

Manual técnico

Aplicación R&G Espinoza

Fecha: 25/Septiembre/2019

# Tabla de contenido

[Tabla de contenido 2](#_Toc19027550)

[Tabla de Ilustraciones 4](#_Toc19027551)

[Herramientas utilizadas para la creación de la aplicación 5](#_Toc19027552)

[Configuraciones previas para trabajar con la aplicación 6](#_Toc19027553)

[Configuraciones previas para subir la aplicación en Testflight 7](#_Toc19027554)

[Recorrido de la aplicación (flujo de pantallas) 12](#_Toc19027555)

[Pantallas de la aplicación 13](#_Toc19027556)

[Ajustador Sin Siniestro Seleccionado 14](#_Toc19027557)

[Ajustador Con Siniestro 18](#_Toc19027558)

[Definición de funciones 37](#_Toc19027559)

[main.js 37](#_Toc19027560)

[utils.js 39](#_Toc19027561)

[login.js 41](#_Toc19027562)

[datosUsuario.js 41](#_Toc19027563)

[baseDatos.js 41](#_Toc19027564)

[bitacora.js 42](#_Toc19027565)

[Nombre 42](#_Toc19027566)

[Parámetros 42](#_Toc19027567)

[Salida 42](#_Toc19027568)

[Descripción 42](#_Toc19027569)

[Buscar.js 44](#_Toc19027570)

[catalagos.js 44](#_Toc19027571)

[datos.js 45](#_Toc19027572)

[documento.js 45](#_Toc19027573)

[fotografias.js 45](#_Toc19027574)

[gastos.js 47](#_Toc19027575)

[inspeccion.js 49](#_Toc19027576)

[pdf.js 51](#_Toc19027577)

[peticiones.js 51](#_Toc19027578)

[ref.js 52](#_Toc19027579)

[uploadFile.js 52](#_Toc19027580)

[Servicios 53](#_Toc19027581)

[Servicios buscar.js 53](#_Toc19027582)

[Servicios catalogos.js 53](#_Toc19027583)

[Servicios datos.js 53](#_Toc19027584)

[Servicios documento.js 53](#_Toc19027585)

[Servicios fotografias.js 53](#_Toc19027586)

[Servicios gastos.js 54](#_Toc19027587)

[Servicios inspeccion.js 55](#_Toc19027588)

[Servicios login.js 56](#_Toc19027589)

[Servicios pdf.js 56](#_Toc19027590)

[Servicios ref.js 56](#_Toc19027591)

[Servicios uploadFile.js 56](#_Toc19027592)

[Anexos 57](#_Toc19027593)

[Control de cambios 57](#_Toc19027594)

# Tabla de Ilustraciones

[Ilustración 1 Flujo de aplicación 12](#_Toc19027525)

[Ilustración 2 Pantalla Inicio 13](#_Toc19027526)

[Ilustración 3 Pantalla principal 14](#_Toc19027527)

[Ilustración 4 Pantalla resultados de búsqueda 15](#_Toc19027528)

[Ilustración 5 Pantalla Gastos del ajustador 16](#_Toc19027529)

[Ilustración 6 Pantalla Buscar inspección sin siniestro 17](#_Toc19027530)

[Ilustración 7 Pantalla Barra de progreso 18](#_Toc19027531)

[Ilustración 8 Pantalla Ref R&G 19](#_Toc19027532)

[Ilustración 10 Pantalla Datos del siniestro 20](#_Toc19027533)

[Ilustración 11 Pantalla Gastos del siniestro 21](#_Toc19027534)

[Ilustración 12 Pantalla Nuevo reporte de gastos 22](#_Toc19027535)

[Ilustración 13 Pantalla Reporte de gasto seleccionado 23](#_Toc19027536)

[Ilustración 14 Pantallas detalle de gasto 1 y 2 24](#_Toc19027537)

[Ilustración 15 Pantalla Enviar reporte de gastos 25](#_Toc19027538)

[Ilustración 16 Pantalla Bitácora 26](#_Toc19027539)

[Ilustración 17 Pantalla Bitácora facturable 27](#_Toc19027540)

[Ilustración 18 Pantalla Bitácora No facturable 28](#_Toc19027541)

[Ilustración 19 Pantalla Inspección 29](#_Toc19027542)

[Ilustración 20 Pantalla Crear nueva inspección 30](#_Toc19027543)

[Ilustración 21 Pantalla Inspección seleccionada 31](#_Toc19027544)

[Ilustración 22 Pantalla Documento 32](#_Toc19027545)

[Ilustración 23 Pantalla Fotografías 33](#_Toc19027546)

[Ilustración 24 Pantalla Cargar fotografías 34](#_Toc19027547)

[Ilustración 25 Pantalla agregar más fotografías 35](#_Toc19027548)

[Ilustración 26 Pantalla Subir PDF firmado 36](#_Toc19027549)

# Herramientas utilizadas para la creación de la aplicación

Los frameworks utilizados para la construcción fue principalmente Apache Cordova, con algunas extensiones o plugins de Phonegap. Para mayor funcionalidad y dinamismo se utilizaron las librerías de jQuery, jQuery Mobile, onsenUi. Para poder funcionar en aplicaciones IOS, se utilizó la herramienta Xcode para la compilación del proyecto.

|  |  |
| --- | --- |
| **Framework / Herramienta** | **Versión** |
| * Cordova | 9.0.0 |
| * Android SDK tools | 19 - 28.0.1 |
| * jQuery | 1.11.1 |
| * jQuery Mobile | 1.4.5 |
| * OnsenUi | 2.0 |
| * Moment Js | 2.24.0 |
| * Numeral Js | 2.0.6 |
| * Xcode | 10.3 |

Tabla 1 Herramientas utilizadas

# Configuraciones previas para trabajar con la aplicación

**Plugins / Complementos:** Se agregaron plugins especiales en la aplicación, según sus necesidades o requerimientos, para su mayor funcionalidad

|  |  |
| --- | --- |
| **Nombre de Plugin** | **Descripción** |
| Cordova-plugin-camera | Este complemento permite acceder a la cámara del dispositivo. |
| Cordova-plugin-device | Este complemento se utiliza para poder acceder ciertas funciones del dispositivo como las acciones o gestos que realiza el usuario. |
| Cordova-plugin-file | Este complemento permite crear, leer y manipular archivos dentro del dispositivo. |
| Cordova-plugin-file-transfer | Este complemento se utiliza para cargar archivos a un servicio REST para descargar un archivo de parte del servidor. |
| Cordova-plugin-file-opener2 | Permite abrir un archivo utilizando su uri dentro del dispositivo. |
| Cordova-plugin-network-information | Este complemento permite conocer información detallada acerca de la conexión a internet que tiene el dispositivo actual. |
| Cordova-plugin-whitelist | Permite manipular la navegación entre páginas de la aplicación. |
| Cordova-plugin-filepath (**Android**) | Permite convertir la uri de un documento / archivo a su url permanente dentro del dispositivo |
| Cordova-plugin-filepicker (**iOS**) | Permite seleccionar los archivos como PDF |
| maurodx-cordova-plugin-file-selector | Permite seleccionar archivos del icloud. |

Tabla 2 Configuración de plugins

**Iconos IOS**: Las siguientes opciones controlan el uso de iconos de la aplicación

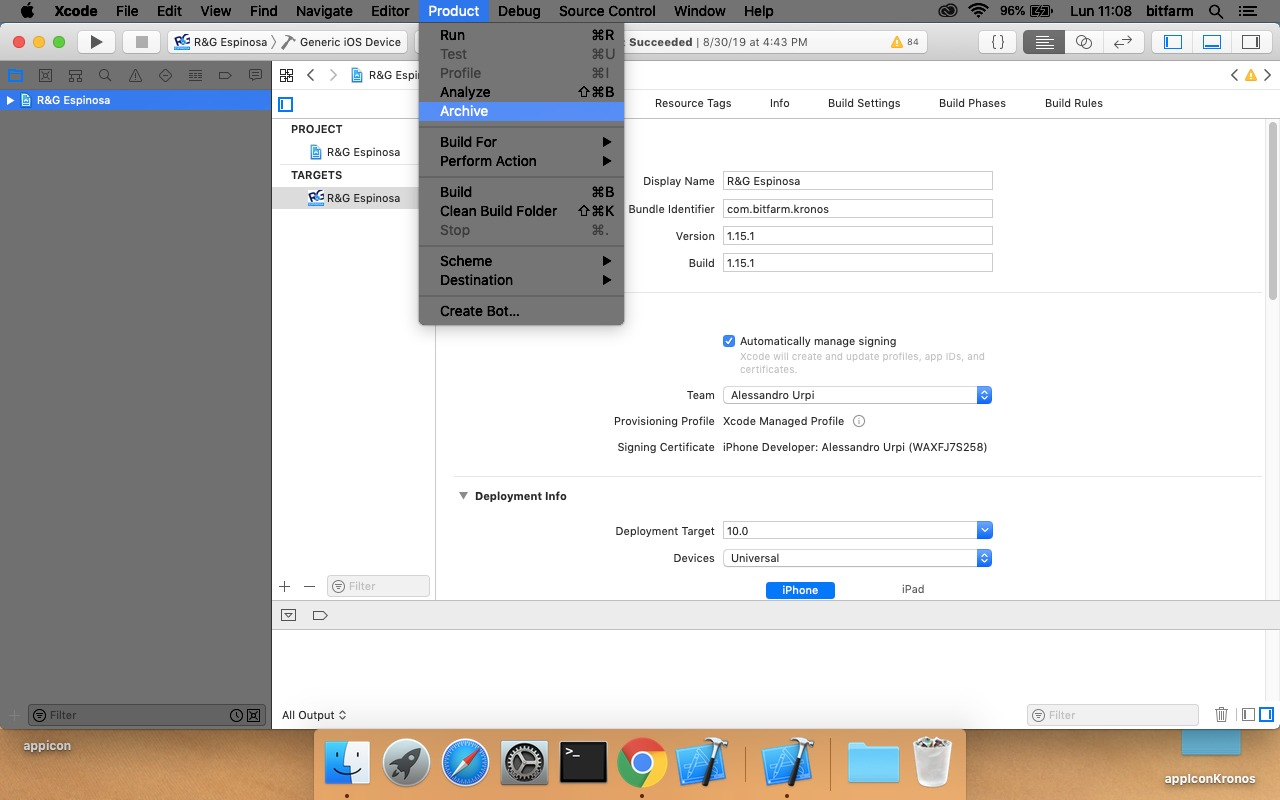
|  |  |
| --- | --- |
| **Targets** | **Descripción** |
| **App Icons Source** | En esta opción escogemos el icono de la aplicación |
| **Launch Images Source** | En esta opción escogemos una imagen que mostrara siempre al inicio de la aplicación |

Tabla 3 Configuración de complementos

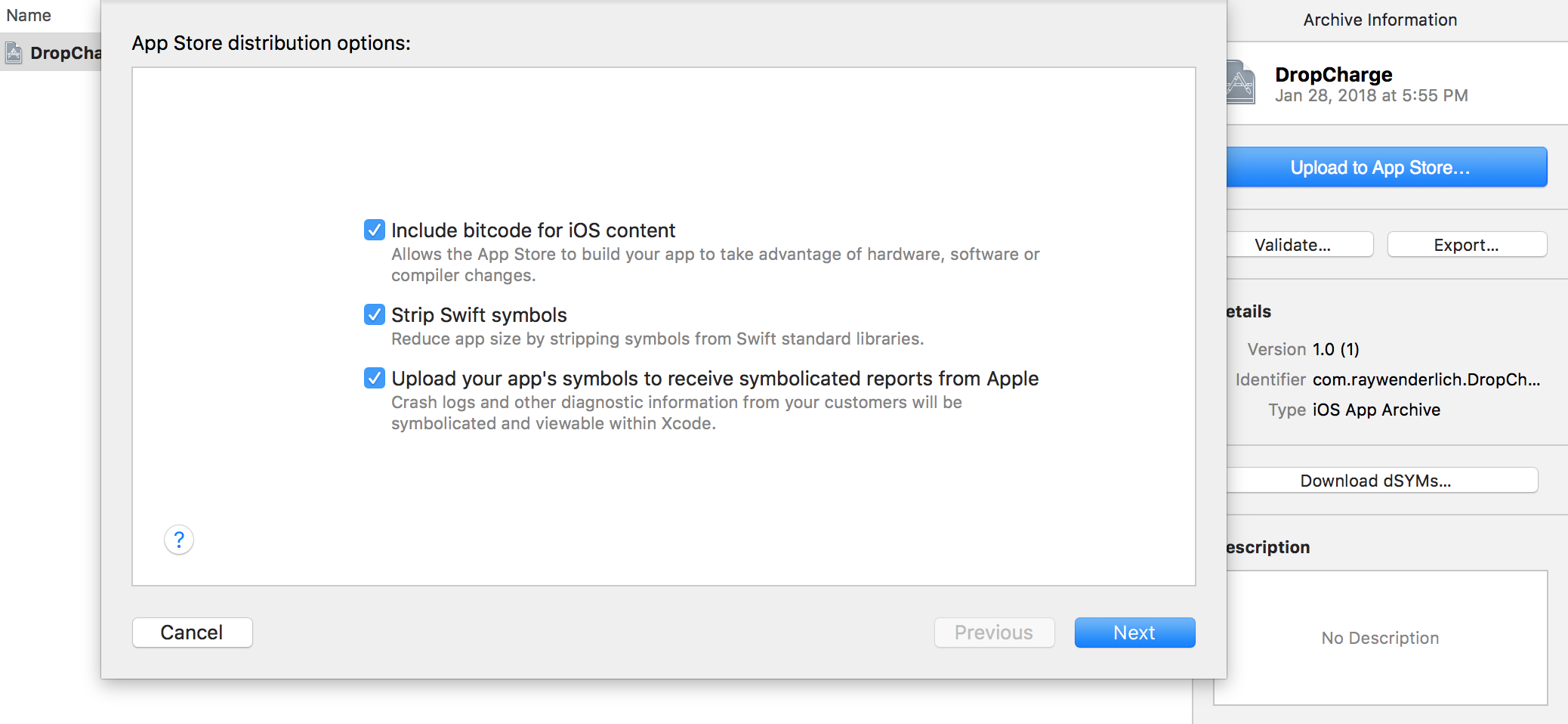
# Configuraciones previas para subir la aplicación en Testflight

Para poder probar la aplicación es necesario subirla a la app store y permitir realizar compilaciones por parte de la herramienta **TestFlight**, a continuación, se mostrarán los pasos para realizar este proceso.

