# Resultados

-Revisar alternativas para móvil

Los desafíos a los que se enfrenta la enseñanza hacen que se busquen continuamente nuevos métodos para hacer frente a estos problemas.

Con la intención de facilitar la programación de aplicaciones móviles surge App Inventor mediante un lenguaje visual basado en bloques.

App Inventor para Android, es un entorno visual de programación en bloques, que posibilita el desarrollo de aplicaciones para dispositivos móviles Android de forma bastante sencilla sobre todo si es comparado con los lenguajes de programación tradicionales.

El entorno App Inventor

App Inventor es un lenguaje visual de programación para la creación de aplicaciones para dispositivos móviles, desarrollada conjuntamente por Google y MIT (Instituto Tecnológico de Massachusetts). App Inventor está basado en un entorno de desarrollo compuesto de bloques, sin embargo, la diferencia fundamental radica en permitir a los usuarios construir aplicaciones que disponen de servicios basados en la web, lectura de códigos de barras, interacción con redes sociales, interacción con sensores de orientación y geolocalización.

La creación de aplicaciones con App Inventor es bastante intuitiva y no requiere conocimientos previos avanzados en programación. Además, las aplicaciones creadas se pueden utilizar en cualquier dispositivo con plataforma Android. App Inventor está basado en componentes los cuales el usuario los manipula mediante la interacción con la aplicación.

La construcción de una aplicación en esta plataforma se realiza a través de dos espacios:

App Inventor Designer y Blocks Editor. App Inventor Designer puede ejecutarse desde cualquier navegador, gracias a él podemos crear visualmente el interfaz de la aplicación simplemente haciendo clic y arrestando componentes como botones, cuadros de texto, figuras, sonidos o animaciones que se encuentran en Palette.

En la ventana de Bock Editor, se encuentran bloques que podemos conectar, estos bloques también se pueden añadir a la pantalla haciendo clic y a arrastrando hacia el espacio de trabajo, con ellos podemos controlar de qué manera se comportaran los componentes que previamente se han definido en la ventana de App Inventor Designer.

La aplicación nos permite realizar pruebas mediante un emulador o instalar la aplicación y ejecutarla directamente desde un dispositivo Android (smartphone o tablet).

Ventajas de utilizar App inventor respecto a otros entornos.

Utilizar App inventor trae consigo ventajas respecto a otras plataformas que podrían dar resultados similares al que se busca en este trabajo.

La ventaja principal de crear una App nativa (es una aplicación desarrollada para smartphones en el lenguaje nativo del propio terminal) respecto a una Web App es que la App nativa una vez descargada e instalada en el smartphone estará siempre a disposición del usuario sin necesidad de acceder a internet cada vez que quiera acceder a ella.

Otras ventajas de que la aplicación esté instalada en el dispositivo móvil son la posibilidad de acceder a elementos como la cámara o la ubicación además de ser más rápida que una web App.

Las ventajas de utilizar App inventor respecto a otras plataformas de programación de App nativas está en que el lenguaje de programación de App inventor consiste en un entorno visual de programación con bloques. Esto hace más sencillo la creación de la aplicación ya que no es necesario saber otros lenguajes de programación como Java, JavaScript, HTML5…

Estas ventajas hacen que utilizar App inventor para realizar la aplicación sea la mejor opción.

-Modelo de base de datos

La base de datos está compuesta por 4 tablas la primera tabla se utiliza para el registro de los usuarios. La segunda tabla está compuesta por el Id de los usuarios, el Id de los fallos que puede tener la máquina, el id de la máquina a la que corresponda el fallo y otros datos que ayuden a resolver el problema. Por último, en las tablas restantes irán los fallos junto con su correspondiente identificador y en la última tabla estarán los identificadores de las maquinas asociadas a los datos correspondientes para su localización.

La tabla de registro está compuesta por la columna id que irá completando automáticamente al recibir el dato de la columna usuario y correo. Cada usuario solo podrá aparecer una vez en la tabla, de este modo únicamente tendrá un valor de id.

La tabla de incidencias recibirá los datos introducidos por el usuario correspondientes a la incidencia, estos datos se relacionarán con la tabla de registro mediante el valor de id del usuario al que irán asociados también a la incidencia en la segunda tabla.

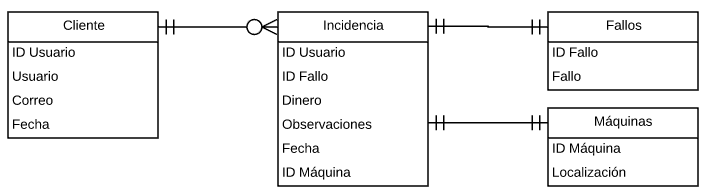
|  |  |  |  |
| --- | --- | --- | --- |
| **Registro** | | | |
| **Id Usuario** | **Usuario** | **Correo** | **Fecha** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Incidencias** | | | | | |
| **ID Usuario** | **ID Fallo** | **Dinero** | **Comentario** | **ID Máquina** | **Fecha** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

|  |  |
| --- | --- |
| **Fallos** | |
| **ID Fallos** | **Fallo** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

|  |  |
| --- | --- |
| **Maquinas** | |
| **ID Máquina** | **Localización** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

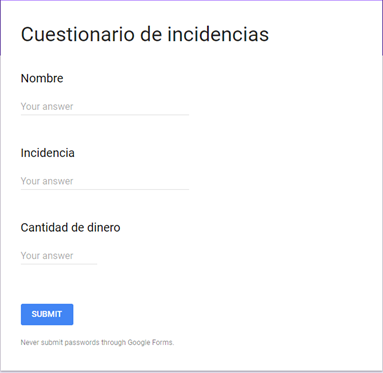
De este modo para un id solo puede existir un único usuario, pero puede haber varias filas en la tabla 2 asociadas a este id.



-Aplicación dos formularios: identificación e incidencias.

Para recibir las incidencias a la base de datos es necesario disponer de alguna plataforma en la que poder publicar las preguntas que está buscando el cliente afectado para que este pueda responder con su problema. Para esto Google Sheets ofrece la posibilidad de crear formularios y almacenar las respuestas a cada pregunta en columnas.

Podríamos crear un cuestionario con los apartados para nombre, incidencia, cantidad de dinero por si ese fuera el problema y quedaría de la siguiente forma.

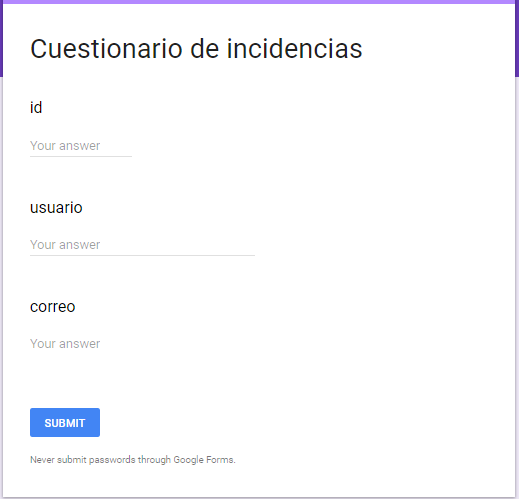


Formulario de Google para registro de usuarios e incidencias

Pero este resultado es bastante primitivo aparte de que lo más correcto sería que en la base de datos estuvieran en tablas separadas los datos del usuario y las incidencias.

