] = useState(null)
  const [title, setTitle] = useState(null)
  const [review, setReview] = useState(null)
  const [isLoading, setIsLoading] = useState(false)
  const { t } = useTranslation()

  ...
}
```

```

        return(
            <ScrollView contentContainerStyle={styles.view}>
                <Text
style={styles.text}>{t('mytrip.review_form.title_lbl')}</Text>
                <View style={{width: "100%", height: 350, marginBottom: 20}}>
                    <View style={styles.view_rating}>
                        <AirbnbRating
                            count={5}
                            reviews={['Terrible', 'OK', 'Good', 'Wow', 'Jesus']}
                            defaultRating={0}
                            size={30}
                            onFinishRating={(value) => {setRating(value)}}
                            defaultRating={completeReview !== null ?
completeReview.rating : 0}
                        />
                    </View>

                    <Input
                        multiline={false}
                        autoCompleteType='off'
                        placeholder = {t('mytrip.review_form.title_placeholder')}
                        containerStyle={styles.text_title_rating}
                        onChangeText={(value) => setTitle(value)}
                        defaultValue= {completeReview !== null ?
completeReview.title : ""}
                    />

                    <Input
                        multiline={true}
                        placeholder = {t('mytrip.review_form.review_placeholder')}
                        autoCompleteType='off'
                        containerStyle={styles.input}
                        onChangeText={(value) => setReview(value)}
                        defaultValue= {completeReview !== null ?
completeReview.review : ""}
                    />
                </View>

                <View style={styles.viewBtn}>
                    <Button
                        title={t('mytrip.review_form.save_btn')}
                        containerStyle={styles.viewBtnContainerSave}
                        buttonStyle={styles.viewBtnSave}
                        onPress={addReview}
                    />
                    <Button
                        title={t('mytrip.review_form.cancel_btn')}
                        containerStyle={styles.viewBtnContainerCancel}
                        buttonStyle={styles.viewBtnCancel}
                        onPress={() => setShowModal(false)}
                    />
                </View>

                <Loading text={t('mytrip.review_form.updating_loading')}>
isVisible={isLoading}</Loading>
            </ScrollView>
        )
    }
}

```

De igual manera se muestra la implementación de listar comentarios cuando se consulta un servicio:

```
function ReviewItem(props) {
    const {key} = props
    const {
        avatar_user,
        createdAt,
        id,
        id_service,
        id_user,
        rating,
        review,
        title
    } = props.review

    const getDateTimeString = (data) => {
        const result = []
        for (let index = 0; index < data.length; index++) {
            if((index != 0) && (index < 5)){
                result.push(data[index])
            }
        }
        return result.join(" ")
    }

    return (
        <View style={styles.view_review} >
            <View style={styles.view_image_avatar}>
                <Avatar
                    size="large"
                    rounded
                    containerStyle={styles.image_avatar_user}
                    source={avatar_user ? {uri : avatar_user} :
require("./../../../../assets/avatar-default.jpg")}>
                />
            </View>
            <View style={styles.view_data_review}>
                <Text style={styles.txt_title_review}>{title}</Text>
                <View style={{flex: 1, flexDirection: "row", justifyContent:
'space-between'}}>
                    <Text style={{flex: 1}}>

{getDateTimeString(createdAt.toDate().toString().split(" "))}

                    </Text>
                    <Rating
                        imageSize={20}
                        startingValue={rating}
                        readonly
                        style={{flex: 1}}
                    />
                </View>
                <Text style={styles.txt_txt_review}>{review}</Text>
            </View>
        </View>
    )
}
```

El resultado de las codificaciones anteriores se muestra a continuación, donde se muestra la ejecución de las pantallas de lista de reviews y formulario de creación de review en sistemas operativos IOS y Android respectivamente.

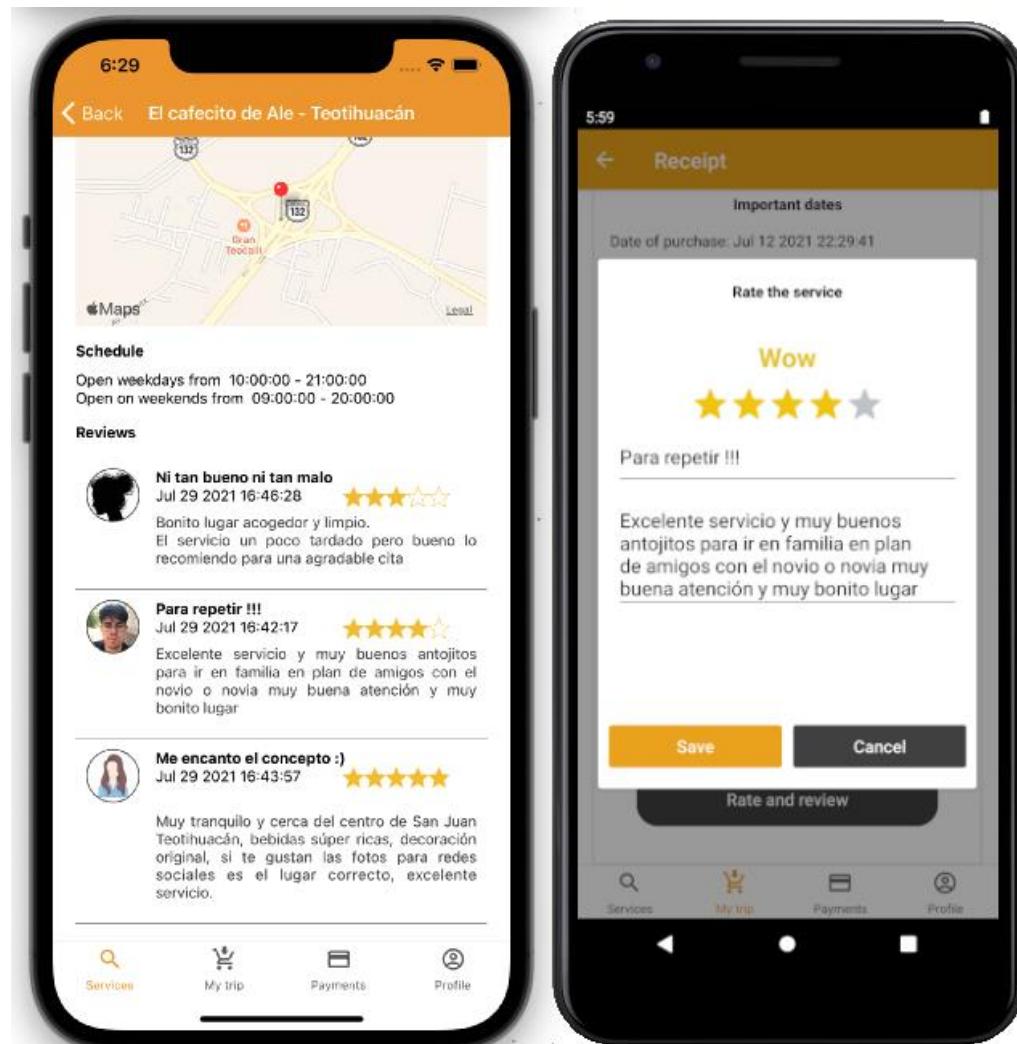


Figura 86. Pantallas de lista y creación de reviews en IOS y Android

5.2.13.Ver ventas

Un usuario vendedor se espera múltiples ventas de cada uno de los servicios que registre en la aplicación, entonces podrá consultarlos en orden descendente acorde con la fecha de la transacción, al igual que al listar servicios comprados en el caso del usuario turista, con el objetivo de mantener un buen rendimiento en la aplicación y no sobrecargar la misma cuando se consultan las compras, se decide mostrar X número de ítems de venta, posterior a visualizar los elementos mostrados, si se hace scroll hacia abajo, se mostrarán nuevamente X nuevas ventas, en el siguiente fragmento de código se implementa esta funcionalidad:

```
export default function Sale(props) {
  const { navigation } = props
  const [user, setUser] = useState(null);

  const [sales, setSales] = useState([])
  const [totalSales, setTotalSales] = useState(0)

  const limitSales = 6
  const [startSale, setStartSale] = useState(null)

  const [isLoading, setIsLoading] = useState(false)

  useEffect(() => {
    firebase.auth().onAuthStateChanged(userInfo) => {
      setUser(userInfo)
    }
  }, [])

  const handleLoadMore = () => {
    const result_sales = []
    //sales.length < totalSales && setIsLoading(true)

    db
      .collection("transactions")
      .where("seller_id", "==", getUser().uid)
      .orderBy("creationAt", "desc")
      .startAfter(startSale.data().creationAt)
      .limit(limitSales)
      .get()
      .then(response => {
        if(response.docs.length > 0){
          setStartSale(response.docs[response.docs.length - 1])
        }else{
          setIsLoading(false)
        }

        response.forEach((doc) => {
          const sale = doc.data()
          sale.id = doc.id
          result_sales.push(sale)
        })
      })

      setSales([...sales, ...result_sales])
  })
}
```

```

        return (
            <View style={styles.view_container}>
                {size(sales) > 0 ?
                    (<FlatList
                        data={sales}
                        renderItem={(sale) => (
                            <SaleItem sale={sale}
                            navigation={navigation}
                        />)}
                        keyExtractor={(item, index) => index.toString()}
                        onEndReachedThreshold={0.00}
                        onEndReached={handleLoadMore}
                        ListFooterComponent={<View style={{height: 20}}/>}
                        ListFooterComponent={<FooterList isLoading={isLoading}/>}
                    />):
                    (<View style={styles.loader_container}>
                        <Text style={styles.txt_loader}>No hay ventas ...</Text>
                    </View> )
                }
            <Icon
                type="material-community"
                name="qrcode-scan"
                color="#404040"
                reverse
                containerStyle={styles.btn_container}
                onPress={() => navigation.navigate("selectNewService")}
            />
        </View>
    )
)
}

```

Al dar clic sobre cualquier ítem de venta, se mostrará de manera detallada el nombre del servicio, el turista que adquirió el servicio, las fechas importantes, el total de lugares a reservar y el total, esta implementación se observa en el siguiente fragmento de código:

```

export default function SeeSale(props) {
    const {route, navigation} = props
    const {creationAt, name_sale, people, seller_id, service_id, status,
total, turist, unit_price, turist_name, type} = route.params.sale

    const InfoHosting = () => {
        return (
            <Card>
                <Card.Title>{name_sale}</Card.Title>
                <Card.Divider style={styles.divider_card}/>
                <View>
                    <Text style={styles.txt_header_card_item}>Turista</Text>
                    <Text style={styles.txt_item}>Nombre: {turist_name}</Text>
                </View>
                <Card.Divider style={styles.divider_card}/>
                <View>
                    <Text style={styles.txt_header_card_item}>Fechas</Text>
                    <Text style={styles.txt_item}>Fecha de compra: {
                        creationAt.toDate().toISOString().substring(0, 10) } </Text>

```

```

                <Text style={styles.txt_item}>Fecha de check in: {  

route.params.sale.checkin_day.toDate().toISOString().substring(0, 10) }  

</Text>  

                <Text style={styles.txt_item}>Fecha de check out: {  

route.params.sale.checkout_day.toDate().toISOString().substring(0, 10) }  

</Text>  

                <Text style={styles.txt_item}>Total de días:  

{Math.abs(route.params.sale.checkout_day.toDate() -  

route.params.sale.checkin_day.toDate())/(1000 * 3600 * 24)}</Text>  

            </View>  

            <Card.Divider style={styles.divider_card}/>  

            <View>  

                <Text style={styles.txt_header_card_item}>Lugares y  

total</Text>  

                <Text style={styles.txt_item}>Número de reservas:  

{people}</Text>  

                <Text style={styles.txt_item}>Precio unitario:  

${unit_price}</Text>  

                <Text style={styles.txt_item}>Precio total: ${total}</Text>  

            </View>  

            <Card.Divider style={styles.divider_card}/>  

            <View style={{alignItems: "center"}}>  

                <Text style={styles.txt_header_card_item}>Status</Text>  

                <Image  

                    style={styles.image}  

                    resizeMode="cover"  

                    source={  

                        status == 1 ?  

require("./../../../../assets/status_pagado_visitado.png")  

                        : status == 2 ?  

require("./../../../../assets/status_pagado_no_visitado.png")  

                        :  

require("./../../../../assets/status_no_pagado_no_visitado.png")  

                    }  

                />  

            </View>  

        </Card>  

    )
}

const InfoOthers = () => {
    return (
        <Card>
            <Card.Title>{name_sale}</Card.Title>
            <Card.Divider style={styles.divider_card}/>
            <View>
                <Text style={styles.txt_header_card_item}>Turista</Text>
                <Text style={styles.txt_item}>Nombre: {turist_name}</Text>
            </View>
            <Card.Divider style={styles.divider_card}/>
            <View>
                <Text style={styles.txt_header_card_item}>Fechas</Text>
                <Text style={styles.txt_item}>
                    {"Compra: "}
{getDateTimeString(route.params.sale.creationAt.toDate().toString().split(" "
))}</Text>
            </View>
        </Card>
    )
}

```

```

        <Text style={styles.txt_item}>
            {"Reservación: "}

{getDateTimeString(route.params.sale.reservation_date_time.toDate().toString()
.split(" "))}
        </Text>
    </View>
    <Card.Divider style={styles.divider_card}/>
    <View>
        <Text style={styles.txt_header_card_item}>Lugares y
total</Text>
        <Text style={styles.txt_item}>Número de reservas:
{people}</Text>
        <Text style={styles.txt_item}>Precio unitario:
${unit_price}</Text>
        <Text style={styles.txt_item}>Precio total: ${total}</Text>
    </View>
    <Card.Divider style={styles.divider_card}/>
    <View style={{alignItems: "center"}}>
        <Text style={styles.txt_header_card_item}>Status</Text>
        <Image
            style={styles.image}
            resizeMode="cover"
            source={
                status == 1 ?
require("./../../../../assets/status_pagado_visitado.png")
                : status == 2 ?
require("./../../../../assets/status_pagado_no_visitado.png")
                :
require("./../../../../assets/status_no_pagado_no_visitado.png")
            }
        />
    </View>
</Card>
)
}
}

return (
    <ScrollView style={styles.container}>
        {type === "hosting" && <InfoHosting />}
        {type !== "hosting" && <InfoOthers />}
    </ScrollView>
)
}
}

```

El resultado de las codificaciones anteriores se muestra a continuación, donde se muestra la ejecución de las pantallas de lista de ventas y detalle de venta en sistemas operativos IOS y Android respectivamente, estas pantallas son la implementación del diseño de la Figura 34. Mockup Lista de ventas y la Figura 35. Mockup Ver venta.

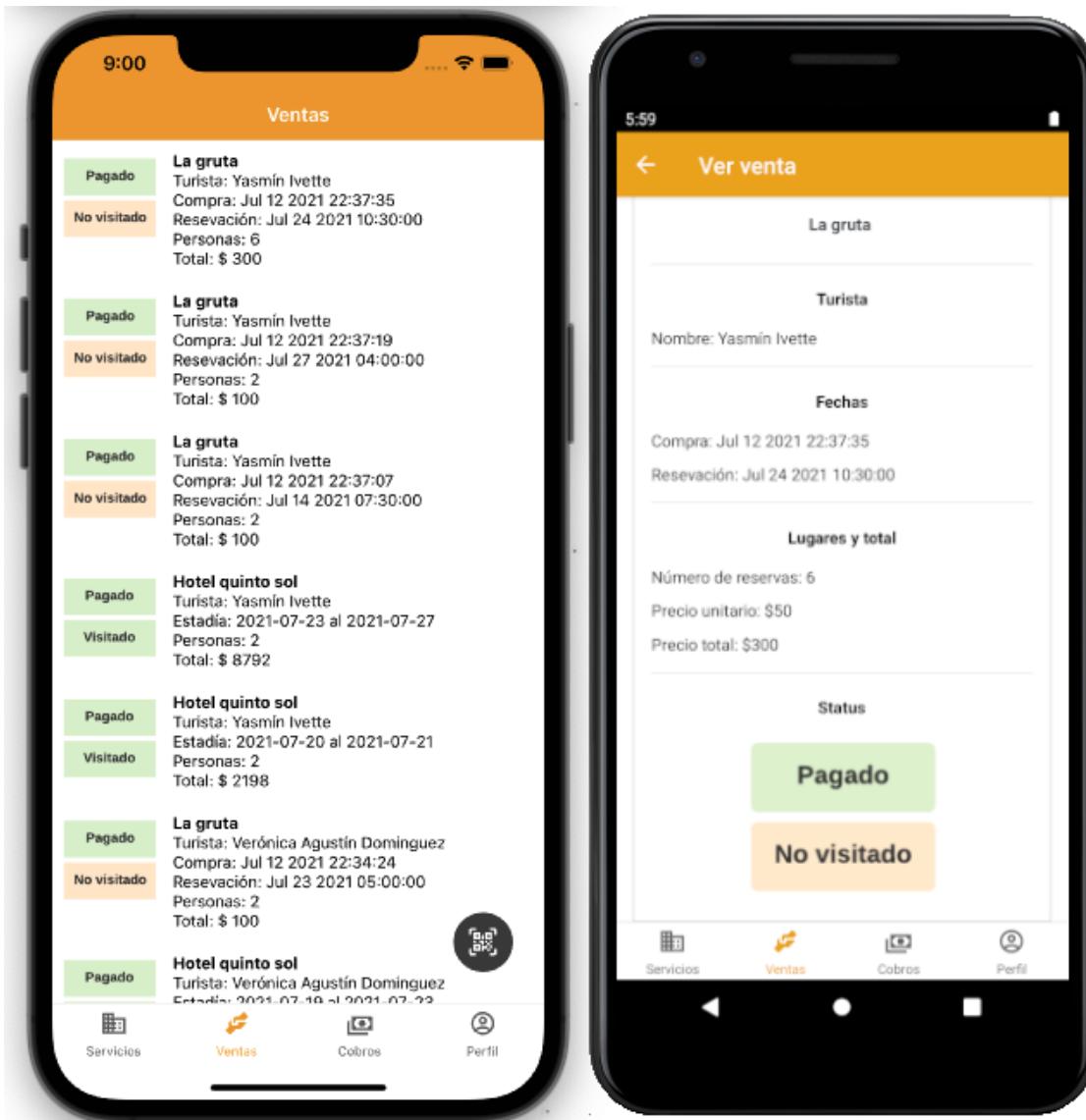


Figura 87. Pantallas de lista de ventas y ver venta a detalle en IOS y Android

5.2.14.Ver perfil de usuario

El usuario turista y vendedor tiene acceso al apartado de “Perfil”, donde se pueden consultar y actualizar los datos de usuario, en el caso del usuario vendedor se visualiza el estatus de validación ante el RFC por parte del administrador, y para el en usuario turista se muestra el idioma en que se encuentra la aplicación, al presionar sobre el ítem de lenguaje se cambiará el idioma, en el siguiente fragmento de código se implementa esta funcionalidad para la aplicación del turista:

```
export default function Profile() {
  const [userInfo, setuserInfo] = useState(null);
  const [userFirebase, setUserFireBase] = useState(null);
  const [loading, setLoading] = useState(false);
  const [loadingText, setLoadingText] = useState("");
  const toastRef = useRef();
  const [reloadInfo, setReloadInfo] = useState(false)

  useEffect(() => {
```

```

// Funcion anonima autoejecutable
(async () => {
    const user = await getUserData()
    setUserInfo(user)

    const user_firebase = await firebase.auth().currentUser;
    setUserFireBase(user_firebase) // only save photo url
})()
setReloadInfo(false)
}, [userFirebase, reloadInfo]) // [] -> Solo se ejecuta una vez

return (
    <View style={styles.view_business_info}>
        {userInfo &&
            <InfoBusiness
                userInfo={userInfo}
                userFirebase={userFirebase}
                toastRef={toastRef}
                setLoading={setLoading}
                setLoadingText={setLoadingText}
            />
        }

        {userInfo &&
            <AccountOptions
                userInfo={userInfo}
                userFirebase={userFirebase}
                toastRef={toastRef}
                setReloadInfo={setReloadInfo}
            />
        }
    </View>
)

```

```

<Button
    title="Cerrar sesión"
    containerStyle={styles.btn_sign_off}
    buttonStyle = {{backgroundColor: "#404040"}}
    onPress={() => signOff()}
/>

<Toast ref={toastRef}
    fadeInDuration={500}
    fadeOutDuration={2500}
    style={{backgroundColor: '#EAA11B', width:'80%'}}
    position="center"
    opacity={0.9}
    textStyle={{color:'#ffffff'}}
```

```

    />

    <Loading text = {loadingText} isVisible={loading}/>
</View>
)
}

```

En el siguiente fragmento de código se implementa esta funcionalidad para la aplicación del vendedor:

```
export default function Profile() {
```

```

const [userInfo, setuserInfo] = useState(null);
const [userFirebase, setUserFireBase] = useState(null);
const [loading, setLoading] = useState(false);
const [loadingText, setLoadingText] = useState("");
const toastRef = useRef();
const [reloadInfo, setReloadInfo] = useState(false)
const { t } = useTranslation()

useEffect(() => {
    // Funcion anonima autoejecutable
    (async () => {
        const user = await getUserData()
        setuserInfo(user)

        const user_firebase = await firebase.auth().currentUser;
        setUserFireBase(user_firebase) // only save photo url
    })()
    setReloadInfo(false)
}, [userFirebase, reloadInfo]) // [] -> Solo se ejecuta una vez

return (
    <View style={styles.view_user_info}>
        {userInfo &&
            <InfoUser
                userInfo={userInfo}
                userFirebase={userFirebase}
                toastRef={toastRef}
                setLoading={setLoading}
                setLoadingText={setLoadingText}
            />
        }

        {userInfo &&
            <AccountOptions
                userInfo={userInfo}
                userFirebase={userFirebase}
                toastRef={toastRef}
                userFirebase={userFirebase}
                setReloadInfo={setReloadInfo}
            />
        }
    </View>
    <Button
        title={t('profile.sign_off_btn')}
        containerStyle={styles.btn_sign_off}
        buttonStyle = {{backgroundColor: "#404040"}}
        onPress={() => signOff()}
    />

    <Toast ref={toastRef}
        fadeInDuration={500}
        fadeOutDuration={2500}
        style={{backgroundColor: '#EAA11B', width:"80%"}}
        position="center"
        opacity={0.9}
        textStyle={{color:'ffffff'}}}
    />

    <Loading text = {loadingText} isVisible={loading}/>

```

```
        </View>
    )
}
```

El resultado de las codificaciones anteriores se muestra a continuación, donde se muestra la ejecución de las pantallas de perfil del usuario vendedor y turista en sistemas operativos IOS y Android respectivamente, estas pantallas son la implementación del diseño de la Figura 39. Mockup Mi perfil vendedor y la Figura 27. MockUp Mi perfil de usuario turista.

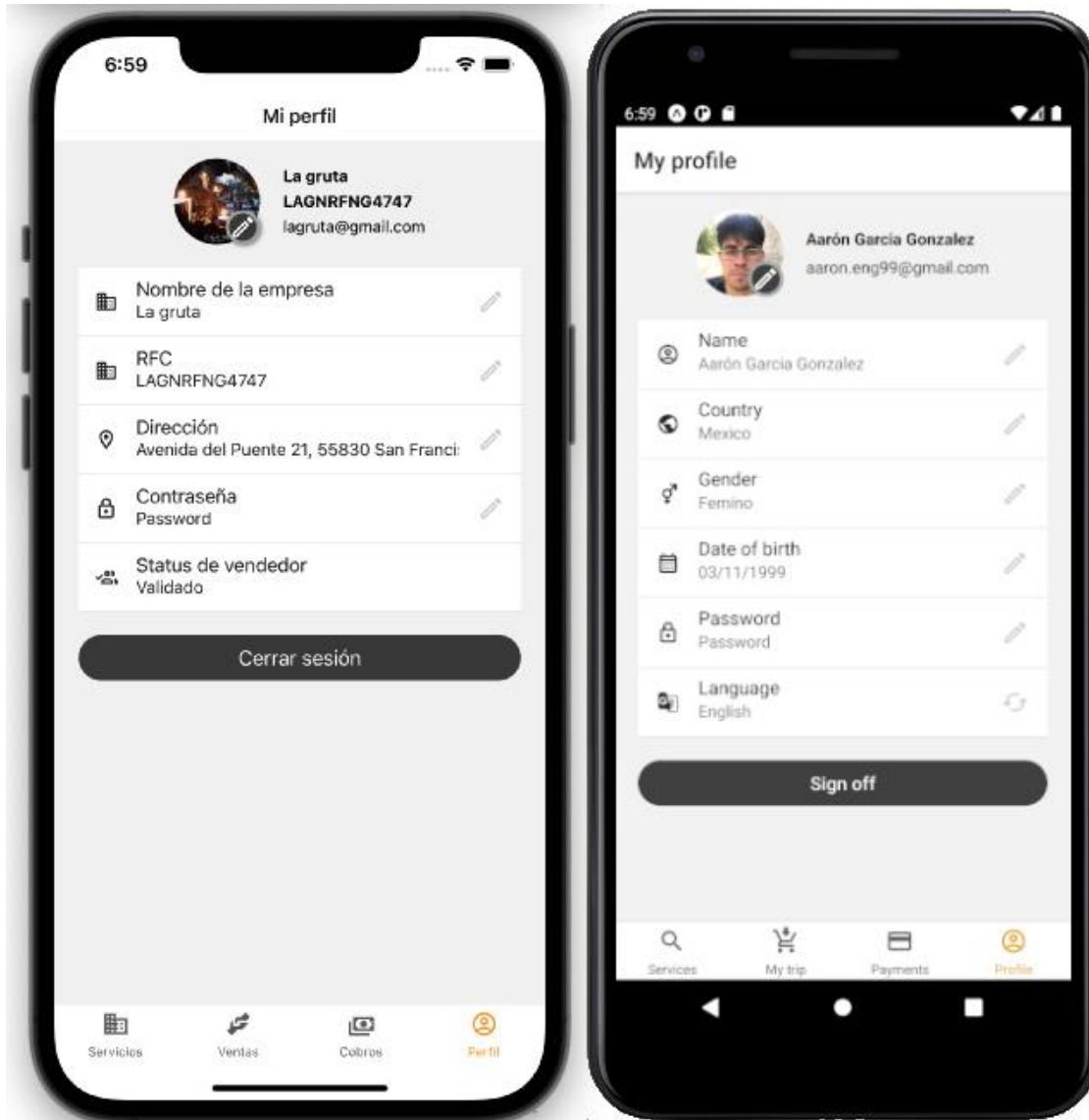


Figura 88. Pantallas de perfil de usuario vendedor y turista en IOS y Android

5.2.15. Aplicación de administrador

El desarrollo de la aplicación web para el usuario administrador es muy similar con el desarrollo de la aplicación móvil, esta aplicación no es la misma que la aplicación móvil con algún tipo de balanceador de usuarios, sino, una totalmente nueva, se desarrolla bajo react, firebase, css, html y javascript puro. La estructura principal de componentes, la observamos a continuación:



Figura 89. Estructura de componentes del proyecto web administrador

5.2.15.1. Inicio de sesión

El proyecto da la posibilidad de tener múltiples usuarios turistas y vendedores, mientras que solo hay un usuario de tipo administrador (este se carga desde base de datos en firebase), por lo que su interfaz de usuario para este ultimo rol, solo se da la posibilidad de iniciar sesión y no un registro de usuario como en la aplicación móvil, en el siguiente fragmento de código se muestra la implementación del componente login:

```
import React from 'react'
import {Form, Button, } from "react-bootstrap"

const LoginForm = (props) => {
    const {
        email, setEmail, password, setPassword, handleLogin, handleSignUp,
        hasAccount, setHasAccount, errorEmail, setErrorEmail,
        errorPassword, setErrorPassword } = props;

    return (
        <div className="container">
            <div style={{height: "100vh", display: "flex", flexDirection:
"column", alignItems: "center", justifyContent: "center"}}>
                <div className="d-grid gap-2" style={{width: "50%"}>
                    <label>Username: </label>
                    <input
                        type="text"
                        autoFocus
                        required
                        value={email}
                        onChange={e => setEmail(e.target.value)} />
                    <p className="errorMsg">{errorEmail}</p>
                    <label>Password: </label>
                    <input
                        type="password"
                        autoFocus
                        required
                        value={password}
                        onChange={e => setPassword(e.target.value)} />
                </div>
            </div>
        </div>
    )
}
```