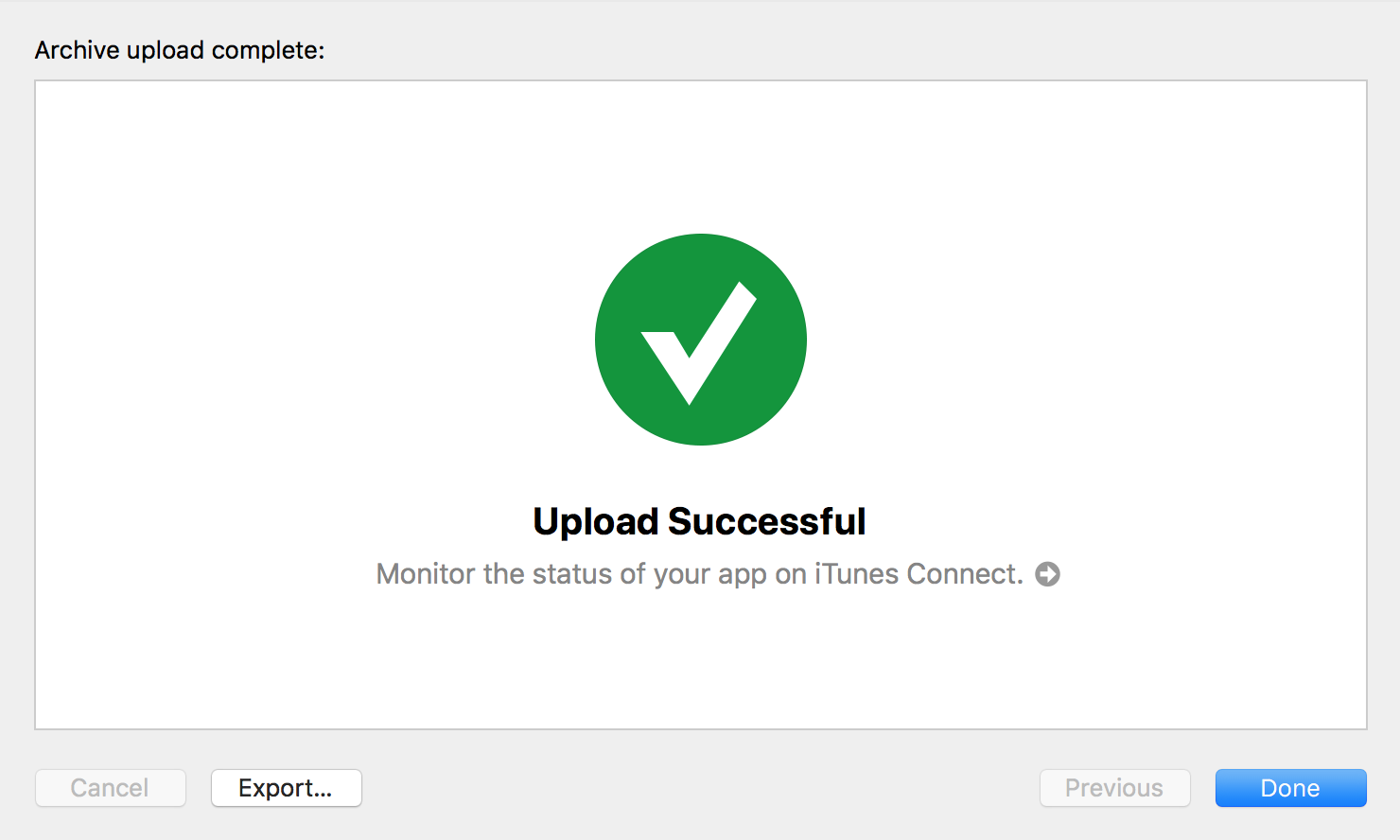
Primero debemos de seleccionar la aplicación y en el apartado de los emuladores seleccionar la opción de **Generic IOS Device** para después dirigirnos a la opción de **Product** y por último **Archive**



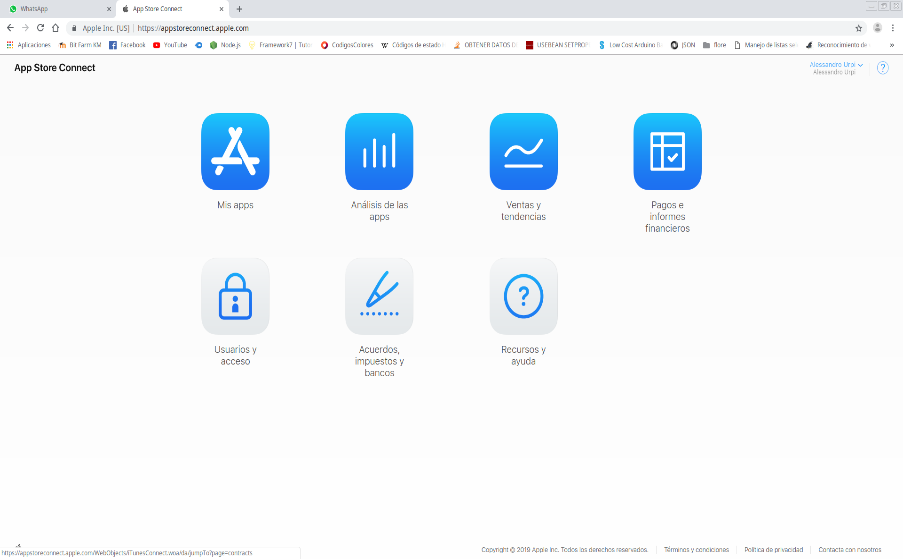
Una vez ya realizando lo anterior nos mostrara una pantalla como la siguiente en donde debemos seleccionar la opción de Upload to App Store para subir la aplicación a la tienda de Apple, cabe aclarar que este proceso suele ser tardado por lo que se recomienda no modificar el proyecto mientras termine de subirse la aplicación.



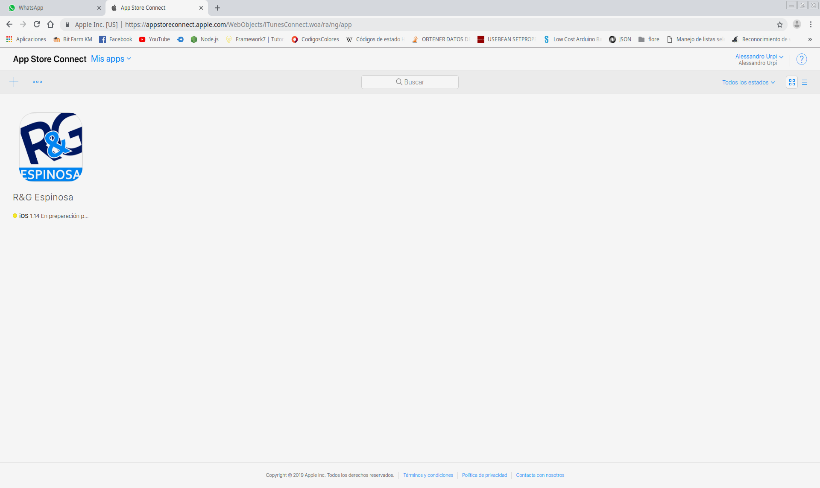
Cuando termine de subirse la aplicación nos mostrara una pantalla como la siguiente el cual nos indica que la aplicación se ha subido correctamente y que podemos dirigirnos a la página de https://appstoreconnect.apple.com/ para vincular la aplicación con TestFlight.



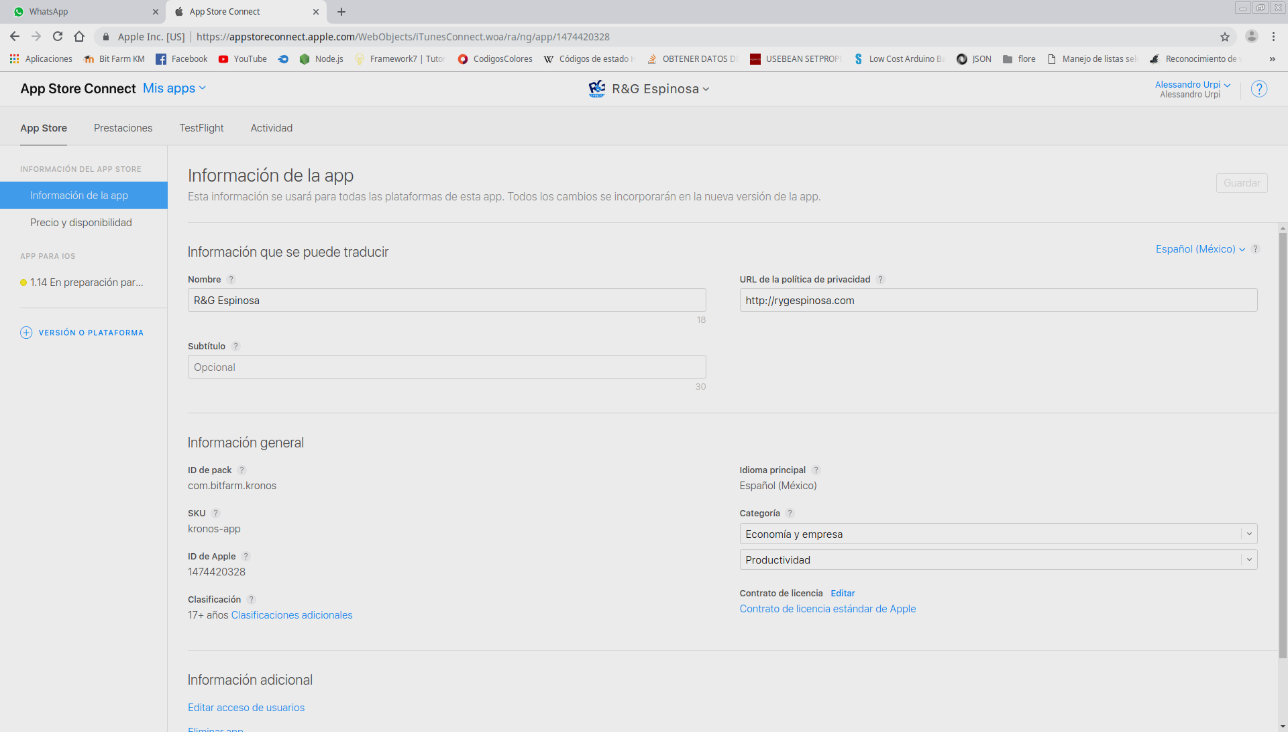
Lo más probable es de que cuando entremos a la liga que esta anteriormente nos solicite nuestro correo y contraseña de la cuenta de app store, al iniciar veremos una pantalla como la siguiente en donde nos vamos a dirigir a la primera opción de **Mis apps.**



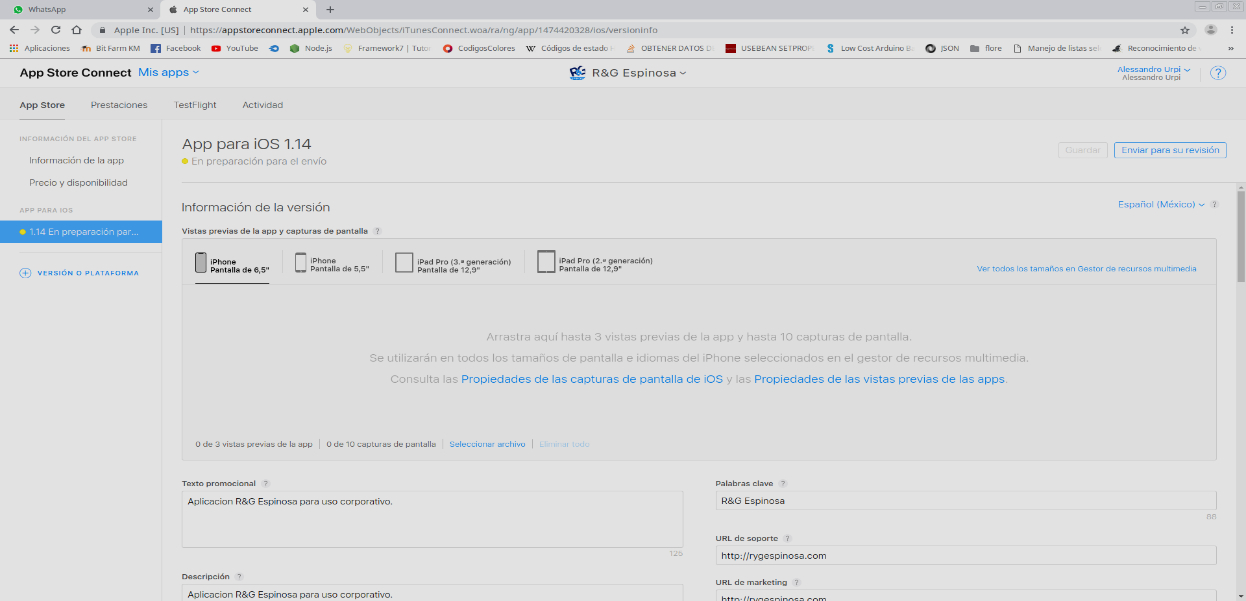
Seleccionamos nuestra aplicación

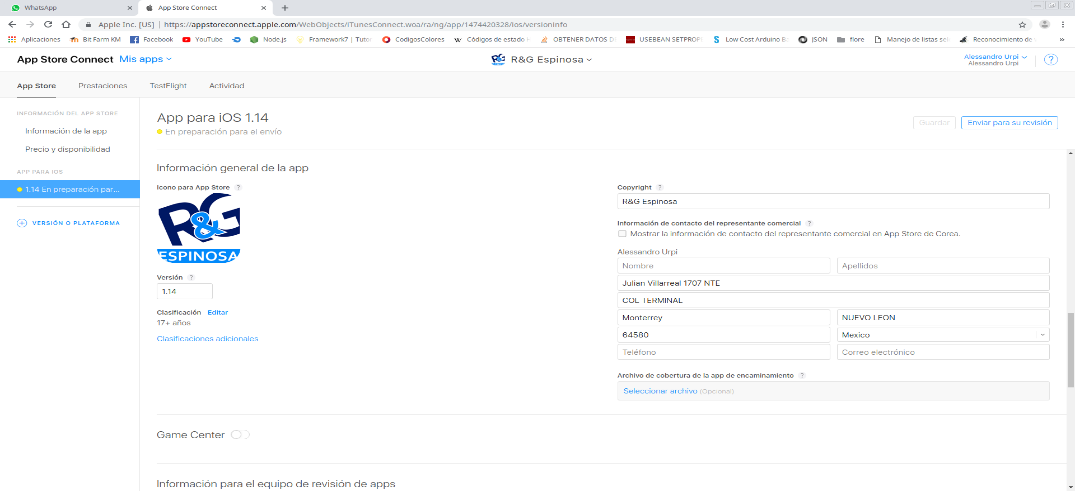


A partir de aquí la aplicación se divide en varias partes, la primea sección **Información de la app** es en donde llenaremos todos los datos importantes que nos solicita para poder registrar la aplicación, cuando terminemos de llenar toda nuestra información daremos clic en **Guardar,** como se muestra en la siguiente imagen.