Por eso crearemos dos formularios, uno para almacenar los datos del usuario, y otro para registrar las incidencias. El objetivo es subir estos dos cuestionarios de alguna manera a la aplicación para almacenar los datos.

El cuestionario elegido tendrá secciones para almacenar: id, usuario y correo. Quedando de la siguiente manera.



Formulario de Google para registro de usuarios

El id será un numero único para cada usuario y será asignado por la base de datos, el usuario será el nombre con el que el cliente afectado quiera que le nombremos y el correo electrónico será necesario para informarle del estado en el que se encuentra su incidencia.

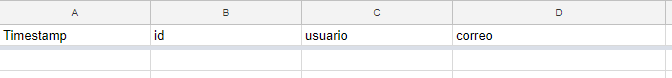


Tabla de Google Sheets para registro de usuarios

Como vemos se ha creado una columna para cada sesión del cuestionario, adema, podemos observar que aparte de crearse estas tres columnas también se ha creado una cuarta con la fecha en el que se envía el formulario. Ya tenemos creada la tabla con los campos necesarios para el registro en la App.

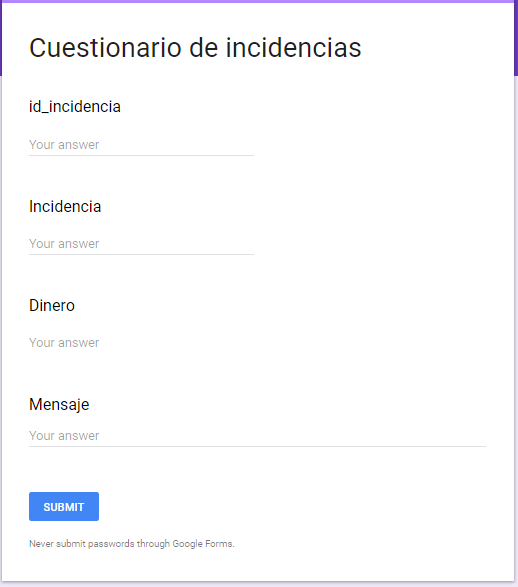
El cuestionario para almacenar las incidencias contendrá los siguientes campos:

-Id\_incidencia, al igual que para el usuario cada incidencia tendrá un id único e irrepetible con el cual se hará referencia al conjunto de la incidencia cuando sea necesario.

-Incidencia, en este espacio el usuario podrá detallar el problema que ha tenido con la máquina.

-Dinero, cuando el contratiempo se deba a una devolución incorrecta de la maquina al cliente el usuario introducirá en este lugar la cantidad no devuelta por la máquina.

-Además, aparecerá otra cuestión llamada mensaje en la que la persona que rellene el formulario podrá añadir más detalles de la incidencia si fuera necesario.



Formulario de Google para registro de incidencias

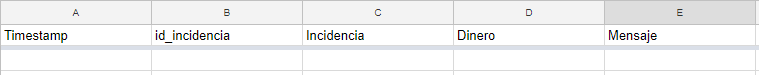


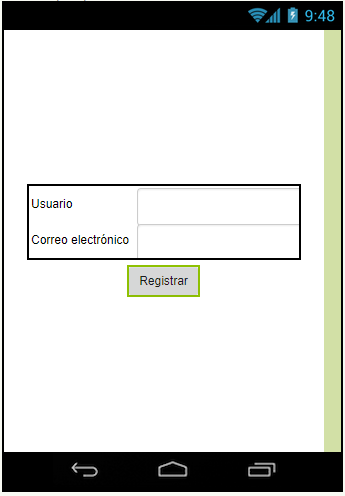
Tabla de Google Sheets para registro de incidencias

Al igual que antes podemos observar que aparte de crearse los encabezados de las columnas que hemos incluido en el formulario también se ha creado una columna con la fecha en la que se envía la incidencia.

Este dato resulta de gran interés ya que si conocemos la fecha en la que se registra la incidencia podremos clasificarla por tiempo o calcular la duración en resolver el problema.

Como dijimos en un punto anterior, utilizaremos App Inventor para crear la aplicación, para ello accedemos a la página web y creamos un nuevo proyecto.

Para crear el formulario de registro de usuario desde el apartado de diseño, arrastraremos a la pantalla dos cajas de texto, dos etiquetas de texto para identificar cada caja, un botón para enviar los datos del formulario y también arrastraremos el elemento Web que se encuentra en el apartado de conectividad, este es un componente no visible que permite gestionar solicitudes HTTP como GET, POST, PUT y DELETE una vez hecho quedara de la siguiente forma.



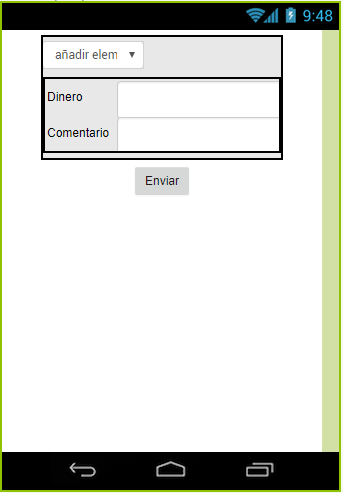
*Pantalla de registro de usuario*

Ahora crearemos otra pantalla en la que construiremos otro formulario, este será par a introducir los datos referentes a la incidencia.

Añadiremos a la pantalla un desplegable en el que introduciremos las incidencias más comunes (no me ha devuelto el cambio, no hay agua, no hay café, no hay vaso, no hay azúcar, el líquido está frío, maquina no operativa y cantidad insuficiente) y la opción otro por si no estuviera entre estas.

Agregaremos también a la pantalla dos cajas de texto, una para introducir la cantidad económica que no ha devuelto la maquina si este fuera el problema y la otra para introducir la incidencia si la opción fuera otra a las que aparecen en la lista. Estas cajas de texto dependeran de la selección de la incidencia, inicialmente no serán visibles y se verán o no en función de la selección del usuario.

También al igual que en el formulario de registro añadiremos la opción web de conectividad y un botón para enviar el formulario.