```

        <p className="errorMsg">{errorPassword}</p>

        <div className="d-grid gap-2">
            <Button variant="warning" size="lg"
onClick={handleLogin}>Sign In</Button>
        </div>
    </div>
</div>
)
}

export default LoginForm

```

El resultado de la codificación anterior se muestra a continuación, donde se muestra la ejecución de la pantalla de login del usuario administrador en versión web, esta pantalla es la implementación del diseño de la Figura 40. MockUp - Administrador, Iniciar sesión:

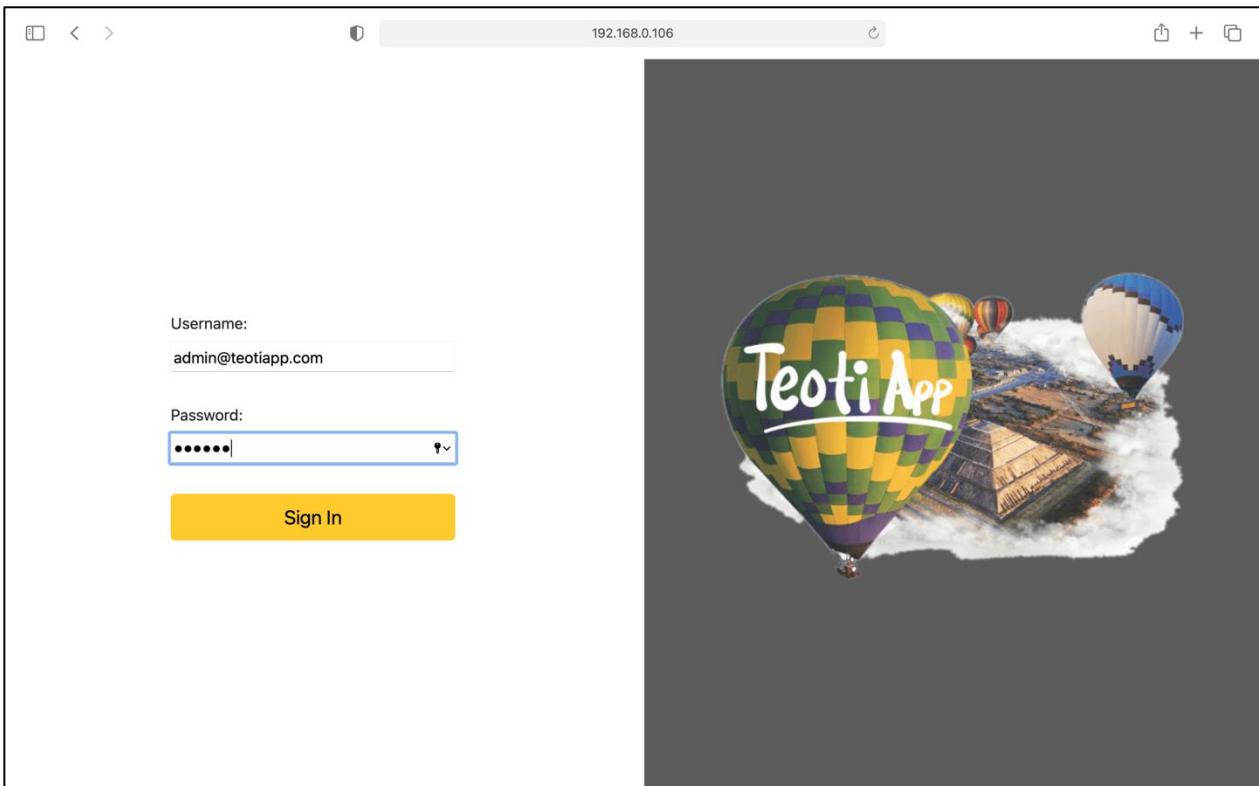


Figura 90. Pantalla de login de administrador

5.2.15.2. Obtención de vendedores

Dada la lógica de negocio, se requieren 3 operaciones básicas; obtención de vendedores pendientes a dictaminar, vendedores dictaminados como aprobados y vendedores dictaminados como rechazados, entonces internamente cada que se registra un vendedor se le asigna un código numérico de status igual con 0, esto representa que esta pendiente a dictaminarse, cuando se dictamina como aprobado, este atributo cambia a valor numérico igual con 1 y finalmente cuando pasa dictaminarse como rechazado, el atributo se actualiza a un valor numérico igual con 2, una vez que ha sido cambiado de status pendiente a cualquiera de los otros 2 valores, ya no será posible regresar a status pendiente, mientras que si es posible pasar de aprobado a rechazado y viceversa.

Para hacer más eficiente y escalable este proceso de consulta de servicios y filtrado por status, se crea la siguiente función, la cual se encarga de consultar y retornar en un arreglo de objetos a los vendedores que cumplen con el criterio que recibe como argumento:

```
import firebaseApp from './Firebase';
import firebase from "firebase/app";
import "firebase/firestore";

const db = firebase.firestore(firebaseApp)

export const handleSellerList = async (status_type) => {
    const final_result = []

    await db
        .collection("users")
        .where("role", "==", "seller")
        .where("status", "==", status_type)
        .get()
        .then((response) => {
            response.forEach((doc) => {
                const seller_item = doc.data();
                seller_item.id = doc.id;
                final_result.push(seller_item);
            })
        })
        .catch((response) => {
            console.log('=====');
            console.log("Error while getting seller list by status equals to " );
            console.log('=====');
        })
        .finally(() => {
            //alert(final_result.length)
        })
}

return final_result
}
```

5.2.15.3. Solicitudes pendientes

Cuando un usuario vendedor se registra en la aplicación móvil IOS o Android, en automático el usuario administrador recibe una solicitud de validación, entonces posterior al iniciar sesión con sus usuario y contraseña, se dirige a la barra de navegación en el ítem “Pendientes”, donde se despliega una tabla, donde cada fila es una solicitud de aprobación de vendedor, donde se incluye un botón para “Aprobar” y otro para “Rechazar” la solicitud, además en el encabezado se muestra un hipervínculo el cual re direcciona a la pagina de validación de RFC ante el SAT, esta pagina se abre en una nueva pestaña, el siguiente fragmento de código representa lo anterior mencionado:

```
import React, {useEffect, useState} from 'react'
import {Table, Button, Col, Row} from "react-bootstrap"
import {handleSellerList, handleUpdateStatusSeller} from
"../../firebase/Actions"
import SatLink from "./SatLink"

function Pending(props) {
    const [items, setItems] = useState([])
    const [hasBeenModified, setHasBeenModified] = useState(false)

    useEffect(async () => {
        const result = await handleSellerList(0)
        setItems(result)
    }, [hasBeenModified])

    const OnPushButton = (button_pressed, seller_doc) => {
        handleUpdateStatusSeller(button_pressed, seller_doc)
        setHasBeenModified(!hasBeenModified)
    }

    return (
        <div className="container">
            <SatLink />
            <Table striped bordered hover size="sm">
                <thead>
                    <tr>
                        <th>Nombre</th>
                        <th>RFC</th>
                        <th>Dirección</th>
                        <th>Email</th>
                        <th>Teléfono</th>
                        <th>Acción</th>
                    </tr>
                </thead>
                <tbody>
                    {items.length > 0 && (
                        items.map((item) => (
                            <tr>
                                <td>{item.name}</td>
                                <td>{item.rfc}</td>
                                <td>{item.address}</td>
                                <td>{item.email}</td>
                                <td>{item.phone}</td>
                                <td>
                                    <div className="btn-group">
```

```

        <Button variant="success" onClick={() =>
OnPushButton(1, item.id)}>Aprobar</Button>
        <Button variant="warning" onClick={() =>
OnPushButton(2, item.id)}>Rechazar</Button>
      </div>
    </td>
  </tr>
)
)
)
}

</tbody>
</Table>
</div>
)
}

export default Pending

```

El resultado de la codificación anterior se muestra a continuación, donde se muestra la ejecución de la pantalla de solicitudes pendientes del administrador en versión web, esta pantalla es la implementación del diseño de la Figura 41. MockUp - Administrador, Selección de operación (Que se muestra como barra de navegación superior) y la Figura 42. MockUp - Administrador, Solicitudes pendientes:

Nombre	RFC	Dirección	Email	Teléfono	Acción
La soberana bar Teotihuacán	JFNRKXKFRN78	Circuito Arqueológico, 55850 San Martín Centro, Méx., Mexico	lasobersnabarteo@gmail.com	5949582225	<button>Aprobar</button> <button>Rechazar</button>
La puerta del sol	GNRNNNDGN7653	Av 16 de Septiembre Sur 62, San Martín Centro, 55850 San Martín Centro, Méx., Mexico	lapuertadelsol@gmail.com	5620976717	<button>Aprobar</button> <button>Rechazar</button>
Asadero y parrilla La Chorcha	HFNFKQJDBC89	Av Tuxpan No 149, 55850 San Martín Centro, Méx., Mexico	ayplc@gmail.com	5520776852	<button>Aprobar</button> <button>Rechazar</button>
Luiggis	BFNRNFMDTB07	Calle 16 De Septiembre 7, San Martín Centro, 55850 San Martín Centro, Méx., Mexico	luiggis@gmail.com	5949585433	<button>Aprobar</button> <button>Rechazar</button>

Figura 91. Pantalla de solicitudes pendientes del administrador

5.2.15.4. Solicitudes aprobadas

Posterior a que el usuario administrador ha dictaminado por primera vez una solicitud, tiene la posibilidad de consultar todas aquellas solicitudes que ha dictaminado como aprobadas anteriormente y cambiar el dictamen a “Rechazado”, se dirige a la barra de navegación en el ítem “Aprobadas”, donde se despliega una tabla, donde cada fila es una solicitud aprobada, donde se incluye un botón para “Rechazar” la solicitud, además en el encabezado se muestra un hipervínculo el cual re direcciona a la pagina de validación de RFC ante el SAT, esta pagina se abre en una nueva pestaña, el siguiente fragmento de código representa lo anterior mencionado:

```
import React, {useEffect, useState} from 'react'
import {Table, Button, Col, Row} from "react-bootstrap"
import {handleSellerList, handleUpdateStatusSeller} from
"../../firebase/Actions"
import SatLink from "./SatLink"

function Approved() {
    const [items, setItems] = useState([])
    const [hasBeenModified, setHasBeenModified] = useState(false)

    useEffect(async() => {
        const result = await handleSellerList(0)
        setItems(result)
    }, [hasBeenModified])

    const OnPushButton = (button_pressed, seller_doc) => {
        handleUpdateStatusSeller(button_pressed, seller_doc)
        setHasBeenModified(!hasBeenModified)
    }

    return (
        <div className="container">
            <SatLink />
            <Table striped bordered hover size="sm">
                <thead>
                    <tr>
                        <th>Nombre</th>
                        <th>RFC</th>
                        <th>Dirección</th>
                        <th>Email</th>
                        <th>Teléfono</th>
                        <th>Acción</th>
                    </tr>
                </thead>
                <tbody>
                    {items.length > 0 && (
                        items.map((item) => (
                            <tr>
                                <td>{item.name}</td>
                                <td>{item.rfc}</td>
                                <td>{item.address}</td>
                                <td>{item.email}</td>
                                <td>{item.phone}</td>
                                <td>
                                    <div className="btn-group">
                                        <Button variant="warning" onClick={() =>
OnPushButton(2, item.id)}>Rechazar</Button>
```

```

        </div>
      </td>
    </tr>
  )
)
}

</tbody>
</Table>
</div>
)
}

export default Approved

```

El resultado de la codificación anterior se muestra a continuación, donde se muestra la ejecución de la pantalla de solicitudes aprobadas del administrador en versión web, esta pantalla es la implementación del diseño de la Figura 41. MockUp - Administrador, Selección de operación (Que se muestra como barra de navegación superior) y la Figura 43. MockUp - Administrador, Solicitudes aprobadas:

Nombre	RFC	Dirección	Email	Teléfono	Acción
La soberana bar Teotihuacán	JFNRKXKFRN78	Circuito Arqueológico, 55850 San Martín Centro, Méx., Mexico	lasobersnabarteo@gmail.com	5949582225	<button>Rechazar</button>
La puerta del sol	GNRNNNDGN7653	Av 16 de Septiembre Sur 62, San Martín Centro, 55850 San Martín Centro, Méx., Mexico	lapuertadelsol@gmail.com	5620976717	<button>Rechazar</button>
Asadero y parrilla La Chorcha	HFNFQJDBC89	Av Tuxpan No 149, 55850 San Martín Centro, Méx., Mexico	ayplc@gmail.com	5520776852	<button>Rechazar</button>
Luiggis	BFNRNFMDTB07	Calle 16 De Septiembre 7, San Martín Centro, 55850 San Martín Centro, Méx., Mexico	luiggis@gmail.com	5949585433	<button>Rechazar</button>

Figura 92. Pantalla de solicitudes aprobadas del administrador

5.2.15.5. Solicitudes rechazadas

Posterior a que el usuario administrador ha dictaminado por primera vez una solicitud, tiene la posibilidad de consultar todas aquellas solicitudes que ha dictaminado como rechazadas anteriormente y cambiar el dictamen a “Aprobado”, se dirige a la barra de navegación en el ítem “Rechazadas”, donde se despliega una tabla, donde cada fila es una solicitud rechazada, donde se incluye un botón para “Aprobar” la solicitud, además en el encabezado se muestra un hipervínculo el cual redirecciona a la pagina de validación de RFC ante el SAT, esta pagina se abre en una nueva pestaña, el siguiente fragmento de código representa lo anterior mencionado:

```
import React, {useEffect, useState} from 'react'
import {Table, Button, Col, Row} from "react-bootstrap"
import {handleSellerList, handleUpdateStatusSeller} from
"../../firebase/Actions"
import SatLink from "./SatLink"

function Rejected() {
    const [items, setItems] = useState([])
    const [hasBeenModified, setHasBeenModified] = useState(false)

    useEffect(async() => {
        const result = await handleSellerList(0)
        setItems(result)
    }, [hasBeenModified])

    const OnPushButton = (button_pressed, seller_doc) => {
        handleUpdateStatusSeller(button_pressed, seller_doc)
        setHasBeenModified(!hasBeenModified)
    }

    return (
        <div className="container">
            <SatLink />
            <Table striped bordered hover size="sm">
                <thead>
                    <tr>
                        <th>Nombre</th>
                        <th>RFC</th>
                        <th>Dirección</th>
                        <th>Email</th>
                        <th>Teléfono</th>
                        <th>Acción</th>
                    </tr>
                </thead>
                <tbody>
                    {items.length > 0 && (
                        items.map((item) => (
                            <tr>
                                <td>{item.name}</td>
                                <td>{item.rfc}</td>
                                <td>{item.address}</td>
                                <td>{item.email}</td>
                                <td>{item.phone}</td>
                                <td>
                                    <div className="btn-group">
```

```

        <Button variant="success" onClick={() =>
OnPushButton(1, item.id)}>Aprobar</Button>
            </div>
        </td>
    </tr>
)
)
)
}

</tbody>
</Table>
</div>
)
}

export default Rejected

```

El resultado de la codificación anterior se muestra a continuación, donde se muestra la ejecución de la pantalla de solicitudes rechazadas del administrador en versión web, esta pantalla es la implementación del diseño de la Figura 41. MockUp - Administrador, Selección de operación (Que se muestra como barra de navegación superior) y la Figura 44. MockUp - Administrador, Solicitudes rechazadas:

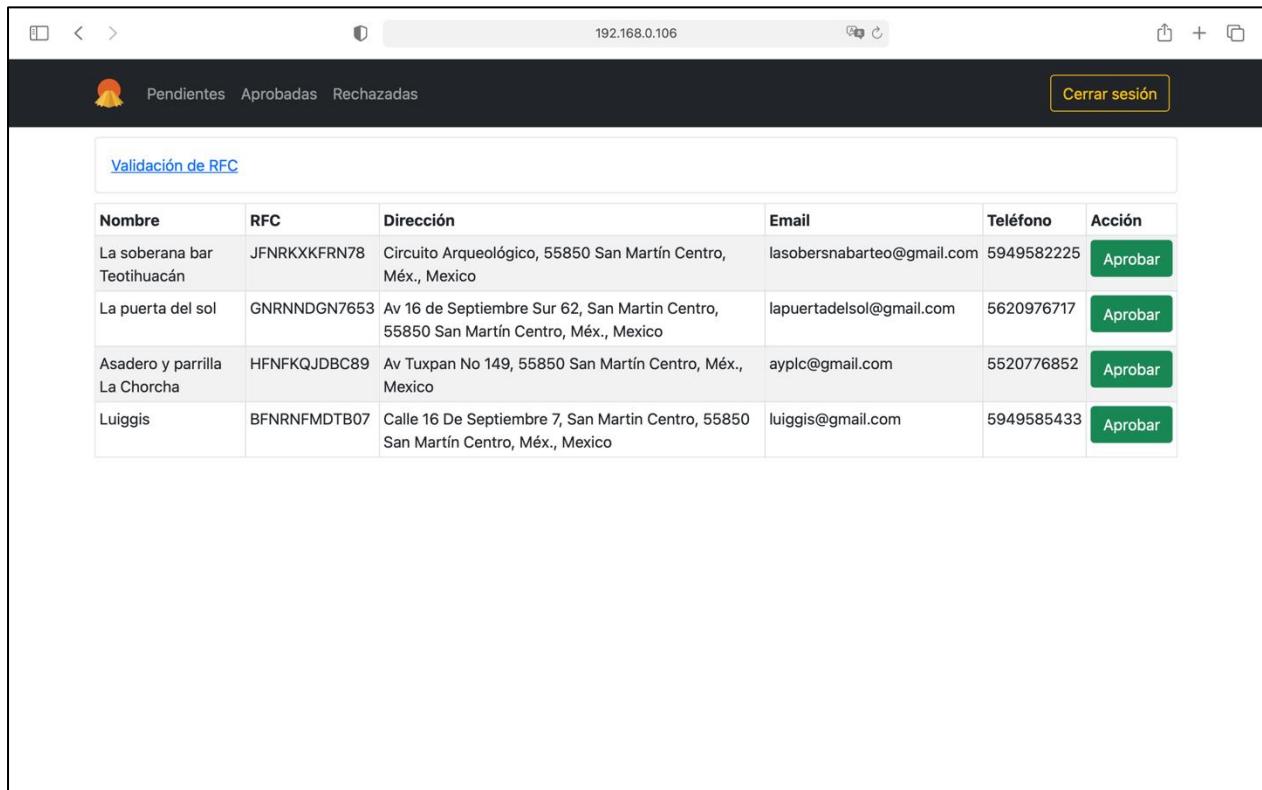


Figura 93. Pantalla de solicitudes rechazadas del administrador

5.3. Submódulo de idioma

5.3.1. Configuración

La biblioteca I18next requiere una única configuración básica que será utilizada en la traducción de toda la aplicación móvil, en esta configuración se inicializa la biblioteca, se define el idioma por defecto, los idiomas con los que va a trabajar y los recursos json de donde obtendrá las equivalencias entre cada idioma, el siguiente fragmento de código es la implementación de lo anterior:

```
import i18next from "i18next"

import spanish from "./spanish.json"
import english from "./english.json"

import {initReactI18next} from "react-i18next"

i18next
  .use(initReactI18next)
  .init({
    lng : 'en',
    keyboard: false,
    interpolation: {
      escapeValue: false
    },
    initImmediate : false,
    resources: {
      en: {
        translation: english
      },
      es : {
        translation: spanish
      }
    }
  })

export default i18next
```

Tal como se menciona arriba, se crean N archivos de equivalencia de idioma según tantos N idiomas se incorporen a la traducción, dada la naturaleza de este proyecto que solo se incluye el idioma inglés y español, se tendrán únicamente 2 archivos, a continuación, se muestra un fragmento del archivo de traducción del idioma español, este archivo tiene su versión espejo en idioma inglés:

```
{
  "login" : {
    "main_btn" : "Iniciar sesión",
    "email_input" : "Correo electrónico",
    "password_input" : "Contraseña",
    "create_account_lbl" : "¿No estás registrado?",
    "forgot_password_lbl" : "¿Olvidaste tu contraseña?",
    "create_account_btn" : "Crear cuenta",
    "reset_password_btn" : "Recuperar contraseña",
    "title_who_are_you_lbl" : "¿Quién eres?",
    "turist_option_btn" : "Turista",
    "seller_option_btn" : "Vendedor",
    "are_you_already_registered_lbl" : "¿Ya estas registrado?",
    "save_btn" : "Guardar",
```

```

        "cancel_btn" : "Cancelar",
        "register_turist": {
            "title" : "Se un turista 360°",
            "placeholder_name" : "Tu nombre",
            "placeholder_email" : "Correo electrónico",
            "placeholder_birthdate" : "Selecciona tu fecha de nacimiento",
            "placeholder_gender" : "Selecciona tu género",
            "placeholder_nationality" : "Selecciona tu nacionalidad",
            "placeholder_password" : "Contraseña",
            "placeholder_repeat_password" : "Confirma tu contraseña",
            "register_btn" : "Registrar"
        }
    }

{
    "login" : {
        "main_btn" : "Sign in",
        "email_input" : "Email",
        "password_input" : "Password",
        "create_account_lbl" : "Are you not registered?",
        "forgot_password_lbl" : "Forgot your password?",
        "create_account_btn" : "Create Account",
        "reset_password_btn" : "Reset your password",
        "title_who_are_you_lbl" : "Who are you?",
        "turist_option_btn" : "Turist",
        "seller_option_btn" : "Seller",
        "are_you_already_registered_lbl" : "Are you already registered?",
        "save_btn" : "Save",
        "cancel_btn" : "Cancel",
        "register_turist": {
            "title" : "Be a tourist 360°",
            "placeholder_name" : "Your name",
            "placeholder_email" : "Email",
            "placeholder_birthdate" : "Select your date of birth",
            "placeholder_gender" : "Select your gender",
            "placeholder_nationality" : "Select your country",
            "placeholder_password" : "Password",
            "placeholder_repeat_password" : "Confirm your password",
            "register_btn" : "Register"
        }
    }
}

```

5.3.2. Cambio de idioma desde login

Cuando la aplicación móvil se abre por primera vez se muestra la interfaz en idioma inglés, la primera pantalla que se muestra (Ver Figura 76. Pantalla de inicio de sesión en IOS y Android) tiene un botón en la parte inferior de la misma, la cual al presionarla se hace el cambio de idioma al correspondiente, el siguiente fragmento de código se implemente lo anterior:

```

import "../../src/language/i18n"
import { useTranslation, Trans } from 'react-i18next'

export default function LoginForm(props) {
    const [email, setEmail] = useState("");

```

```

const [password, setPassword] = useState("");
const { t, i18n } = useTranslation()

const changeLanguage = () => {

    if(i18n.language == "es"){
        i18n.changeLanguage("en")
    } else {
        i18n.changeLanguage("es")
    }
}

return (
    <View style={styles.container}>
        ...
        ...

        <View style={styles.btn_login}>
            <TouchableOpacity style={styles.btn_login_social}
                onPress={() => changeLanguage()}>
                <Icon
                    size={24}
                    type="material-community"
                    name = "google-translate"
                    color = "#FFFFFF"
                    background = "transparent"
                />
            </TouchableOpacity>
        </View>
    </View>
)
}

```

5.3.3. Cambio de idioma desde perfil

Una vez que la aplicación móvil se abre por primera vez, se registra o inicia sesión como usuario turista, es posible cambiar su idioma en cualquier momento, para lograrlo hay que presionar sobre “Perfil” o “Profile” según sea el idioma actual (Ver Figura 87. Pantallas de perfil de usuario vendedor y turista en IOS y Android), después sobre el ítem de idioma, el cual al presionarlo se hace el cambio de idioma al correspondiente, el siguiente fragmento de código se implemente lo anterior:

```

import { ListItem, Icon } from "react-native-elements"
import { map } from "lodash"
import "../../language/i18n"
import { useTranslation, Trans } from 'react-i18next'

export default function AccountOptions(props) {
    const { userInfo, userFirebase, toastRef, setReloadInfo } = props;
    const [showModal, setShowModal] = useState(false);
    const [renderComponent, setRenderComponet] = useState(null);

    const selectComponent = (key) => {
        switch (key) {
            case "Display name":
                ...
                break;

```

```

        case "Display country":
            ...
            break;

        case "Display gender":
            ...
            break;
        case "Display date of birth":
            ...
            break;
        case "Display password":
            ...
            break;
        case "Display language":
            setRenderComponet(
                <Text>Changing language</Text>
            )
            setShowModal(true);
            break;
        default:
            setRenderComponet(null);
            setShowModal(false);
            break;
    }
}

const menuOptions = generateOptions(userInfo, selectComponent);

return (
    <View style={styles.view_container}>
        {menuOptions.map((item, i) => (
            <ListItem key={i} bottomDivider onPress={item.onPress}>
                <Icon type="material-community" name={item.icon}/>
                <ListItem.Content>
                    <ListItem.Title>{item.title}</ListItem.Title>
                    <ListItem.Subtitle>{item.data}</ListItem.Subtitle>
                </ListItem.Content>
                <ListItem.Chevron type="material-community"/>
            </ListItem>
        ))}
        {renderComponent && (
            <Modal isVisible={showModal} setIsVisible={setShowModal}>
                {renderComponent}
            </Modal>
        )}
    </View>
)
}

function generateOptions(userInfo, selectComponent) {
    const { t, i18n } = useTranslation()

    const changeLanguage = () => {
        console.log(i18n.language)

        if(i18n.language == "es") {

```

```
        i18n.changeLanguage("en")
    } else {
        i18n.changeLanguage("es")
    }
}

return [
{
    ...
},
{
    title: t('profile.show.language_lbl'),
    data : i18n.language == "es" ? "Español" : "English" ,
    icon: "google-translate",
    rightIcon: "cached",
    onPress: () => changeLanguage(),
}
];
}
```

5.4. Submódulo de generación de código QR

En esta sección se muestra el desarrollo de la generación y lectura de códigos QR con el objetivo de agilizar el proceso de compra/venta que contempla el proyecto, los códigos QR serán generados/leídos por el usuario turista y vendedor en sus requerimientos correspondientes:

Tabla 23. Escenarios de lectura y generación de códigos QR

Usuario	Generar código QR	Leer código QR
Turista	Posterior al pago del servicio se mostrará en un comprobante de pago en forma de código QR. [a]	A partir de [b] se escanea el código QR, el cual le permite a la aplicación móvil del turista obtener los datos del pago a realizar. [c]
Vendedor	El usuario turista solicita la compra de servicio en la sucursal física y hay disponibilidad de este, entonces el vendedor mediante la aplicación móvil genera un código QR el cual es una solicitud de pago que incluye los datos para realizar el pago correspondiente. [b]	Cuando un turista llega a la sucursal correspondiente, muestra su comprobante de pago [a], donde la aplicación del vendedor compara algunos de los datos recabados en la lectura de código QR, respecto con la consulta de sus servicios y transacciones, por ejemplo: fechas, identificador de transacción, costo, identificador del turista, entre otros datos. [d]

5.4.1. Generación de comprobante de pago – Turista

Un turista puede comprar/reservar cualquier servicio turístico, posterior a ello, consultar su compra y dentro de esta compra se muestran los datos más relevantes y un código QR que incluye la información más importante de la misma, esto es lo siguiente:

```
const object_to_validate = {
    "creationAt": route.params.shopping.creationAt,
    "people": route.params.shopping.people,
    "reservation_date_time": route.params.shopping.reservation_date_time,
    "checkin": "llegada",
    "checkout": "salida",
    "seller_id": route.params.shopping.seller_id,
    "service_id": route.params.shopping.service_id,
    "status": route.params.shopping.service_id,
    "total": route.params.shopping.total,
    "turist": route.params.shopping.turist,
    "id": route.params.shopping.id
}
```

Se incluye esta información en el código QR con el objetivo que cuando se escanee dicho comprobante de pago por el usuario vendedor, cargue inmediatamente dicha información y no pasar por una consulta de información en firebase. En el siguiente fragmento de código se implementa la generación de código QR:

```

export default function Receipt(props) {
  const {route, navigation} = props
  const { creationAt, name_sale, people, seller_id, service_id, status, total,
  tourist, unit_price, tourist_name, type, id } = route.params.shopping

  const ReviewMarkButton = (props) =>{
    const {t} = props

    return (
      <Button
        title={t('mytrip.receipt.rate_review_btn')}
        containerStyle={styles.viewBtnContainerReview}
        buttonStyle={styles.viewBtnReview}
        onPress={() => setShowModal(true)}
      />
    )
  }

  const ShoppingHostingInfo = (props) => {
    const {t} = props

    return (
      <Card>
        <Card.Title>{name_sale}</Card.Title>
        <Card.Divider style={styles.divider_card}/>
        <View style={styles.container_qr}>
          <Text
            style={styles.txt_header_card_item}>{t('mytrip.receipt.qr_title')}</Text>
          <QRCode
            value = {JSON.stringify(object_to_validate)}
            size= {windowWidth * 0.85 }
            enableLinearGradient ={true}
            linearGradient={[ "#EAA11B", "#404040"]}
            gradientDirection={[0,0,0,1]}
          />
        </View>
        <Card.Divider style={styles.divider_card}/>
        <View>
          <Text
            style={styles.txt_header_card_item}>{t('mytrip.receipt.turist_title')}</Text>
          <Text
            style={styles.txt_item}>{t('mytrip.receipt.turis_name_lbl')}
            {tourist_name}</Text>
          </View>
          <Card.Divider style={styles.divider_card}/>
          <View>
            <Text
              style={styles.txt_header_card_item}>{t('mytrip.receipt.dates_title')}</Text>
              <Text style={styles.txt_item}>
                {t('mytrip.receipt.date_purchase_lbl')}

```