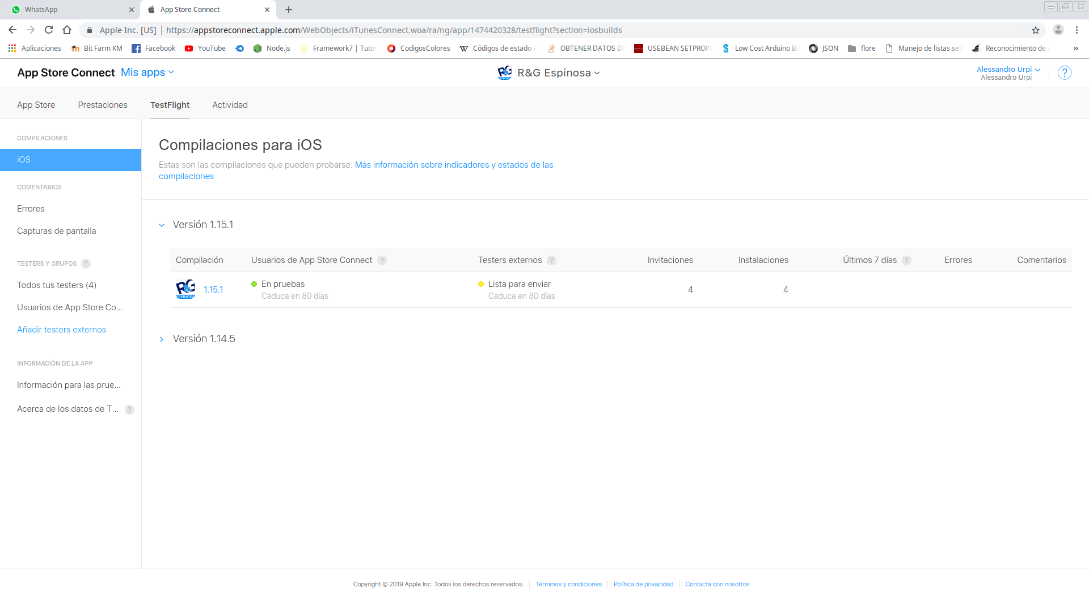


Después en la sección de **(Versión) En preparación para el envío** debemos de especificar más a detalle datos respecto al vendedor, también debemos de declarar la versión de la aplicación, que tipo de clasificación se considera y respectivas fotografías para la App Store, cuando se termine de llenar todo lo que se solicita, daremos clic en **Guardar** y **Enviar para su revisión**, como se muestra en la siguiente imagen.

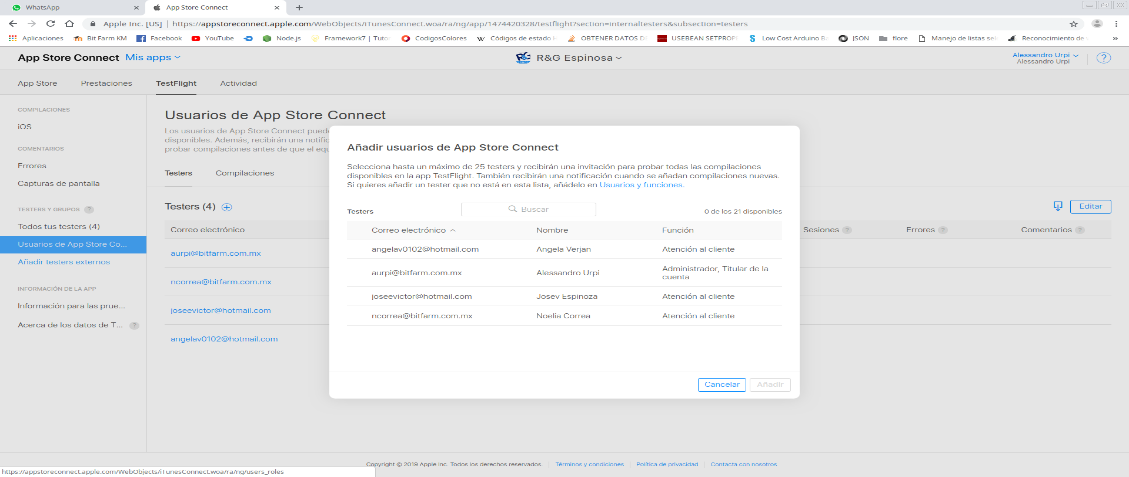




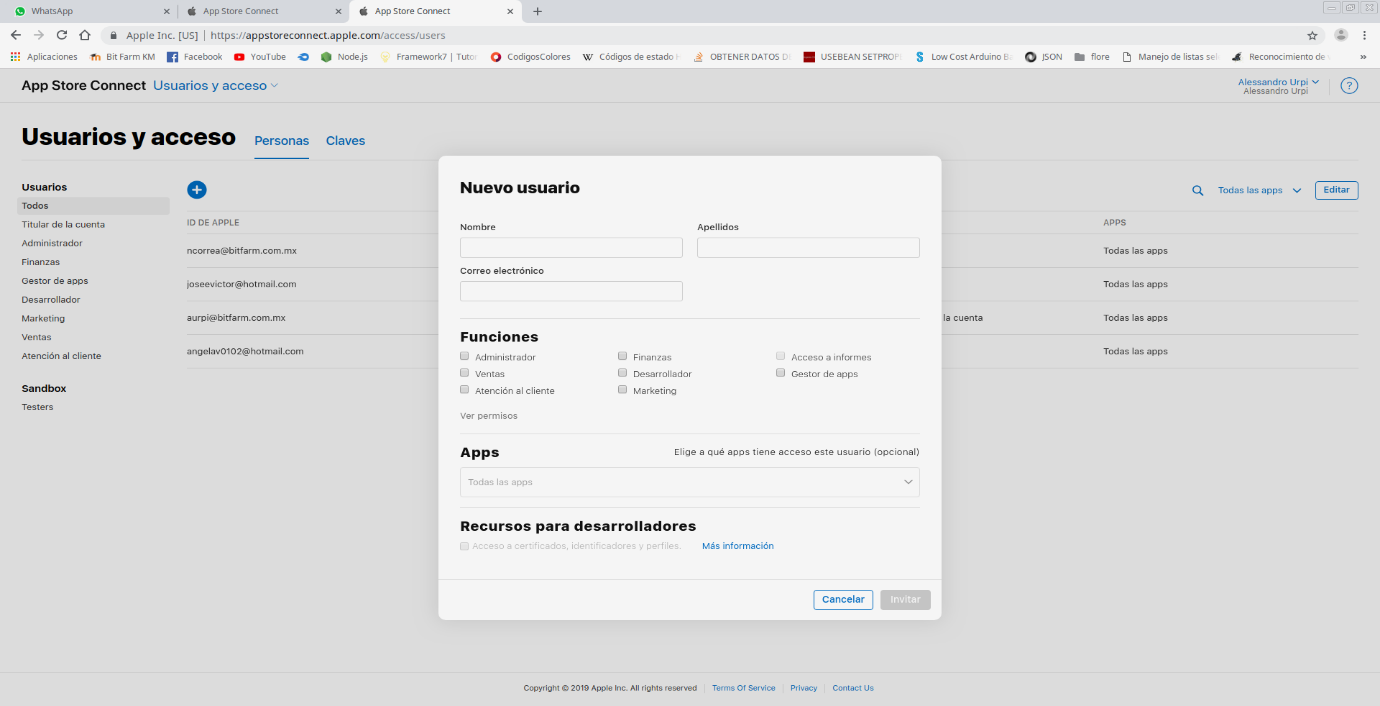
Luego nos vamos a la opción de **TestFlight** para empezar a realizar las pruebas de la aplicación, en la sección de **COMPILACIONES/ IOS** y seleccionamos nuestra aplicación como se muestra en la siguiente imagen.



Nos dirigimos a la opción de **Usuarios de App Store Connect,** a lado del título **Testers (0)** aparecerá un icono de **+** el cual vamos a dar clic y nos abrirá una pantalla en donde podremos seleccionar a las personas que les vamos otorgar permiso e invitación para que puedan probar la aplicación que seleccionamos anteriormente, en caso de que el usuario no aparezca en la lista podremos agregarlo enviándole una invitación, para hacer esto nos vamos a la opción de **Usuarios y funciones**



Cuando entremos a la opción de **Usuarios y acceso** para agregar a personas a pertenecer al grupo de pruebas, nos vamos al icono de **+** y llenamos toda la información que nos solicita y procedemos a invitar



Cuando la invitación ya está enviada solo nos queda esperar a que la acepte y listo, los demás pasos vienen a partir del correo

# Recorrido de la aplicación (flujo de pantallas)

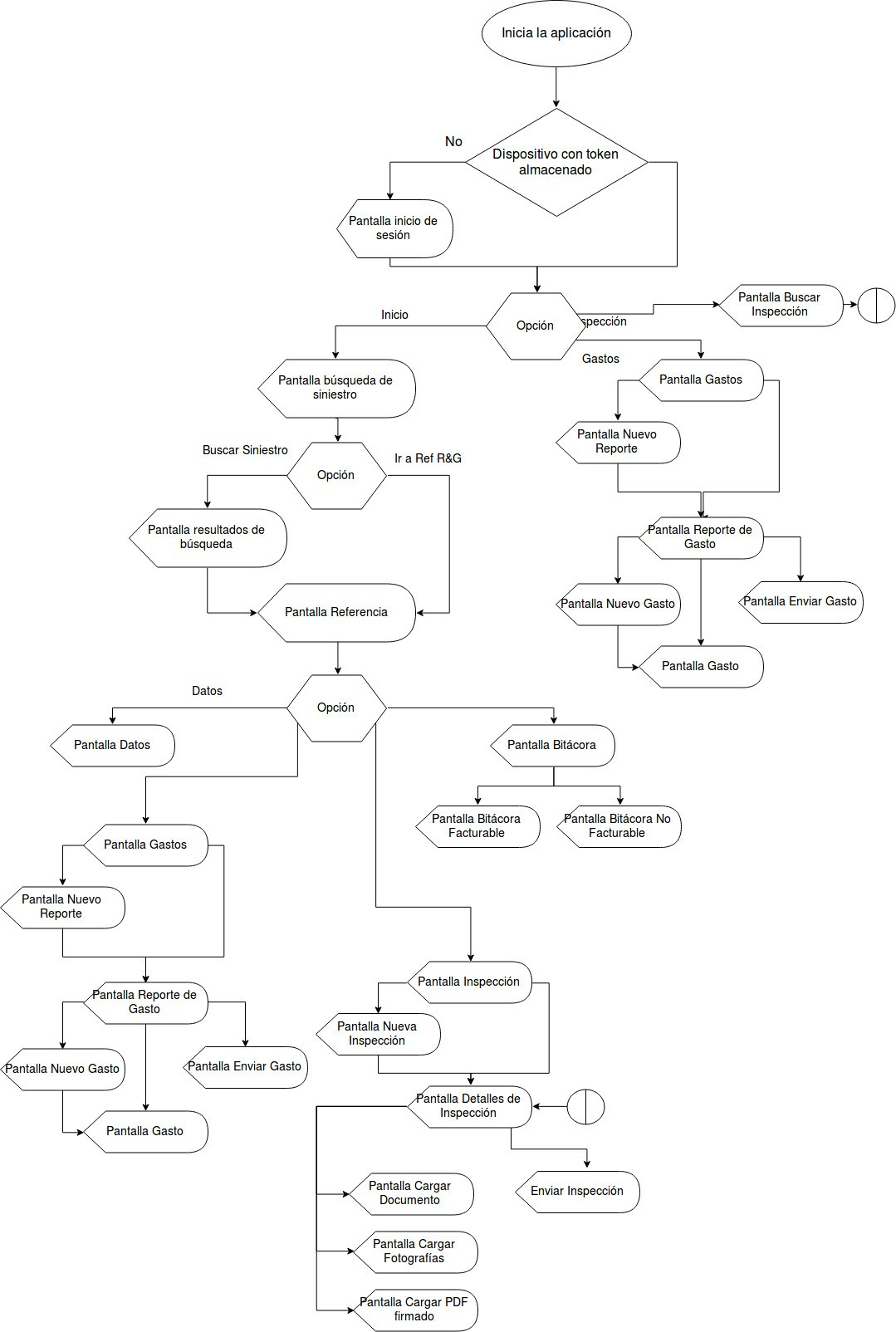


Ilustración 1 Flujo de aplicación

# Pantallas de la aplicación

**Pantalla:** Login / Inicio de sesión

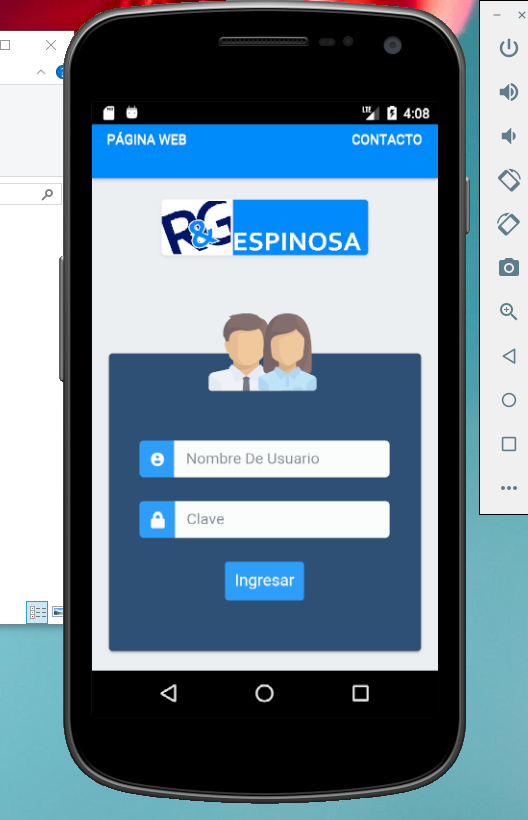


Ilustración 2 Pantalla Inicio

|  |
| --- |
| Html **(index.html)**  **Id=”login.html”** |
| **login.js** |
| **main.js** |
| **utils.js** |
| **baseDatos.js** |

Tabla 4 código pantalla inicio

## Ajustador Sin Siniestro Seleccionado

**Pantalla**: Página Principal



Ilustración 3 Pantalla principal

|  |
| --- |
| **Html (index.html)**  **Id=”buscar.html”** |
| **main.js** |
| **utils.js** |
| **baseDatos.js** |
| **ref.js** |
| **buscar.js** |

Tabla 5 Código página principal

**Pantalla:** Resultados de Búsqueda



Ilustración 4 Pantalla resultados de búsqueda

|  |
| --- |
| **Html (index.html)**  **Id=”resultadosSiniestro.html”** |
| **main.js** |
| **utils.js** |
| **baseDatos.js** |
| **buscar.js** |

Tabla 6 Código página resultados de búsqueda

**Pantalla:** Gastos del ajustador



Ilustración 5 Pantalla Gastos del ajustador

|  |
| --- |
| **Html (index.html)**  **Id=”gastos.html”** |
| **main.js** |
| **utils.js** |
| **baseDatos.js** |
| **gastos.js** |

Tabla 7 Código pantalla gastos del ajustador

**Pantalla:** Buscar Inspección Sin Siniestro



Ilustración 6 Pantalla Buscar inspección sin siniestro

|  |
| --- |
| **Html (index.html)**  **Id=”inspeccionSin.html”** |
| **main.js** |
| **utils.js** |
| **baseDatos.js** |
| **inspeccion.js** |

Tabla 8 Código pantalla buscar inspección

## Ajustador Con Siniestro

**Pantalla:** Barra de progreso

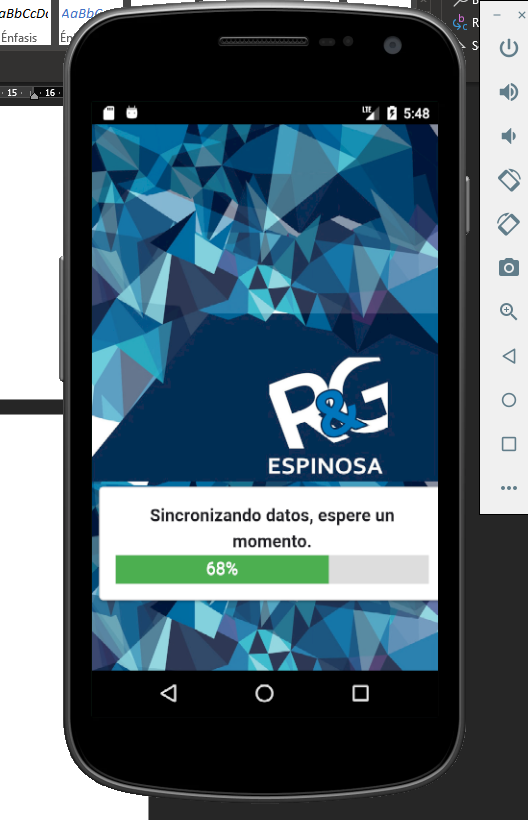


Ilustración 7 Pantalla Barra de progreso

|  |
| --- |
| **Html (index.html)**  **Id=”ref.html”** |
| **utils.js** |
| **baseDatos.js** |
| **inspeccion.js** |
| **datos.js** |
| **gastos.js** |
| **bitacora.js** |