Pantalla de registro de incidencias

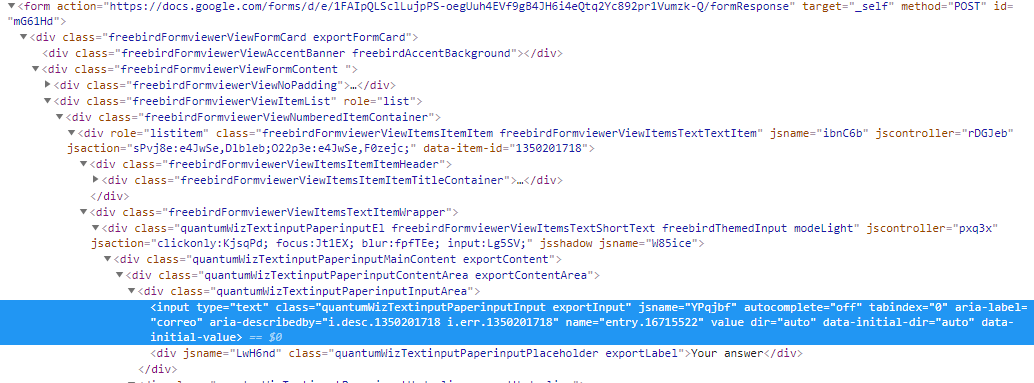
Desde la aplicación enviaremos los datos a la base de datos mediante un POST, es decir enviaremos los datos desde un enlace.

Para ello lo primero que haremos será ir al formulario en vivo y mirar el código fuente del formulario. Encontraremos la URL de publicación en la etiqueta acción del formulario y los elementos de la lista.



Código fuente del formulario de Google. URL

Para obtener los elementos de la lista, buscaremos la etiqueta de nombre hasta localizar un valor como “entry.16715522”.



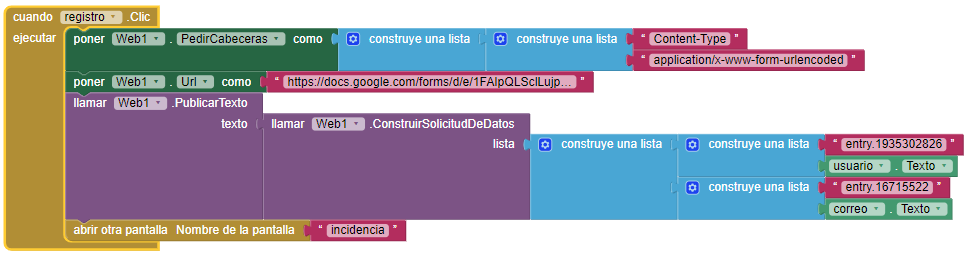
Código fuente del formulario de Google. Elementos

Esto lo tendremos que hacer tanto en el formulario de registro como en el de envío de incidencias.

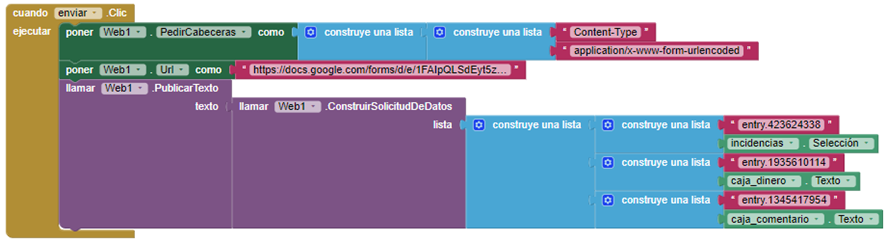
Después iremos a la hoja de cálculo y seleccionaremos Archivo – Compartir, hacemos clic en “avanzado” y cambiamos el acceso de privado a cualquiera con enlace y cambiamos el tipo de acceso de “puede ver” que viene por defecto a “se puede editar” y hacemos clic en guardar.

Ahora seleccionamos archivo – publicar en la web en la siguiente pantalla hacemos clic en publicar y luego en aceptar y cerrar.

Hechos esto ahora construiremos los bloques. Realizaremos el envío de los datos de registro del usuario mediante un post utilizando el enlace obtenido del código fuente del formulario y los nombres de los elementos que pusimos en el de la siguiente forma.

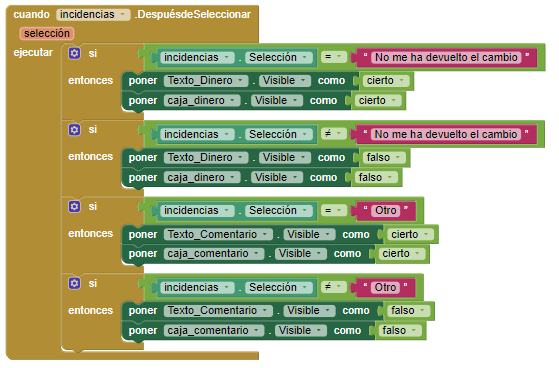


Bloques del botón registro de pantalla de registro de usuarios

Los datos referentes a las incidencias los enviaremos de la misma forma que los de registro de usuario, obtendremos el enlace del formulario y el nombre de los elementos que queremos conseguir. 

Bloques del botón enviar, pantalla de registro de usuarios

En este formulario sin embargo veremos las cajas de texto de dinero y de comentarios en función de la selección como podemos ver en la imagen.



Bloques de incidencias, pantalla de incidencias

-Relación usuario id

Anteriormente tanto en la tabla de registro de usuarios como en la de incidencias hemos creado un campo identificado como id. Este campo será muy útil para poder identificar cada incidencia o cada usuario por un número que será único en cada caso.

Este campo queremos que se cree automáticamente, para ello crearemos un script con un programa que nos realizará este trabajo.

Para ello desde la hoja de cálculo iremos a herramientas, script editor y crearemos un nuevo script.



Script crear id

Este programa lee el último dato de la columna de id y le suma +1. Este proceso se hará automáticamente cada vez que se envíe un formulario de usuario o de incidencias.

De este modo ya tenemos automatizada la creación del id para cada incidencia o usuario.

-Usuario recordado

Para que el uso de la aplicación sea lo más cómodo posible para el usuario lo ideal es que una vez registrado, la aplicación tiene que ser capaz de recordar sus datos.

Las aplicaciones creadas con App Inventor se inicializan cada vez que son ejecutadas. Esto quiere decir que si una aplicación fija el valor de una determinada variable y el usuario cierra la aplicación el valor de esta variable no será recordado la próxima vez que se inicialice la aplicación.

Para ello App Inventor dispone de TinyDB, este es un componente invisible que sirve para almacenar datos para la aplicación.