```

        route.params.shopping.checkin_day.toDate().toISOString().substring(0, 10) }
    </Text>
        <Text
            style={styles.txt_item}>{t('mytrip.receipt.check_out_date_lbl')} {
        route.params.shopping.checkout_day.toDate().toISOString().substring(0, 10) }
    </Text>
        <Text
            style={styles.txt_item}>{t('mytrip.receipt.total_days_lbl')} {Math.abs(route.params.shopping.checkout_day.toDate() - route.params.shopping.checkin_day.toDate())/(1000 * 3600 * 24)}</Text>
        </View>
        <Card.Divider style={styles.divider_card}/>
        <View>
            <Text
                style={styles.txt_header_card_item}>{t('mytrip.receipt.places_total_title')}</Text>
            <Text
                style={styles.txt_item}>{t('mytrip.receipt.places_number_lbl')} {people}</Text>
            <Text
                style={styles.txt_item}>{t('mytrip.receipt.unit_price_lbl')} ${unit_price}</Text>
            <Text
                style={styles.txt_item}>{t('mytrip.receipt.total_price_lbl')} ${total}</Text>
            </View>
            <Card.Divider style={styles.divider_card}/>
            <View style={{alignItems: "center"}}>
                <Text style={styles.txt_header_card_item}>Status</Text>
                <Image
                    style={styles.image}
                    resizeMode="cover"
                    source={...}
                />
            </View>

            <Card.Divider style={styles.divider_card}/>
            <View style={{alignItems: "center"}}>
                <Text
                    style={styles.txt_header_card_item}>{t('mytrip.receipt.review_title')}</Text>
                    {completeReview ?
                        <Text>{t('mytrip.receipt.review_yes')}</Text>
                        : <Text>{t('mytrip.receipt.review_no')}</Text>
                    }
                    <ReviewMarkButton t={t}/>
            </View>
        </Card>
    )
}

return (
    <ScrollView style={styles.view_container}>
        {type === "hosting" && <ShoppingHostingInfo t={t}/>}
        {type !== "hosting" && <ShoppingOtherInfo t={t}/>}

        <Modal isVisible={showModal} setIsVisible={setShowModal}>
            <ReviewMark
                id_service ={service_id}
                setShowModal={setShowModal}
                completeReview = {completeReview ? completeReview : null}

```

```

        setCompleteReview = {setCompleteReview}
        hasBeenModifiedReview = {hasBeenModifiedReview}
        setHasBeenModifiedReview = {setHasBeenModifiedReview}
    />
  </Modal>
</ScrollView>
}

}

```

El resultado de las codificaciones anteriores se muestra a continuación, donde se observa la ejecución de las pantallas de perfil del usuario vendedor y turista en sistemas operativos IOS y Android respectivamente, estas pantallas son la implementación del diseño de la Figura 24. Mockup Comprobante de pago para servicio.

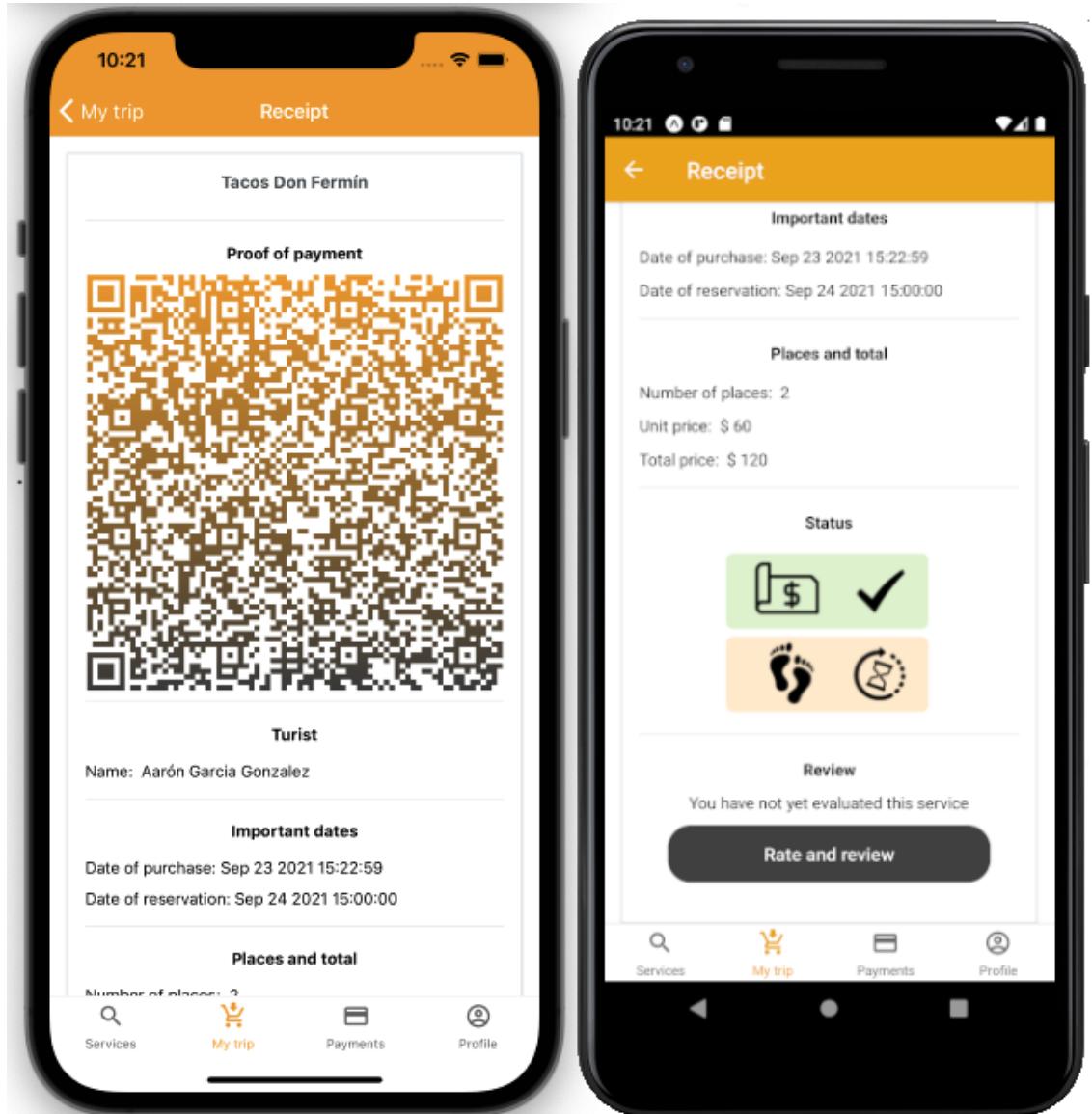


Figura 94. Pantalla de comprobante de pago - turista en IOS y Android

5.4.2. Lectura de comprobante de pago – Vendedor

Cuando un vendedor recibe a un turista en sus instalaciones y el turista solicita acceso debido que ya ha realizado su compra previamente mediante la aplicación, el vendedor requerirá poder autenticar dicha compra, es por ello que le solicitará el comprobante de pago (Figura 36. Mockup Escanear comprobante de pago realizado), de ser correcto, el vendedor otorgará el acceso, o bien, en caso de no ser valido, lo indicará la aplicación. En el siguiente fragmento de código se implementa la generación de código QR:

```
export default function Receipt(props) {
  const ReviewMarkButton = (props) =>{
    const {t} = props

    return (
      <Button
        title={t('mytrip.receipt.rate_review_btn')}
        containerStyle={styles.viewBtnContainerReview}
        buttonStyle={styles.viewBtnReview}
        onPress={() => setShowModal(true)}
      />
    )
  }

  const ShoppingHostingInfo = (props) => {
    const {t} = props

    return (
      <Card>
        <Card.Title>{name_sale}</Card.Title>
        <Card.Divider style={styles.divider_card}>
        <View style={styles.container_qr}>
          <Text
            style={styles.txt_header_card_item}>{t('mytrip.receipt.qr_title')}</Text>
          <QRCode
            value = {JSON.stringify(object_to_validate)}
            size= {windowWidth * 0.85 }
            enableLinearGradient ={true}
            linearGradient={[ "#EAA11B", "#404040"]}
            gradientDirection={[0,0,0,1]}
          />
        </View>
        <Card.Divider style={styles.divider_card}>
        <View>
          <Text
            style={styles.txt_header_card_item}>{t('mytrip.receipt.turist_title')}</Text>
          <Text
            style={styles.txt_item}>{t('mytrip.receipt.turis_name_lbl')}
```

```

        </Text>
        <Text
style={styles.txt_item}>{t('mytrip.receipt.check_in_date_lbl')} {
route.params.shopping.checkin_day.toDate().toISOString().substring(0, 10) }
</Text>
        <Text
style={styles.txt_item}>{t('mytrip.receipt.check_out_date_lbl')} {
route.params.shopping.checkout_day.toDate().toISOString().substring(0, 10) }
</Text>
        <Text
style={styles.txt_item}>{t('mytrip.receipt.total_days_lbl')}
{Math.abs(route.params.shopping.checkout_day.toDate() -
route.params.shopping.checkin_day.toDate())/(1000 * 3600 * 24)}</Text>
        </View>
        <Card.Divider style={styles.divider_card}/>
        <View>
            <Text
style={styles.txt_header_card_item}>{t('mytrip.receipt.places_total_title')}</Text>
            <Text
style={styles.txt_item}>{t('mytrip.receipt.places_number_lbl')}
{people}</Text>
            <Text
style={styles.txt_item}>{t('mytrip.receipt.unit_price_lbl')} ${unit_price}</Text>
            <Text
style={styles.txt_item}>{t('mytrip.receipt.total_price_lbl')} ${total}</Text>
            </View>
            <Card.Divider style={styles.divider_card}/>
            <View style={{alignItems: "center"}}>
                <Text style={styles.txt_header_card_item}>Status</Text>
                <Image
                    style={styles.image}
                    resizeMode="cover"
                    source={}
                />
            </View>

            <Card.Divider style={styles.divider_card}/>
            <View style={{alignItems: "center"}}>
                <Text
style={styles.txt_header_card_item}>{t('mytrip.receipt.review_title')}</Text>

                {completeReview ?
                    <Text>{t('mytrip.receipt.review_yes')}</Text>
                    : <Text>{t('mytrip.receipt.review_no')}</Text>
                }
                <ReviewMarkButton t={t}/>
            </View>
        </Card>
    )
}

return (
    <ScrollView style={styles.view_container}>
        {type === "hosting" && <ShoppingHostingInfo t={t}/>}
        {type !== "hosting" && <ShoppingOtherInfo t={t}/>}

        <Modal isVisible={showModal} setIsVisible={setShowModal}>
            <ReviewMark

```

```

        id_service ={service_id}
        setShowModal={setShowModal}
        completeReview = {completeReview ? completeReview : null}
        setCompleteReview = {setCompleteReview}
        hasBeenModifiedReview = {hasBeenModifiedReview}
        setHasBeenModifiedReview = {setHasBeenModifiedReview}
    />
</Modal>
</ScrollView>

}

}

```

El resultado de las codificaciones anteriores se muestra a continuación, donde se observa la ejecución de las pantallas de perfil del usuario vendedor y turista en sistemas operativos IOS y Android respectivamente, estas pantallas son la implementación del diseño de la Figura 36. Mockup Escanear comprobante de pago realizado.

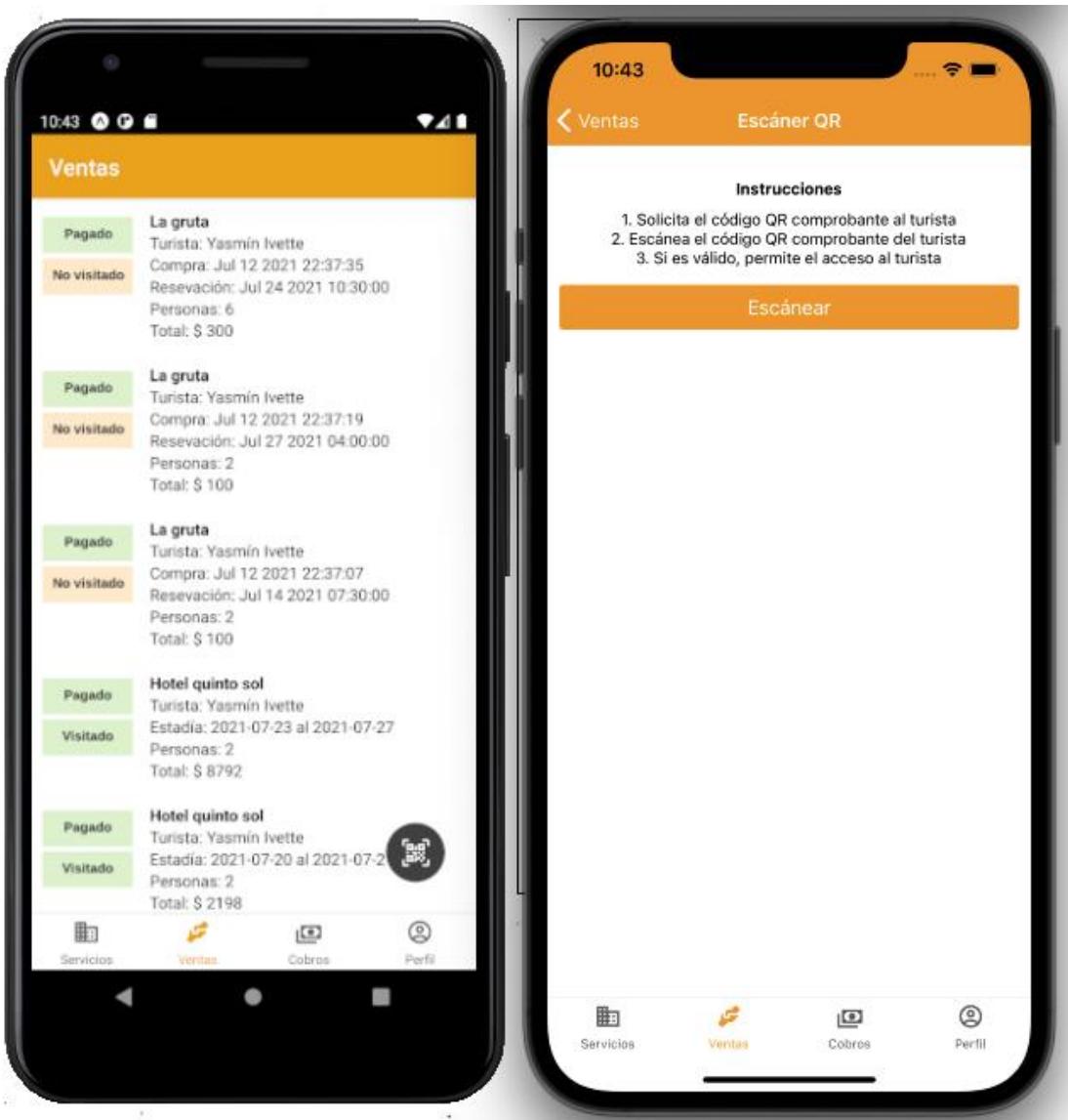


Figura 95. Pantalla de escaneo de comprobante de pago - vendedor en IOS y Android

5.4.3. Generación de orden de pago – Vendedor

Cuando un turista por alguna razón no ha comprado un servicio y se dirige a un establecimiento para adquirirlo, el turista tendrá 2 opciones: usar su aplicación y adquirir el servicio o bien, indicarle al vendedor sus necesidades, donde el vendedor generará una orden de pago mediante un código QR que incluye los datos necesarios, estos datos son los siguientes:

```
const object_to_validate = {
    "creationAt": new Date(),
    "reservation_date_time": new Date(moment(new Date(checkIn).getTime()).add(reservationHourToMinutes(reservationHour), "minutes")),
    "checkin_day": checkIn,
    "checkout_day": checkOut,
    "seller_id": getUser().uid,
    "service_id": serviceSelected.id,
    "status": 1,
    "total": total,
}
```

Se incluye esta información en el código QR con el objetivo que cuando se escanee dicho comprobante de pago por el usuario vendedor, cargue inmediatamente dicha información y no pasar por una consulta de información en firebase. En el siguiente fragmento de código se implementa la generación de código QR:

```
export default function Charge() {
    ...
    return (
        <ScrollView containerStyle={styles.general_container}
style={styles.style_container}>
            <View style={styles.container}>
                <Text style={styles.text}>Genera tu orden de pago</Text>

                <Input
                    placeholder = "Selecciona servicio"
                    editable={false}
                    autoCapitalize='none'
                    autoCompleteType='off'
                    containerStyle = {styles.input}
                    leftIcon ={{{
                        type: "material-community",
                        name: "store-outline",
                        color: "#404040",
                    }}}
                    rightIcon = {{{
                        size: 30,
                        type: "material-community",
                        name: "gesture-tap",
                        color: "#404040",
                        onPress: () => {
                            setRenderComponet(<SelectService
                                myServices={myServices}
                                setShowModal={setShowModal}
                                serviceSelected={serviceSelected}
                                setServiceSelected={setServiceSelected}
                                setServiceName={setServiceName}
                            />)
                        }
                    }}}
                </Input>
            </View>
        </ScrollView>
    )
}
```

```

        setShowModal(true)
        clearSelectionModal()
    }
}
value={serviceName}
/>

{serviceName ?
<View style={styles.container_optionals}>
    {serviceSelected.type.includes("hotel") ?
        <View>
            <Input
                containerStyle = {styles.input}
                placeholder = {'Selecciona Check In'}
                editable={false}
                autoCompleteType='off'
                leftIcon ={{{
                    type: "material-community",
                    name: "ray-start-arrow",
                    color: "#404040",
                }}}
                rightIcon = {{{
                    type: "material-community",
                    name: "calendar-month-outline",
                    color: "#404040",
                    onPress: () => {
                        setRenderComponet(
                            <CalendarSelection
                                setShowModal={setShowModal}
                                    setCheck={setCheckIn}
                                    t={t}
                                />
                            )
                        setShowModal(true)
                    }
                }}}
                value = {checkIn}
            />
            <Input
                containerStyle = {styles.input}
                placeholder = {'Selecciona Check Out'}
                editable={false}
                autoCompleteType='off'
                leftIcon ={{{
                    type: "material-community",
                    name: "ray-end-arrow",
                    color: "#404040",
                }}}
                rightIcon = {{{
                    type: "material-community",
                    name: "calendar-month-outline",
                    color: "#404040",
                    onPress: () => {
                        setRenderComponet(
                            <CalendarSelection
                                setShowModal={setShowModal}
                                    setCheck={setCheckOut}

```

```

        t={t}
    />
)
setShowModal(true)
}
}
value = {checkOut}
/>
</View>:

```

El resultado de las codificaciones anteriores se muestra a continuación, donde se observa la ejecución de las pantallas de perfil del usuario vendedor y turista en sistemas operativos IOS y Android respectivamente, estas pantallas son la implementación del diseño de la Figura 37. Mockup Generar orden de pago para servicio de hotel.

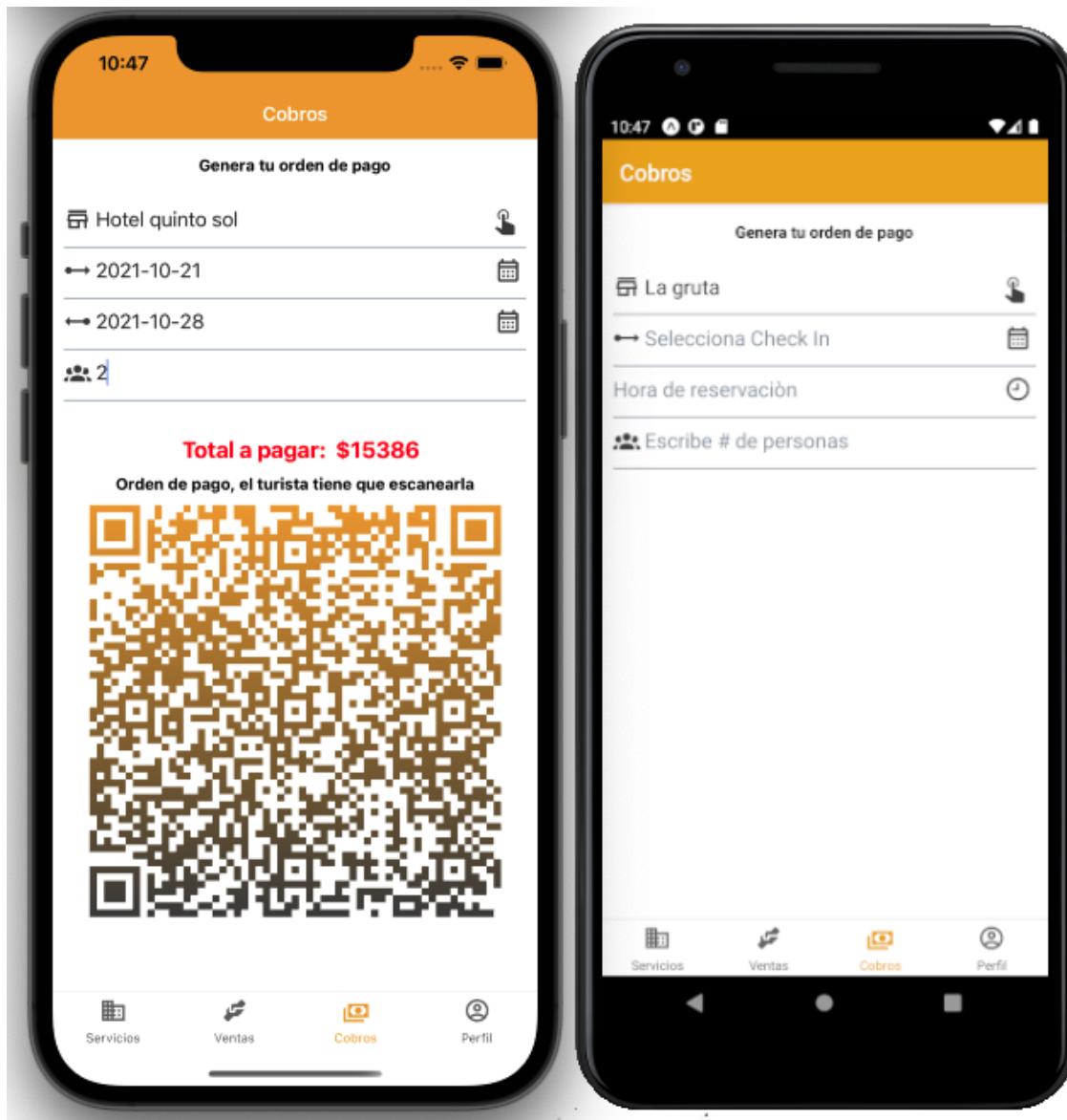


Figura 96. Pantalla de generación de orden de pago - vendedor en IOS y Android

5.4.4. Lectura de orden de pago – Turista

En este escenario se basa en la generación de orden de pago que realiza el usuario vendedor, el usuario tendrá la posibilidad de escanear el QR que el vendedor le muestra, a partir de ello, el usuario turista podrá ver los detalles de pago y a partir de esto, decidir continuar con el pago y, por ende, la compra. En el siguiente fragmento de código se implementa la lectura de este código QR:

```
useEffect(() => {
    //const axu = await()

    if(serviceValue) {
        if(user === scannedValue.seller_id) {
            if(serviceValue.service_id === scannedValue.service_id) {
                if(serviceValue.status === 2){
                    setMessage(messages.pagado_no_visitado)
                } else if(serviceValue.status === 1){
                    setMessage(messages.pagado_visitado)
                } else {
                    setMessage(messages.no_pagado_no_visitado)
                }
            } else {
                setMessage(messages.otro_servicio_propio)
            }
        } else {
            setMessage(messages.no_coincidencias_servicios)
        }
    }
}, [serviceValue])

useEffect(() => {

    Alert.alert(
        message.title,
        message.content,
        [
            {
                text: 'OK',
                //onPress: () =>
                setGoBackFromQRStatus(!goBackFromQRStatus)
            },
        ]
    )
}, [message])

const handleBarCodeScanned = ({ type, data }) => {
    setDefaultQRData() // setea el escaner
    setScannedValue(JSON.parse(data)) // obtiene y setea el objeto leido
};

const setDefaultQRData = () => {
    setScanned(false)
    setOpenScanner(false)
}
```

El resultado de las codificaciones anteriores se muestra a continuación, donde se observa la ejecución de las pantallas de perfil del usuario vendedor y turista en sistemas operativos IOS y Android respectivamente, estas pantallas son la implementación del diseño de la Figura 25. MockUp Pago de servicio presencial.

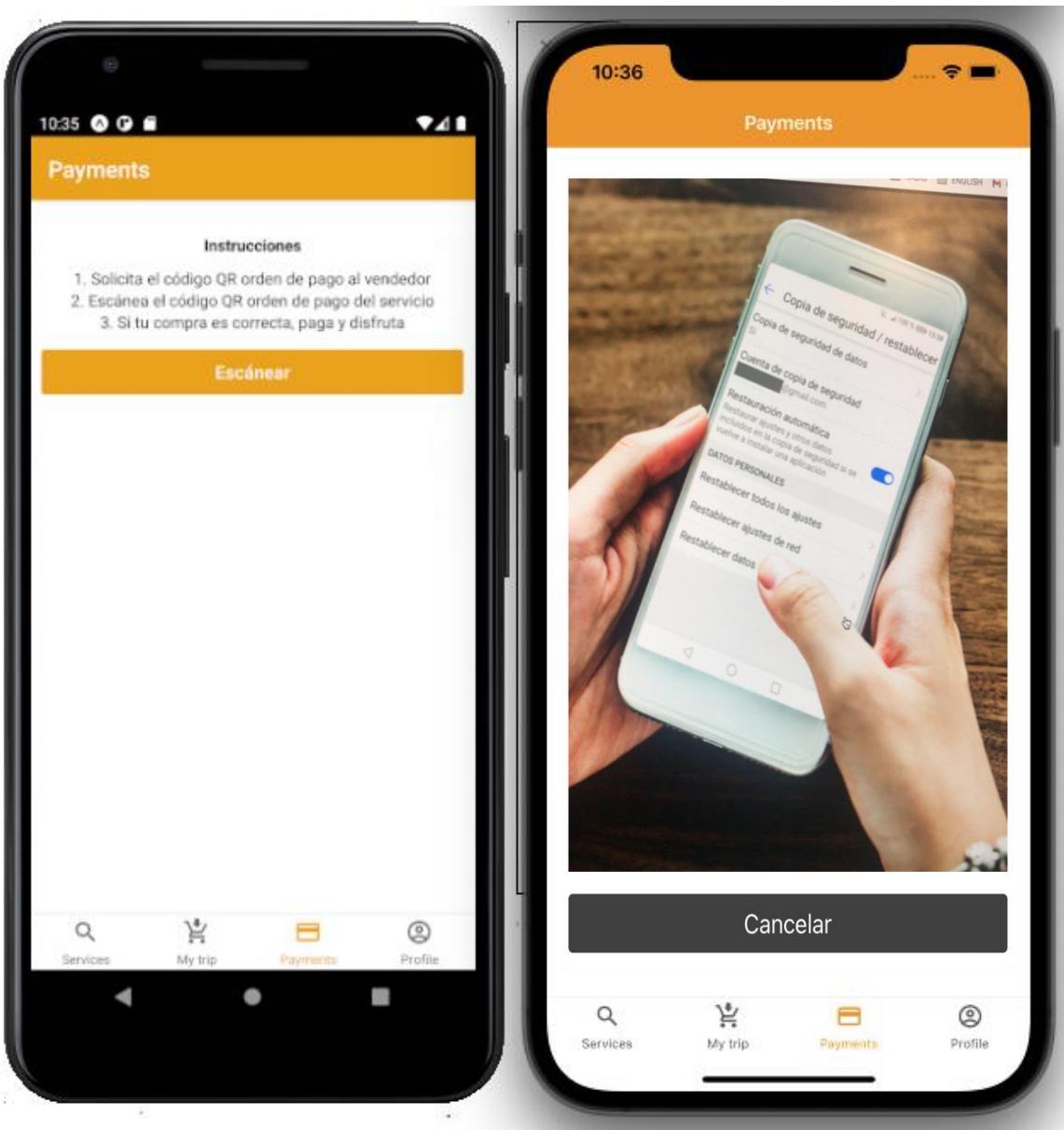


Figura 97. Pantallas de Lectura de Orden de pago - turista en IOS y Android

5.5. Submódulo de pagos

Finalmente, en esta sección se aborda el último modulo de desarrollo, se hace hincapié en el modulo anterior “Generación/lectura de código QR” ya que es la continuación y la fuente de datos para realizar la transacción monetaria.

Un pago electrónico puede hacerse desde los siguientes 2 escenarios:

1. El turista busca un servicio, selecciona el de su preferencia y decide adquirirlo, entonces procede al pago.
2. El turista acude físicamente a la sucursal de un servicio determinado, le solicita al vendedor que le genere una orden de pago, el vendedor genera la orden de pago y la muestra en forma de código QR, posteriormente el turista escanea el código y corrobora los datos de su pedido, entonces de ser correcto procede con el pago.

Estos 2 escenarios son diferentes hasta el punto en que el turista ve el botón de pagar, ya que, a partir de presionar el botón de pagar, el proceso que se sigue para efectuar el pago es el mismo en ambos, es por ello que se realiza una única codificación en forma de componente, el cual es posible reutilizar tantas veces como se quiera.

5.5.1. Obtención de credenciales para cuenta de developer

Como primer punto se crea la aplicación desde el dashboard de desarrollo, el nombre se genera en automático, por lo que hay que seleccionar el país/region, el tipo de aplicación (Como vendedor o como tienda, se elige tienda), y presionar en crear app.

Create New App

Before you create your new app, let us know what kind of solution you're looking for.

Application Details

App Name ?

Platform Partner App

App Type

Merchant – Accept payments as a merchant (seller)

Platform – Move payments to sellers as a platform (marketplace, crowdfunding, or e-commerce platform)

Country / Region

United States

Figura 98Crear App en paypal

Una vez creada la aplicación se puede observar la comprobación de esta en la siguiente pantalla:

My apps & credentials

Sandbox Live

REST API apps

Get started by clicking **Create App**. **PayPal Commerce Platform for Business** users can get started quickly by using the **Default Application** credentials to test PayPal REST APIs in our sandbox.

App name	Actions
Default Application	System generated, no actions available.
Platform Partner App - 4811676593306681281	trash

Create App

Figura 99. App paypal creada correctamente

Las siguientes claves son vitales para la conexión desde react native con el api de PayPal en forma develop, donde el sandbox account es el correo que usaremos a modo de prueba, el cliente id es el identificador de cliente que se ocupa en múltiples operaciones que se detallan mas adelante.

SANDBOX API CREDENTIALS

Sandbox account
sb-g...@business.example.com

Client ID
AVi5UMjNXzzmf2KZHjh6Df...OG-YZt

Secret
[Hide](#)

Note: When you generate a new secret, you still maintain the original secret. The maximum number of client secrets is two. A client secret is either in enabled or disabled state.

Created	Secret	Status	Action
Nov 15, 2021	EKPIvoAq73bSYf9... WxVhGLbVrttgdtN...	Enabled	...

Generate new secret

Figura 100. Claves y accesos paypal develop

5.5.2. Lógica del proceso

Se usará el api rest de paypal para la implementación de pagos, el proceso descrito en su página web es el siguiente:

1. Creación el objeto de pago y el método de pago
2. Elegir URL's de cancelación y retorno
3. Declarar detalles de pago
4. Declarar cantidad
5. Crear transacción
6. Añadir detalles de pago y declarar como intento de autorización
7. Crear el contexto mediante el cliente id y el secret id
8. Crear pago y obtener el id de pago
9. Establecer el id de pago al objeto de ejecución
10. Ejecutar pago y obtener autorización

La función principal es la siguiente, la cual llama a la función “createOrder(order)”, la cual recibe como argumento al objeto de pago a realizar el mismo, posteriormente redirige a la aplicación web de paypal para ejecutar el pago:

```
export async function pay(data_order) {
  const url = await createOrder(data_order)

  Linking.openURL(url);
  //executePayment(result["id"])
}
```

A continuación, se muestra la codificación de la función que crea la orden de pago:

```
async function createOrder(data_order) {
  const {service_name, total} = data_order
  const redirectUrl = Linking.createURL('')
  const token = await getToken()

  let result = await fetch(PAYPAL_API_SANBOX+'/v2/checkout/orders',
  {
    method: 'POST',
    headers: {
      'Accept': 'application/json',
      'Content-Type': 'application/json',
      'Authorization': `Bearer ${token}`
    },
    body: JSON.stringify({
      intent: 'CAPTURE',
      purchase_units: [
        {
          amount: {
            currency_code: "MXN",
            value: total
          },
          description: 'Servicio Compra-Venta'
        }
      ],
      application_context: {
        user_action: 'PAYMENT',
       急行：上記のコードを修正して、`急行`と`急行`を削除する。
        payment_method: 'IMMEDIATE_PAYMENT_REQUIRED'
      }
    })
  }
}
```

```

        brand_name : 'TeotiApp',
        landing_page : 'NO_PREFERENCE',
        user_action : 'PAY_NOW',
        return_url: redirectUrl, // Url despues de realizar el pago
        cancel_url: redirectUrl // Url despues de realizar el pago
    }
}
)
.then((response) => {
    //console.log(response);
    return response.json()
})
.then((responseJson) => {
    //console.log(responseJson);
    return responseJson
})
.catch((response) => {
    console.log(response);
})
)

return result["links"][1]["href"]
}

```

La función que codificada para la obtención del token es la siguiente, misma donde se usa la versión 2 del api, también el cliente y el secret id descrito anteriormente.

```

export async function getToken() {
    console.log("getting token");

    let result = await fetch('https://api-m.sandbox.paypal.com/v1/oauth2/token',
    {
        method: 'POST',
        headers: {
            'Accept': 'application/json',
            'Accept-Language': 'en_US',
            'Content-Type': 'application/x-www-form-urlencoded',
            'Authorization': 'Basic ' + btoa(CLIENT + ':' + SECRET)
        },
        body: 'grant_type=client_credentials'
    })
    .then((response) => {
        return response.json()
    })
    .then((responseJson) => {
        return responseJson["access_token"]
    })
    .catch((error) => {
        console.error(error);
    });
}

return result;
}

```

Finalmente, la función que ejecuta el pago, esto debido a que ya fue creada y autorizado el pago, entonces se puede proceder:

```

export async function executePayment(id) {
    console.log("Authorizing payment");

```

```

let url = 'https://api-
m.sandbox.paypal.com/v2/checkout/orders/'+id+'/authorize'

console.log("url: " + url);