Tabla 9 Código pantalla barra de progreso

**Pantalla:** Página Principal Referencia R&G

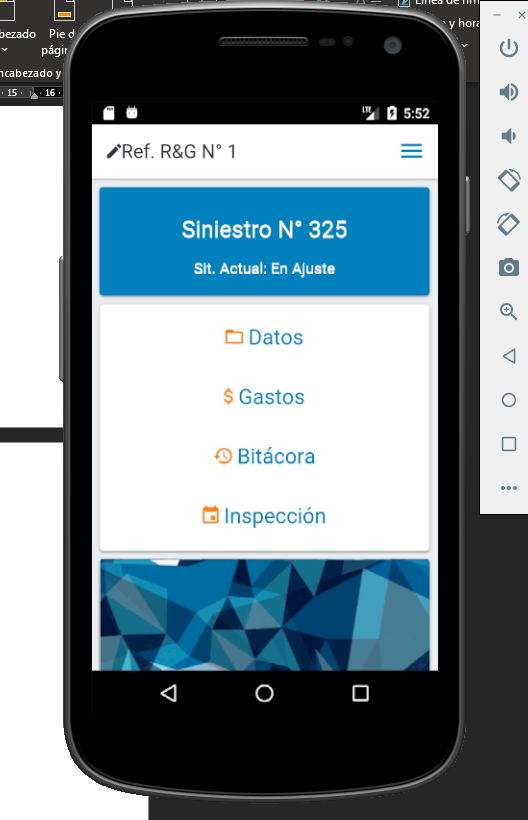


Ilustración 8 Pantalla Ref R&G

|  |
| --- |
| **Html (index.html)**  **Id=”ref.html”** |
| **utils.js** |
| **baseDatos.js** |
| **inspeccion.js** |
| **datos.js** |
| **gastos.js** |
| **bitacora.js** |

Tabla 10 Código pantalla Ref. R&G

**Pantalla:** Datos del siniestro

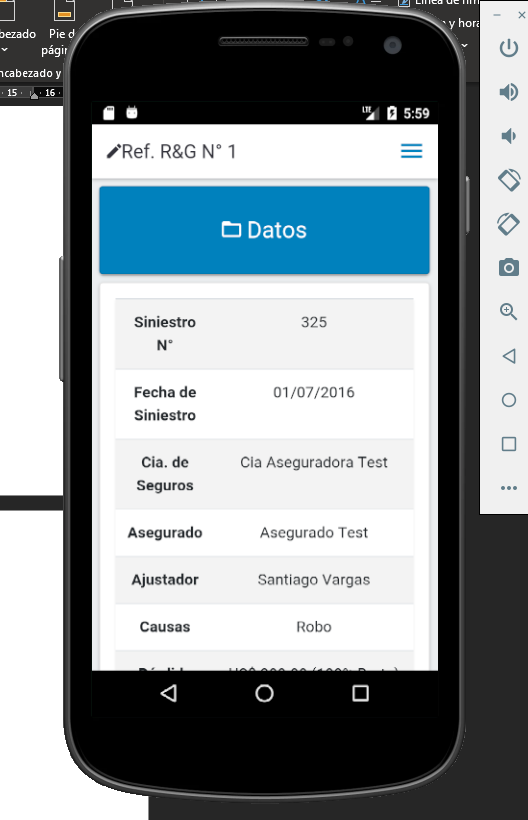


Ilustración 9 Pantalla Datos del siniestro

|  |
| --- |
| **Html (index.html)**  **Id=”datos.html”** |
| **utils.js** |
| **baseDatos.js** |
| **datos.js** |
| **main.js** |

Tabla 11 Código pantalla Datos del siniestro

**Pantalla:** Gastos del siniestro

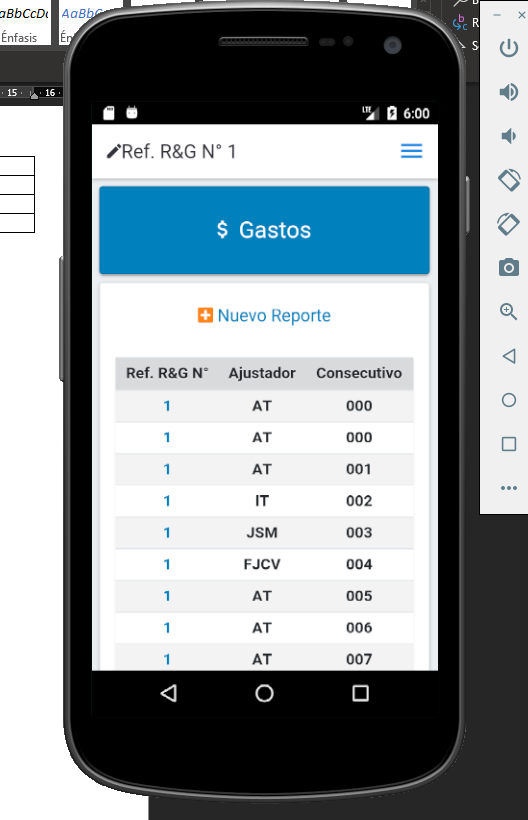


Ilustración 10 Pantalla Gastos del siniestro

|  |
| --- |
| **Html (index.html)**  **Id=”gastos.html”** |
| **utils.js** |
| **baseDatos.js** |
| **gastos.js** |
| **main.js** |

Tabla 12 Código pantalla gastos

**Pantalla:** Nuevo Reporte de Gastos

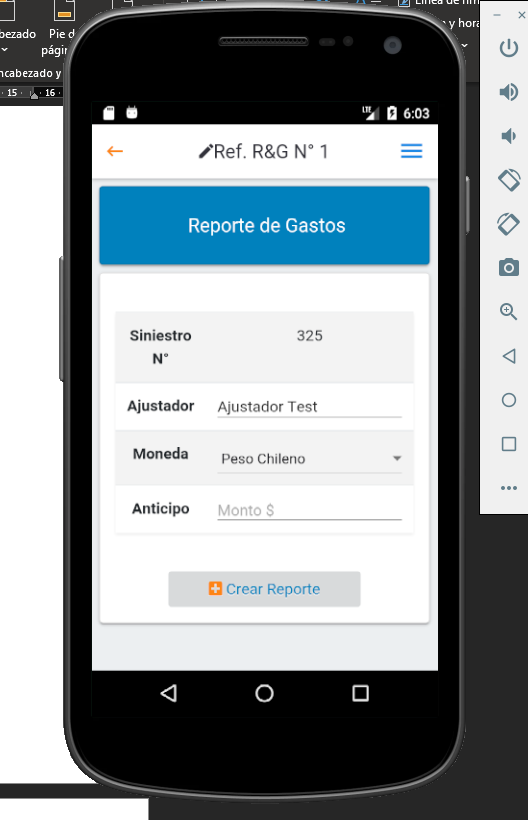


Ilustración 11 Pantalla Nuevo reporte de gastos

|  |
| --- |
| **Html (index.html)**  **Id=”nuevoReporteGasto.html”** |
| **utils.js** |
| **baseDatos.js** |
| **gastos.js** |
| **main.js** |

Tabla 13 Código pantalla nuevo reporte de gastos

**Pantalla:** Reporte de Gasto Seleccionado

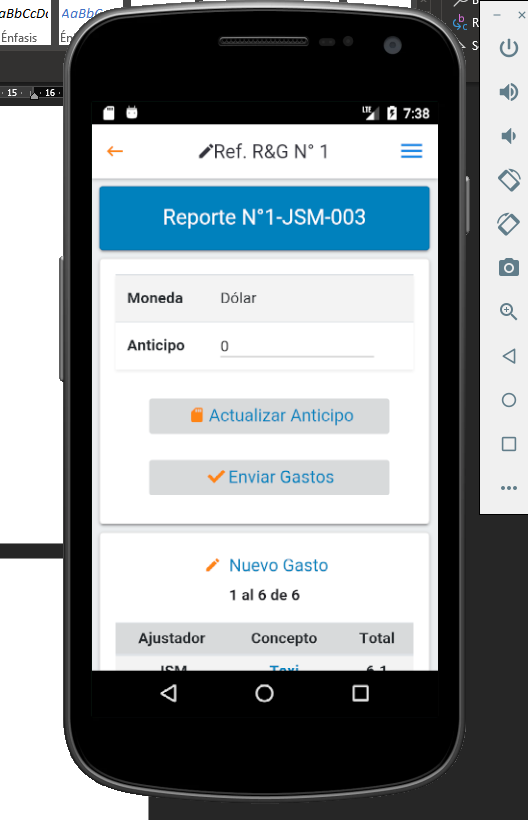


Ilustración 12 Pantalla Reporte de gasto seleccionado

|  |
| --- |
| **Html (index.html)**  **Id=”reporteGasto.html”** |
| **utils.js** |
| **baseDatos.js** |
| **gastos.js** |
| **main.js** |

Tabla 14 Código pantalla de reporte de gasto seleccionado

**Pantalla:** Gasto Seleccionado

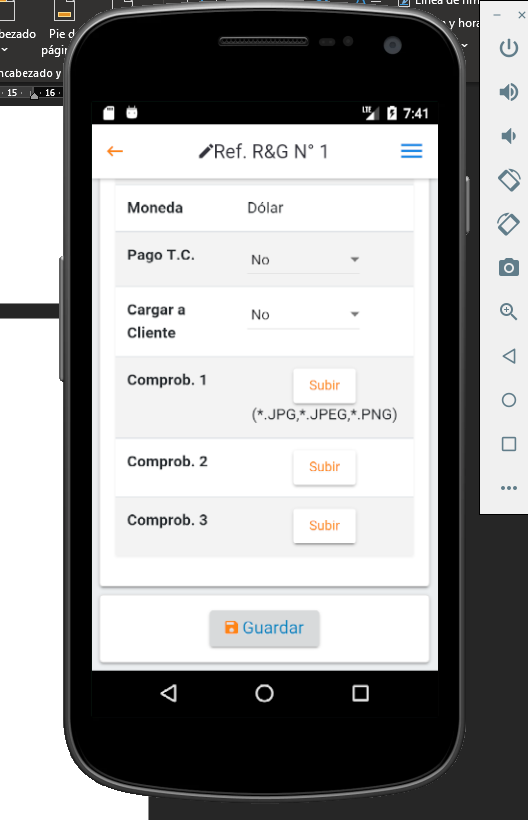
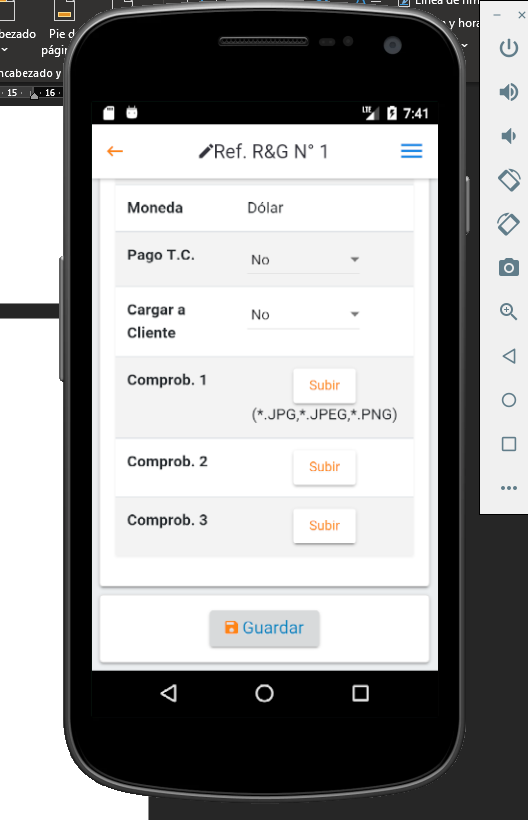
 

Ilustración 13 Pantallas detalle de gasto 1 y 2

|  |
| --- |
| **Html (index.html)**  **Id=”detalleReporteGasto.html”** |
| **utils.js** |
| **baseDatos.js** |
| **gastos.js** |
| **catalago.js** |
| **main.js** |

Tabla 15 Código de pantalla detalle de gasto

**Pantalla:** Enviar Reporte de Gastos



Ilustración 14 Pantalla Enviar reporte de gastos

|  |
| --- |
| **Html (index.html)**  **Id=”sendGastos”** |
| **utils.js** |
| **baseDatos.js** |
| **gastos.js** |
| **main.js** |

Tabla 16 Código de pantalla enviar reporte de gastos

**Pantalla:** Bitácora

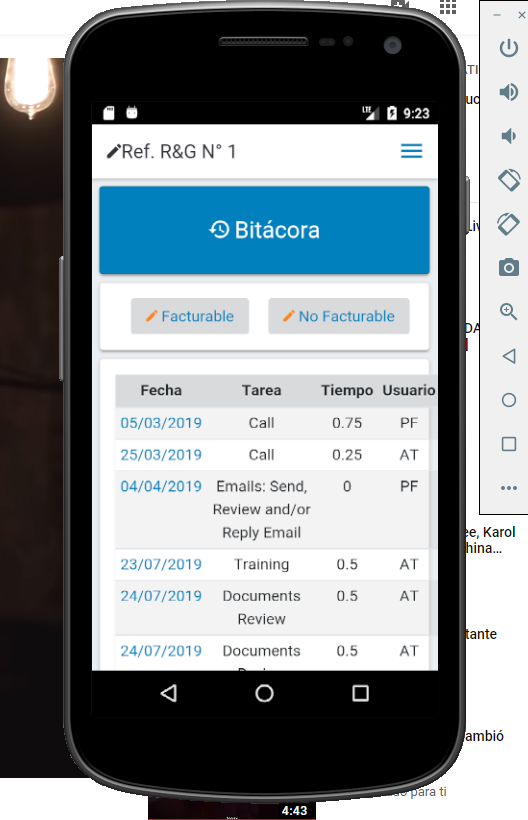


Ilustración 15 Pantalla Bitácora

|  |
| --- |
| **Html (index.html)**  **Id=”bitácora.html”** |
| **utils.js** |
| **baseDatos.js** |
| **bitacora.js** |
| **main.js** |