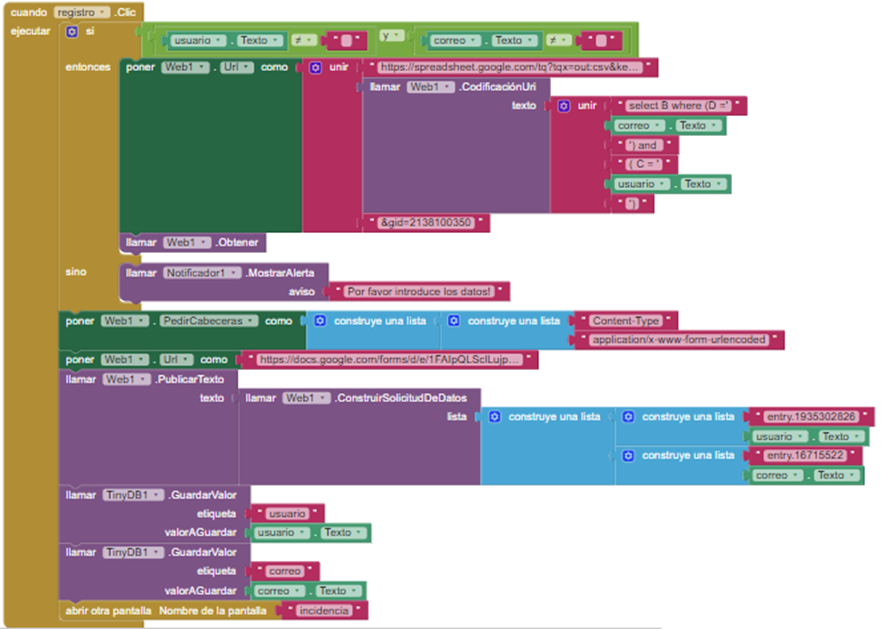
Los datos almacenados en TinyDB estarán disponibles cada vez que se abra la aplicación.

Los datos se guardan mediante valores con etiquetas. Para guardar un elemento hay que especificar la etiqueta con la que quiere guardarse. Luego para recuperar el valor solo hay que poner la etiqueta con la que hemos guardado ese dato.

Para añadir este componente simplemente lo arrastramos desde los componentes de almacenamiento hasta nuestra pantalla, una vez añadido el componente nos vamos a los bloques donde lo configuraremos.

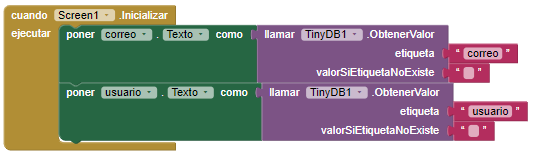
Añadiremos dos bloques de TinyDB en los cuales el valor a guardar será el texto escrito en los campos de texto de usuario y correo con las etiquetas usuario y correo que serán los nombres con los que se identificarán estas variables

Cuando el usuario haga clic en el botón de registrar automáticamente los valores de usuario y correo quedaran guardados en el dispositivo.



Bloques del botón registro con TinyDB, pantalla de registro de usuarios.

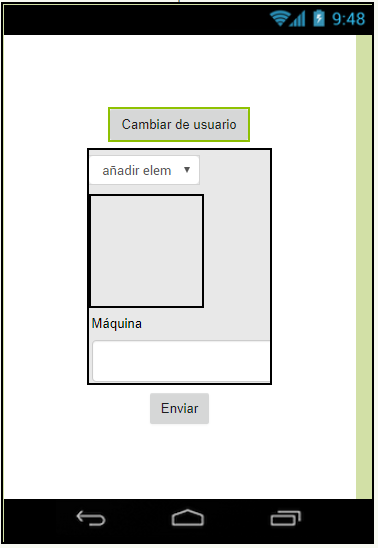
Ahora que ya podemos guardar los datos en el dispositivo crearemos un bloque para recordar los datos por defecto al abrir la aplicación, de este modo no será necesario registrarse una y otra vez, sino que la aplicación se abrirá con los datos que nos hemos registrado.



Bloques de inicialización, pantalla de registro de usuarios

Estos datos nos aparecerán siempre hasta una vez que los cambiemos. Para ello pondremos un acceso a la página de registro desde la página en la que registramos las incidencias.

Para ello arrastramos un botón desde la Paleta de componentes hasta nuestra pantalla y añadimos un bloque para realizar la función de pasar a la pantalla de registro de usuario.



Pantalla de registro de usuarios con opción de cambiar de usuario



Bloques del botón de cambiar de usuario, pantalla de incidencias

-Crear lista de incidencias

El sistema de automatización de incidencias que crearemos estará compuesto también de una página web en la que se podrán ver los resultados de la aplicación.

Se podrán ver los datos de forma sencilla como incidencias y estadísticas que facilitarán la resolución de los problemas ocasionados por las máquinas de café.

La página web mostrará los datos de las tablas que se encuentran en Google Sheets, dicho de otro modo, la intención es mostrar los datos que tenemos en Google Sheets pero de una forma más simplificada y atractiva.

Crearemos unos archivos para almacenar el código que será necesario para la página web, estos archivos los guardaremos en el repositorio GitHub.

GitHub es una plataforma de desarrollo colaborativo de software para guardar proyectos utilizando el sistema de versiones Git.

|  |
| --- |
| No sé si debo mostrar los archivos la página web o que |

Para crear la página web con estos archivos utilizaremos una aplicación llamada RawGit, esta aplicación actúa como un servidor de almacenamiento en caché para archivos que estén subidos a GitHub. Es básicamente una aplicación de internet que sirve archivos de GitHub a CDN externos. RawGit reenvía todas las solicitudes de usuario a GitHub y luego almacena en cache las respuestas a su navegador.

Para obtener nuestra página web únicamente tenemos que obtener la dirección del archivo en GitHub, ir a la página de RawGit pegar el enlace en el apartado de archivos en bruto y automáticamente se creará una URL que será la dirección de la web.

El aspecto de la página web que hemos creado es siguiente, en la parte superior hay un menú en el que aparecen las opciones incidencias, estadísticas y mantenimiento para poder movernos fácilmente.



Pagina web. Inicio

Tal como hemos visto en la página hay una sección para incidencias, en ellas se irán publicando tablas de incidencias con diversos motivos. Uno de estos gráficos que se publicarán será el de las ultimas incidencias. Pero publicar los gráficos tal cual no es estético, ni ofrece una vista cómoda de los datos, por ello hará que solucionar este problema.

La web nos proporciona infinidad de soluciones con las que resolver este problema, entre todas las posibilidades que hay en este caso se ha optado por utilizar Awesome Table.

Awesome table es una aplicación capaz de convertir una información simple y aburrida de una hoja de datos de Google Sheets en una vista atractiva.

Awesome table se integra perfectamente en el entorno de Google, la única condición para utilizar la aplicación web es tener una cuenta de Google.

Con esta aplicación es posible dar un determinado formato a los datos para que se vean de la manera que nosotros queramos, además estos datos se pueden ordenar forma sencilla mediante filtros interactivos.

Aparte de todo esto, esta aplicación web nos permite que incrustemos en cualquier página web el resultado que hemos obtenido.

Accedemos a la página web de Awesome table, ingresamos con nuestro usuario de Google, hacemos clic en crear una nueva vista, elegimos la hoja de Google Sheets de la que queremos extraer los datos y nos aparecerá la tabla en esta web.

Imagen de tabla de incidencias

La tabla que nos aparece a simple vista a cambiado muy poco, en las características podemos elegir el rango de datos de la tabla que queremos mostrar.