let result = await fetch('https://api-
m.sandbox.paypal.com/v2/checkout/orders/'+id+'/authorize',{
  method: 'POST',
  headers: {
    'Content-Type': 'application/json',
    'Authorization': 'Basic ' + btoa(CLIENT + ':' + SECRET)
  },
})
.then((response) =>{
  return response.json()
})
.then((responseJson)=>{
  console.log(responseJson);
  return responseJson
})
.catch((error) => {
  console.log(error);
})
}
}

```

A continuación, se muestra la pantalla donde está el botón de pagar después de llenar la información requerida, en este caso para un servicio de hospedaje, al presionar sobre este, se redirige al navegador por defecto, en caso de tener una cuenta de paypal asociada se tomará el login correspondiente:

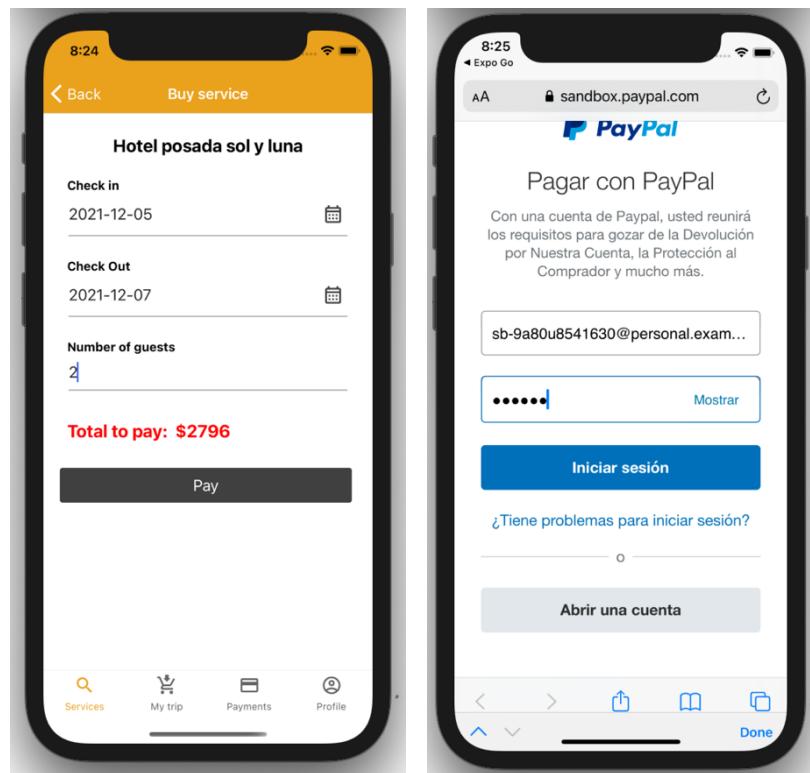


Figura 101. Redirección y login en Paypal

Posteriormente al hacer login en paypal, se muestran las siguientes pantallas, la que esta a la izquierda muestra que esta cargando y preparando los datos para realizar el pago que se muestra en la pantalla del medio, donde se puede apreciar el total y el nombre del servicio a comprar, y al dar clic en “Pagar ahora” se realiza el pago, se muestra la pantalla de la derecha, donde indica mediante la leyenda “Gracias por usar ¡Payapal!” y un modal que indica que se abrirá la aplicación “TeotiApp”, es decir la aplicación que la envió a Paypal, y los datos de pago de igual manera se almacenan en la base de datos de la aplicación.

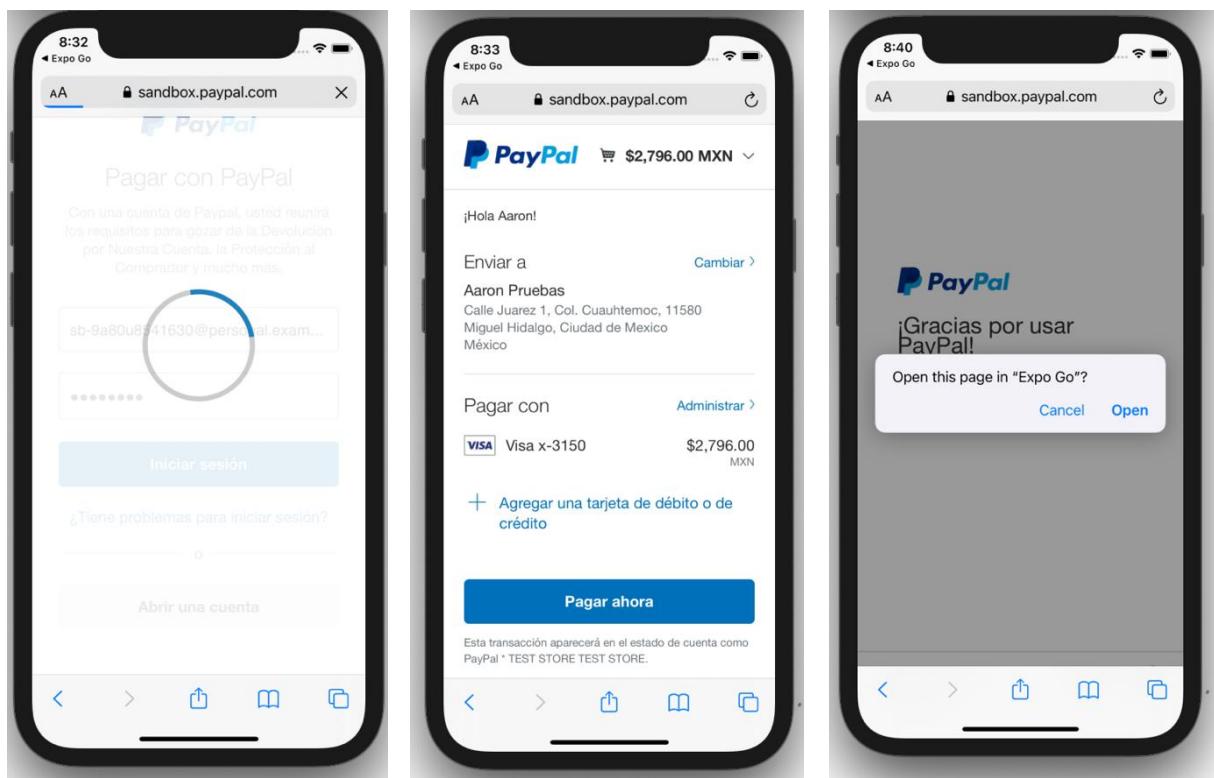


Figura 102. Pago en Paypal

6. Pruebas

6.1. Persistencia de datos

Desde la interfaz de google firebase, se muestran a continuación algunos ejemplos más representativos del almacenamiento de los datos que se obtienen a partir de los múltiples clientes (usuario administrador, turista y vendedor).

En la colección “users” se asigna de manera automática un identificador para cada documento que se agrega a la misma, en esta colección se almacenan todos los usuarios que incluye el sistema, es decir el administrador, turistas y vendedores, a continuación, se muestra el documento para el usuario vendedor cuyo correo electrónico es “rbicisteo@gmail.com”:

tteotiapp > users > 13e005p9KIMB...		
+	users	13e005p9KIMBaiCyCZ0IeZ2A5lm2
+ Iniciar colección	+ Agregar documento	+ Iniciar colección
reviews	13e005p9KIMBaiCyCZ0IeZ2A5lm2 >	+ Agregar campo
roles	1NrKqjFtDcVSzEEbkL4RfLUCjbv1	address: "México 132, Estado de México, Mexico"
services	5A8ID1W1TFY8YmqqkgLBV0f1Ak2	creation_date: 20 de marzo de 2021, 17:14:47 UTC-6
transactions	7oMPPE0m6nP49f4NRbszC3srjn1	email: "rbicisteo@gmail.com"
users >	81CU04UD67SEGJEFaEWRihPDae82	locationSeller
	9qAyndfkGufmtoyUiKwxKT2jEaG3	latitude: 19.68447930163126
	C24N3EQRqlYDfen3RV03BAMY3Pu2	latitudeDelta: 0.005932174563227477
	CaW6w9Fuy5aVlqinNwlq0i67VWW2	longitude: -98.85269991117754
	Csn1K3tQ0tSXOaNRGd0fMxxsY8W2	longitudeDelta: 0.005099053933037112
	DFKNxL08DnWs9D6EtB6xnspd2102	name: "Renta de bicicletas"
	JzJq5tVAm6Sa6R1WyzPJuEEzgM42	phone: "5521727420"
	KWBFCkVN6sXnhov2R31Jn6jyz0g1	rfc: "JFNRCNCIDK683"
	OWS2DkShxJUVziQPQ40vsfPv47B2	role: "seller"
	RaJKiGnS1VeChvX9na7ZK0d1S02	

Figura 103. Persistencia de datos, usuarios

En la colección “roles” se almacenan todos los roles del sistema, esta colección funciona como un catálogo, es la colección más sencilla que incluye el sistema, a continuación, se muestra el documento para el rol vendedor:

tteotiapp > roles > 3		
+	roles	3
+ Iniciar colección	+ Agregar documento	+ Iniciar colección
reviews	1	+ Agregar campo
roles >	2	name: "turist"
services	3 >	
transactions		
users		

Figura 104. Persistencia de datos, roles

En la colección “services” se asigna de manera automática un identificador para cada documento que se agrega a la misma, en esta colección se almacenan todos los servicios que los usuarios vendedores registran en la aplicación móvil sin importar el tipo de servicio que se registra (consumo, hospedaje u otros), a continuación, se muestra el documento para el servicio “Tacos Don Fermin”:

tteotiapp	services	05FZmq9R0XospI5R8b2D
+ Iniciar colección	+ Agregar documento	+ Iniciar colección
reviews	05FZmq9R0XospI5R8b2D >	+ Agregar campo
roles	5IFSjLGRkNqV6B9I29YA	availability: "8"
services >	50knKjkG5an52Rnqxk8c	createBy: "aJJPLAz0FNpVeFwiqSZpSar80N2"
transactions	kpsRgh8hS3adDYGIVVyQ	creationAt: 28 de mayo de 2021, 12:35:36 UTC-5
users	qSWSvBp2HwL41DT4WoMT	description: "Tacos de pastor, carnes rojas, bebidas, música y lugar tranquilo, en el corazon de Otumba."
	safrDBPA6pPI6QlU2edH	► images: [https://firebasestorage....]
		isOpenOnWeekDays: false
		isOpenOnWeekends: true
		► location: {latitude: 19.699535288122...}
		name: "Tacos Don Fermin"
		price: "60"
		quantity_voting: 0
		rating: 0

Figura 105. Persistencia de datos, servicios

En la colección “reviews” se asigna de manera automática un identificador para cada documento que se agrega a la misma, en esta colección se almacenan todos los review, a continuación, se muestra el documento cuyo rating es 3 para el usuario “ZHc2sHbhkGYptOuEbtjb6S3GIAV2” (este edificador es únicamente conocido de manera interna en la aplicación, es decir el usuario final jamás tendrá esta información):

tteotiapp	reviews	bxqzp0ACGeG80QltyqT6
+ Iniciar colección	+ Agregar documento	+ Iniciar colección
reviews >	bxqzp0ACGeG80QltyqT6 >	+ Agregar campo
roles	mn9x0E4uFrRvW9m0R6NP	avatar_user: "https://firebasestorage.googleapis.com/v0/b/tteotiapp.appspot.com/..., alt=media&token=169869af-23eb-4235-8a9e-fa63d04581e1"
services	srvfJAMMP8mHULEx9qsS	createAt: 29 de julio de 2021, 16:46:28 UTC-5
transactions		id_service: "kpsRgh8hS3adDYGIVVyQ"
users		id_user: "ZHc2sHbhkGYptOuEbtjb6S3GIAV2"
		rating: 3
		review: "Bonito lugar acogedor y limpio. El servicio un poco tardado pero bueno lo recomiendo para una agradable cita"
		title: "Ni tan bueno ni tan malo"

Figura 106. Persistencia de datos, reviews

En la colección “transacciones” se asigna de manera automática un identificador para cada documento que se agrega a la misma, en esta colección se almacenan todas las transacciones de compra/venta de servicios sin importar el tipo de este, a continuación, muestra el documento cuyo servicio es “La gruta”, efectuado el “12 e3 Julio de 2021 a las 22:34:24 hrs”, con 2 reservaciones para el “23 de Julio de 2021 a las 17:00 hrs”:

transactions		LVNiFtKxTWTizP0gCdMA
reviews	5m0t3WJU9bAMNojrupsF	creationAt: 12 de julio de 2021, 22:34:24 UTC-5
roles	71jBpN6jCLNTJ6kFvKka	name_sale: "La gruta"
services	E3yB1R3omSd6bS3c2q8z	people: 2
transactions	LVNiFtKxTWTizP0gCdMA	reservation_date_time: 23 de julio de 2021, 05:00:00 UTC-5
users	QEJ0JdxpzPL8ho0WGsc0	seller_id: "oYGeDHpv5VUoBkmydpGAN6ING63"
	QPtIgKoSsJhUo79D8EPY	service_id: "qSWSvBp2HwL41DT4WoMT"
	ZKzjdDasV3LACUymBdea	status: 2
	asr25Zk47fiz1q080K2r	total: 100
	c0dMRTXjmy9DqDWfkUYh	turist: "9qAymdfkGufmttoyUiKwxKT2jEaG3"
	filuU28gokyw5THjf1UK	turist_name: "Verónica Agustín Dominguez"
	hh5VqPdMNP1JwQhVogJz	type: "consumption"
	10RUf1Ph1CEd6rEz7yF5	unit_price: 50
	p3zthH5bUs1JTnA6A9Ue	
	n98cwoPfkat10YhmKyz1	

Figura 107. Persistencia de datos, transacciones

6.2. Happie path registro de usuario

Las siguientes imágenes son simulaciones reales en dispositivos virtuales y físicos en iphone 12 pro max y iphone 7 para algunos casos.

Se comienza con el registro de usuarios, primeramente, con el registro de usuario de tipo vendedor, donde se al abrir la aplicación por primera vez se muestra la pantalla de login, posterior a ello se hace clic en “Crear cuenta”, seleccionar el tipo de usuario, en este caso “vendedor”, inmediatamente cambia a una pantalla con un formulario para el registro de datos del vendedor.

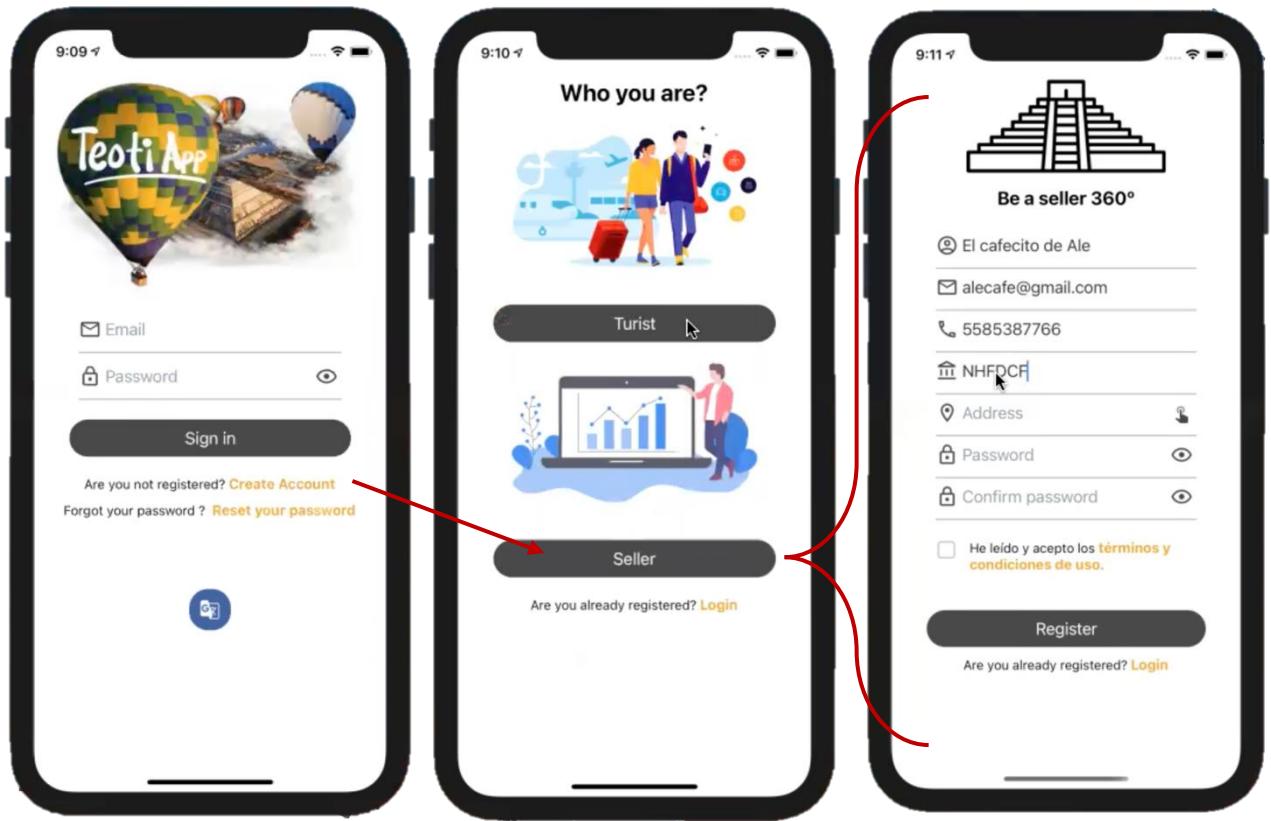


Figura 108. Pruebas - Registro de usuario parte 1

El llenado de datos tiene validaciones para cada campo, todos los campos son requeridos, de lo contrario se muestra una leyenda que indica que todos los campos son obligatorios, hasta no cumplir con dicha validación no se continua con el proceso de registro de usuario.

Uno de los campos más interesantes de ejecutar y describir, es la selección de la ubicación por medio del api de google, al presionar sobre el icono de selección táctil, se abre un modal con la posibilidad de seleccionar en tiempo real la ubicación.

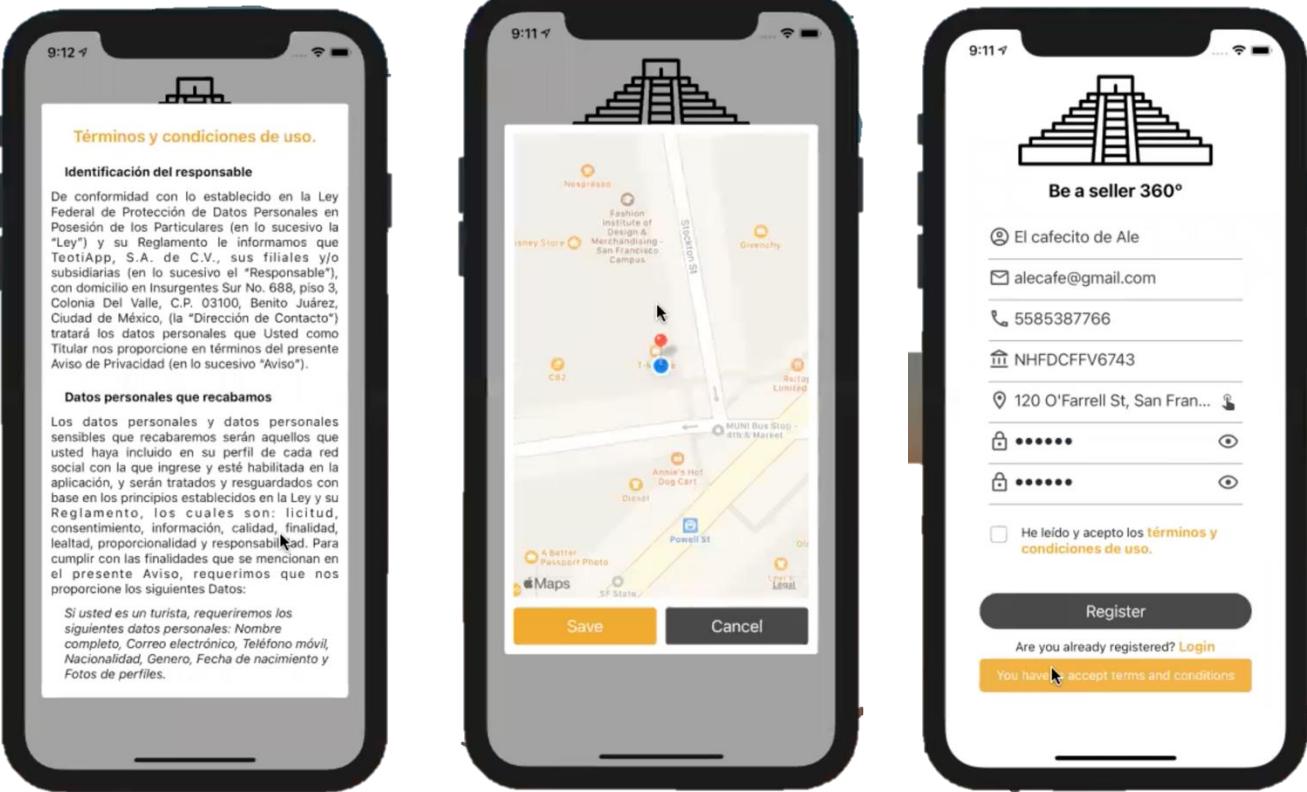


Figura 109. Pruebas - Registro de usuario parte 2

El apartado “términos y condiciones” tiene validación, pues al no aceptarlos no permite el registro de usuario, y muestra una leyenda que explica el motivo por el cual no se ha podido concretar el registro de usuario. Por otro lado, al presionar sobre el botón “Registrar” si todos los campos se han completado, el correo electrónico tiene un formato correcto, la dirección esta en una zona alrededor de la zona teotihuacana y las contraseñas coinciden, se procede con el registro de usuario en el sistema y continua con el acceso a la aplicación.

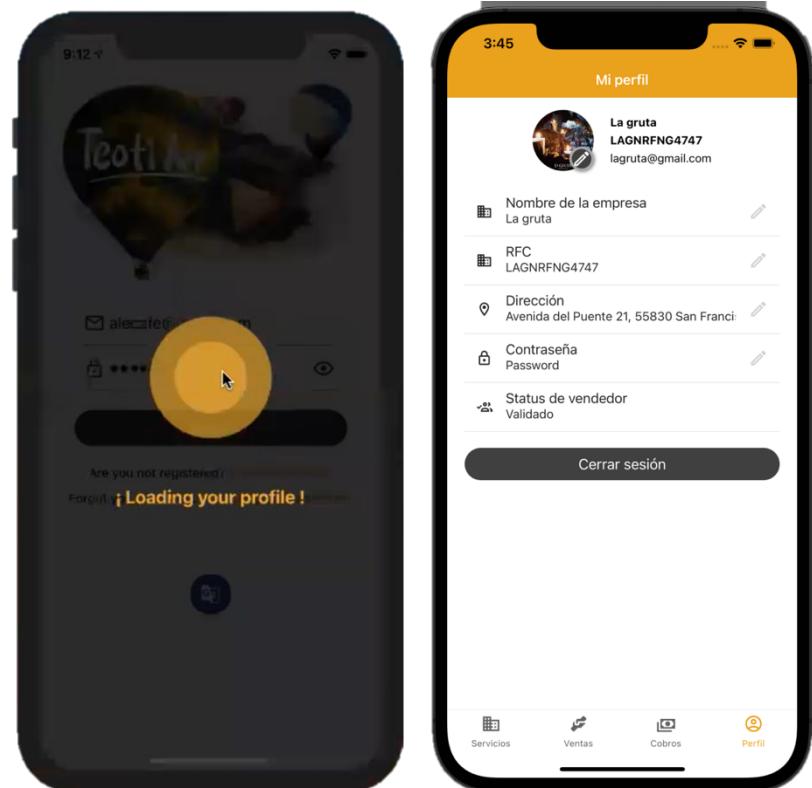


Figura 110. Pruebas - Registro de usuario parte 3

Para el caso de registro de usuario de tipo turista se realiza exactamente lo mismo, a diferencia que el formulario de registro de usuario solicita algunos otros datos distintos, por ejemplo, el selector de fecha de nacimiento, genero y nacionalidad, donde todos los campos son obligatorios.

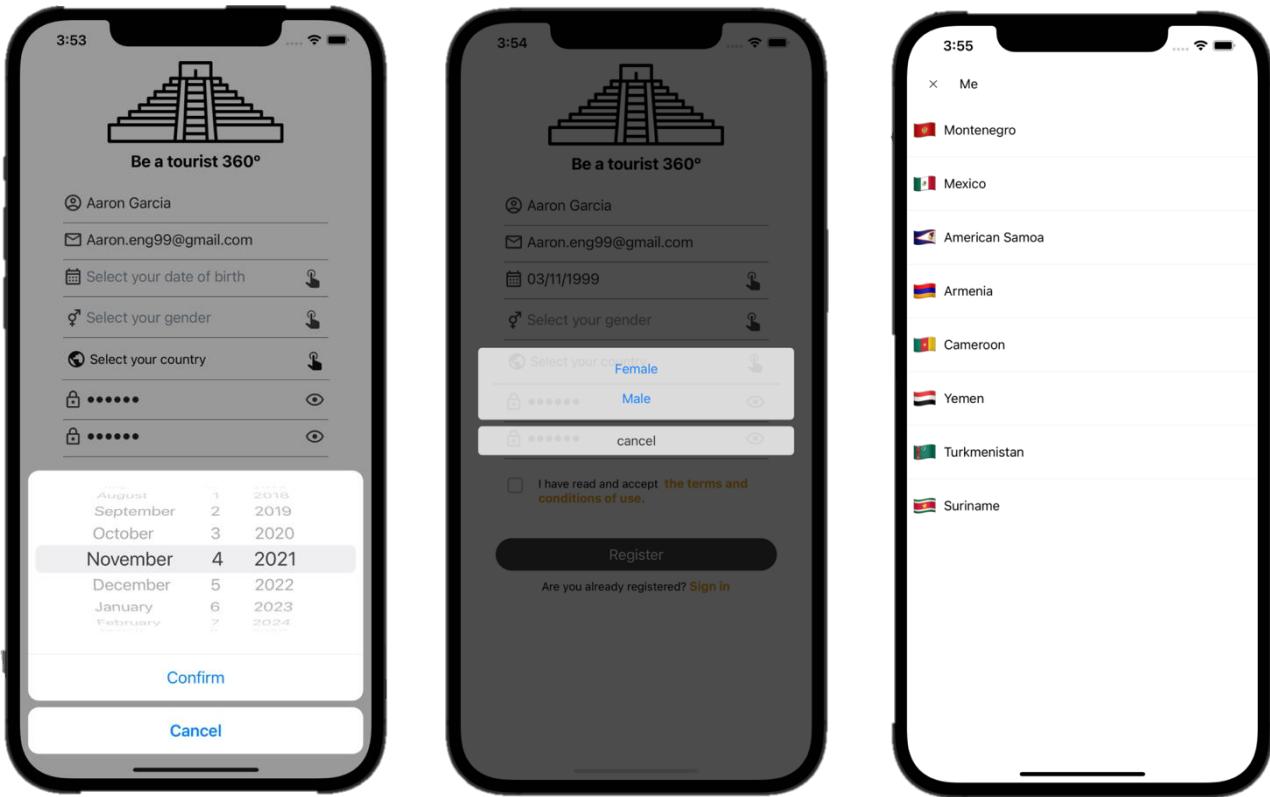


Figura 111. Pruebas - Registro de usuario parte 4

Al presionar en el botón “Registrar” se valida que todos los campos estén capturados, tengan el formato correcto, y las contraseñas coinciden, de todo salir bien, se realiza el registro del usuario en el sistema y continua con el acceso a la aplicación para el usuario correspondiente.

6.3. Happie path de creación de servicio

Se creará un servicio de tipo hospedaje, el cual llevará por nombre “Hotel quinto Sol”, en el cual se cargarán 5 fotografías, una descripción de este que contenga lo siguiente: “El mejor hotel en la zona! A unos osos de la urbe teotihuacana y de los pueblos mágicos, hotel de lujo, dentro podrás disfrutar de gastronomía internacional, alberca, spa y mucho más. ¡Ven y hospédate en un hotel a la altura! Grandes personalidades se han hospedado con nosotros.”, con disponibilidad unitaria y actual de 75 y costo unitario por \$1099.

Para ello al presionar sobre “Servicios” en la barra de navegación y posteriormente en el botón con el ícono de “+”, posteriormente seleccionar el tipo de servicio a crear, en este caso es hospedaje.

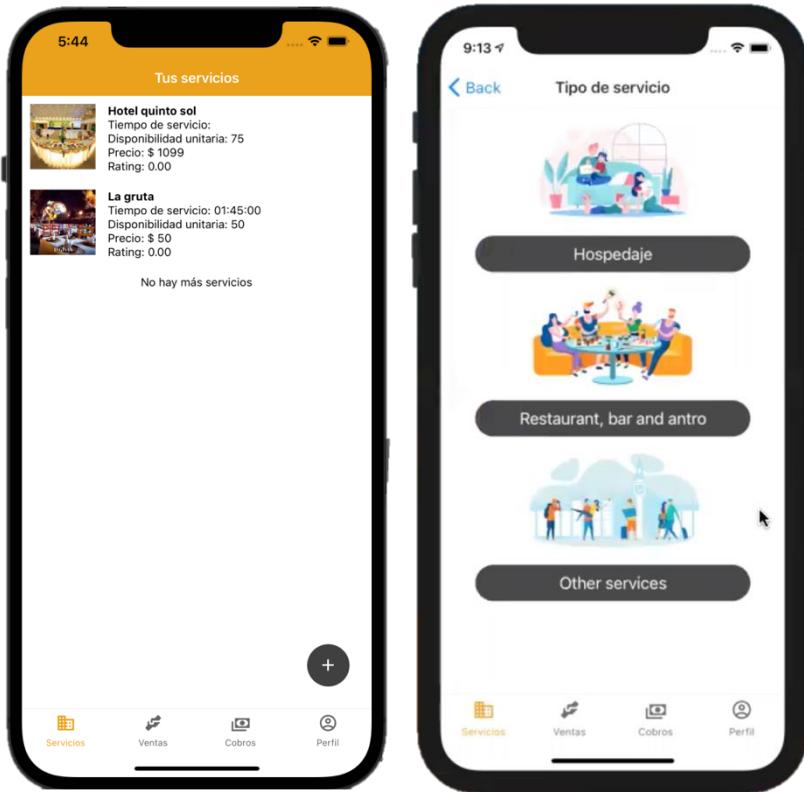


Figura 112. Pruebas - Creación de servicio parte 1

Inmediatamente aparecerá el formulario a llenar para la creación de dicho servicio, este es el mismo procedimiento en caso de ser otro tipo de servicio, el llenado de este formulario es obligatorio para todos los campos, se deberá seleccionar al menos 1 fotografía para poder continuar con el registro del servicio, la primera vez que se ingresa se solicita permisos para acceder a la galería de fotografías por lo que hay que dar acceso a la misma.

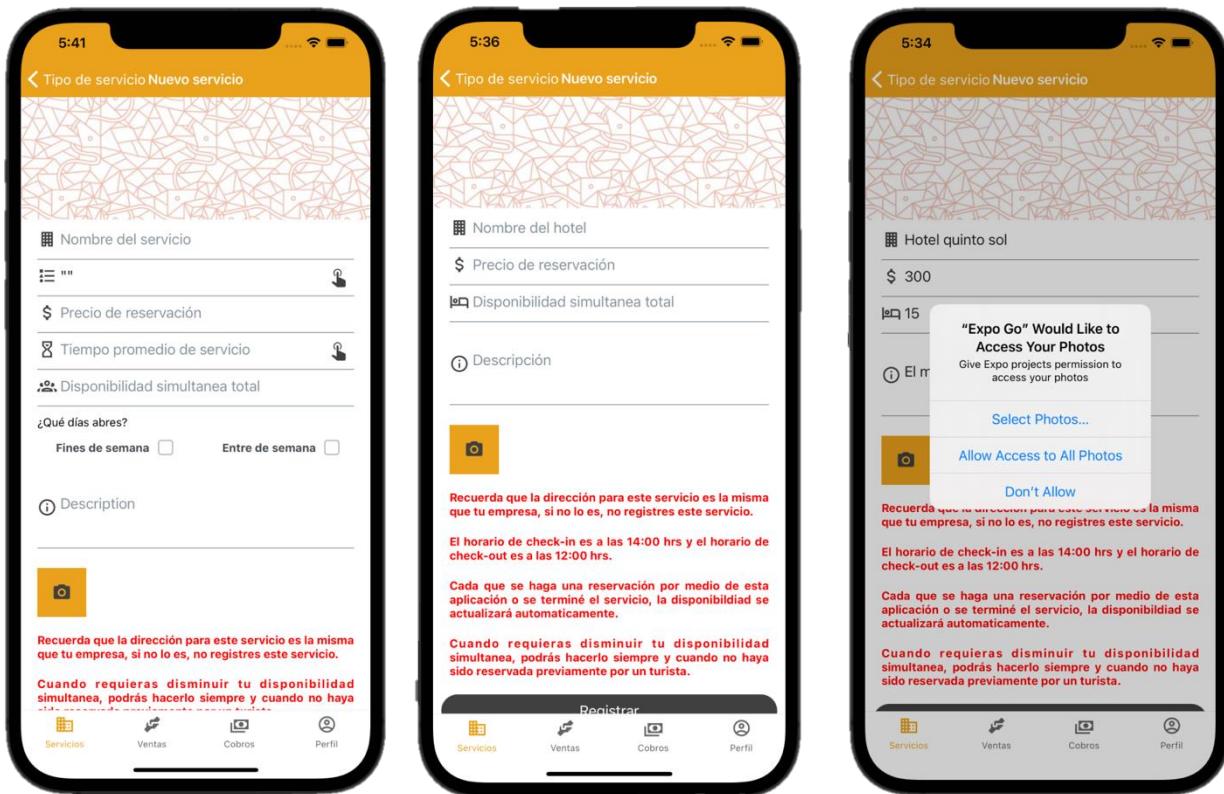


Figura 113. Pruebas - Creación de servicio parte 2

Una vez que se ha registrado el servicio es posible consultar y actualizar en tiempo real las características de este:

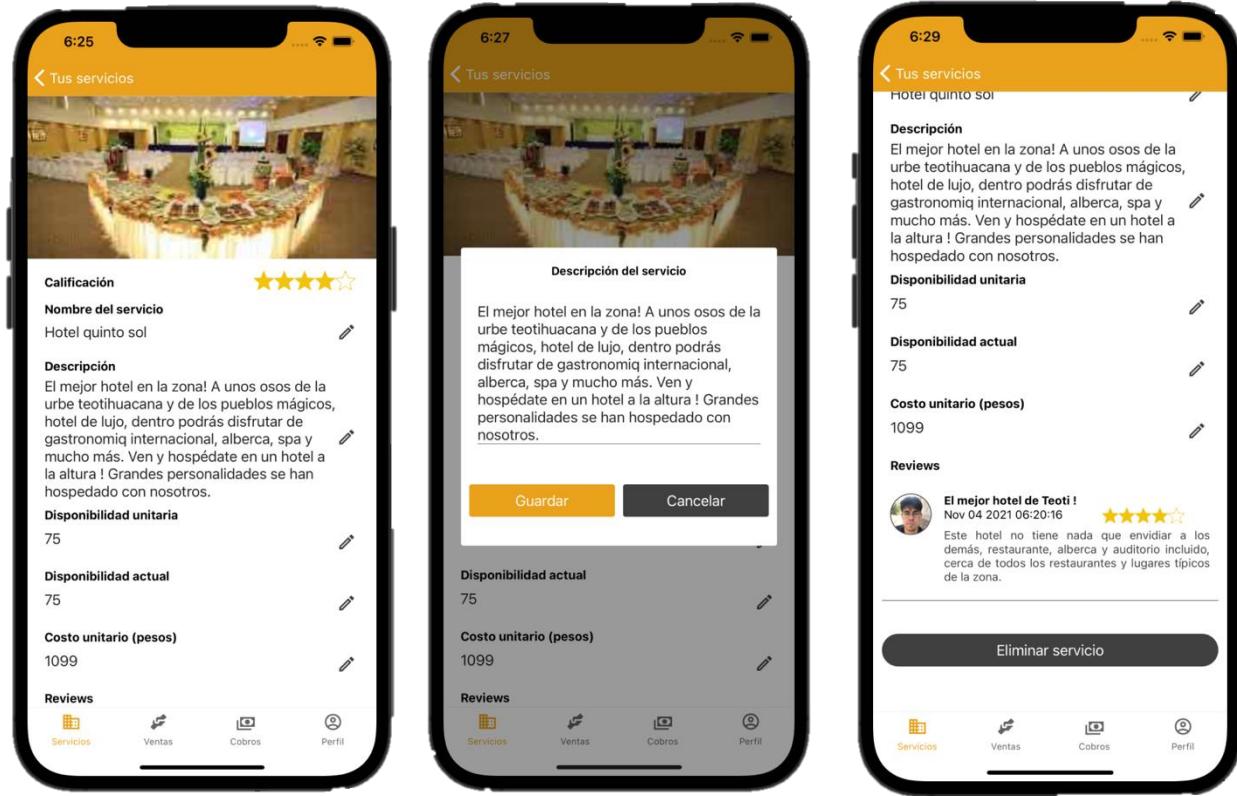


Figura 114. Pruebas - Creación de servicio parte 3

Se hace la actualización de datos de todos los campos uno a la vez, donde cada actualización es reflejada en la aplicación móvil y en la base de datos en firebase, donde se cumple con las reglas ACID. El único dato que no se puede actualizar ni eliminar son los comentarios o calificaciones que realizan los usuarios que han adquirido el servicio previamente.