Tabla 17 Código de pantalla Bitácora

**Pantalla:** Bitácora Facturable

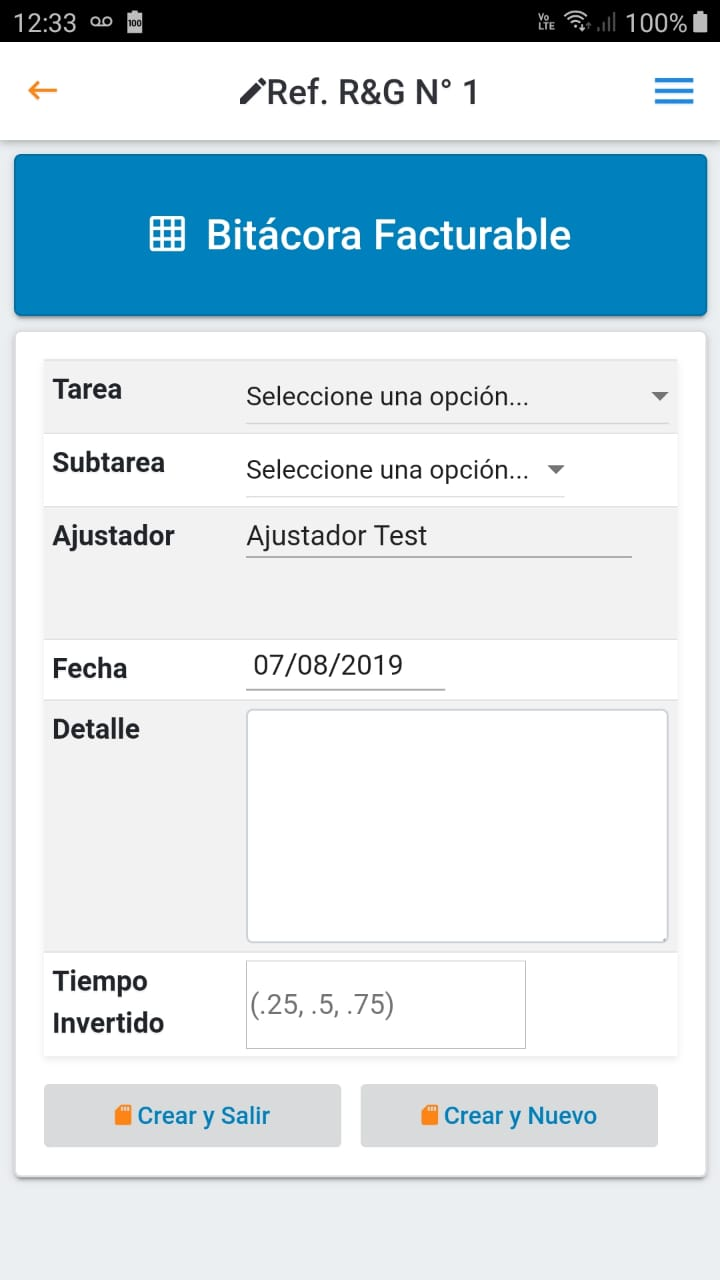


Ilustración 16 Pantalla Bitácora facturable

|  |
| --- |
| **Html (index.html)**  **Id=”bitFacturable.html”** |
| **utils.js** |
| **baseDatos.js** |
| **bitacora.js** |
| **main.js** |

Tabla 18 Código de pantalla Bitácora facturable

**Pantalla**: Bitácora No Facturable



Ilustración 17 Pantalla Bitácora No facturable

|  |
| --- |
| **Html (index.html)**  **Id=”bitNoFacturable.html”** |
| **utils.js** |
| **baseDatos.js** |
| **bitacora.js** |
| **main.js** |

Tabla 19 Código pantalla Bitácora No facturable

**Pantalla**: Inspección



Ilustración 18 Pantalla Inspección

|  |
| --- |
| **Html (index.html)**  **Id=”inspeccion.html”** |
| **utils.js** |
| **baseDatos.js** |
| **inspeccion.js** |
| **main.js** |

Tabla 20 Código de pantalla Inspección

**Pantalla**: Crear nueva inspección

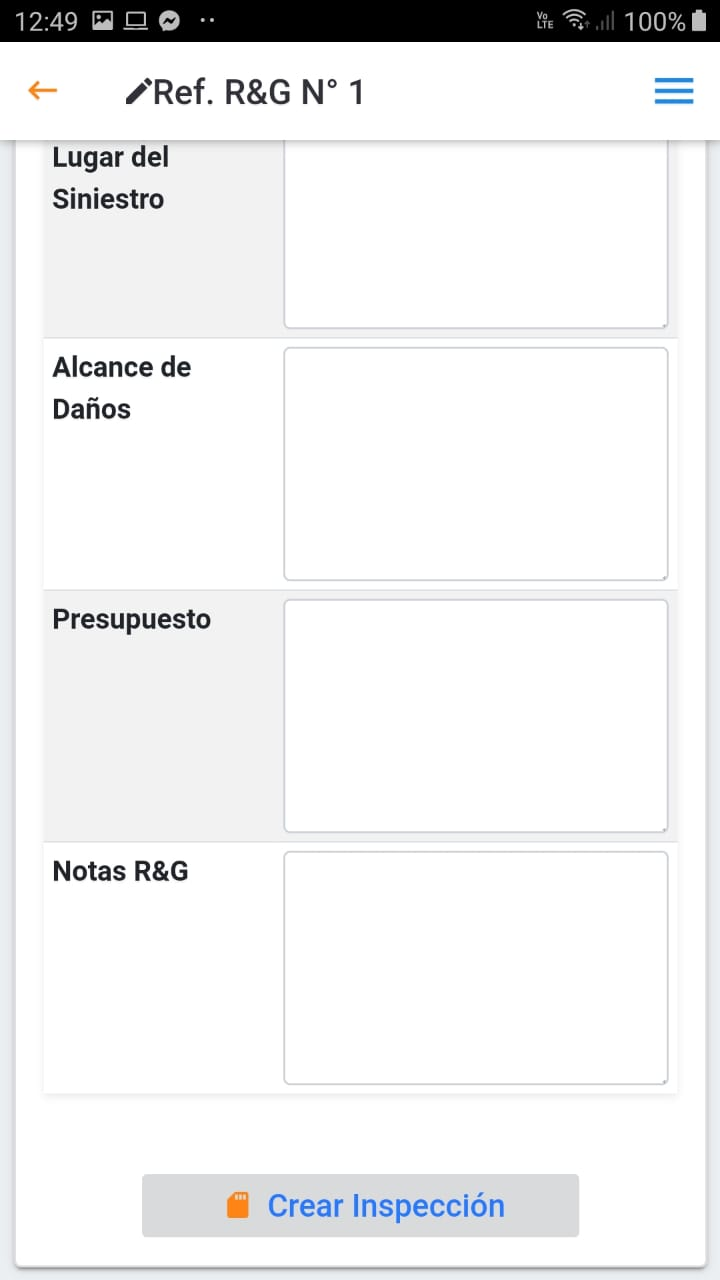


Ilustración 19 Pantalla Crear nueva inspección

|  |
| --- |
| **Html (index.html)**  **Id=”nuevaInspeccion.html”** |
| **utils.js** |
| **baseDatos.js** |
| **inspeccion.js** |
| **main.js** |

Tabla 21 Código pantalla nueva inspección

**Pantalla:** Inspección Seleccionada

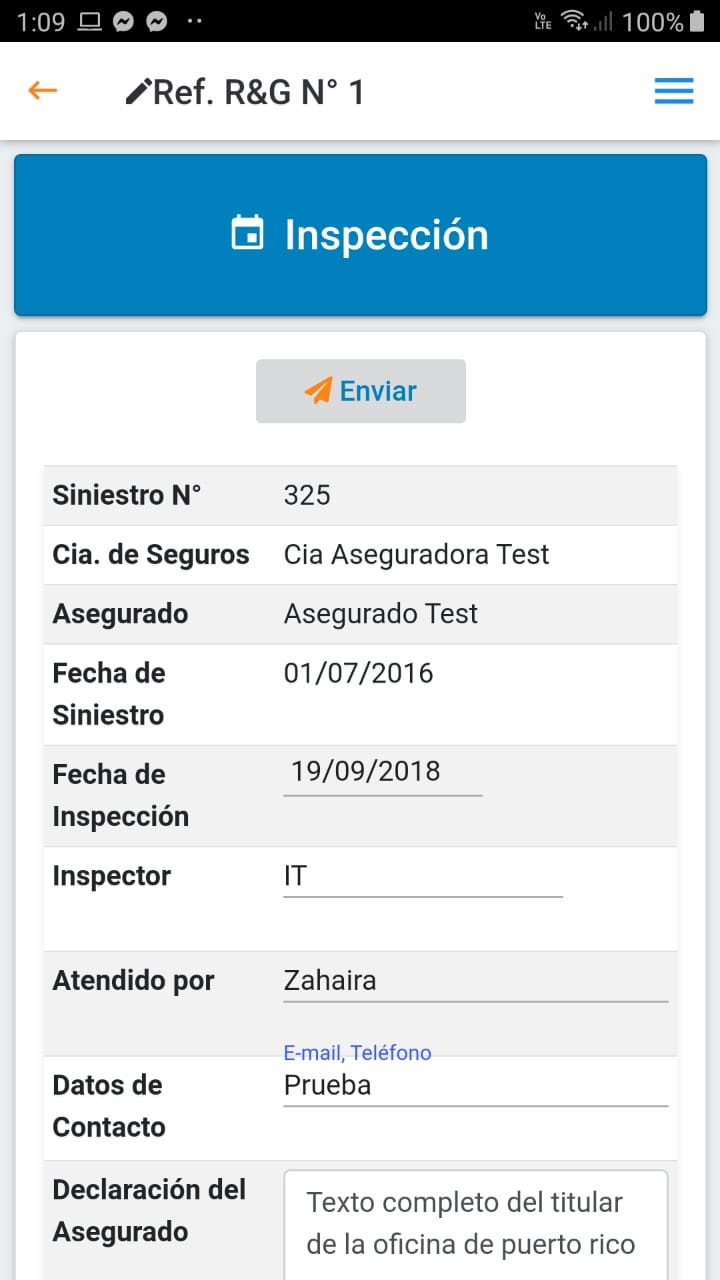


Ilustración 20 Pantalla Inspección seleccionada

|  |
| --- |
| **Html (index.html)**  **Id=”inspeccionSeleccionada.html”** |
| **utils.js** |
| **baseDatos.js** |
| **inspeccion.js** |
| **main.js** |

Tabla 22 Código de pantalla inspección seleccionada

Pantalla: Inspección / Documento



Ilustración 21 Pantalla Documento

|  |
| --- |
| **Html (index.html)**  **Id=”documento.html”** |
| **utils.js** |
| **baseDatos.js** |
| **documento.js** |
| **main.js** |

Tabla 23 Código pantalla Documento

**Pantalla:** Inspección / Listado de fotografías

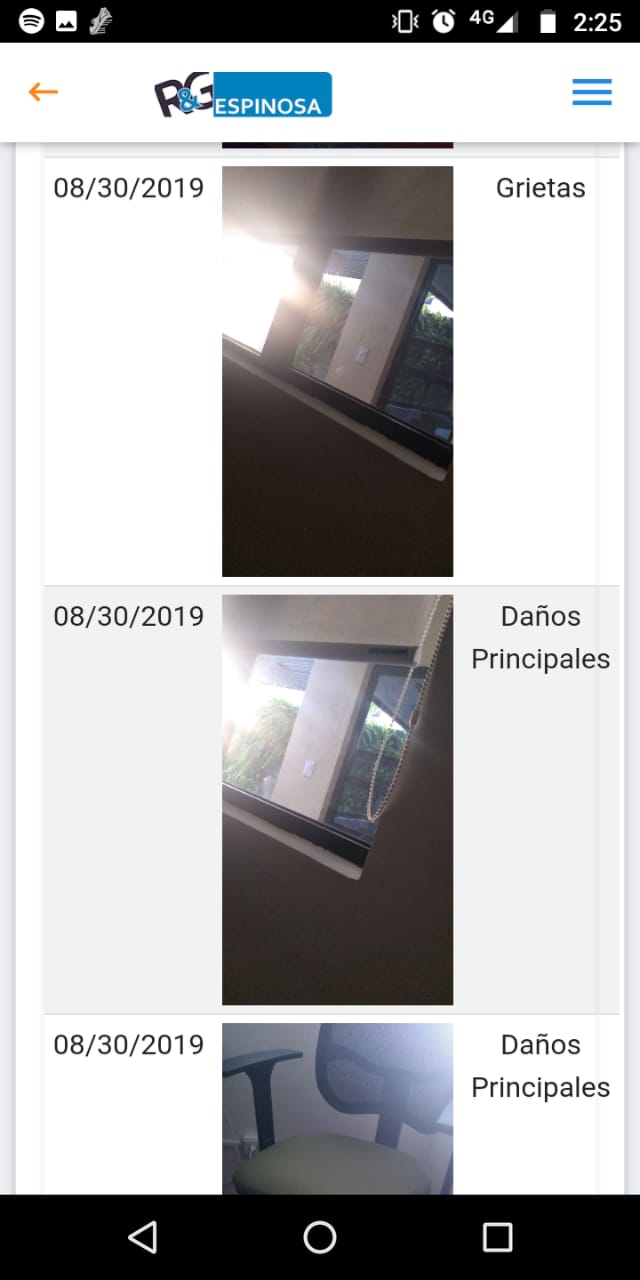
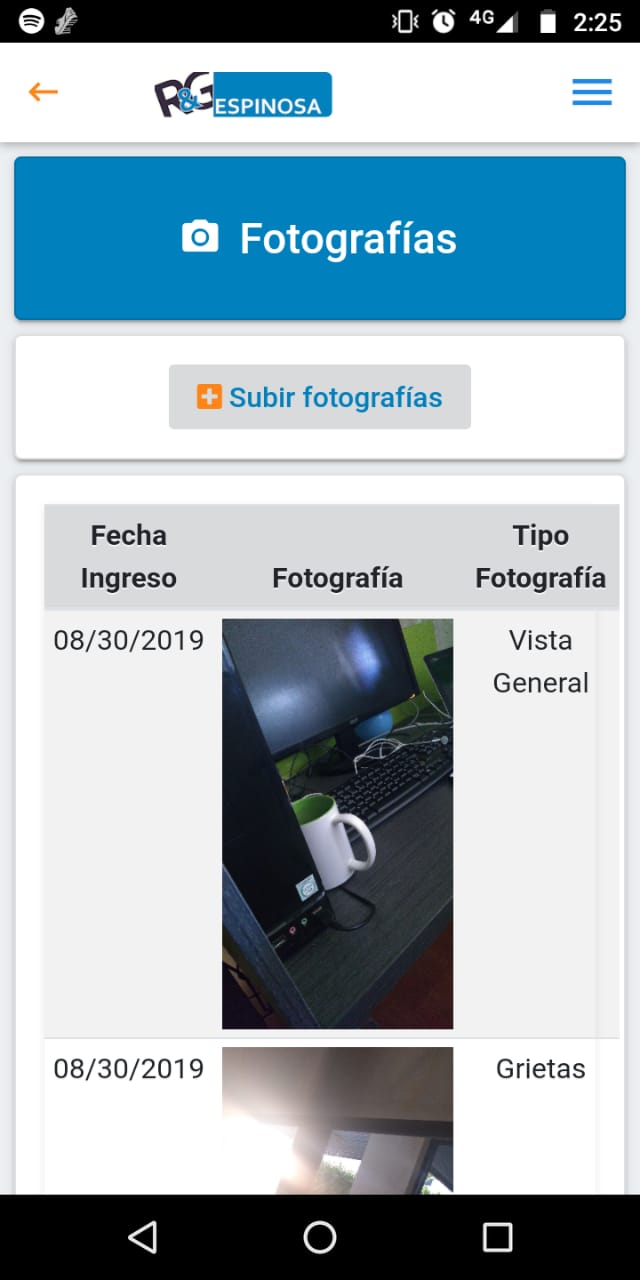


Ilustración 22 Pantalla Fotografías

|  |
| --- |
| **Html (index.html)**  **Id=”listFotografias.html”** |
| **utils.js** |
| **baseDatos.js** |
| **fotografias.js** |
| **main.js** |