En parámetros avanzados podemos añadir un formato para nuestra tabla, esta apariencia estará dada por un programa que hayamos escrito en una hoja de Google Sheets, este se aplicará agregando la dirección de la celda en el que lo hayamos guardado.

Existen muchas posibilidades para crear una apariencia más atractiva para la tabla, en este caso el aspecto de los datos lo dará el siguiente texto en lenguaje html.

|  |
| --- |
| <p>-------------------------------------------------------------------------------------------</p>  </p><span style="font-size:20px">Fecha: ${"Timestamp"}</span><br>  <span style="font-size:20px">Usuario: ${"Usuario"}</span><br>  <span style="font-size:20px"><p>Incidencia: ${"Incidencia"}</span> <br>  <span style="font-size:20px">Cantidad: ${"Dinero"}</span> |

Plantilla para tabla de incidencias

Al aplicar esta configuración veremos la tabla con el siguiente aspecto.

Imagen de tabla de incidencias como etiquetas

Una de las ventajas de utilizar Awesome table es que podemos empotrar las tablas que creemos en cualquier página web que diseñemos. Para ello hacemos clic en compartir, después en empotrar y copiamos el código que nos aparece, este lo añadiremos al de nuestra página web.

-Crear lista de incidencias de última semana

A parte de la lista completa de incidencias también es interesante poder filtrar los datos por fecha, como por ejemplo realizar una búsqueda de las incidencias ocurridas en la última semana.

Para ello creamos otra hoja en Google Sheets y hacemos una consulta a la hoja en la que se almacenan los datos de las incidencias, la consulta será: =query(Formulario incidencias!A:D), formulario incidencias es la hoja en la que se encuentran los datos y A:D se refiere al rango de los datos que queremos extraer, que son la columnas de usuario, incidencia, dinero y timestamp.

Una vez hecha la consulta ya tenemos los datos que queremos en nuestra hoja actual, ahora solo tenemos que filtrar los datos por fecha.

Hacemos clic en el icono de crear un filtro, y veremos cómo nos han aparecido unos marcadores al lado de los encabezados de las columnas.



Tabla de incidencias de última semana

Hacemos clic en el marcador de Timestamp y le damos a filtro por condición, después a de la fecha es anterior a, y elegimos la opción de la semana pasada, le damos a ok y automáticamente aparecerán únicamente las incidencias de la semana pasada.

|  |
| --- |
| Tabla con incidencias de la semana pasada |

El objetivo de esta tabla es alojarla en la página web, al igual que en el punto anterior la apariencia de la tabla actual es bastante mejorable. Por ello realizaremos los pasos del punto anterior para transformar los datos de la hoja de Google Sheets en una Awesome table, quedando de la siguiente forma.

|  |
| --- |
| Awesome table de datos de la semana pasada |

Le damos a compartir, copiamos el código de nuestra Awesome table y lo pegamos junto con el código de nuestra página web.

-Cerrar incidencias desde la web mediante el mantenimiento de las máquinas

Además de proporcionar los datos generados por el sistema también funcionará como vínculo con el personal de mantenimiento proporcionando una pantalla útil para llevar un seguimiento del mantenimiento de las maquinas.

Dentro de la página web crearemos una sección para llevar el mantenimiento de la máquina, para ello deberemos crear otro archivo html, copiaremos el encabezado de la página principal y añadiremos un formulario, quedando de la siguiente forma.

|  |
| --- |
| Archivo html de mantenimiento |



Formulario pagina web, mantenimiento

El formulario que hemos creado dispondrá de un menú en el que el operario de mantenimiento podrá actualizar el estado de las maquinas, una vez que ha realizado su puesta a punto. Así una vez que se ha actualizado el estado de una máquina, todas las incidencias de esta se darán por finalizadas.

La función de este formulario será enviar el numero de la maquina junto con la fecha a una hoja de Google Sheets mediante el método POST.

El formulario de la página estará enlazado con el siguiente archivo de Google Script:

|  |
| --- |
| var sheetName = 'anular'  /\*  Obtiene un almacén de propiedades al que todos los usuarios pueden acceder, pero solo dentro de este script.  https://developers.google.com/apps-script/reference/properties/properties-service#getScriptProperties ()  \*/  var scriptProp = PropertiesService.getScriptProperties()  /\*  Esta es la función de configuración inicial. Obtiene el ID de SpreadsheetApp activo y lo agrega a nuestro PropertiesService.  https://developers.google.com/apps-script/reference/spreadsheet/spreadsheet-app#getactivespreadsheet  \*/  function intialSetup () {  var activeSpreadsheet = SpreadsheetApp.getActiveSpreadsheet()  scriptProp.setProperty('key', activeSpreadsheet.getId())  }  function doPost (e) {  /\*  Obtiene un bloqueo que impide que cualquier usuario ejecute simultáneamente una sección de código. Una sección de código  protegido por un bloqueo de script no se puede ejecutar simultáneamente, independientemente de la identidad del usuario.  https://developers.google.com/apps-script/reference/lock/lock-service#getScriptLock ()  \*/  var lock = LockService.getScriptLock()  /\*  Intenta adquirir el bloqueo y se agota el tiempo de espera con una excepción después del número de milisegundos proporcionado.  Este método es el mismo que tryLock (timeoutInMillis), excepto que lanza una excepción cuando el bloqueo  No se pudo adquirir en lugar de devolver el falso.  https://developers.google.com/apps-script/reference/lock/lock#waitLock(Integer)  \*/  lock.tryLock(10000)  try {  /\*  Abre la hoja de cálculo con la ID dada. Un ID de hoja de cálculo se puede extraer de su URL. Por ejemplo,  El ID de la hoja de cálculo en la URL https://docs.google.com/spreadsheets/d/abc1234567/edit#gid=0 es "abc1234567".  https://developers.google.com/apps-script/reference/spreadsheet/spreadsheet-app#openbyidid  \*/  var doc = SpreadsheetApp.openById(scriptProp.getProperty('key'))  /\*  Devuelve una hoja con el nombre dado. Si varias hojas tienen el mismo nombre,  Se devuelve el de la izquierda. Devuelve nulo si no hay ninguna hoja con el nombre dado.  https://developers.google.com/apps-script/reference/spreadsheet/spreadsheet#getSheetByName(String)  \*/  var sheet = doc.getSheetByName(sheetName)  /\*  Devuelve el rango con la celda superior izquierda en las coordenadas dadas y con el número dado de filas.  https://developers.google.com/apps-script/reference/spreadsheet/sheet#getRange(Integer,Integer)  Luego devuelve la posición de la última columna que tiene contenido.  https://developers.google.com/apps-script/reference/spreadsheet/sheet#getlastcolumn  Luego devuelve la cuadrícula rectangular de valores para este rango (una matriz bidimensional de valores, indexada por fila y luego por columna).  https://developers.google.com/apps-script/reference/spreadsheet/range#getValues ()  \*/  var headers = sheet.getRange(1, 1, 1, sheet.getLastColumn()).getValues()[0]  // Obtiene la última fila y luego agrega una  var nextRow = sheet.getLastRow() + 1  /\*  Asigna la matriz de encabezados a una nueva matriz. Si el valor de un encabezado es 'timestamp', entonces  devuelve un nuevo objeto Date (), de lo contrario, devuelve el valor del parámetro de URL correspondiente  https://developers.google.com/apps-script/guides/web  \*/  var newRow = headers.map(function(header) {  return header === 'timestamp' ? new Date() : e.parameter[header]  })  /\*  Obtiene un rango desde la fila siguiente hasta la fila final según la cantidad de elementos que hay en newRow  luego establece los nuevos valores de toda la matriz a la vez.  https://developers.google.com/apps-script/reference/spreadsheet/range#setValues(Object)  \*/  sheet.getRange(nextRow, 1, 1, newRow.length).setValues([newRow])  /\*  Devolver resultados exitosos como JSON  https://developers.google.com/apps-script/reference/content/content-service  \*/  return ContentService  .createTextOutput(JSON.stringify({ 'result': 'success', 'row': nextRow }))  .setMimeType(ContentService.MimeType.JSON)  }  catch (e) {  return ContentService  .createTextOutput(JSON.stringify({ 'result': 'error', 'error': e }))  .setMimeType(ContentService.MimeType.JSON)  }  finally {  /\*  Libera el bloqueo, permitiendo que otros procesos que esperan en el bloqueo continúen.  https://developers.google.com/apps-script/reference/lock/lock#releaseLock ()  \*/  lock.releaseLock()  }  } |