Se realiza la eliminación y sustitución de imágenes, esto se logra al dar presionar sobre la imagen y dar en aceptar, a lo sumo se permiten 5 imágenes, cuando hay menos de esta cantidad, se incluye una imagen con el ícono de agregar imagen y al presionar sobre esta, se accede nuevamente a la galería de fotos, en caso de cancelar la operación se despliega una alerta indicando que no se ha seleccionado ninguna imagen y permite continuar con la navegación/actualización de los campos del servicio en cuestión.

6.4. Happie path de compra de servicio

Se accede como usuario turista, se elije el servicio a adquirir, se presiona sobre el botón que incluye el costo de reservación, se muestra un formulario donde se solicita el día de checkin, el día de checkout (en caso de ser hospedaje), la hora de llegada (en caso de ser un servicio distinto a hospedaje) y el numero de personas, una vez capturados todos los datos anteriores, se muestra el total a pagar.

Como parte del proceso de pruebas se intenta pagar con campos vacíos, con fechas incorrectas (que el día de salida sea antes que el de llegada al servicio de hospedaje) y so n superadas correctamente gracias a las validaciones implementadas, por lo que no permite la compra de servicios con campos incorrectos.

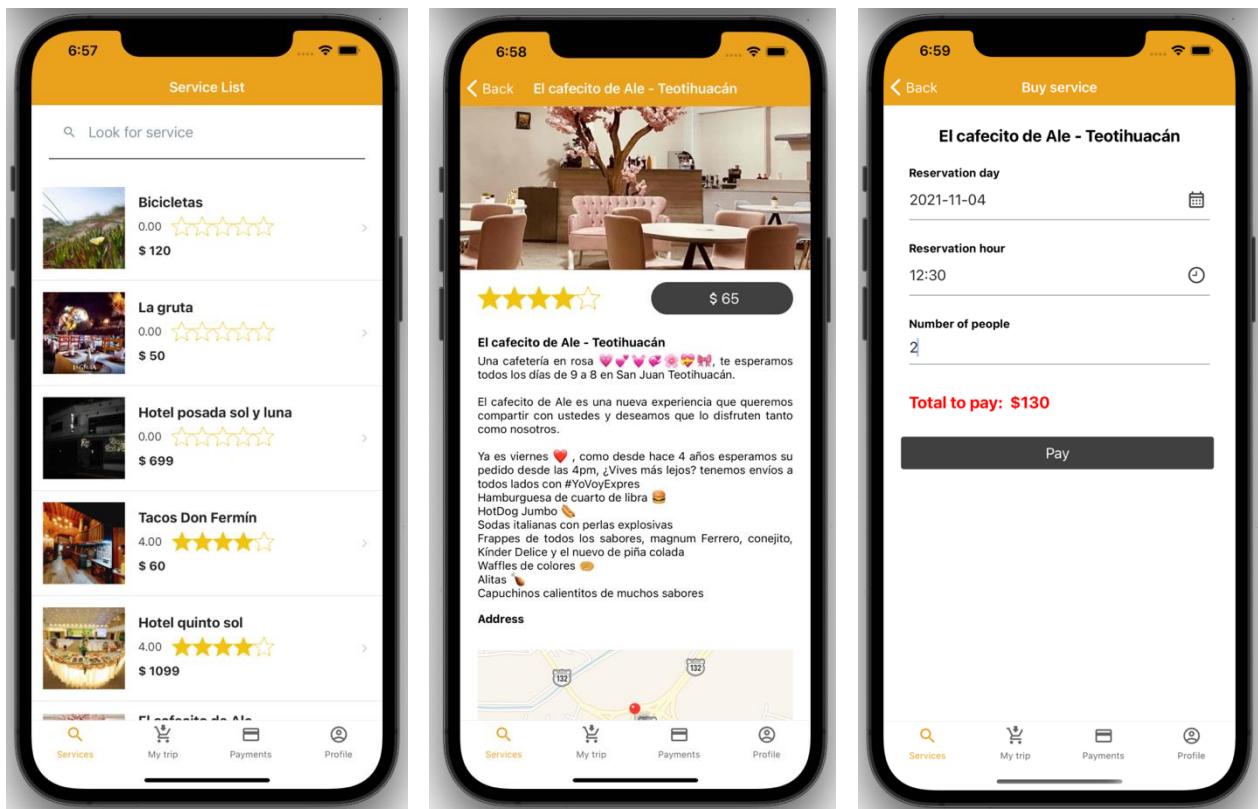


Figura 115. Pruebas - Compra de servicio parte 1

Posterior a la compra del servicio, se redirige a la pestaña de “Mi viaje”, la primera compra es la mas reciente, por lo que al presionar sobre este se muestran los detalles de compra realizada, donde se puede ver cada detalle y además realizar y editar la evaluación del servicio.

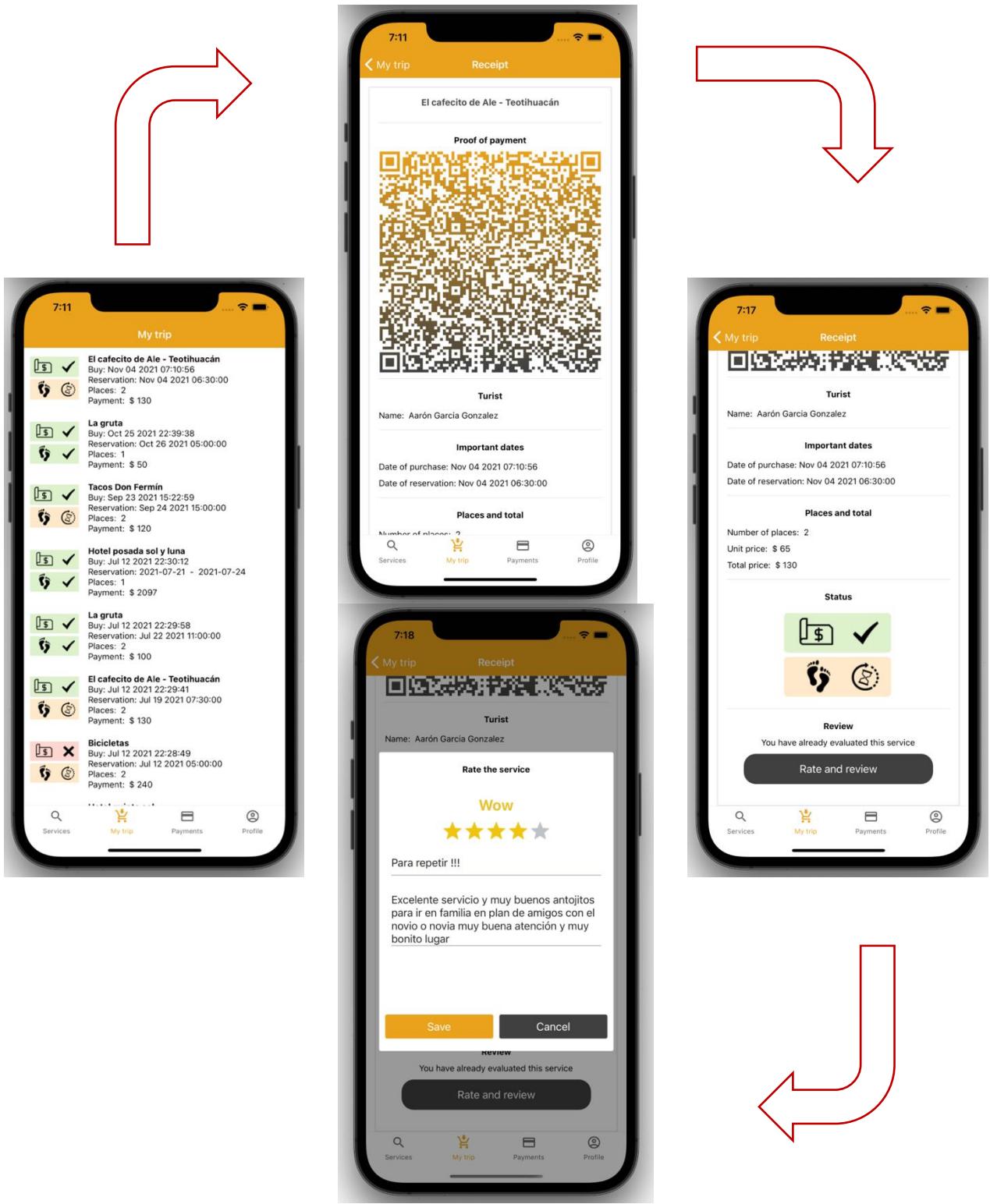


Figura 116. Pruebas - Compra de servicio parte 2

6.5. Happie path de comprobante de pago

En la prueba anterior se genero el comprobante de pago de la compra, este comprobante el turista lo muestra al vendedor en tiempo y forma para tener acceso y uso del servicio adquirido, en la prueba anterior se muestran algunos de los escenarios que se presentan en este proceso de acceso y autenticación de transacciones.

En la siguiente imagen, del lado derecho se muestra la simulación en un iphone 12 pro la aplicación del turista donde muestra su comprobante de pago QR, del lado derecho se muestra el iphone 7 del vendedor, donde ya escaneo el servicio, pero la validación no fue exitosa, esto debido a que el servicio a dar acceso es “El cafecito de Ale – Teotihuacán”, pero el vendedor no es responsable/dueño de este servicio, es por ello que no se le da acceso.

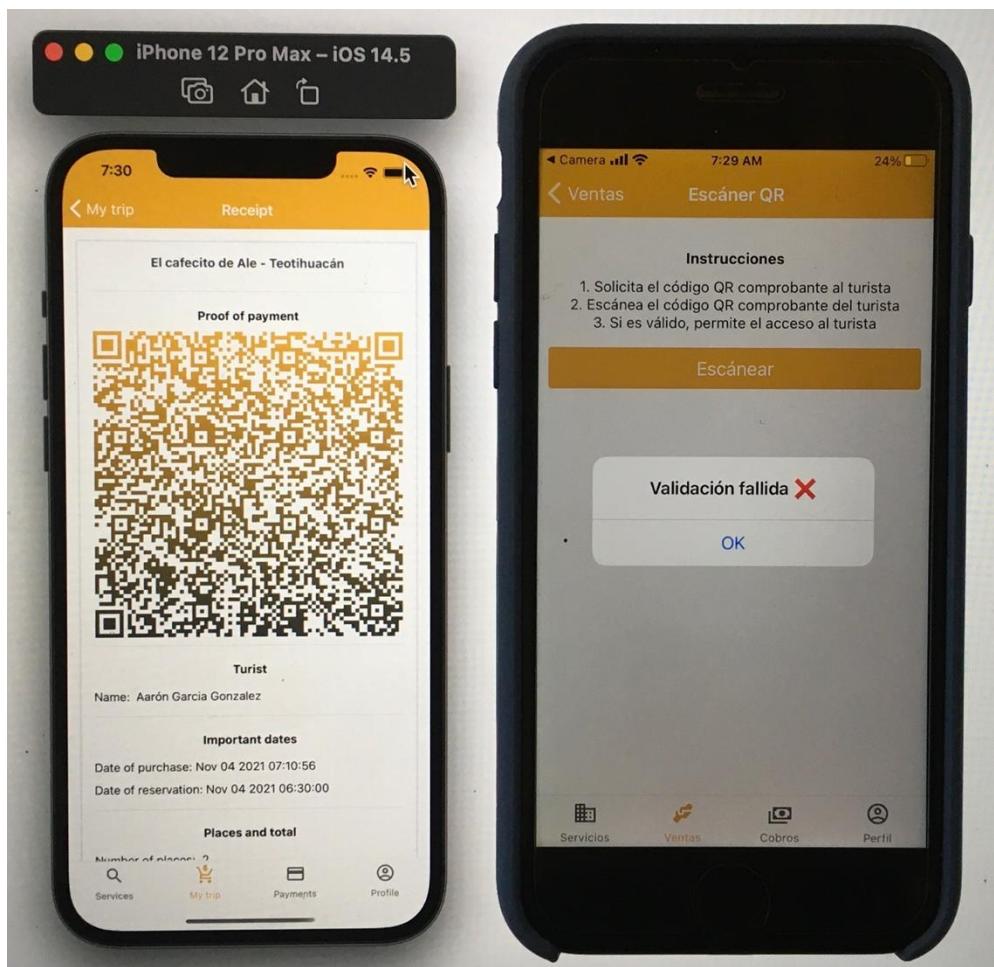


Figura 117. Pruebas - Validación con fallo por escaneo de otro servicio

En la siguiente imagen, del lado derecho se muestra la simulación en un iphone 12 pro la aplicación del turista donde muestra su comprobante de pago QR, del lado derecho se muestra el iphone 7 del vendedor, donde ya escaneo el servicio, pero la validación no fue exitosa, esto debido a que el servicio a dar acceso es “La gruta” específicamente esta compra ya fue escaneada previamente, por lo que ya no es valido hacerlo por segunda vez.

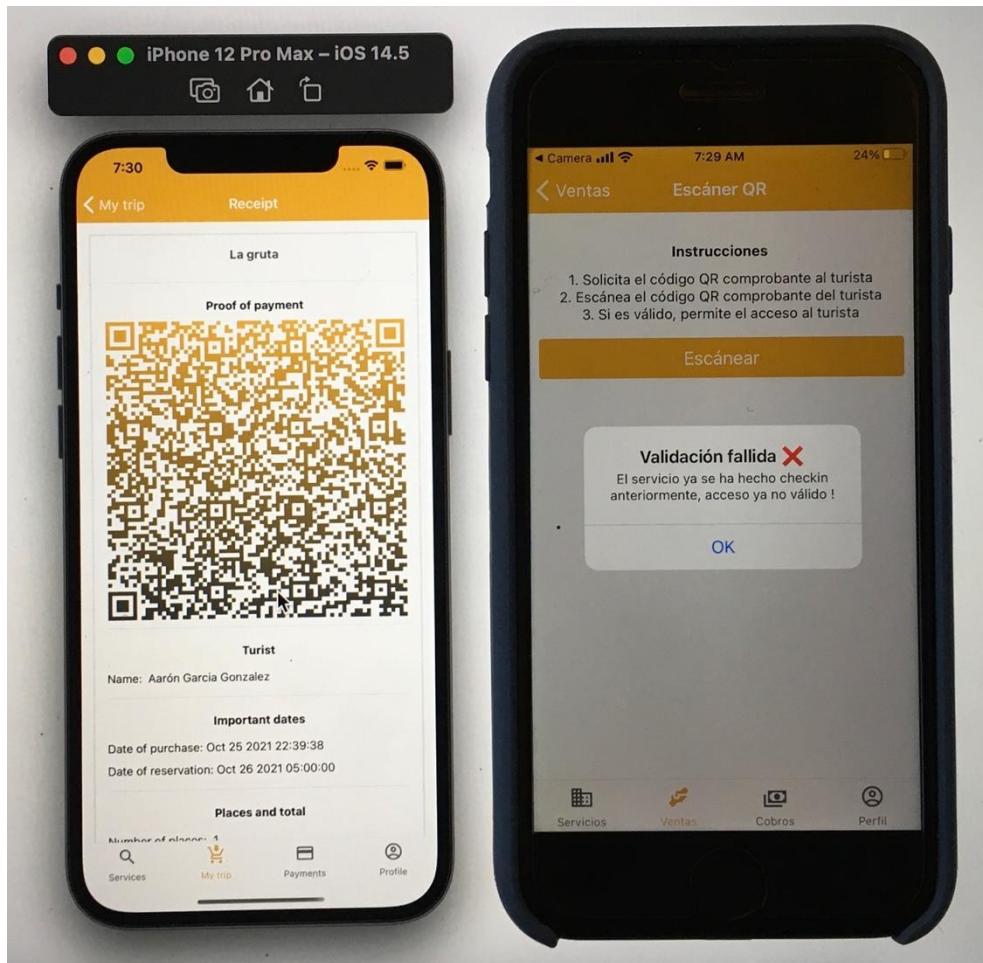


Figura 118. Pruebas - Validación con fallo por doble escaneo de comprobante de pago

En la siguiente imagen, del lado derecho se muestra la simulación en un iphone 12 pro la aplicación del turista donde muestra su comprobante de pago QR, del lado derecho se muestra el iphone 7 del vendedor, donde ya escaneo el servicio, con la validación exitosa, esto debido a que el vendedor es propietario del servicio, y jamás se ha registrado el ingreso con esta transacción.

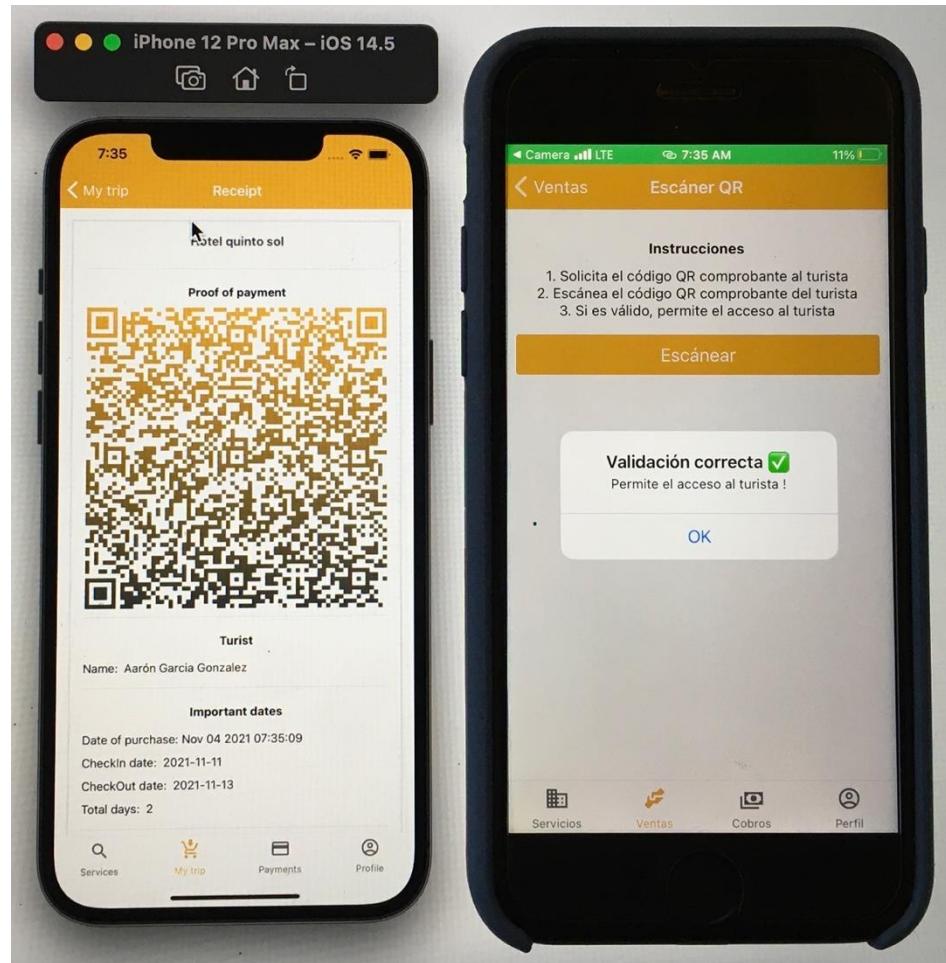


Figura 119. Pruebas - Validación exitosa de escaneo de comprobante de pago

6.6. Happie path de generación y escaneo de orden de pago

Se simula el que un turista le solicita a un vendedor que le genere una orden de pago para el servicio de “Hotel quinto sol”, donde que se requiere una reservación entre el 04 y 11 de Noviembre del año 2021 para 2 personas, esto mencionado anteriormente se realiza en el lado derecho por el usuario vendedor, en el lado derecho se encuentra el usuario turista, donde ya ha escaneado el código QR, se muestra la orden de pago donde se corroboran los datos de la reservación, de ser correcto se procede a “Pagar” mismo donde se presiona en el botón “Pay”.

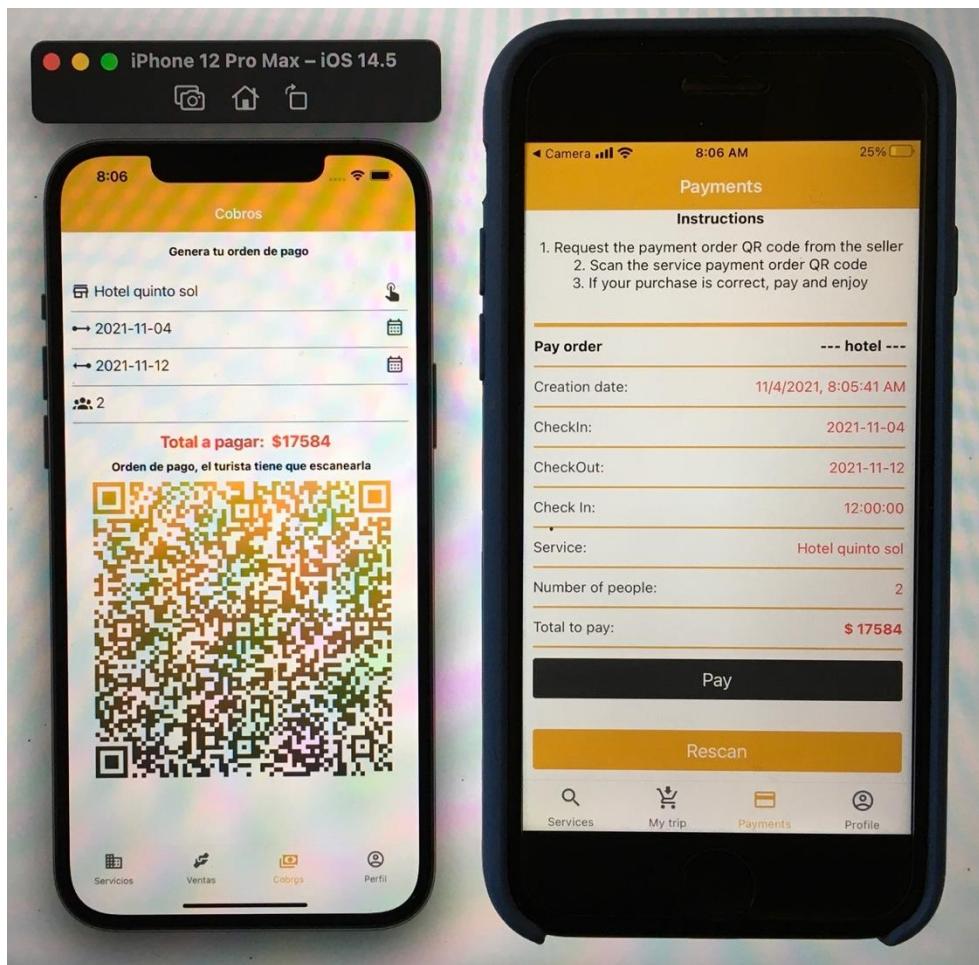


Figura 120. Pruebas - Generar y escánean orden de pago

Trabajo a futuro

Esta aplicación es una idea de negocio compleja que podría aprovechar la ausencia de necesidades no cubiertas en este nicho, el trabajo a futuro dentro de esta aplicación es muy amplio, es una aplicación como Facebook o mercado libre, que inicialmente fue lanzada en una versión y ha ido creciendo a lo largo del tiempo, entendido esto, se entiende que esta es una versión piloto lista para ser desarrollada agregando 1 y 1000 funcionalidades, a continuación, se describen algunas de estas:

- Módulo de notificaciones
La aplicación móvil sería importante que mandara notificaciones de cualquier cambio/actualización tanto a la versión móvil de turista y vendedor como de administrador.
- Añadir mas plataformas de pago
Además de la pasarela de pago usada, se le brindaría un plus a la aplicación móvil el poder ejecutar pagos con otros proveedores, por ejemplo, OpenPay de BBVA, mercado pago, stripe, codi QR, o pago con tarjetas bancarias.
- Mas idiomas
Los idiomas en la versión actual es inglés y español, el agregar otros idiomas de los países cuyas nacionalidades de turistas, haría más atractiva esta aplicación, por ejemplo, francés, italiano, chino, ruso, portugués, hindi, árabe, etc.
- Módulo de analítica de datos para comentarios y posicionamiento
Una versión de la aplicación móvil para los vendedores donde se les ayuda a lograr posicionamiento y competitividad respecto a la competencia, donde incluya analítica de datos que presenta gráficos, resúmenes, etc.
- Integración/registro con redes sociales
Al registrar o iniciar sesión, tal como lo hacen las aplicaciones más grandes, el poder incluir este proceso mediante redes sociales como Facebook y google.
- Conexión a servicio de transporte
Un servicio importante para esta zona es el servicio de transporte, este es un caso especial, ya que prácticamente es desarrollar una aplicación como Uber, pero la posible integración con algo ya existente podría ser una buena opción o bien un modulo que se encargue de ello.

Impacto ambiental

El uso de esta aplicación móvil podría ser un efecto domino en ayuda del medio ambiente en la zona turística, pequeños pero simples detalles que fortalecen el cuidado del ambiente, por ejemplo, en los siguientes puntos:

Disminución de uso de dinero físico, lo cual implica menor demanda/cambio de dinero, que involucra menor uso de cajeros automáticos, producción de billetes, uso de banca en general. De igual manera estamos en un tiempo de pandemia, por lo que el tocar dinero físico involucra estar expuesto a este virus mediante el contacto, entonces, el uso de esta aplicación involucra menor consumo de gel antibacterial, toallas desinfectantes, etc.

Dado que la aplicación móvil incluye adecuación al lenguaje, hace no ser necesario el uso de un traductor y todos los recursos que esto implica.

El simple hecho de poder visualizar los servicios que están disponibles en la aplicación hace que el turista no tenga que desplazarse y gastar combustible para encontrar que el lugar está cerrado, ya no existe o simplemente no le gusta, entonces es un ahorro de recursos redondos.

Conclusiones

Este proyecto inicio como una idea dentro de una de las asignaturas que lleve en la carrera, “gestión empresarial”, inicio como una idea hasta ser propuesta con los directores de este proyecto, es una idea de negocio que tiene razón de ser y utilidad en el mundo real.

A lo largo del desarrollo de este proyecto se presentaron situaciones de diversos índoles, el aprender React Native a la par del desarrollo, hizo lento el desarrollo al principio del proyecto, especialmente en la implementación del primer módulo, donde siendo muy realistas algunas funcionalidades consumieron grandes cantidades de tiempo, luego al tener que documentar en paralelo, hubo algunos días que se dejaba de lado la programación y cuando se retomaba de nuevo, la noción y práctica adquirida días antes parecía disminuir, estos fueron los pininos en el proyecto, el tiempo siguió avanzando, y se aprendió de ello, ya que como único miembro del equipo, de manera autónoma se tomaron las decisiones pertinentes, por ello se comenzó a documentar de manera semanal o cada que había algo que documentar de manera considerable, y la programación al ser una metodología ágil y extrema, se atendieron los cambios y sugerencias que fueron detectadas, pese a tener un diseño previo.

Las principales dificultades en el desarrollo fueron familiarizarse con la conexión a firebase y la forma de acceder a los datos, la utilización de los componentes de selección de fecha, hora, listas desplegables, y manejo de imágenes en para su manipulación.

Fue buena estrategia el desarrollar bajo estos 5 módulos, ya que la aplicación en este punto se puede ver a grandes rasgos el funcionamiento, que al final del día, ese es uno de los principios del desarrollo ágil, funcionalidad constante, ahora ya se tiene cierto grado de práctica, por lo que los módulos con mayor grado de dificultad, se les reduce el peso del lado de programación ahora al saber usar React Native.

Otro punto que considerar, son las observaciones realizadas por los profesores, las cuales fueron atendidas y aplicadas en la medida de lo posible, por ahora estoy satisfecho con los resultados, continuaré trabajando duro para culminar el proyecto en tiempo y forma.

Por el numero de tecnologías que aprendí y utilice en el desarrollo del proyecto, tanto de desarrollo y gestión, puedo decir que fue un proyecto integral que a modo de reflexión personal, pase por gran parte de las áreas que involucran proyectos de tecnologías actuales, este proyecto me genero cierto grado de madurez profesional, al comenzar a buscar trabajo los reclutadores siempre me preguntaron cual ha sido mi mayor reto académico y profesional, dado que este proyecto lo realice en tiempo y forma aplicando los conocimientos adquiridos en la escuela, otros de manera autodidacta, siempre lo mencione y explique como algo que se pudo superar, aprender y dejar huella.

Referencias

- [1] Instituto Nacional de Antropología e Historia, «www.inah.gob.mx,» 18 Diciembre 2020. [En línea]. Available: <https://www.inah.gob.mx/zonas/23-zona-arqueologica-de-teotihuacan>. [Último acceso: 10 Febrero 2021].
- [2] Universidad Autónoma del Estado de Hidalgo UAEH, «“Motivaciones turísticas del Corredor del Valle de Teotihuacán: estudio de caso en Zona Arqueológica de Teotihuacán, Estado de México,» *Boletín Científico INVESTIGIUM de la Escuela Superior de Tizayuca*, vol. 10, pp. 45-50, 2020.
- [3] Organización para la Cooperación y el Desarrollo Económicos OCDE, «Estudio de la Política Turística de México,» Secretaría de Turismo de los Estados Unidos Mexicanos , 2017.
- [4] C. d. A. d. S. d. Turismo, «Nuestro turismo "El gran motor de la economía nacional", 5 objetivos de la política pública,» Editorial Raices, CDMX, 2018.
- [5] J. P. J. P. BUAP, «El impulso del turismo a través de las prácticas asociacionistas, el caso del corredor turístico del valle de Teotihuacán,» *Topofilia, Revista de Arquitectura, Urbanismo y Territorios* , vol. 5, pp. 109-137, 2015.