Tabla 24 Código de pantalla fotografías

**Pantalla:** Inspección / Cargar fotografías



Ilustración 23 Pantalla Cargar fotografías

|  |
| --- |
| **Html (index.html)**  **Id=”fotografias.html”** |
| **utils.js** |
| **baseDatos.js** |
| **fotografias.js** |
| **uploadFile.js** |
| **main.js** |

Tabla 25 Código pantalla cargar fotografías

**Pantalla:** Inspección / Agregar más fotografías

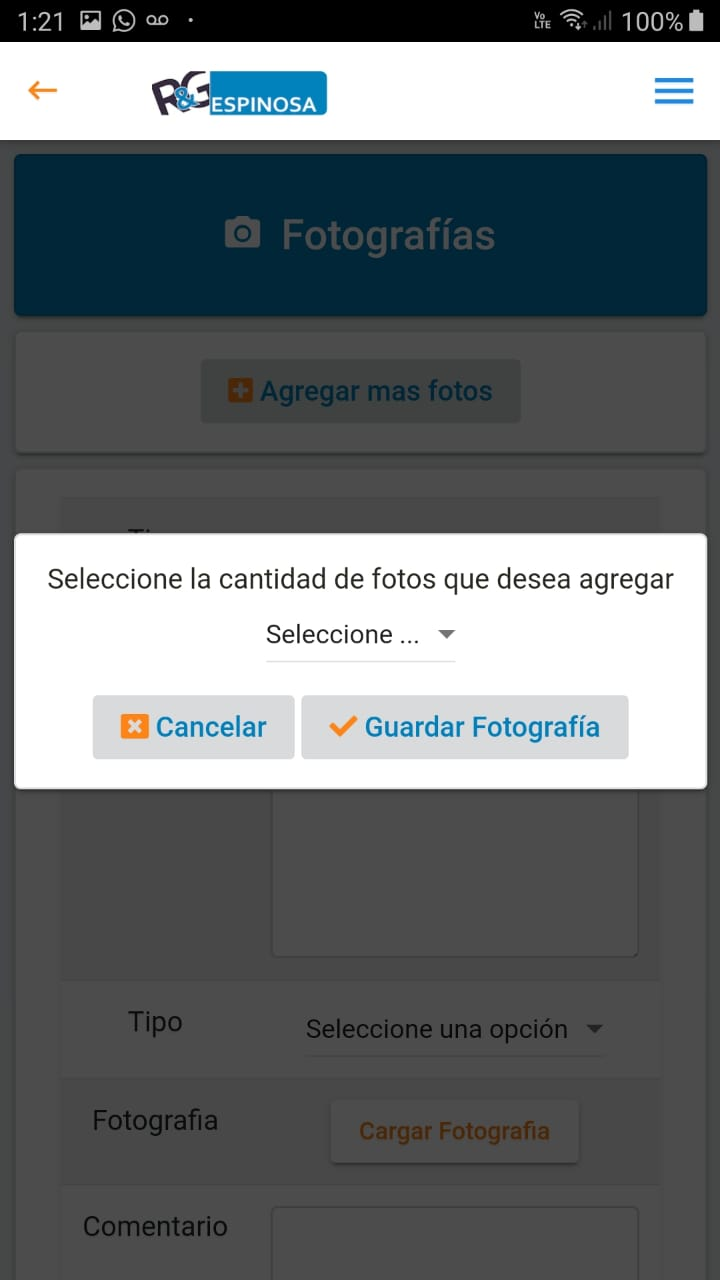


Ilustración 24 Pantalla agregar más fotografías

|  |
| --- |
| **Html (index.html)**  **Id=”modalFotografias”** |
| **utils.js** |
| **baseDatos.js** |
| **fotografias.js** |
| **uploadFile.js** |
| **main.js** |

Tabla 26 Código pantalla agregar más fotografías

**Pantalla:** Subir PDF firmado

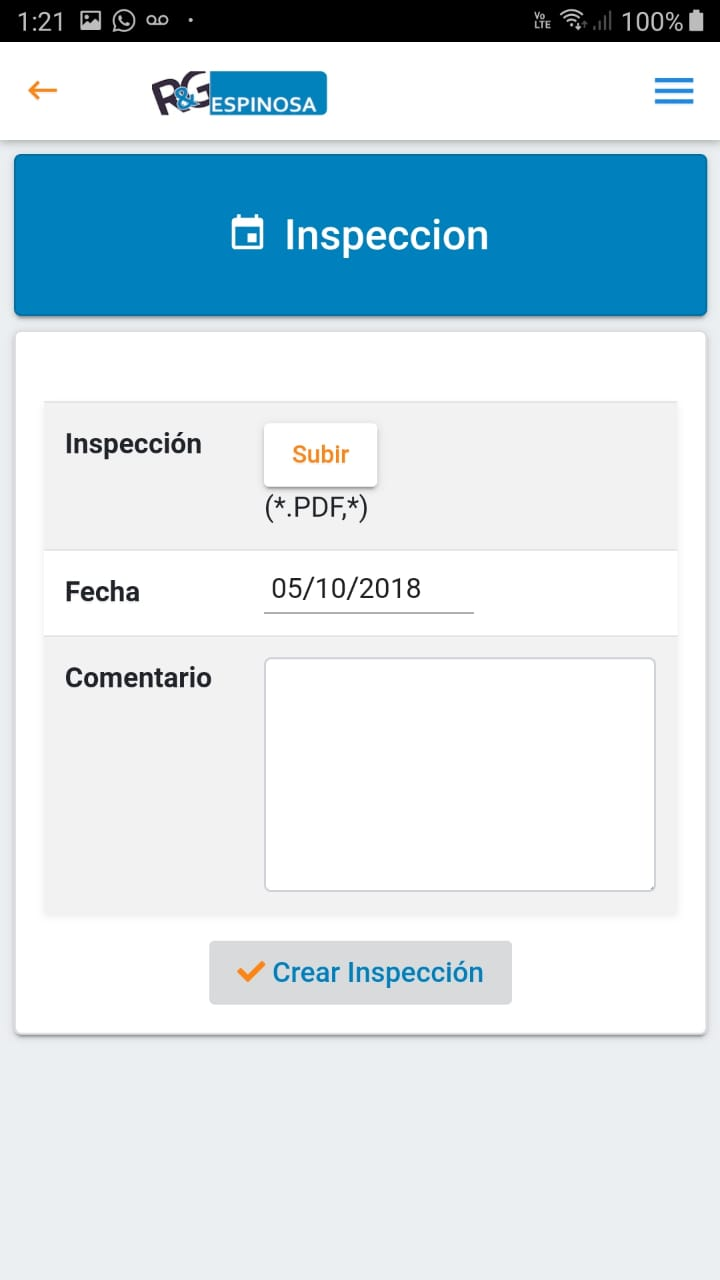


Ilustración 25 Pantalla Subir PDF firmado

|  |
| --- |
| **Html (index.html)**  **Id=”pdf.html”** |
| **utils.js** |
| **baseDatos.js** |
| **pdf.js** |
| **uploadFile.js** |
| **main.js** |

Tabla 27 Código pantalla subir PDF firmado

# Definición de funciones

|  |  |  |  |
| --- | --- | --- | --- |
| **main.js** | | | |
| **Nombre** | **Parámetros** | **Salida** | **Descripción** |
| **onDeviceReady** | N/A | Parámetros  globales del  sistema | Función que se ejecuta al detectar  que se inicia la aplicación |
| **addEventListener()** | “event”,  successFunction | Ejecuta la función  especificada | Función que se ejecuta al arrancar la  aplicación y ejecuta la función  initPage para ejecutar la primera  función operativa de la aplicación |
| **sendGetRequest()** | Url,  successHandler,  errorHandler,  async, dataType | sendRequest() | Envía parámetros a la función  sendReques() |
| **sendPostRequest()** | url, dataParam,  successHandler,  errorHandler | sendRequest() | Envía parámetros a la función sendReques() |
| **sendRequest()** | type, url,  dataParam,  successHandler,  errorHandler,  async, dataType | {idUser: 124, estatus: "true"} | Hace la petición vía AJAX al  servidor según sea GET o POST el  parámetro que se le haya enviado |
| **showLoading()** |  |  | Muestra la animación de carga |
| **hideLoading()** |  |  | Oculta la animación de carga |
| **httpErrorHandler()** | JqXHR,  textStatus,  errorThrown | Alert o imprime  aviso de error con  401, 403 o 0. | Muestra los mensajes de error por  parte de las peticiones HTTP y los  códigos correspondientes |
| **showError()** | message | Muestra un aviso de  error | Imprime un aviso con prioridad de Error |
| **showSuccess()** | message | Muestra un aviso  exitoso | Imprime un aviso con prioridad de exitoso |
| **showNotice()** | message | Muestra un aviso  ocasional | Imprime un aviso con prioridad normal |
| **functionInit()** | event | Funciones diversas | Función que detecta que página es  la que se está iniciando al cargar  una nueva página |
| **backButton()** | N/A |  | Función que establece la función del  back button para una acción  determinada |
| **jQuery.extend(jQuery.validator.messages)** | N/A | N/A | Mensajes de validación para los  formularios |
| **onBackKeyDown** | N/A |  | Establece una función para cuando el back button es presionado |
| **verificarConexion()** |  | True / False | Función que verifica el estatus actual de la conexión a Internet |
| **onDeviceOnline()** | N/A | Ejecuta una función para iniciar el envío de peticiones. | Función que se activa cuando el dispositivo se conecta a internet. |
| **onDeviceOffline()** | N/A | Ejecuta una función para detener el envío de peticiones. | Función que se activa cuando el dispositivo se desconecta de internet. |

|  |  |  |  |
| --- | --- | --- | --- |
| **utils.js** | | | |
| **Nombre** | **Parámetros** | **Salida** | **Descripción** |
| **fn.open()** | N/A | Evento JavaScript | Función que muestra el panel lateral. |
| **fn.load()** | Page: Indica la página que se cargará. | Funciones JavaScript | Función que muestra las páginas de la aplicación, recibe el id de la página y posteriormente la muestra en el contenido. |
| **fn.pagination()** | TotalRegistros: Número total de los registros de la lista.  PageOriginal: Página actual en donde se encuentra el usuario. | Botones de navegación para las listas de la aplicación. | Función que recibe el total de registros y la página actual para determinar la cantidad de páginas totales del listado. |
| **fn.getTotalPages()** | AllData: Cantidad total de registros de una lista.  Limit: Cantidad de registros que se tiene como límite para mostrar en una lista. | Obtiene la cantidad de páginas totales de un listado. | Función que recibe la cantidad total de registros, y la cantidad de registros que se mostrarán por página para obtener el número de páginas requeridas. |
| **fn.formatDate()** | Date: formato de fecha en YYYY-MM-DD HH:mm:ss  Spliter: Separador del formato de fecha. | Regresa el formato de fecha en “DD/MM/YYYY” | Función que recibe un formato de fecha especifico, separa la fecha y la transforma a un formato solicitado. |
| **fn.formatDate2()** | Date: formato de fecha en YYYY-DD-MM Spliter: Separador del formato de fecha | Regresa el formato de fecha en “DD/MM/YYYY” | Función que recibe un formato de fecha especifico, separa la fecha y la transforma a un formato específico. |
| **fn.getDateToString()** | Date: Objeto JavaScript de fecha. | Regresa el formato de fecha en YYYY-DD-MM | Función que recibe un objeto Fecha de JavaScript y lo transforma a una fecha en un String con formato: “YYYY-DD-MM”. |
| **fn.getDateYMD()** | Date: Objeto JavaScript de fecha. | Regresa el formato de fecha en YYYY-MM-DD | Función que recibe un objeto date de JavaScript y lo transforma en un String con formato “YYYY-MM-DD” |
| **fn.dataViewed()** | TotalRegistros: Cantidad de registros de una lista.  PageActual: Número de página actual del usuario.  Limit: Limite de registros por página. | Imprime un mensaje en donde se indica la cantidad de registros que se han visto. | Función que recibe totalRegistros, pageActual y limit para calcular la cantidad de registros que el usuario a visualizado. |
| **fn.dinamicLimit()** | TotalRegistros: Cantidad de registros de una lista. | Regresa un número que indica el límite de registros por página. | Función que recibe el totalRegistros para verificar cual es el límite de registros mostrados en cada página, en caso de ser menos del límite especificado, solo muestra el total de registros existentes. |
| **fn.JsonToArray()** | json\_ada: Objeto JSON | Regresa los valores del JSON en un Array | Función que recibe un objeto JSON, lee sus valores y los coloca en un Array. |
| **fn.closeModal()** | idModal: identificador del modal. | Evento cerrar modal | Función que recibe el id del modal especificado y lo cierra. |
| **fn.openModal()** | IdModal: identificador del modal. | Evento abrir modal | Función que recibe el id del modal especificado y lo muestra en el content. |
| **progressBar()** | N/A | Muestra barra de progreso. | Función que imprime una barra de progreso mientras se cargan los datos. |
| **getFechaActual()** | Date: fecha actual del dispositivo (objeto Date).  Spliter: Separador del formato de fecha. | Obtiene la fecha actual en formato “YYYY-MM-DD hh:mm:ss” | Función que recibe un objeto fecha con la fecha más reciente en el dispositivo y devuelve un formato de fecha específico. |

|  |  |  |  |
| --- | --- | --- | --- |
| **login.js** | | | |
| **Nombre** | **Parámetros** | **Salida** | **Descripción** |
| **revisarLogin()** | N/A | Función revisa sesión | Función que revisa si todos los parámetros están correctos para iniciar sesión |
| **iniciarSesion()** | N/A | Función iniciar sesión | Función que inicia sesión para que el usuario ingrese al sistema |
| **verificarSesion()** | N/A | Función que valida sesión | Función que valida si el usuario existe en el sistema |
| **onInitApp()** | N/A | Función existe siniestro | Función que verifica si existe un siniestro guardado |
| **cerrarSesion()** | N/A | Función que cierra la sesión | Función que cierra la sesión del usuario |
| **lastUserAct()** | LastPageUser: nombre de la última página en la que estuvo el usuario. | Función o pantalla en donde estuvo el usuario la última vez. | Función que recibe la última página en donde estuvo el usuario y determina a que página redirigirlo al expirar el token y volver iniciar sesión. |

|  |  |  |  |
| --- | --- | --- | --- |
| **datosUsuario.js** | | | |
| **Nombre** | **Parámetros** | **Salida** | **Descripción** |
| **saveDataUser()** | dataUser | Función guardar datos del usuario | Almacena los datos de usuario en la base de datos |
| **panelUsuario()** | N/A | Función que imprime el panel con opciones | Función para imprimir el panel de usuario |
| **CambiarSiniestro()** | N/A | Función cambiar siniestro | Función que cambia el siniestro por otro actual |

|  |  |  |  |
| --- | --- | --- | --- |
| **baseDatos.js** | | | |
| **Nombre** | **Parámetros** | **Salida** | **Descripción** |
| **createDatabase()** | N/A | Función que crea la base de datos | Función que inicializa las tablas de la base de datos |
| **dropDatabase()** | N/A | Función que elimina las tablas | Función que elimina las tablas  de la base de datos |
| **deleteDatabase()** | N/A | Función que elimina la información | Función para eliminar toda la información de la base de datos |
| **sqlQuery()** | query, params, callback, insert | Función que realiza las acciones a la base de datos | Función para realizar consultas SQL en la base de datos local |
| **sqlError()** | e: mensaje de error | Función que manda errores | Función que muestra el mensaje de error para los querys |