Script, enviar id de máquinas

De esta forma escribiremos una fila con la maquina actualizada y la fecha en la que se ha realizado el mantenimiento en la hoja de Google Sheets que hemos llamado “anular” quedando de la siguiente forma:

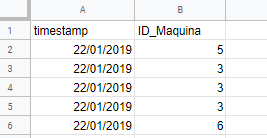
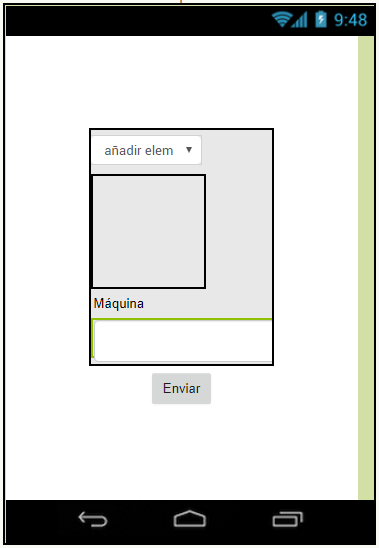


Tabla de máquinas actualizadas

-Añadir opción de máquinas en la aplicación

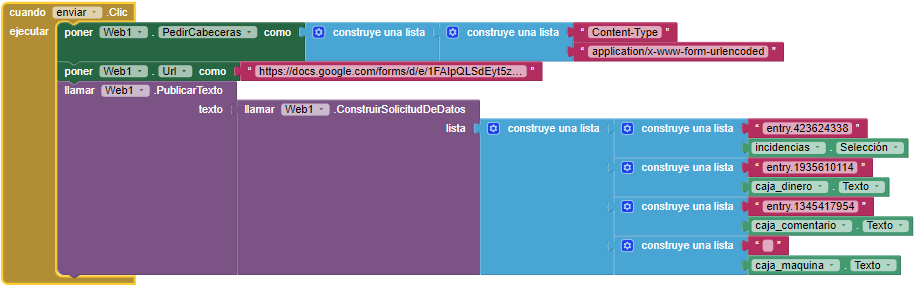
Para especificar mejor el problema ocurrido al usuario de las máquinas y añadir detalles que precisen mejor la incidencia para poder facilitar su resolución, añadiremos la opción para elegir la máquina que en concreto a tenido el problema.

Para ello volveremos a la página de App Inventor para ampliar el formulario de las incidencias. En el apartado de diseño añadiremos una etiqueta de texto y una caja de texto para añadir el numero de la máquina. Renombramos los componentes añadidos a la pantalla y en las propiedades de la caja de texto marcamos la opción solo números, ya que el identificativo de cada máquina será un número.



Pantalla de incidencias, añadido introducir id de máquina

Una vez añadidos los componentes pasamos a añadirles funcionalidades en la zona de bloques. Añadimos a la lista de los datos que queremos enviar el dato que se introduzca en la caja de texto de caja\_maquinas.

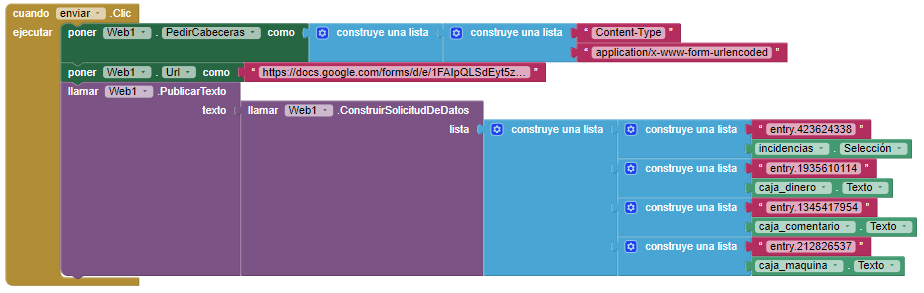


Bloque de botón enviar con máquina, pantalla de incidencias

El hueco que queda en blanco será la etiqueta de acción del formulario de la pregunta correspondiente a la opción máquina que crearemos.

Una vez editado el formulario de Google vemos el código fuente y copiamos la acción del formulario del apartado máquina.

Una vez pegado quedara de la siguiente forma.



Bloque botón enviar con máquina, pantalla de incidencias II

Ahora ya tenemos la tabla con la columna Máquina en la que se irán almacenando los id de las máquinas relacionadas con algún fallo que se introduzcan en la aplicación.

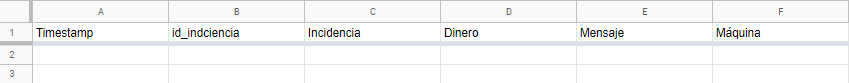
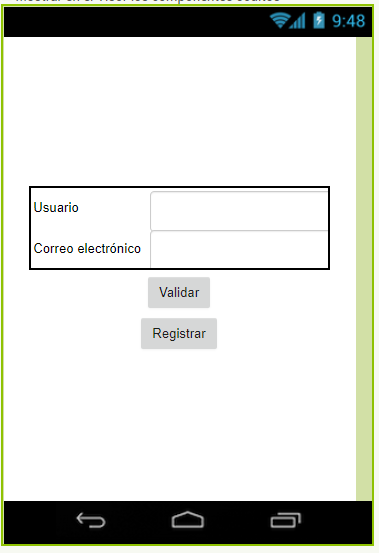


Tabla de incidencias con columna para máquinas

## -No repetir usuario

Para evitar que haya usuarios repetidos, pero con diferente id habrá que evitar que un mismo usuario se escriba dos veces en la base de datos. Para ello añadimos a la pantalla un botón al que llamaremos validar y deshabilitaremos el botón registrar modificando sus propiedades. La intención es que no nos deje registrar el usuario y el correo sin antes comprobar si está ya en la base de datos o no.