- [6] Secretaría de Turismo SETUR, «La actividad turística en México hoy tiene una nueva dimensión social,» 23 Enero 2020. [En línea]. Available: <https://www.gob.mx/sectur/prensa/la-actividad-turistica-en-mexico-hoy-tiene-una-nueva-dimension-social-232853?idiom=es>. [Último acceso: Febrero 2021].
- [7] Sistema Nacional de Información Estadística y Geográfica, «DATATUR: Análisis integral del turismo,» 01 Enero 2016. [En línea]. Available: <http://www.datatur.sectur.gob.mx/SitePages/ActividadesCulturales.aspx>. [Último acceso: 11 Febrero 2021].
- [8] Quadratín Estado de México, «Quadratin EDOMEX,» Agencia de noticias Quadratin, 18 Noviembre 2016. [En línea]. Available: <https://edomex.quadratin.com.mx/van-calendario-unico-actividades-teotihuacan-san-martin/>. [Último acceso: 11 Febrero 2021].
- [9] E. J. A. y. J. C. T. Muñoz, «Sistema de difusión cultural interactivo con enfoque Regional Mexicano,» Instituto Politécnico Nacional, IPN ESCOM, CDMX, México, 2016.
- [10] Secretaría de Turismo, «www.visitmexico.com,» Gobierno de México, [En línea]. Available: <https://www.visitmexico.com/>. [Último acceso: 11 Febrero 2021].
- [11] Despegar.com, «[despegar.com](http://www.despegar.com),» [En línea]. Available: <https://www.despegar.com.mx/>. [Último acceso: 11 Febrero 2021].
- [12] Expedia Inc, «Paquetes Vacacionales Incluyen Hotel y Vuelo a Teotihuacán,» [En línea]. Available: <https://www.expedia.mx/Teotihuacan.d182187.Guia-de-vacaciones?pwaLob=wizard-package-pwa>. [Último acceso: 11 Febrero 2021].
- [13] Tours and travel Teotihuacán, «teotihuacan.com.mx,» [En línea]. Available: <https://teotihuacan.com.mx/>. [Último acceso: 11 Febrero 2021].
- [14] Explora Teotihuacan, [En línea]. Available: <http://explorateotihuacan.com.mx/>. [Último acceso: 11 Febrero 2021].
- [15] Sky Balloons México, [En línea]. Available: <https://www.skyballoons.mx/>. [Último acceso: 11 Febrero 2021].
- [16] Secretaría de Gobernación de México (SEGOB), «ACUERDO por el que se establecen los Lineamientos generales para la incorporación y permanencia al Programa Pueblos Mágicos.,» *Diario Oficial de la Federación (DOF)*, 26 Septiembre 2014.

- [17] Dirección de Legalización y del Periódico Oficial “Gaceta del Gobierno”, «<http://legislacion.edomex.gob.mx/>,» 7 Octubre 2014. [En línea]. Available: [/sites/legislacion.edomex.gob.mx/files/files/pdf/gct/2014/oct072.PDF](http://sites/legislacion.edomex.gob.mx/files/files/pdf/gct/2014/oct072.PDF). [Último acceso: 22 Marzo 2021].
- [18] Honorable Cámara de Diputados , «<http://www.diputados.gob.mx/>,» 05 07 2010. [En línea]. Available: <http://www.diputados.gob.mx/LeyesBiblio/pdf/LFPDPPP.pdf>. [Último acceso: 04 2021].
- [19] Debitoor, «E-commerce - ¿Qué es el e-commerce?,» Sumup, 01 01 2021. [En línea]. Available: <https://debitoor.es/glosario/definicion-e-commerce>. [Último acceso: 09 04 2021].
- [20] Market Hax, «Pasarelas de pago para ecommerce en México,» MarketHax, 21 04 2020. [En línea]. Available: https://markethax.com/pasarelas-de-pago-para-ecommerce-en-mexico/#%C2%BFQue_son_las_pasarelas_de_pago. [Último acceso: 09 04 2021].
- [21] D. L. G. d. l. Fraga, «<http://delta.cs.cinvestav.mx/>,» 08 05 2015. [En línea]. Available: <http://delta.cs.cinvestav.mx/~fraga/Cursos/SistemasOperativos/2015/sisOperativos.pdf>. [Último acceso: 04 2021].
- [22] MDN Web Docs, «JavaScript,» Mozilla and individual contributors, 01 01 2021. [En línea]. Available: <https://developer.mozilla.org/es/docs/Web/JavaScript>. [Último acceso: 05 04 2021].
- [23] Profile, «12 librerías JavaScript que deberías conocer,» Profile Software Services, 08 07 2020. [En línea]. Available: <https://profile.es/blog/librerias-javascript/>. [Último acceso: 05 04 2021].
- [24] MDN Web Docs, «CSS,» Mozilla and individual contributors, 01 01 2021. [En línea]. Available: <https://developer.mozilla.org/es/docs/Web/CSS>. [Último acceso: 05 04 2021].
- [25] MDN Web Docs, «HTML: Lenguaje de etiquetas de hipertexto,» Mozilla and individual contributors. , 04 04 2021. [En línea]. Available: <https://developer.mozilla.org/es/docs/Web/HTML>. [Último acceso: 05 04 2021].
- [26] MDN Web Docs, «Entendiendo los frameworks de JavaScript del lado del cliente,» Mozilla and individual contributors, 04 04 2021. [En línea]. Available: https://developer.mozilla.org/es/docs/Learn/Tools_and_testing/Client-side_JavaScript_frameworks. [Último acceso: 05 04 2021].
- [27] Deloitte, «¿Qué es React Native?,» Deloitte, 01 01 2021. [En línea]. Available: <https://www2.deloitte.com/es/es/pages/technology/articles/que-es-react-native.html>. [Último acceso: 05 04 2021].
- [28] Oracle México, «Base de datos,» Oracle, 01 01 2021. [En línea]. Available: <https://www.oracle.com/mx/database/what-is-database/>. [Último acceso: 06 04 2021].
- [29] I18next, «I18next documentation,» GitBook, 01 02 2021. [En línea]. Available: <https://www.i18next.com/>. [Último acceso: 06 04 2021].
- [30] Xataka, «API: qué es y para qué sirve,» Webedia, 19 08 2019. [En línea]. Available: <https://www.xataka.com/basics/api-que-sirve>. [Último acceso: 06 04 2021].
- [31] AGILEMANIFESTO, «Manifesto for Agile Software Development,» artwork , 01 01 2001. [En línea]. Available: <http://agilemanifesto.org/>. [Último acceso: 12 04 2021].
- [32] Universidad Politécnica de Valencia, «Métodologías Ágiles en el Desarrollo de Software,» 04 04 2011. [En línea]. Available: <http://aleteya.cs.buap.mx/~jlavalle/papers/agileMethodology/TodoAgil.pdf>. [Último acceso: 12 04 2021].

- [33] Glassdoor, «Sueldos para desarrollador de aplicaciones móviles,» Glassdooe, 01 01 2021. [En línea]. Available: https://www.glassdoor.com.mx/Sueldos/desarrollador-de-aplicaciones-m%C3%B3viles-sueldo-SRCH_KO0,37.htm. [Último acceso: 10 04 2021].
- [34] Cámara de diputados del H. Congreso de la unión, «www.diputados.gob.mx,» 07 Julio 2019. [En línea]. Available: http://www.diputados.gob.mx/LeyesBiblio/pdf/LGT_310719.pdf. [Último acceso: 13 Febrero 2021].
- [35] QuestionPro, «www.questionpro.com,» Software para encuestas QuestionPro, [En línea]. Available: <https://www.questionpro.com/es/features/valoracion-por-estrellas/>. [Último acceso: 16 Febrero 2021].
- [36] Secretaria de Turismo, «Nuestro turismo "El gran motor de la economía nacional", 5 Objetivos de Política Pública,» Gobierno de México, CDMX, México, 2018.
- [37] Servicio de Administración Tributaria , «Validador de RFC's,» [En línea]. Available: <https://portalsat.plataforma.sat.gob.mx/ConsultaRFC/>. [Último acceso: 19 Febrero 2021].

Apéndice A. Aviso de privacidad en encuestas aplicadas

Se aplicaron 2 encuestas diferentes: vendedores y turistas, para lo que se informó al usuario el siguiente aviso de privacidad previo a llenar la encuesta correspondiente.

“Por medio del Instituto Politécnico Nacional (IPN) y Escuela Superior de Cómputo (ESCOM), se propone el proyecto de trabajo terminal B004 - Aplicación móvil de comercio para servicios turísticos en zona de Teotihuacán.

Este formulario tiene el objetivo de recabar información sobre la percepción turística ofertada en la zona de Teotihuacán, con el propósito de conocer sobre la interacción entre vendedores y turistas, toma de decisiones, análisis e impacto del desarrollo de este proyecto.

Las siguientes preguntas se utilizarán únicamente para uso analítico general en ambiente académico. Sus respuestas individuales no se entregarán a ningún tercero en absoluto. Proceder a la encuesta implica que usted comprende y acepta las disposiciones de este descargo de responsabilidad.”

Apéndice B. Aviso de privacidad en aplicación móvil

I. Identificación del responsable

De conformidad con lo establecido en la Ley Federal de Protección de Datos Personales en Posesión de los Particulares (en lo sucesivo la “Ley”) y su Reglamento le informamos que TeotiApp, S.A. de C.V., sus filiales y/o subsidiarias (en lo sucesivo el “Responsable”), con domicilio en Insurgentes Sur No. 688, piso 3, Colonia Del Valle, C.P. 03100, Benito Juárez, Ciudad de México, (la “Dirección de Contacto”) tratará los datos personales que Usted como Titular nos proporcione en términos del presente Aviso de Privacidad (en lo sucesivo “Aviso”).

II. Datos personales que recabamos

Los datos personales y datos personales sensibles que recabaremos serán aquellos que usted haya incluido en su perfil de cada red social con la que ingrese y esté habilitada en la aplicación, y serán tratados y resguardados con base en los principios establecidos en la Ley y su Reglamento, los cuales son: licitud, consentimiento, información, calidad, finalidad, lealtad, proporcionalidad y responsabilidad. Para cumplir con las finalidades que se mencionan en el presente Aviso, requerimos que nos proporcione los siguientes Datos:

- a) Si usted es un turista, requeriremos los siguientes datos personales
Nombre completo, Correo electrónico, Teléfono móvil, Nacionalidad, Genero, Fecha de nacimiento y Fotos de perfiles.
- b) Si usted es un vendedor, requeriremos los siguientes datos de su organización
Ubicación geográfica, correo electrónico, nombre, RFC, numero telefónico, costos de reservación o adquisición de servicio, horarios de trabajo, descripción de sus servicios, aforo simultaneo que dispone e imágenes de sus servicios.

III. Finalidades del tratamiento de sus datos personales

El responsable, sus filiales y/o subsidiarias, recaban sus datos personales y datos personales sensibles con el objeto de utilizarlos para los siguientes fines:

- a) Obtener información y datos estadísticos de hábitos de navegación
- b) Identificar perfiles que ingresan a la aplicación

IV. Transferencia de sus datos personales

Le informamos que sus datos personales serán transferidos dentro y fuera del país, a los siguientes destinatarios, y con las siguientes finalidades: Empresas del mismo grupo del responsable, con las finalidades de mercadotecnia, publicidad y proyección comercial. Por lo anterior solicitamos su consentimiento expreso para la transferencia de sus datos. Por favor seleccione “Si acepto las condiciones descritas anteriormente” al final del aviso de privacidad, o bien “No acepto las condiciones descritas anteriormente”.

V. Medios para limitar el uso o divulgación de sus datos personales

Hacemos de su conocimiento que sus datos personales y datos personales sensibles serán resguardados bajo estrictas medidas de seguridad administrativas, técnicas y físicas las cuales han sido implementadas, en términos del Reglamento, con el objeto de proteger y garantizar sus datos personales contra daño, pérdida, alteración, destrucción o el uso, acceso o tratamiento no autorizados.

VI. Medios para revocar el consentimiento para el tratamiento de sus datos personales

Usted podrá revocar su consentimiento para el tratamiento de sus datos personales de la misma forma por la cual otorgo su consentimiento. Asimismo Usted tiene derecho de: (i) Acceder a sus datos personales en nuestro poder y conocer los detalles del tratamiento de los mismos, (ii) Rectificarlos en caso de ser inexactos o incompletos, (iii) Cancelarlos cuando considere que no se requieren para alguna de las finalidades señaladas en el presente Aviso, estén siendo utilizados para finalidades no consentidas o haya finalizado la relación contractual o de servicio, u (iv) Oponerse al tratamiento de los mismos para fines específicos, según lo disponga la Ley"). Para el ejercicio de sus Derechos, deberá presentar una Solicitud al siguiente correo electrónico: datospersonales@teotiapp.com.mx acompañada de la siguiente información y documentos:

- a) Su nombre, domicilio y correo electrónico para poder comunicarle la respuesta a la Solicitud