|  |  |  |  |
| --- | --- | --- | --- |
| **bitacora.js** | | | |
| Nombre | Parámetros | Salida | Descripción |
| **bitacora()** | N/A | Función que regresa datos de la bitácora | Función que muestra los datos de la bitácora |
| **guardarBitacoras()** | LstBitacoras: listado de las bitácoras.  Callback: función de retorno. | Función callback de retorno | Función que recibe el listado de las bitácoras, las guarda en la base de datos y al terminar devuelve a la función callback. |
| **getBitacoraData()** | N/A | Función que consulta la bitácora de acuerdo el id | Función que consulta el listado de bitácoras en base al id del siniestro actual |
| **resultadosBitacora()** | bitácoras | Función que muestra los resultados | Función para procesar los datos del siniestro |
| **nuevaBitacoraFacturable()** | N/A | Función que abre una nueva bitácora facturable | Función que inicializa la pantalla de nueva bitácora facturable |
| **nuevaBitacoraNoFacturable()** | N/A | Función que abre una nueva bitácora no facturable | Función que inicializa la pantalla de bitácora no facturable |
| **initBitacora()** | tipoBitacora | Función que carga los datos a la pantalla | Inicializa los datos generales de la pantalla para capturar una nueva bitácora |
| **verBitacora()** | IdBitacora: identificador de la bitácora a visualizar | Función que carga la información de la bitácora | Función que recibe el identificador de la bitácora y prepara la carga de la información para dicha bitácora. |
| **loadBitacora()** | IdBitacora: identificador de la bitácora. | Muestra los datos de la bitácora | Función que obtiene los datos de la bitácora y los muestra en pantalla. |
| **optionBitaFact()** | Accion: indica la acción de la bitácora (si es nueva o edita) | Botones de la pantalla para agregar una bitácora. | Función que recibe la acción a realizar e imprime los botones correspondientes según el tipo de acción a realizar. |
| **loadSubtareas()** | N/A | Función que tiene los datos del select | Función que cambia el select de subtareas |
| **updateBitacora()** | N/A | Actualiza datos de la bitácora | Función que actualiza los datos de la bitácora que se hayan editado. |
| **sendCambiosBitacora()** | BitacoraParams: objeto con los parámetros respectivos de la bitácora | Envía / actualiza los datos de la bitácora actual en el servidor. | Función que obtiene los parámetros de la bitácora modificada y envía los datos al servidor o guarda la petición en la base de datos. |
| **deleteBitacora()** | N/A | Elimina una bitácora | Función que elimina una bitácora del sistema. |
| **cambiarTareaNoFact()** | N/A | Actualiza la información de la bitácora a no facturable | Función para cambiar la tarea de una bitácora a no facturable. |
| **updateBitaNoFact()** | IdBitacora: identificador de la bitácora. | Muestra la página de bitácoras (*bitácora()*) | Función para actualizar una bitácora a no facturable en la base de datos local. |
| **guardarCambioBitacora()** | IdBitacora: identificador de la bitácora. Params: parametros correspondientes a la bitácora | *updateBitaNoFact()* | Función para guardar los cambios de una bitácora facturable a una no facturable. |
| **guardarBitacoraFacturableYSalir()** | N/A | Función que guarda una bitácora facturable | Guarda una bitácora facturable y regresa a la pantalla de bitácoras |
| **guardarBitacoraFacturableYNueva()** | N/A | Función que guarda una bitácora no facturable | Guarda una bitácora facturable y prepara la pantalla para capturar una nueva bitácora |
| **guardarBitacoraNoFacturableYSalir()** | N/A | Función que guarda una bitácora no facturable y regresa a la pantalla | Guarda una bitácora no facturable y regresa al listado de bitácoras |
| **guardarBitacoraNoFacturableYNueva()** | N/A | Función que guardar una bitácora facturable y abre otra pantalla con la información captura | Guarda una bitácora no facturable y prepara la pantalla para capturar una nueva bitácora |
| **guardarBitacora()** | url/callback | Función que guarda la bitácora | Función que inserta una nueva bitácora en la base de datos |
| **getBitacoraParams()** | Gestión, subtarea, user\_iniciales | Función que obtiene los datos de la bitácora | Función que obtiene los datos para agregar una nueva bitácora del siniestro |
| **limpiarCampos()** | N/A | Función que limpia los campos | Limpia los campos del formulario |
| **validacionNuevaBitacora()** | N/A | Función que valida que este todo correcto | Validad que estén los datos completos en el formulario |

|  |  |  |  |
| --- | --- | --- | --- |
| **Buscar.js** | | | |
| **Nombre** | **Parámetros** | **Salida** | **Descripción** |
| **initResultadosSiniestro()** | page | Función que valida si un numero ha sido ingresado | Función que valida si un número se ingreso |
| **buscarSiniestro()** | page, numSin | Función que busca un siniestro | Función para buscar un siniestro en el servidor |
| **resultadosBusqueda()** | siniestro, data, pageActual | Función que muestra los resultados | Función para procesar el resultado de la lista de siniestros |

|  |  |  |  |
| --- | --- | --- | --- |
| **catalagos.js** | | | |
| **Nombre** | **Parámetros** | **Salida** | **Descripción** |
| **getCatalogosVersion()** | Callback (opcional): función de retorno | Función callback | Función que obtiene los datos de los catálogos en base a la versión actual y las guarda en el dispositivo local. |
| **guardarMonedas()** | CatalogoMoneda: catalogo con la información de las monedas. | N/A | Función que guarda el catálogo de monedas en la base de datos. |
| **guardarConceptos()** | CatalogoConcepto: listado de conceptos del sistema. | N/A | Función que almacena el catálogo de conceptos. |
| **guardarEstadoInspeccion()** | CatalogoEstatus: listado de estatus. | N/A | Función que almacena el catálogo de estatus de inspección en el dispositivo local. |
| **guardarGrupoDocumento()** | CatalogoGrupoDoc: listado de tipos de documentos. | N/A | Función que almacena el catálogo de grupo documento. |
| **guardarTipoAnexo()** | CatalogoAnexo: listado de los tipos de documentos. | N/A | Función que almacena el catálogo de los tipos de anexo |
| **guardarTipoFoto()** | CatalogoFoto: listado de los asuntos y/o titulos de las fotografías. | N/A | Función que almacena el catálogo de los tipos de foto |
| **guardarTarea()** | CatalogoTarea: listado del catálogo de tareas y/o gestión | N/A | Función que guarda el catálogo de tarea |
| **guardarSubtarea()** | CatalogoSubtarea: listado del catálogo de subtarea y/u objetivo. | N/A | Función que guarda el catálogo de subtarea |
| **getConceptoPorId()** | IdConcepto: identificador del concepto a buscar.  Callback: función de retorno. | Objeto concepto seleccionado. | Función que obtiene el concepto por el identificador de dicho concepto. |
| **getMonedaPorId()** | IdMoneda: identificador de la moneda a buscar.  Callback: función de retorno. | Objeto moneda seleccionado en la función de retorno. | Función que obtiene la moneda por el id |

|  |  |  |  |
| --- | --- | --- | --- |
| **datos.js** | | | |
| **Nombre** | **Parámetros** | **Salida** | **Descripción** |
| **datos()** | N/A | Función que guarda todos los datos | Función que muestra los datos del siniestro |
| **resultadosDatos()** | siniestro | Función que muestra los datos | Función para procesar los datos del siniestro |
| **verifySiniestroData()** | IdSiniestro: identificador del siniestro | Redirige a la pantalla de carga, mientras que carga la información del siniestro. | Función que, por medio del identificador, obtiene todos los datos del siniestro. |

|  |  |  |  |
| --- | --- | --- | --- |
| **documento.js** | | | |
| **Nombre** | **Parámetros** | **Salida** | **Descripción** |
| **documento()** | N/A | Función que consulta el documento | Función que consulta, guarda e imprime documento |
| **cargarSelect()** | N/A | Función que carga Select | Función que carga los datos del Select |
| **crearDocumento()** | N/A | Función que crea documento | En esta función se guardarán los datos del documento |
| **docFotoToJson()** | documento | Función que convierte en json los parámetros | Función que recibe y devuelve en json los parámetros |

|  |  |  |  |
| --- | --- | --- | --- |
| **fotografias.js** | | | |
| **Nombre** | **Parámetros** | **Salida** | **Descripción** |
| **openFotografia()** | Catalogo: listado de asuntos de fotografía. | Función que muestra el formulario para subir las fotografías. | Función que despliega el formulario para subir o cargar las fotografías. |
| **openFotoInspeccion()** | Element: elemento del formulario en donde se carga la fotografía. | Función para seleccionar la foto de la inspección. | Función que carga la foto de inspección |
| **onPhotoInsSuccess()** | FotoUri: ubicación de la fotografía. IdInput: id del input en donde se cargó la fotografía. | Muestra que la fotografía se ha cargado correctamente. | Función que muestra cuando una fotografía ya se ha cargado. |
| **onPhotoInsError()** | N/A | Función que marca error de la foto | Función que se encarga de mandar un mensaje de que la foto no se ha podido cargar |
| **showModal()** | N/A | Función que abre un modulo | Función que abre una pequeña pantalla |
| **closeModal()** | N/A | Función que cierra el modulo | Función que cierra la pequeña pantalla |
| **addFotos()** | N/A | Función agrega módulos | Función que agrega la cantidad de módulos que se le especifica |
| **openFotoInspeccionPlus()** | N/A | Función que carga la foto de inspección plus | Función que carga la foto de inspección plus |
| **validateNuevasFotografias()** | N/A | Función que valida las fotografías | Función que inicializa las validaciones para el formulario de agregar nuevas fotografías |
| **submitNuevasFotografias()** | N/A | Función que valida las fotografías | Función para validar el formulario de nuevas fotografías |
| **saveFotografia()** | N/A | Función que guarda las fotografías | En esta función se guardarán las fotografías |
| **fotoInspeccionToJson()** | N/A | Función que devuelve un JSON | Función que devuelve un JSON de foto inspección |
| **openFotografiasCargadas()** | N/A | Muestra la lista de fotografías. | Función que muestra las fotografías cargadas desde el dispositivo local. |
| **showFotosCargadas()** | LstFotos: lista de fotografías. | Imprime la lista de fotografías en pantalla. | Función que recibe la lista de fotografías tomadas desde el dispositivo y las muestra en pantalla. |
| **viewFotoSelected()** | IdFoto: identificador de la fotografía cargada. | Muestra la fotografía seleccionada. | Muestra una vista aumentada de la fotografía cargada. |

|  |  |  |  |
| --- | --- | --- | --- |
| **gastos.js** | | | |
| **Nombre** | **Parámetros** | **Salida** | **Descripción** |
| **consultarGastos()** | page | Función que consulta gastos | Función que consulta la lista de gastos en el servidor |
| **getGastoData()** | idSiniestro, page | Función que obtiene los gastos y los imprime | Función que obtiene los datos de la base de datos y prepara para mostrar en pantalla |
| **guardarGastos()** | listaGastos, callback | Función que guarda gastos | Almacena la lista de gastos en la base de datos |
| **mostrarGastos()** | listaGastos, pageActual | Función que muestra gastos | Función que muestra el listado de gastos |
| **verReporteGasto()** | gastos\_repor\_id, idGasto | Función que busca el reporte | Función que busca en el servidor el reporte general de un gasto |
| **getDataReporteGasto()** | idGasto | Función que obtiene datos de gastos | Función que obtiene los datos del reporte de gastos para imprimirlos en pantalla |
| **guardarDetalleGasto()** | detalleReporte, callback, idGasto | Función que guarda el detalle de gasto | Función que guarda el reporte general de un gasto |
| **mostrarReporteGasto()** | reporteGasto, detalleGasto | Función que muestra el reporte | Función que muestra el reporte del gasto |
| **initNuevoGasto()** | N/A | Función que carga la pantalla nuevo gasto | Función para cargar la pantalla de un nuevo gasto |
| **validateNuevoRepGasto()** | N/A | Función que valida el nuevo reporte | Función que inicializa las validaciones para el formulario de agregar un nuevo reporte gasto |
| **submitRepoGasto()** | N/A | Función para validar el formulario | Función para validar el formulario de nuevo reporte gasto |
| **guardarNuevoGasto()** | N/A | Función que guarda el nuevo gasto | Función para guardar un nuevo gasto |
| **updatePagesGasto()** | Callback: funcion de retorno. | Llama a la función callback. | Función para actualizar las páginas de cada gasto. |
| **getGastoParams()** | consecutivo, moneda | Función que obtiene los datos | Función para obtener los parámetros de un |
| **updateAnticipo()** | N/A | Función que actualiza el anticipo | Función para actualizar el anticipo del gasto |
| **initSendGastos()** | N/A | Función que abre el modal | Función que inicializa el modal para enviar los gastos |
| **validateGastosReport()** | N/A | Función sendMail() | Función que válida el formulario para enviar un correo sobre el reporte de gasto. |
| **submitValidateGastosReport()** | N/A | Submit al formulario | Función que realiza submit al formulario para validar los campos. |
| **sendMail()** | N/A | Función que envía correos | Función para enviar mails con los gastos correspondientes |
| **verDetalleGasto()** | idDetalleGasto, idGasto | Función que consulta el detalle | Función que consulta con el servidor el detalle de un gasto |
| **mostrarDetalleGasto()** | detalleGasto | Función que muestra el detalle | Función que permite mostrar la información detallada del reporte de un gasto |
| **backReporteGasto()** | N/A | Función obtiene el id | Función que obtiene el id del reporte |
| **initNuevoDetalle()** | N/A | Función que inicializa los datos del formulario para agregar un nuevo detalle gastos | Función que inicia los datos |
| **optionsDetalleGasto()** | tipo | Función que muestra las opciones del detalle gasto | Función que muestra las opciones de la pantalla detalle gastos |
| **validateNuevoDetalleGasto()** | N/A | Función que valida el nuevo detalle gasto | Función que valida el formulario para agregar un nuevo detalle gasto |
| **submitNuevoDetalleSalir()** | N/A | Función que valida el nuevo detalle y sale | Función que valida el nuevo detalle y sale |
| **submitNuevoDetalleNuevo()** | N/A | Función que valida el nuevo detalle y sale | Función que valida el nuevo detalle y sale |
| **guardarNuevoDetalle()** | accion | Función que guarda un nuevo detalle gasto | Función para guardar un nuevo detalle gasto |
| **getDetalleGastoParams()** | concepto | Función que obtiene el detalle gasto | Función para obtener los parámetros de detalle gasto |
| **updateDetalleGasto()** | N/A | Función que actualiza el detalle | Función para actualizar el detalle de los gastos |
| **deleteDetalleGasto()** | N/A | Función que muestra el modal | Función para mostrar el modal para confirmar la acción |
| **confirmDeleteGasto()** | N/A | Función que elimina el detalle gasto | Función para eliminar el detalle del gasto |
| **detalleGastoToJson()** | detalleGasto | Función que devuelve JSON | Función que obtiene los parámetros de detalles y los devuelve en forma de JSON |
| **documentoGastoToJson()** | documentoGasto | Función que devuelve JSON | Función que obtiene los parámetros de documentos y los devuelve en forma de JSON |