Pantalla de registro de usuario con botón validar

Después de hacer esta modificación en el diseño de la pantalla modificaremos los bloques para que realicen la función deseada.

Crearemos dos variables a las que inicializaremos sin ningún valor a las que llamaremos id y repetido.

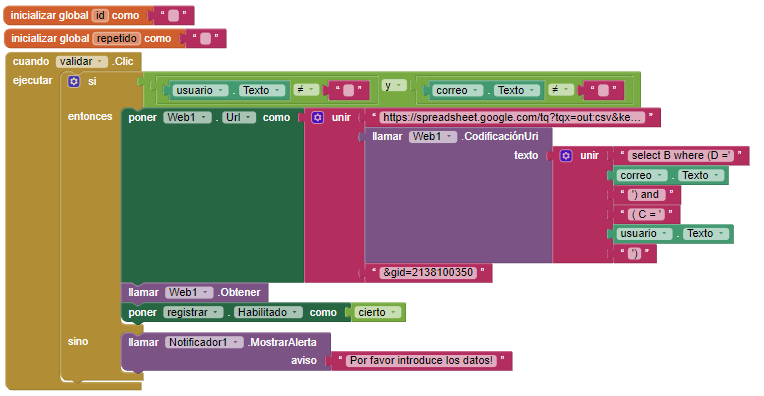
-La variable id obtendrá el valor de id de usuario en el caso de que el usuario y correo que introduzcamos ya se encuentre en la base de datos.

-La variable repetido podrá tener dos valores, verdadero si el correo y usuario ya se encuentra registrado o falso si por el contrario el usuario y correo que escribimos en las cajas de texto son nuevos y aun no se han registrado.

Cuando hacemos clic en validar con algún campo de texto vacío no nos dejará continuar y nos saltará una alerta que indicará que introduzcamos todos los datos.

Una vez que hemos introducido los datos y hecho clic en el botón validar el siguiente bloque realizará una consulta en nuestra base de datos mediante una url. Esta consulta se realizará uniendo un enlace da acceso a la tabla en la que queremos hacer la consulta y el código que nos resulte del transformador de código uri de la consulta que queremos realizar.

Para terminar con este bloque habilitaremos el botón registrar que hasta este momento se encontraba deshabilitado.

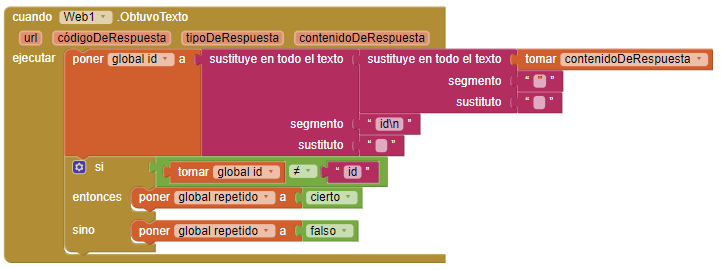


Bloque de botón validar, pantalla de registro de usuarios

Una vez mandada la consulta nos dará una respuesta que será el id de los datos si estos ya se encuentran en la base de datos o nada si estos datos aún no se han registrado.

El problema es que la respuesta vendrá en forma de columna y con los datos entre comillas, por ello sustituiremos las comillas y la palabra id por un texto vacío mediante una combinación de bloques para quedarnos con el valor de id si este la existe o la palabra id si este aun no existiera.

Para finalizar con este bloque pondremos la variable repetido en falso si el resultado de la combinación de bloques anteriores nos hubiera quedado “id” o en cierto si fuera diferente, en este caso el resultado habrá sido el id de los datos ya existentes en la base de datos.

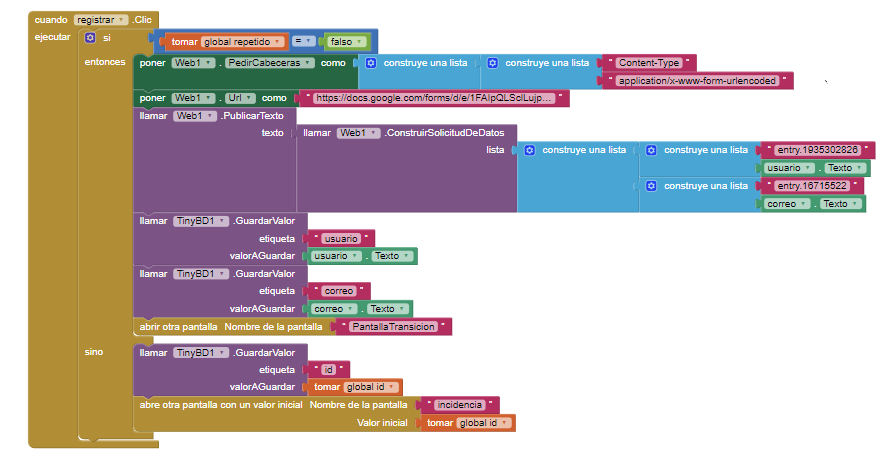


Bloque respuesta Web, pantalla de registro de usuarios

Modificaremos el bloque de las funciones del botón registrar realizado en un punto anterior, ahora al hacer clic en el botón registrar podremos encontrarnos con dos casos:

Si la variable repetido tiene el valor “falso” enviaremos los datos a la base de datos tal y como está hecho. La diferencia es que tras enviar los datos iremos a una página que crearemos a continuación que llamaremos PantallaTransicion con la intención de hallar el id de los datos que acabamos de mandar.

Si la variable repetido tiene el valor “cierto” almacenaremos el valor de id en TinyDB e iremos directamente al formulario de incidencias.



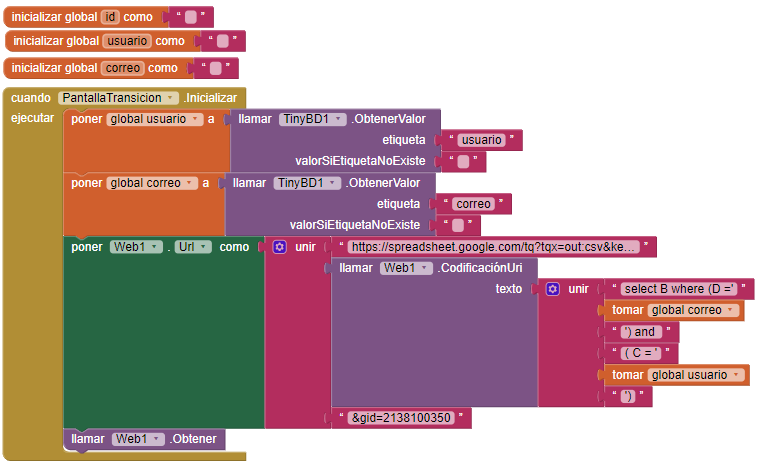
Bloque enviar con máquina, pantalla de incidencias

Como hemos dicho anteriormente crearemos una pantalla en la que hallaremos el id del usuario en los casos en los que el usuario no esté registrado y tengamos que haber mandado los datos.