- b) Una copia de los documentos que acrediten su identidad (copia de IFE, pasaporte o cualquier otra identificación oficial) o en su caso, los documentos que acrediten su representación legal, cuyo original deberá presentar para poder recibir la respuesta del responsable.
- c) Una descripción clara y precisa de los datos personales respecto de los cuales busca ejercer alguno de los Derechos
- d) Cualquier documento o información que facilite la localización de sus datos personales;
- e) En caso de solicitar una rectificación de sus datos personales, deberá de indicar también, las modificaciones a realizarse y aportar la documentación que sustente su petición.

El Oficial de Privacidad responderá y dará seguimiento a su Solicitud de Derechos en los plazos establecidos por la Ley. De igual manera, Usted podrá revocar su consentimiento para el tratamiento de sus datos personales siguiendo el mismo procedimiento que el mencionado para el ejercicio de sus Derechos.

VII. Cambios o modificaciones al presente aviso

El responsable se reserva el derecho de efectuar en cualquier momento modificaciones o actualizaciones al presente Aviso, en el entendido de que toda modificación al mismo se hará de su conocimiento por correo electrónico. Solicitamos su consentimiento expreso para el tratamiento de sus datos personales. Por favor seleccione "Si acepto las condiciones descritas anteriormente" al final del aviso de privacidad, o bien "No acepto las condiciones descritas anteriormente".

Apéndice C. Teoría - Tamaño de la muestra

La probabilidad de que se conteste una encuesta sin ser tendencioso hacia el objetivo de la encuesta determina las variables de P y Q, que en este caso se considerarán con un valor de 0.5, a fin de que se cumpla la igualdad que se establece en todo proceso estocástico.

$$P + Q = 1$$

Donde:

- Siendo “1” la Probabilidad de todo el Universo
- P = Probabilidad de que sucedan los eventos
- Q = Probabilidad de que no sucedan los eventos

Para determinar el número de encuestas que se deberían de aplicar,

- **Para una población finita (<=100,000 habitantes)**

$$n = \frac{k^2 P Q N}{e^2(N-1) + k^2 P Q}$$

- **Para una población infinita (>100,000 habitantes)**

Se toma el límite cuando $N \rightarrow \infty$

$$n = \frac{\frac{k^2 P Q N}{N}}{\frac{e^2(N-1)}{N} + \frac{k^2 P Q}{N}} = \frac{k^2 P Q}{e^2}$$

Como se observa, la expresión se reduce para muestras infinitas.

Donde:

- N = Total tamaño de la población
- k= Nivel de confianza. Que los individuos contesten con certeza siempre y cuando hayan entendido la pregunta.
- P = proporción de individuos que poseen en la población la característica de estudio esperada (en este caso 50% = 0.5).
- Q = proporción de individuos que no poseen en la población la característica de estudio esperada (en este caso 50% = 0.5).
- e = Margen de error en porcentaje, es un error que se puede arrastrar por entrevistar a una persona que no corresponda a la población, o una mala interpretación de lo pregunta, diseño de esta junto con sus respuestas. Puede tener valores de $0 < e < 1$. Aunque se recomienda que este valor sea entre 5% y 10%. Entre menor sea el margen de error, mayor será el tamaño de la muestra.
- n = Tamaño de la muestra.

Para determinar el nivel de confianza, se emplean las tablas de probabilidad de distribución normal, de donde obtenemos la información que a continuación se cita:

Tabla 24. Tabla de probabilidad de distribución

%	k
75.0	1.15
80.0	1.28
85.0	1.49
90.0	1.65
95.0	1.96
95.5	2.00
97.5	2.24
99.0	2.58

Apéndice D. Encuesta a turistas

Datos estadísticos

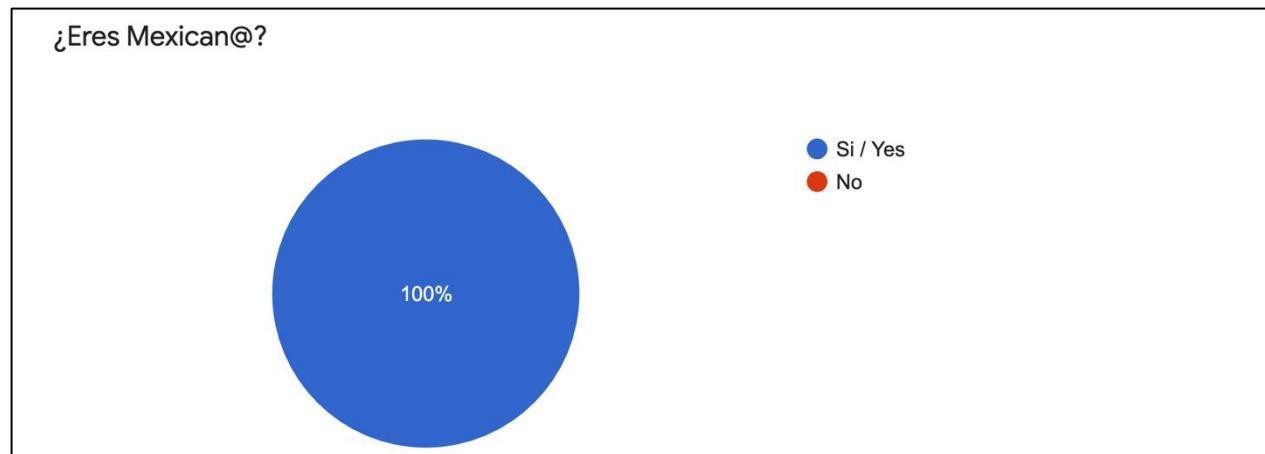


Figura 121. Encuesta a turistas - Pregunta #1

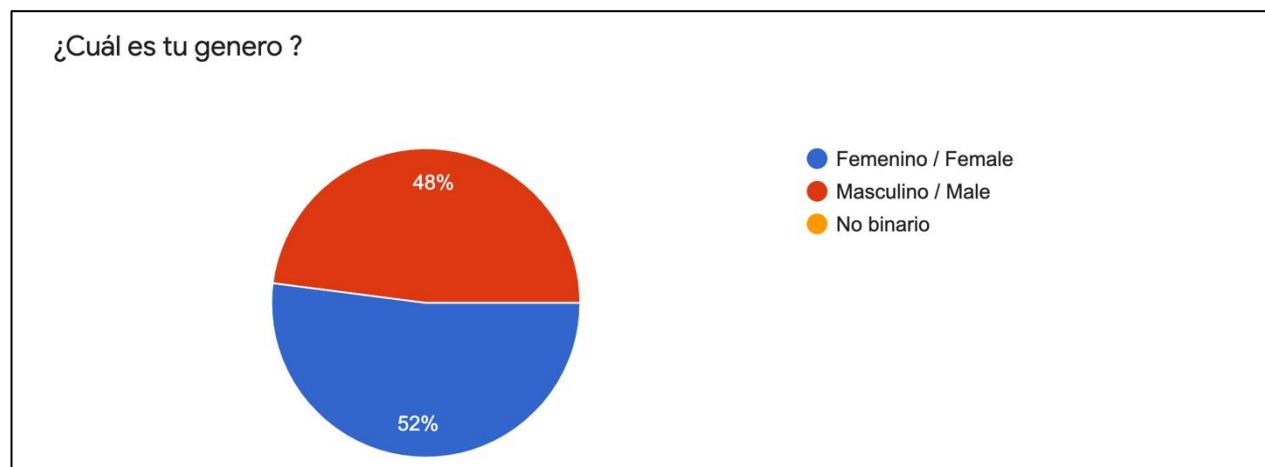


Figura 122. Encuesta a turistas - Pregunta #2

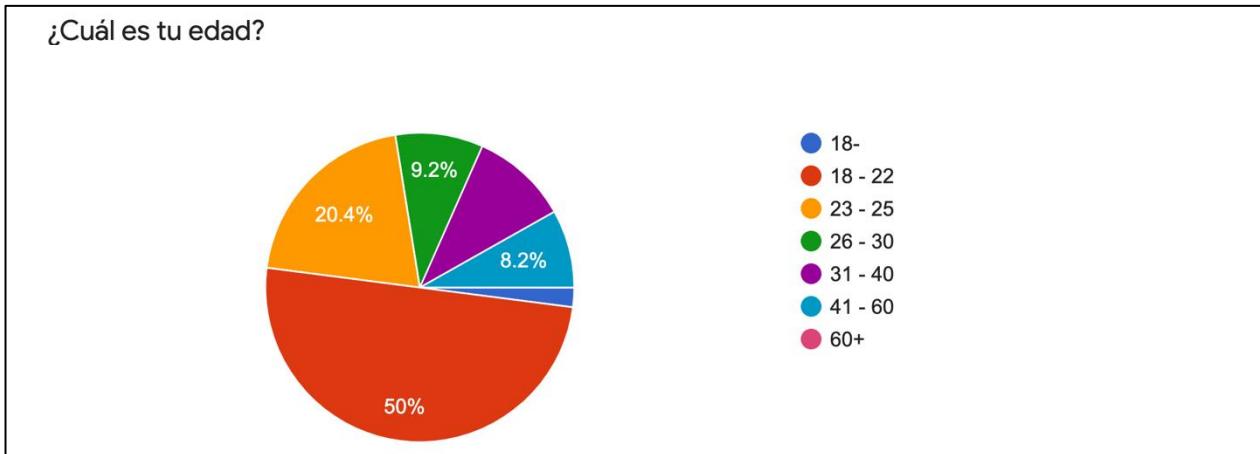


Figura 124. Encuesta a turistas - Pregunta #3

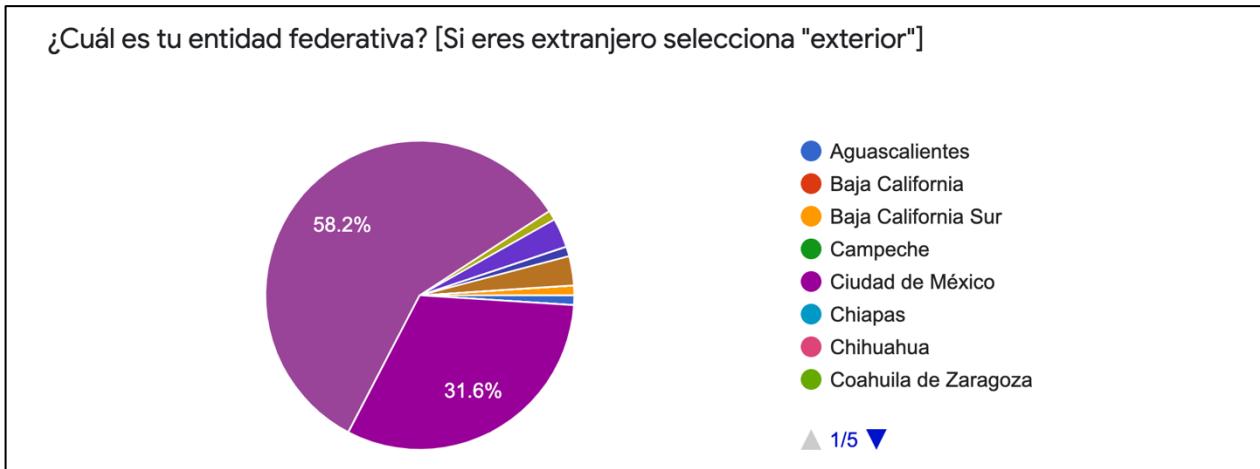


Figura 123. Encuesta a turistas - Pregunta #4

¿Has visitado Teotihuacán?

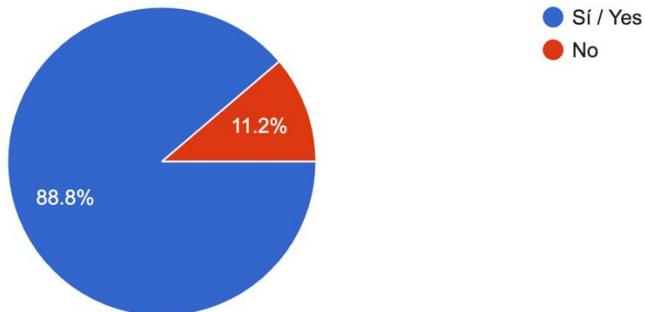


Figura 125. Encuesta a turistas - Pregunta #5

¿Conoces, has oido hablar o visitado los pueblos mágico y pueblos con encanto vecinos de Teotihuacán?

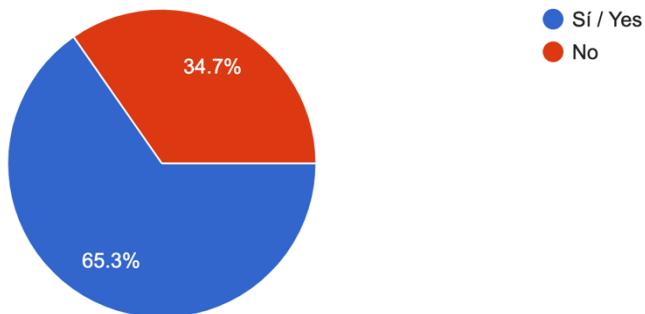


Figura 126. Encuesta a turistas - Pregunta #6

¿Cuántas veces has visitado Teotihuacán?

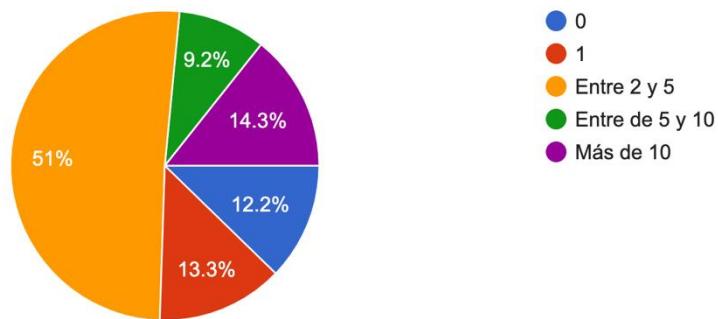


Figura 127. Encuesta a turistas - Pregunta #7

¿Cuáles de estos servicios te gustaría conocer o volver a disfrutar en Teotihuacán? [Puedes seleccionar más de uno]

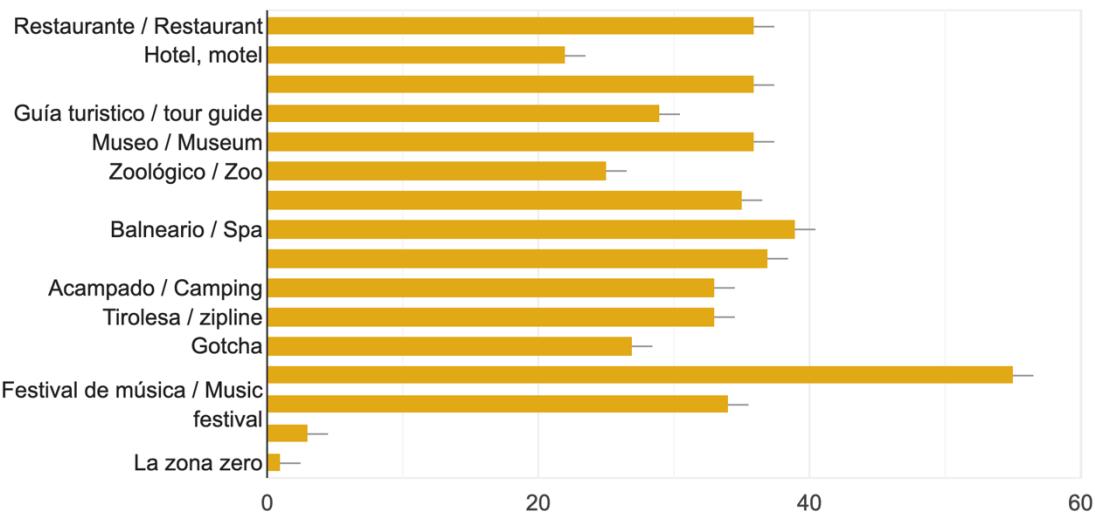


Figura 130. Encuesta a turistas - Pregunta #10

¿Has usado alguna de estas aplicaciones de viaje? [Puedes seleccionar más de una.]

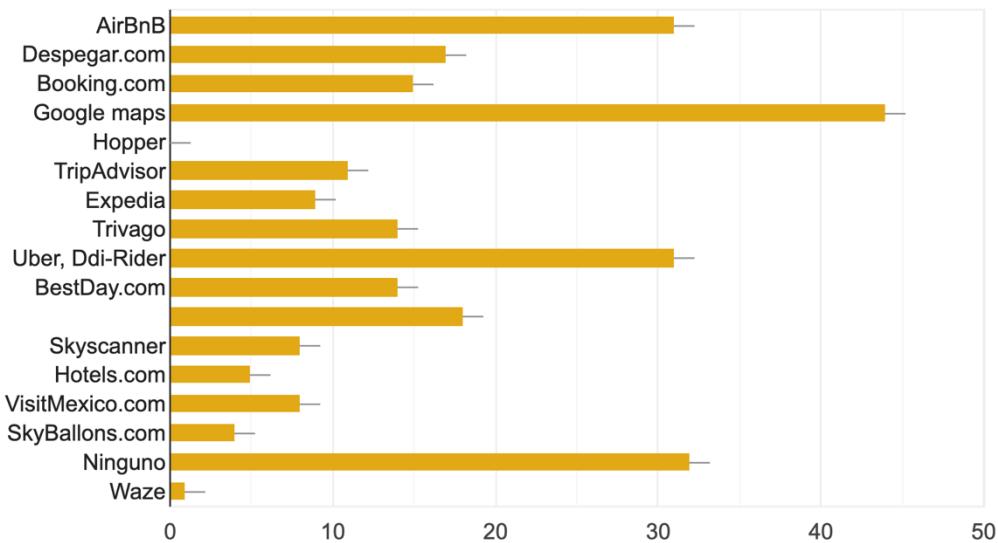


Figura 131. Encuesta a turistas - Pregunta #11

¿Qué medio de transporte utilizas o utilizarías para llegar a la zona de Teotihuacán?

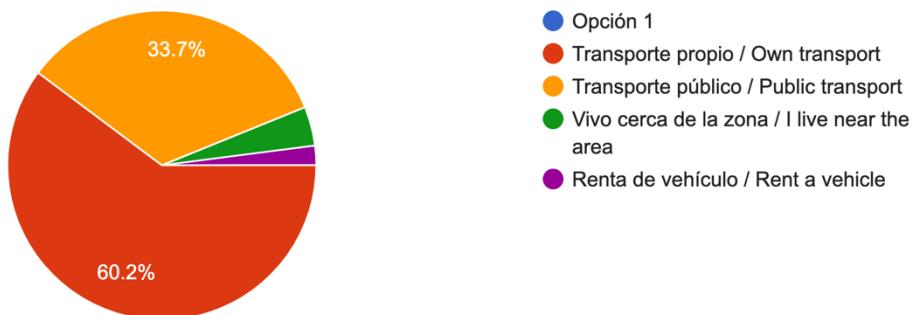


Figura 132. Encuesta a turistas - Pregunta #12

Una nueva opción

Si hubiese una aplicación móvil que reuniera la mayor cantidad de servicios turísticos que hay en la zona (No solo los lugares típicos), que te permitiera...d de tus compras, ¿La usarías en tu próximo viaje?

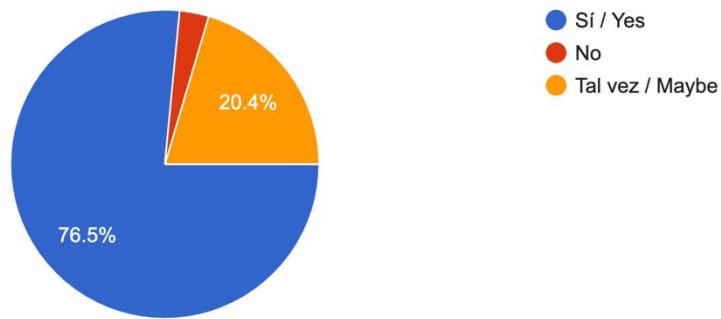


Figura 133. Encuesta a turistas - Pregunta #13

Cuando realizas algún tipo de pago por un servicio, en general ¿En qué momento prefieres pagar por el mismo?



Figura 134. Encuesta a turistas - Pregunta #14

¿Te gustaría que se hiciera realidad esta propuesta?

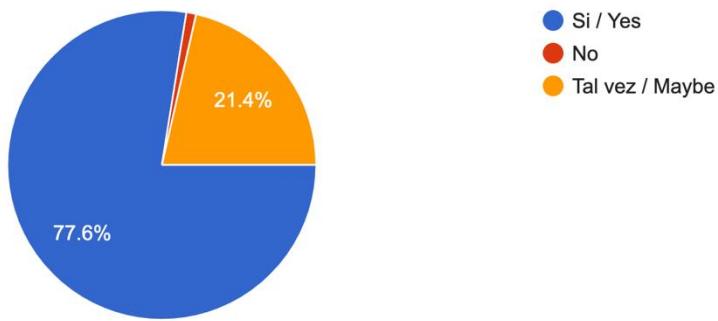


Figura 136. Encuesta a turistas - Pregunta #15

¿Qué sistema operativo usas en tu teléfono móvil?

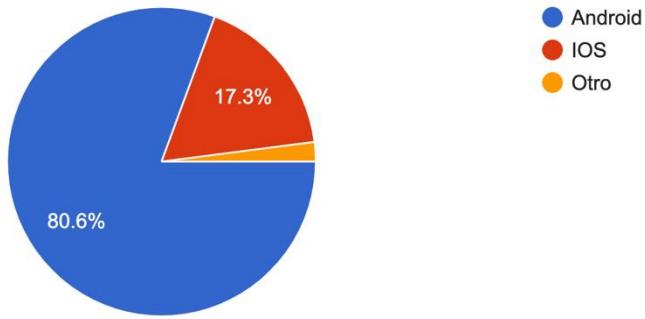


Figura 135. Encuesta a turistas - Pregunta #16

¿Cuál de las siguientes es más cómoda para ti?



Figura 137. Encuesta a turistas - Pregunta #17

¿Qué método de pago usas más?



Figura 138. Encuesta a turistas - Pregunta #18

¿Confías en los pagos electrónicos?

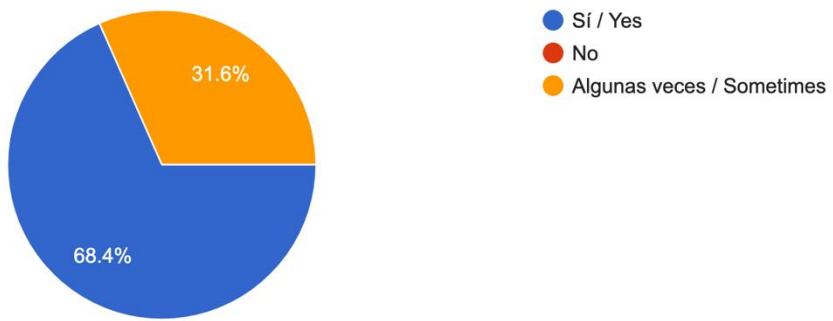


Figura 139. Encuesta a turistas - Pregunta #19

Extranjeros

Si conocieras a alguna persona extranjera que visita Teotihuacán, ¿Le sugerirías o recomendarías una aplicación como esta o similar? Considera que ...en si tú eres una persona extranjera, ¿La usarías?

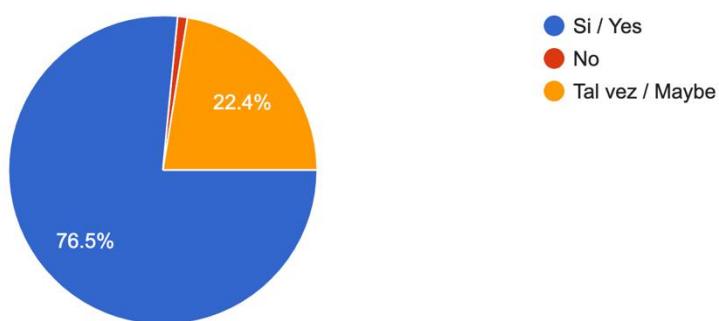


Figura 140. Encuesta a turistas - Pregunta #20

¿Tienes amigos o conocidos extranjeros?

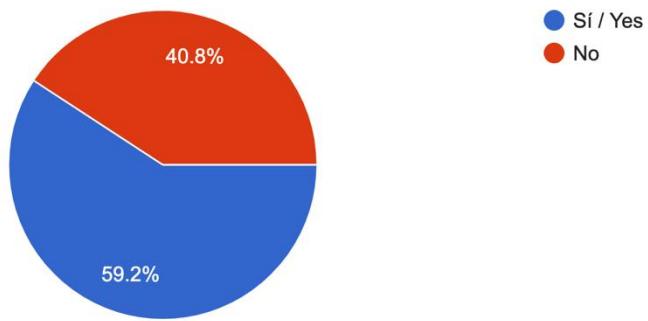


Figura 141. Encuesta a turistas - Pregunta #21

¿Y la pandemia?

¿Visitarías Teotihuacán mientras la pandemia COVID-19 es parte de nuestras vidas?

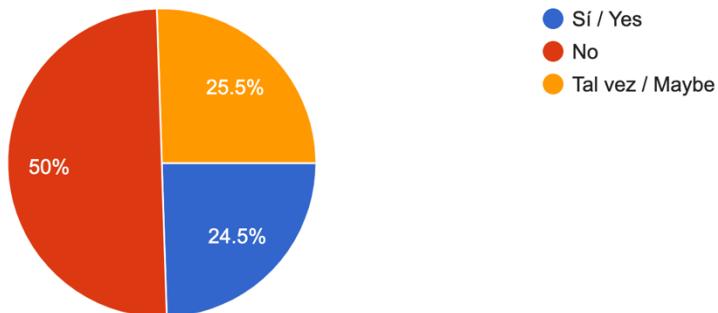


Figura 142. Encuesta a turistas - Pregunta #22

Algún día cuando la pandemia termine o no represente un peligro, ¿Te gustaría salir a conocer nuevos lugares, de vacaciones o realizar alguna actividad relacionada al turismo?

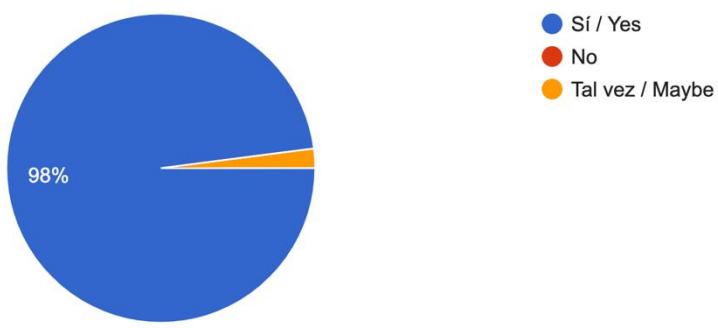


Figura 143. Encuesta a turistas - Pregunta #23

Apéndice E. Encuesta a vendedores

Sobre tu servicio

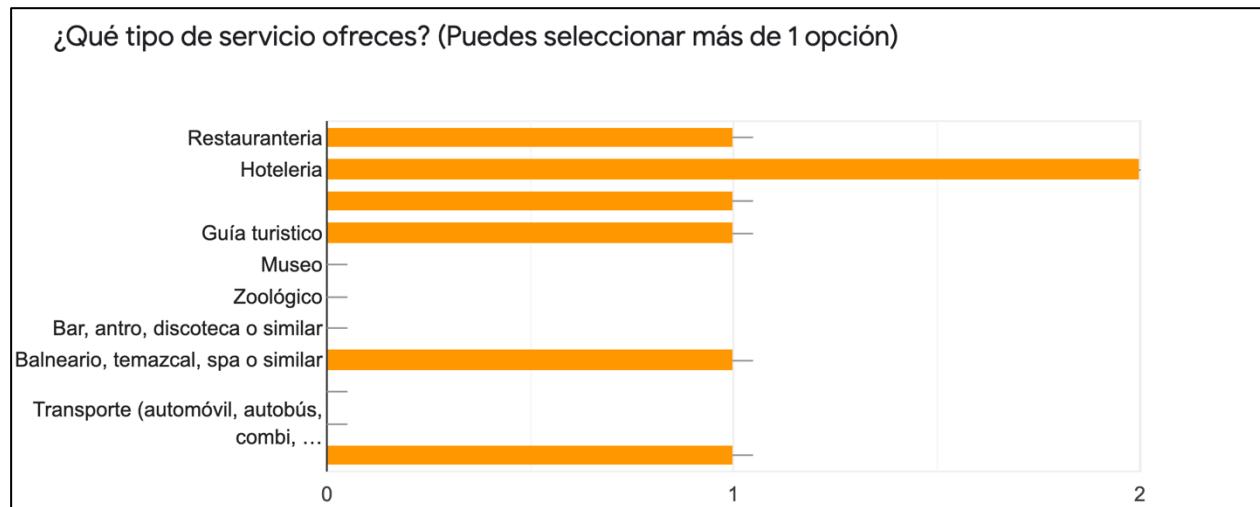


Figura 144. Encuesta a vendedores - Pregunta #01

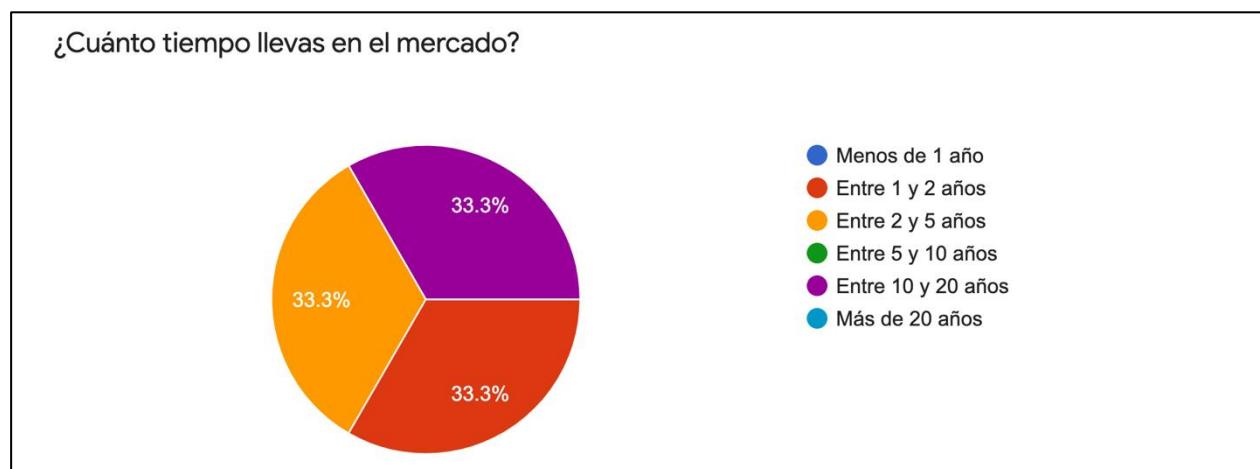


Figura 145. Encuesta a vendedores - Pregunta #02

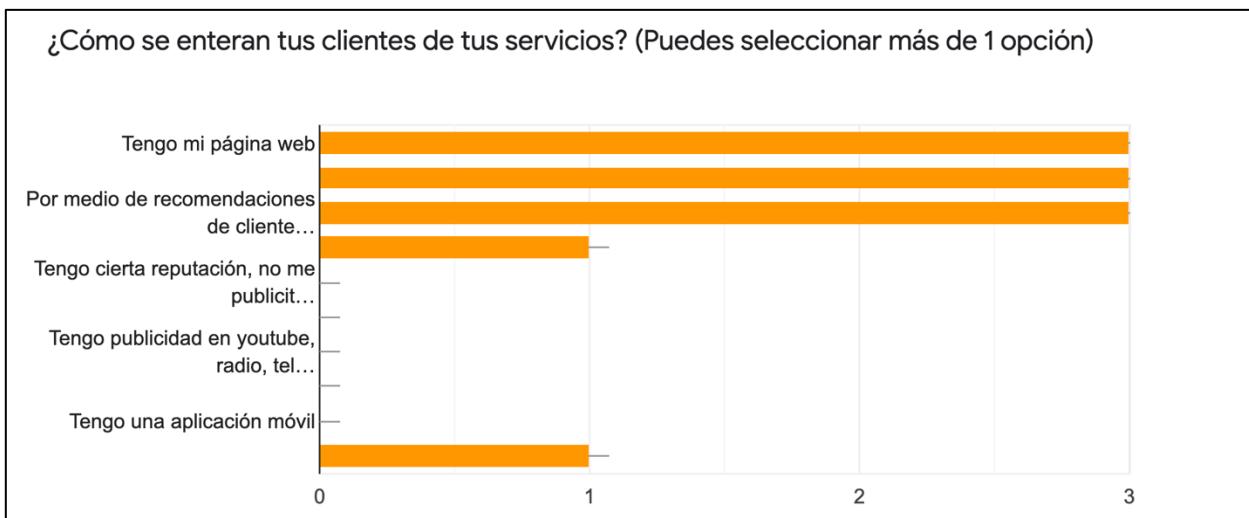


Figura 146. Encuesta a vendedores - Pregunta #03

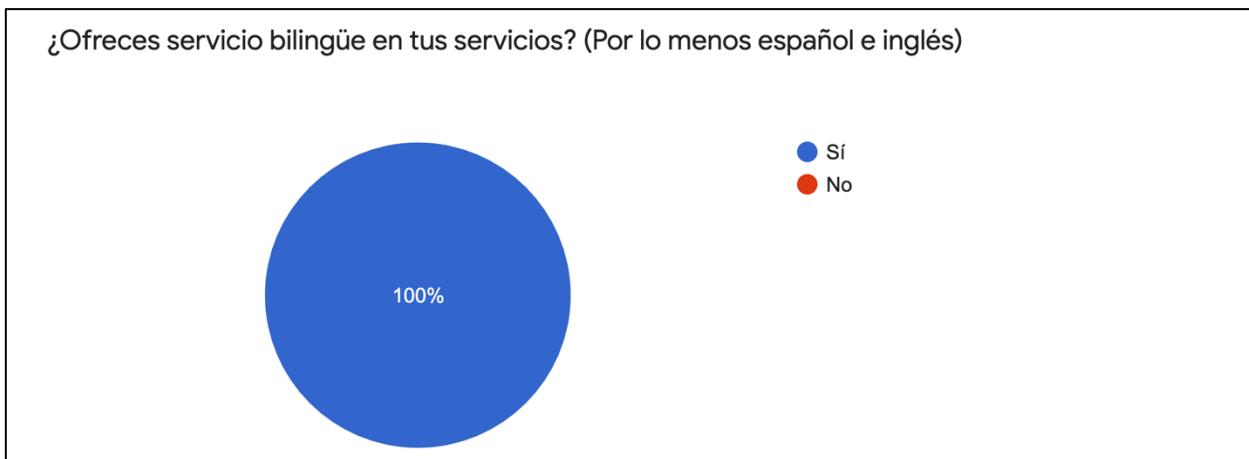


Figura 147. Encuesta a vendedores - Pregunta #04

¿Qué medios de pago utilizas?(Puedes seleccionar más de uno)

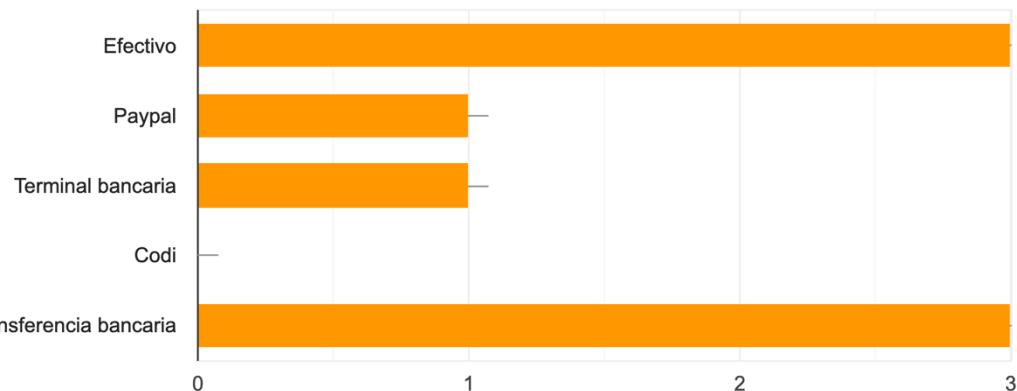


Figura 148. Encuesta a vendedores - Pregunta #05

¿En que municipio radicas?

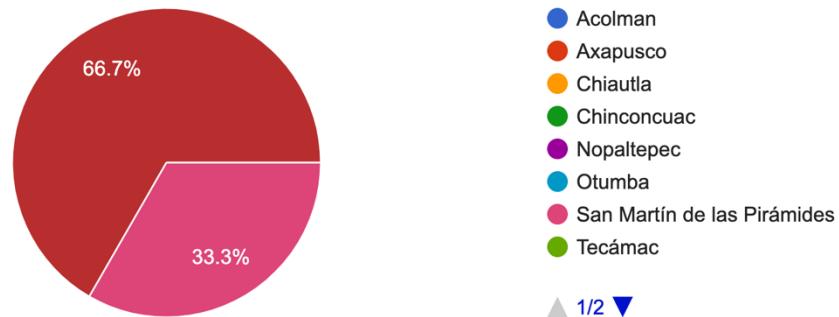


Figura 149. Encuesta a vendedores - Pregunta #06

¿Haces reservaciones o citas para tu servicio?



Figura 150. Encuesta a vendedores - Pregunta #07

¿Qué días abres tus servicios?

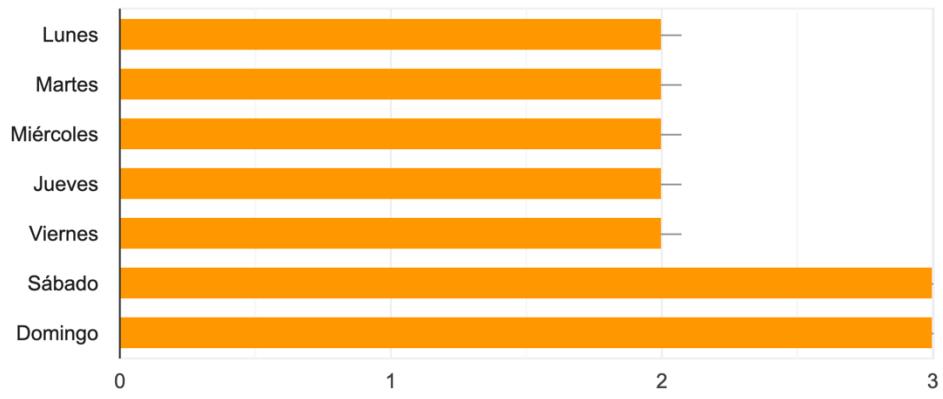


Figura 151. Encuesta a vendedores - Pregunta #08

¿Cuál de los siguientes incluye tu servicio? (Puedes seleccionar más de una opción)

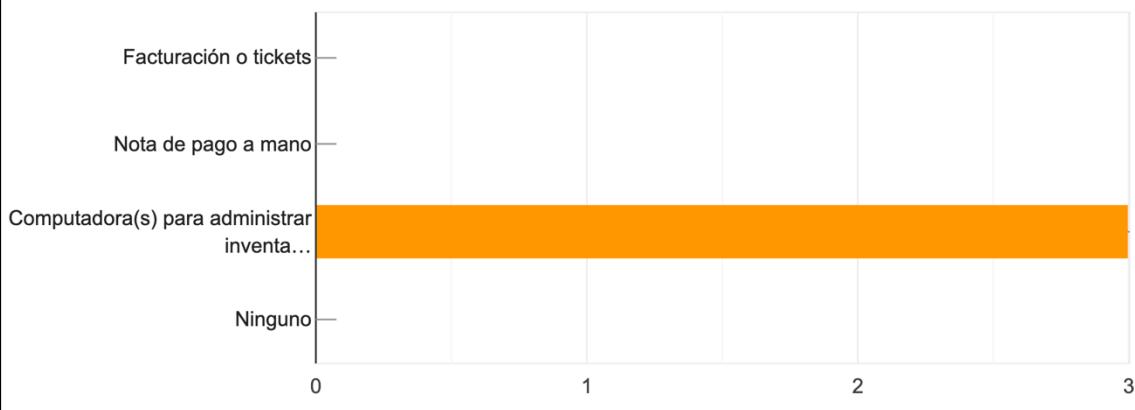


Figura 152. Encuesta a vendedores - Pregunta #09

¿Tu servicio esta registrado en google maps?

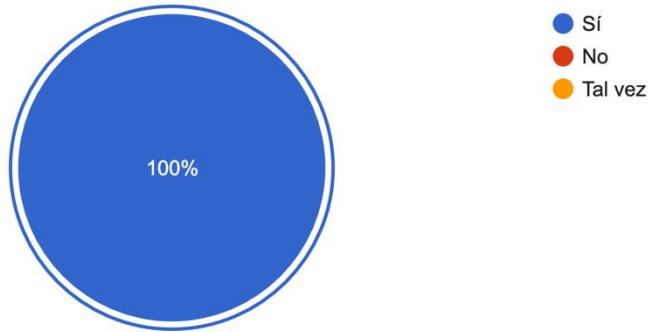


Figura 153. Encuesta a vendedores - Pregunta #10

Seguridad y confianza

¿Has sufrido algún tipo de fraude, falsificación de dinero o pagos no cubiertos por clientes ?

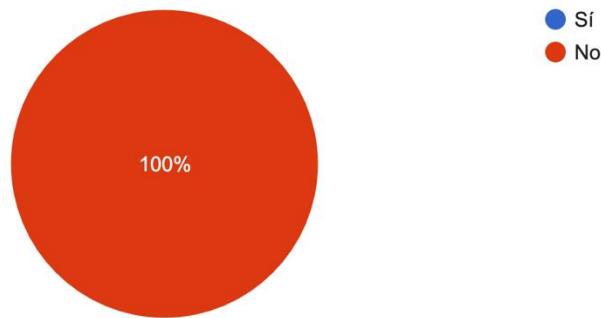


Figura 154. Encuesta a vendedores - Pregunta #11

¿En qué momento realizas el cobro de tus servicios?

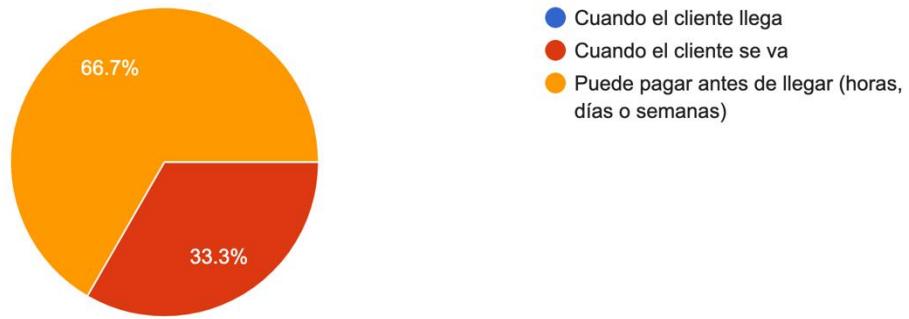


Figura 155. Encuesta a vendedores - Pregunta #12

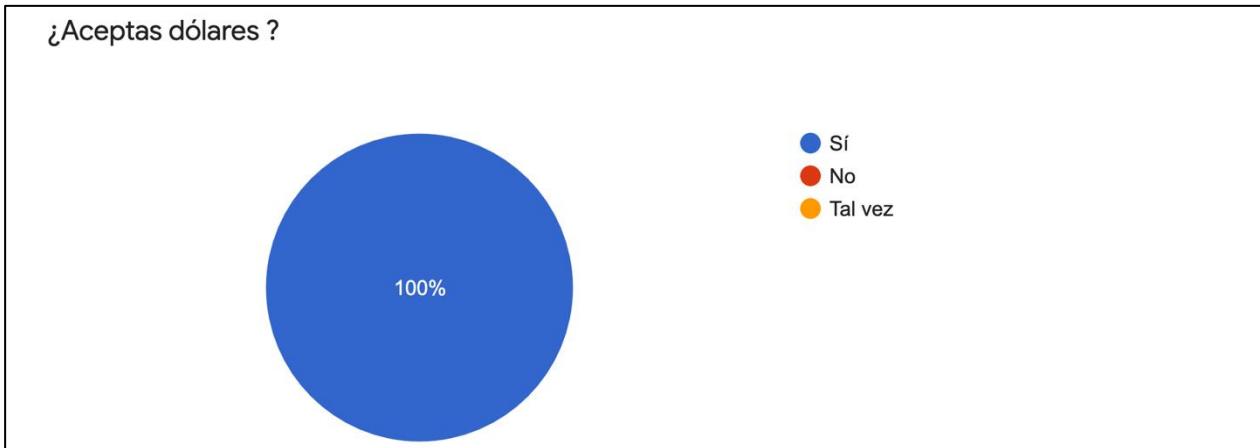


Figura 156. Encuesta a vendedores - Pregunta #13

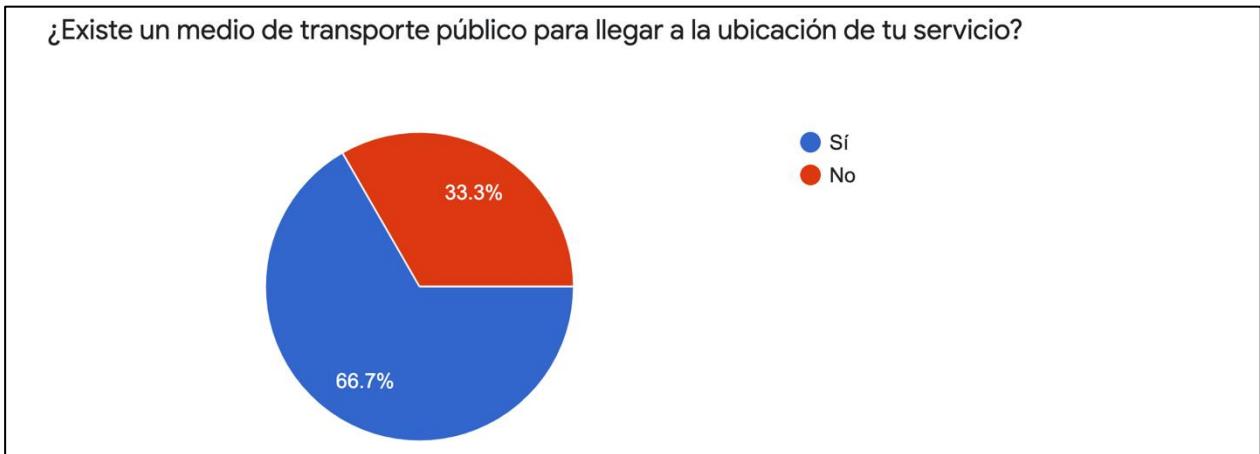


Figura 157. Encuesta a vendedores - Pregunta #14

Seguridad y confianza

De cada 10 turistas que atiendes, Aproximadamente ¿Cuántos son extranjeros?

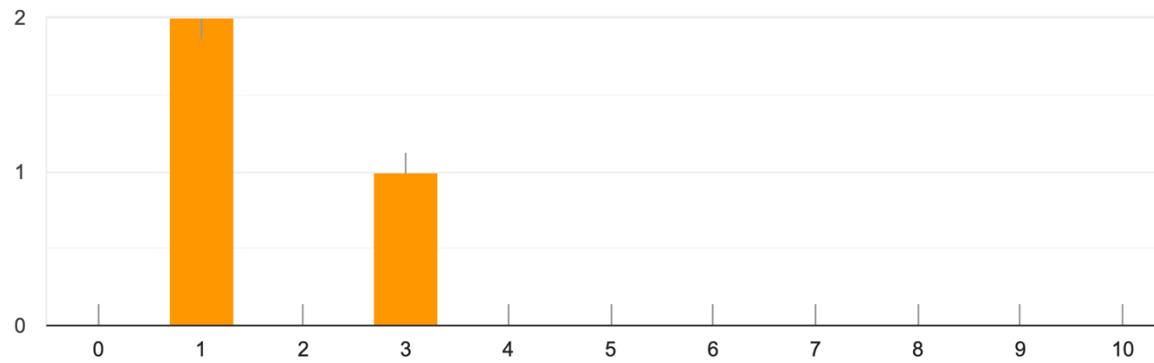


Figura 158. Encuesta a vendedores - Pregunta #15

De cada 10 turistas que atiendes, Aproximadamente ¿Cuántos son nacionales?

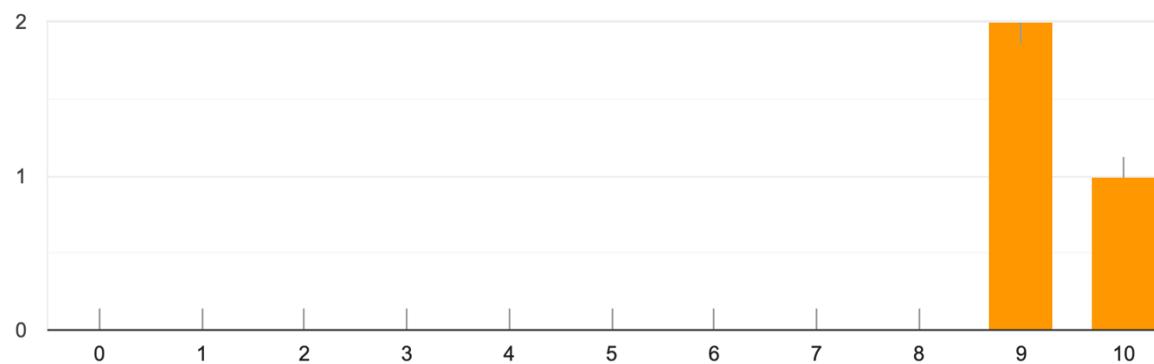


Figura 159. Encuesta a vendedores - Pregunta #16

¿Consideras que el precio que tienen tus servicios es accesible para una persona mexicana promedio?

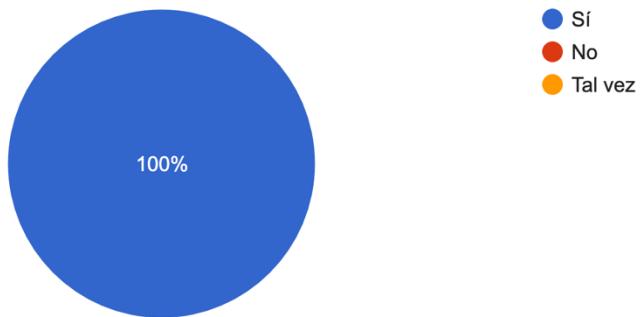


Figura 160. Encuesta a vendedores - Pregunta #17

Sobre la contingencia sanitaria COVID 19

Recuerda cuando el covid no existía, supón que en ese entonces tenías 10 clientes al día, ¿Cuántos clientes atiendes ahora en promedio en un día?

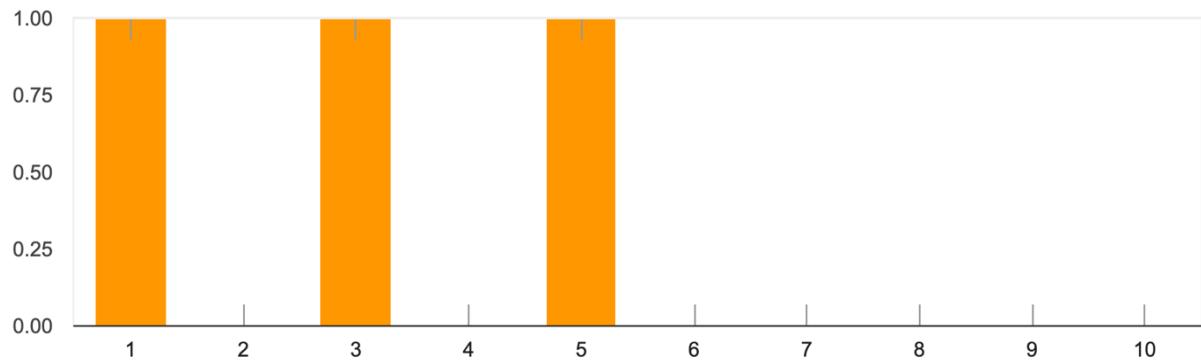


Figura 161. Encuesta a vendedores - Pregunta #18

¿Haz adecuado tus instalaciones para hacer frente a la pandemia?

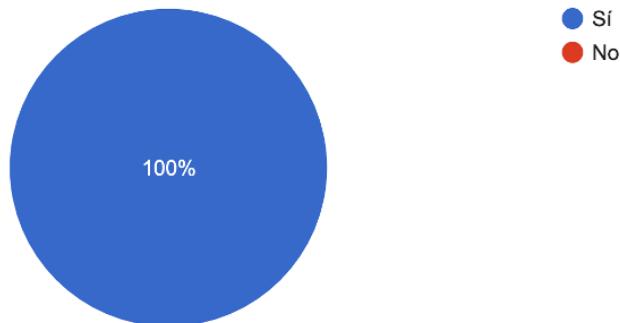


Figura 162. Encuesta a vendedores - Pregunta #19

¿El propietario del servicio, trabajadores o personal de apoyo se han contagiado de covid mientras laboran en tus instalaciones?

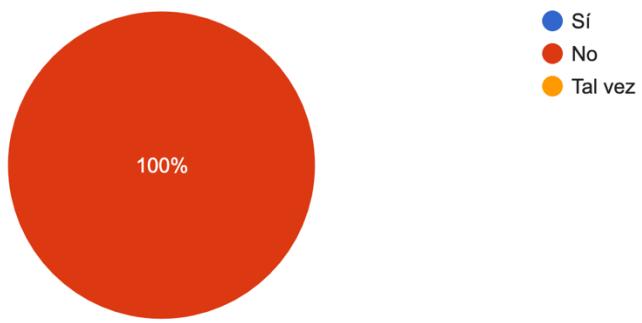


Figura 163. Encuesta a vendedores - Pregunta #20

Sobre una aplicación propuesta

Se propone una aplicación móvil IOS-Android, en la cual tu como vendedor puedes poner a la venta tus servicios turísticos en zona de Teotihuacán. Tu como empresa de cualquier tamaño puedes registrarte y comenzar a ofertar tus servicios. Los turistas nacionales e internacionales pueden ver tu información de los servicios que ofertas en inglés o español, pueden hacerte pagos electrónicos antes de llegar a tu locación o al momento en que consuman el servicio tu podrás solicitar el pago por medio de tu teléfono móvil mismo que con solo mostrar tu teléfono al turista, el sabría el monto y lo que le solicitas.

El objetivo de este proyecto es reunir la mayor cantidad de servicios turísticos de la zona, de tal manera que aumente la visibilidad de tus servicios, dar mayor confianza a los turistas facilitando la manera en que pagan y conocen tus servicios ¡Sin personas intermediarias!

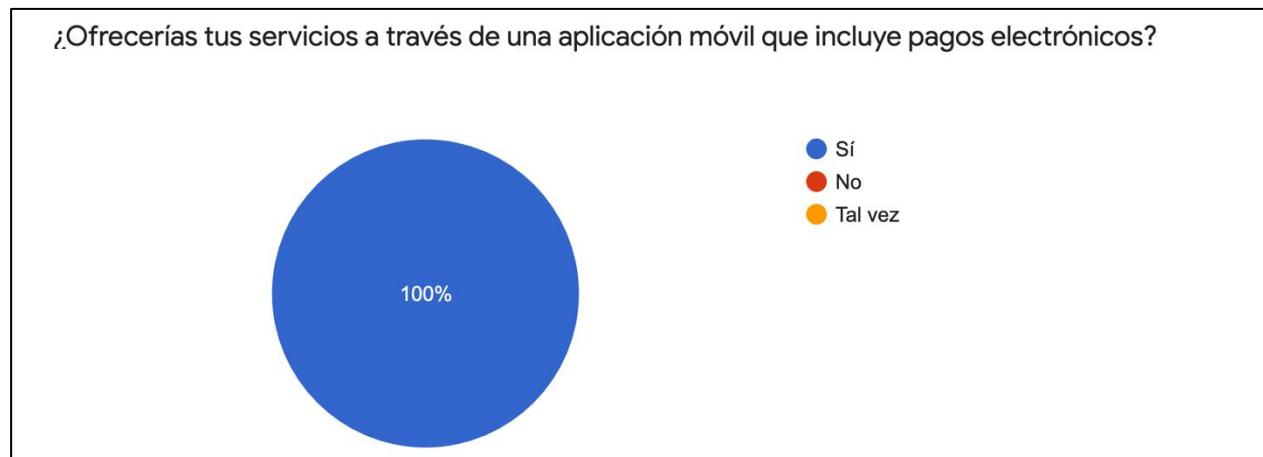


Figura 164. Encuesta a vendedores - Pregunta #21

Si ofrecieras tus servicios al utilizar una aplicación móvil ¿Qué plan de negocio es más rentable para tu empresa?



Figura 165. Encuesta a vendedores - Pregunta #22

¿Qué medio de pago es más eficiente para tu negocio o tienes más experiencia?

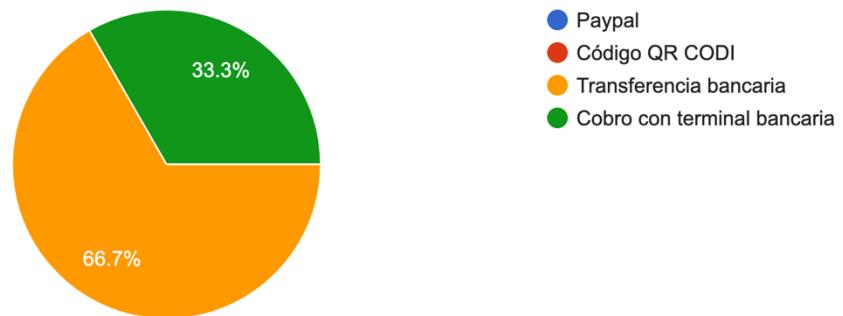


Figura 166. Encuesta a vendedores - Pregunta #23

¿Te interesa una aplicación como esta o similar? ¿Te interesaría que se hiciera realizad?

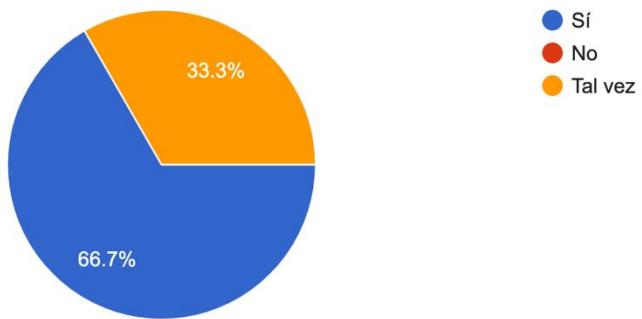


Figura 167. Encuesta a vendedores - Pregunta #24

¿Te interesa tener más clientes que sean turistas internacionales?

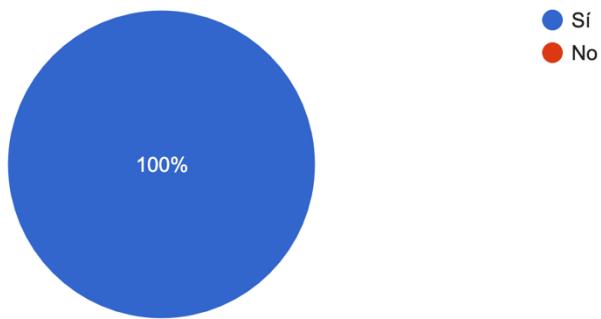


Figura 168. Encuesta a vendedores - Pregunta #25