|  |  |  |  |
| --- | --- | --- | --- |
| **inspeccion.js** | | | |
| **Nombre** | **Parámetros** | **Salida** | **Descripción** |
| **Inspeccion()** | Page: indica la página actual a consultar. | Función resultadosInspeccion() | Función que obtiene la lista de inspecciones del siniestro desde el servidor, y las almacena en el dispositivo local. |
| **updatePagesInspeccion()** | Callback: función de retorno. | Función callback de retorno. | Función que reorganiza las páginas del módulo de inspección. |
| **resultadosInspeccion()** | Siniestros: lista de siniestros, pageActual: pagina actual que se esta consultando, numSin: identificador del siniestro actual. | Listado de inspecciones en pantalla. | Función que recibe la lista de inspecciones, la página actual y el número de siniestro para imprimir en pantalla el listado de inspecciones |
| **verInspeccion()** | inspeccion\_id: identificador de la inspeccion desde el servidor. idInspeccion: identificador de la inspección local.  numSin: identificador del siniestro actual. | Función para ver una inspección en especifico | Función para consultar los datos específicos de una inspección |
| **getInspeccionData()** | IdInspeccion: identificador local de inspección. inspeccion\_id: identificador del servidor de la inspección. IdSiniestro: identificador del siniestro. DetalleInspeccionLINE: determina si el detalle de la inspección se obtuvo directamente del servidor. | Función que obtiene datos | Función para obtener los de una inspección |
| **mostrarInspeccion()** | results, siniestro, detalles | Función que imprime los datos en la pantalla | Función que imprime los datos en la pantalla |
| **updateEstado()** | N/A | Función que actualiza | Función que actualiza el estado de la inspección |
| **openEnviarIns()** | N/A | Función que abre pantalla para enviar correo | Abre la pantalla para enviar la inspección por correo |
| **abrirNuevaInspeccion()** | N/A | Función que abre una nueva pantalla | Función que abre una nueva inspección |
| **mostrarNuevaInspeccion()** | siniestro | Función que abre una nueva pantalla | Función que muestra una nueva inspección |
| **crearInspeccion()** | N/A | Función para crear una nueva inspección | Función para crear una nueva inspección junto con su detalle inspección |
| **guardarCambios()** | N/A | Función que guarda cambios | Función que guarda los cambios y modificaciones del detalle de la inspección |
| **guardarDetalleInspección()** | detalleInspeccion, inspeccion\_id, idInspeccion, callback | Función que guarda detalle inspección | Función que guarda detalle inspección |
| **validateFormNuevaInspeccion()** | N/A | Función que valida el formulario | Función que valida el formulario de nueva inspección |
| **submitNuevaInspeccion()** | N/A | Función que valida nueva inspección | Función que valida nueva inspección |
| **inspeccionBuscarSin()** | N/A | Función que carga la pantalla | Función que cambia la pantalla |
| **validateFormInspeccionSeleccionada()** | N/A | Función que valida el formulario | Función que valida el formulario de una inspección seleccionada |
| **submitInspeccionSeleccionada()** | N/A | Función que valida inspección seleccionada | Función que valida inspección seleccionada |
| **validarInspeccionSinSiniestro()** | N/A | Función que valida inspección sin siniestro | Función que valida inspección sin siniestro |
| **buscarInspeccionSinSiniestro()** | numSin | Función que busca una inspección sin siniestro | Función para buscar inspecciones (sin siniestro seleccionado) en el servidor |
| **resultadosInspeccionSin()** | siniestros, pageActual, numSin | Función que muestra resultados. | Función que muestra resultados de inspección sin siniestro. |
| **verInspeccionSin()** | idInspeccion, numSin | Función para ver una inspección. | Función para ver una inspección sin siniestro. |
| **openEnviarIns()** | N/A | Carga la pantalla para enviar una inspección. | Función para cargar la pantalla para enviar una inspección por email. |
| **validateEnviarIns()** | N/A | Valida el formulario para enviar inspección | Función para validar y marcar campos con error o succes del formulario para enviar una inspección. |
| **submitEnviarIns()** | N/A | Submit al formulario enviar inspección. | Función para cargar la validación del formulario para enviar una inspección. |
| **saveEmail()** | N/A | Función para guardar un correo. | Función para guarda un correo. |
| **backVerInspeccion()** | N/A | Función para obtener valores. | Función para obtener valores. |

|  |  |  |  |
| --- | --- | --- | --- |
| **pdf.js** | | | |
| **Nombre** | **Parámetros** | **Salida** | **Descripción** |
| **openPDF()** | N/A | Función que consulta información | Función que consulta la información |
| **mostrarPDF()** | data | Función que imprime información | Función que imprime datos en la pantalla |
| **subirPDF()** | N/A | Función para subir PDF | Subir para subir un PDF |
| **pdfToJson()** | Pdf | Retorna JSON | Función que devuelve los parámetros en JSON |
| **createDirectory()** | N/A | Crea un directorio | Función que crea un directorio |
| **getFsDirectory()** | Fs | Dirección | Función que se dirige al directorio |
| **downloadPDF()** | N/A | PDF | Función que descarga el PDF |
| **createDocument()** | N/A | documento | Función que crea el documento |
| **gotFs()** | Fs | Obtiene directorio | Función que obtiene el directorio |
| **gotDir()** | dirEntry | Obtiene directorio | Función que obtiene directorio |
| **Fail()** | Error | Mensaje | Función que indica un error en caso de que algo falle |
| **Download()** | fileEntry, uri | Descarga | Función que descargue |
| **readBinaryFile()** | FileEntry | Formato PDF | Función convierte al formato PDF |
| **writeFile()** | fileEntry, dataObj | PDF escrito | Función que escribe el PDF |

|  |  |  |  |
| --- | --- | --- | --- |
| **peticiones.js** | | | |
| **Nombre** | **Parámetros** | **Salida** | **Descripción** |
| **guardarPeticion()** | url, metodo, nombreTabla, nombreIdentificador, idEntidad, callback | Regresa una petición insertada | Función para guardar las peticiones pendientes por sincronizar |
| **deletePeticion()** | NombreTabla, nombreIdentificador, idEntidad, estatus, callback | Función callback de retorno. | Función que elimina una petición de la base de datos. |
| **getPeticionesPorEstatus()** | estatus | Respuesta de estatus | Obtiene las peticiones de acuerdo con su estatus |
| **sendPeticiones()** | petición | peticiones | Se encarga de obtener los parámetros y enviar las peticiones al servidor |
| **actualizarEstatusPeticion()** | idPeticion, estatus, callback | estatus | Función que actualiza la petición |
| **updateLocalData()** | petición, entidad, response | Datos guardados | Función que actualiza los datos dependiendo de las tablas |

|  |  |  |  |
| --- | --- | --- | --- |
| **ref.js** | | | |
| **Nombre** | **Parámetros** | **Salida** | **Descripción** |
| **initRefSin()** | N/A | Datos guardados | Función que guarda todos los datos del siniestro |
| **buscarReferencia()** | numSin | Datos | Función que busca el siniestro por medio de su referencia |
| **setDataSiniestro()** | caso\_id, n\_siniestro, estadoSin | Imprimir | Función para setear los títulos e información del siniestro actual |
| **guardarSiniestro()** | data | Insert en la base de datos | Función que guarda toda la información de un siniestro |

|  |  |  |  |
| --- | --- | --- | --- |
| **uploadFile.js** | | | |
| **Nombre** | **Parámetros** | **Salida** | **Descripción** |
| **openPhoto()** | N/A | Archivo cargado | Función que abre el módulo para seleccionar una foto |
| **setOptions()** | srcType | Configuraciones | Función que declara las configuraciones de la foto |
| **onGetPhotoSuccess()** | imageUri | Archivo cargado | Función que obtiene los datos de la foto |
| **onPhotoError()** | N/A | Mensaje | Función que imprime un error en caso de fallar |
| **getFileEntry()** | imgUri | Archivo cargado | Función que carga el archivo |
| **openFileXml()** | N/A | Datos actualizados | Función que actualiza e inserta en la base de datos |
| **openFile()** | N/A | Función que captura la foto | Función que abre la foto y la guarda |
| **openFileDocumentos()** | N/A | Función que guarda el archivo | Función que abre la foto y la guarda |
| **uploadFile()** | urlController, params, fileKey, metodoHttp, mimeType, succesFileTransfer, errorFileTransfer | Función que guarda en la base de datos y al servidor | Función que guarda en la base de datos y se manda la foto al servidor |

# Servicios

|  |  |  |  |
| --- | --- | --- | --- |
| **Servicios buscar.js** | | | |
| Url | Tipo | Parámetros | Respuesta |
| Busqueda.php?caso= | GET | Caso: indica el número de siniestro que se está buscando. | Listado de siniestros. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Servicios catalogos.js** | | | |
| **Url** | **Tipo** | **Parámetros** | **Respuesta** |
| **Catalogos.php** | GET | V: Indica la versión actual que se encuentra en el dispositivo móvil. | Catálogos del sistema |

|  |  |  |  |
| --- | --- | --- | --- |
| **Servicios datos.js** | | | |
| **Url** | **Tipo** | **Parámetros** | **Respuesta** |
| **Siniestro.php?caso=** | GET | Caso: Indica el número de siniestro a consultar | Datos del siniestro actual |

|  |  |  |  |
| --- | --- | --- | --- |
| **Servicios documento.js** | | | |
| **Url** | **Tipo** | **Parámetros** | **Respuesta** |
| **anexos.php** | POST | {edita\_dato, grupo, fecha, anio, mes, dia, selecsecundario, up\_documento, anexos\_observaciones} | ok |

|  |  |  |  |
| --- | --- | --- | --- |
| **Servicios fotografias.js** | | | |
| **Url** | **Tipo** | **Parámetros** | **Respuesta** |
| **fotos.php** | POST | {fotos\_nombre, fotos\_observaciones,  up\_documento, edita\_dato} | ok |

|  |  |  |  |
| --- | --- | --- | --- |
| **Servicios gastos.js** | | | |
| **Url** | **Tipo** | **Parámetros** | **Respuesta** |
| **siniestro\_gastos.php?caso=** | GET | Caso: indica el número de siniestro para consultar los gastos. | Lista de gastos |
| **ajustados\_gastos.php** | GET |  | Lista de gastos |
| **gastos\_reporte.php?id=** | GET | Id: Indica el id del reporte que se esta consultando | Se obtiene una lista de gastos relacionados con el reporte. |
| **ajustador\_gastos.php** | POST | {correspon\_caso,  gastos\_repor\_consecu,  gastos\_repor\_moneda,  gastos\_repor\_anticipo} | Id del reporte de gasto |
| **ajustador\_gastos.php** | PUT | {correspon\_caso,  gastos\_repor\_consecu,  gastos\_repor\_moneda,  gastos\_repor\_anticipo} | Regresa un estatus Ok |
| **envio\_reporte.php** | POST | {anio,  mes,  dia,  anio\_2,  mes\_2,  dia\_2,  gastos\_repor\_nombre,  gastos\_repor\_mail,  gastos\_repor\_coment,  gastos\_repor\_id\_anti} | Estatus Ok |
| **gasto\_detalle.php** | POST | {gastos\_movil\_caso,  anio,  mes,  dia,  gastos\_repor\_id,  gastos\_valor,  gastos\_cargar,  gastos\_moneda,  gastos\_emisor,  gastos\_descripcion,  gastos\_concepto,  gastos\_razon\_social,  gastos\_amex,  up\_documento\_2} | {gastos\_id,  gastos\_caso,  gastos\_user,  gastos\_ajustador,  gastos\_visada,  gastos\_id\_reporte,  gastos\_valor,  gastos\_cargar,  gastos\_moneda,  gastos\_fecha\_real,  gastos\_emisor,  gastos\_descripcion,  gastos\_concepto,  gastos\_razon\_social,  gastos\_viaje\_ini,  gastos\_viaje\_fin,  gastos\_amex,  gastos\_nulo,  gastos\_archivo,  gastos\_archivo\_xml,  gastos\_archivo\_3,  gastos\_fecha,  user\_nombre} |

|  |  |  |  |
| --- | --- | --- | --- |
| **Servicios inspeccion.js** | | | |
| **Url** | **Tipo** | **Parámetros** | **Respuesta** |
| **siniestro\_inspecciones?caso=** | GET | Caso: indica el número de siniestro para consultar las inspecciones. | Lista de inspecciones |
| **inspeccion\_detalle.php?id=** | GET | Id: indica el id de la inspeccion que se esta consultando | Información detallada sobre la inspección actual. |
| **inspeccion\_detalle.php** | POST | {  correspon\_caso,  inspeccion\_tipo,  inspeccion\_contacto,  inspeccion\_mail,  inspeccion\_daño\_estr,  inspeccion\_declaracion,  inspeccion\_descrip,  inspeccion\_comentario,  inspeccion\_almacen  } | {  inspeccion\_id,  inspeccion\_caso,  inspeccion\_estado,  correspon\_caso,  inspeccion\_tipo,  inspeccion\_contacto,  inspeccion\_mail,  inspeccion\_daño\_estr,  inspeccion\_declaracion,  inspeccion\_descrip,  inspeccion\_comentario,  inspeccion\_almacen  } |
| **inspeccion\_detalle.php** | PUT | {  correspon\_caso,  inspeccion\_tipo,  inspeccion\_contacto,  inspeccion\_mail,  inspeccion\_daño\_estr,  inspeccion\_declaracion,  inspeccion\_descrip,  inspeccion\_comentario,  inspeccion\_almacen  } | {  inspeccion\_id,  inspeccion\_caso,  inspeccion\_estado,  correspon\_caso,  inspeccion\_tipo,  inspeccion\_contacto,  inspeccion\_mail,  inspeccion\_daño\_estr,  inspeccion\_declaracion,  inspeccion\_descrip,  inspeccion\_comentario,  inspeccion\_almacen  } |
| **envio\_inspeccion.php** | POST | {  casos,  contacto\_mail  } | Regresa un estatus Ok |

|  |  |  |  |
| --- | --- | --- | --- |
| **Servicios login.js** | | | |
| **Url** | **Tipo** | **Parámetros** | **Respuesta** |
| **Login.php** | POST | {  Rut: nombre de usuario,  clave: contraseña del usuario  } | Token de sesión |

|  |  |  |  |
| --- | --- | --- | --- |
| **Servicios pdf.js** | | | |
| **Url** | **Tipo** | **Parámetros** | **Respuesta** |
| **Documento.php** | POST | {  documentos\_caso,  up\_documento,  documentos\_observaciones  } | {  status,  documentos\_id  } |
| inspeccion\_pdf.php?inspeccion= | GET | Inspeccion: id de la inspeccion correspondiente. | Documento pdf de la inspección correspondiente. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Servicios ref.js** | | | |
| **Url** | **Tipo** | **Parámetros** | **Respuesta** |
| **siniestro\_full.php?caso=** | GET | Caso: id o número de siniestro. | Información completa de los datos del siniestro, bitácoras, gastos e inspecciones. |
| **inspeccion\_pdf.php?inspeccion=** | GET | Inspeccion: id de la inspeccion correspondiente. | Documento pdf de la inspección correspondiente. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Servicios uploadFile.js** | | | |
| **Url** | **Tipo** | **Parámetros** | **Respuesta** |
| **archivos\_gasto.php** | POST | {  gasto\_id,  filetype,  up\_documento  } | {  status  } |

# Anexos

## Control de cambios

|  |  |  |
| --- | --- | --- |
| **No. Revisión** | **Contenido** | **Realiza** |
| **1.0** | Documentación de aplicación | Bitfarm |