Tras crear la pantalla PantallaTransicion, desde la ventana de diseño, arrastraremos de la paleta de componentes a nuestra pantalla los elementos TinyDB y Web.

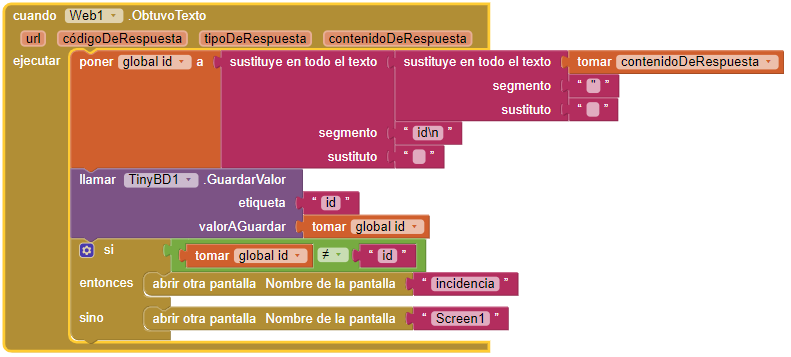
Luego desde la ventana de bloques crearemos las variables id, usuario y correo.

Después, crearemos un bloque en el que al inicializarse la pantalla pondremos las variables usuario y correo los valores almacenados en TinyDB y realizaremos una consulta mediante una URL de la misma manera que hemos hecho anteriormente.



Bloque iniciar PantallaTransicion, de la PantallaTransicion

Tras hacer la consulta obtenemos el valor de id y pasamos a la pantalla de incidencias.



Bloque respuesta Web, PantallaTransicion

-Ver mis últimas incidencias desde la aplicación

En este punto añadiremos a la aplicación una funcionalidad que podría ser de gran utilidad, incorporaremos una opción para poder ver todas las incidencias que se han introducido a la base de datos con el usuario que este activo en ese momento en la aplicación.

Para crear esta opción añadiremos otra página a la aplicación, la llamaremos mis\_incidencias. Arrastramos de la paleta de componentes dos botones, TinyDB, el componente web y un VisorWeb.

Renombraremos los botones, uno será el encargado de realizar la función de volver a la página anterior y lo llamaremos “volver” y el otro se encargará de mostrar las ultimas incidencias y lo nombraremos como “histórico”.

VisorWeb es un componente que nos permite ver páginas web, utilizaremos esta funcionalidad para realizar una búsqueda en nuestras tablas de Google Sheets mediante un enlace.

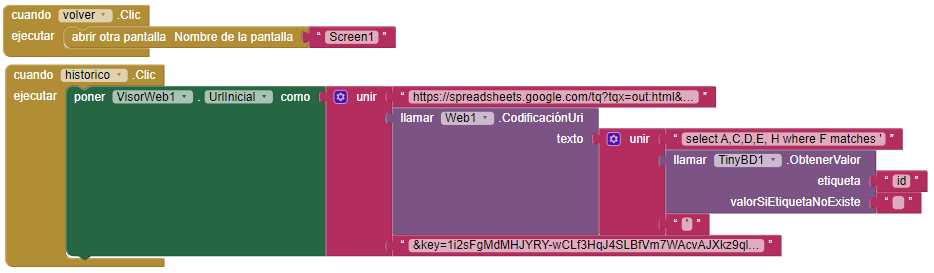


Pantalla mis\_incidencidas

La consulta que realizaremos a nuestra base de datos en la hoja de incidencias será:

select A,C,D,E, H where F matches 'id'

para realizar esto utilizaremos un enlace para acceder a la tabla y a este le añadiremos la consulta anterior mediante la configuración uri.



Bloque botón histórico, pantalla mis\_incidencias

Como podemos ver, cuando pulsamos el botón que hemos llamado histórico ponemos en el visor web el enlace que formamos al unir el acceso a la tabla y la consulta transformada a codificación uri mediante el componente web.

Automáticamente nos aparecerá en el visor web una vista de los datos que hemos buscado, en este caso las incidencias registradas por el usuario.

También se hemos creado un bloque con el cual podremos volver a la pantalla anterior.

## -Crear código para anular incidencia

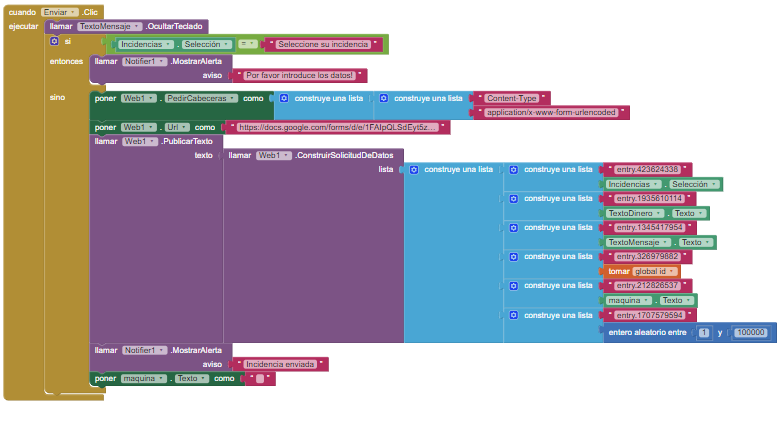
Hasta el momento en la tabla de incidencias estamos enviando datos como el problema sufrido por el cliente, la fecha y la máquina en la que se había ocasionado el problema.

Como lo que pretendemos es solucionar estos problemas, además de los datos descritos anteriormente añadiremos un código que podrá ser utilizado a la hora de solucionar la incidencia. Este código lo generaremos mediante bloques en la aplicación.

Realizaremos los mismos pasos que para enviar el id de la máquina, primero ampliamos el formulario de Google añadiendo una pregunta para el código, luego vamos al formulario, vemos el código fuente y copiamos la acción del formulario del apartado código.

Ahora modificaremos el bloque del botón enviar, ampliaremos la lista de datos que enviaremos añadiendo el código y pegaremos la etiqueta de acción que hemos copiado anteriormente.

Como hemos dicho antes el código lo crearemos desde la aplicación en este caso mediante un bloque que genera un numero aleatorio entre uno y 100000.



Bloque de botón enviar enviando código, pantalla mis\_incidencias



Tabla de incidencias con columna código

## -Enviar correo electrónico al iniciar incidencia.

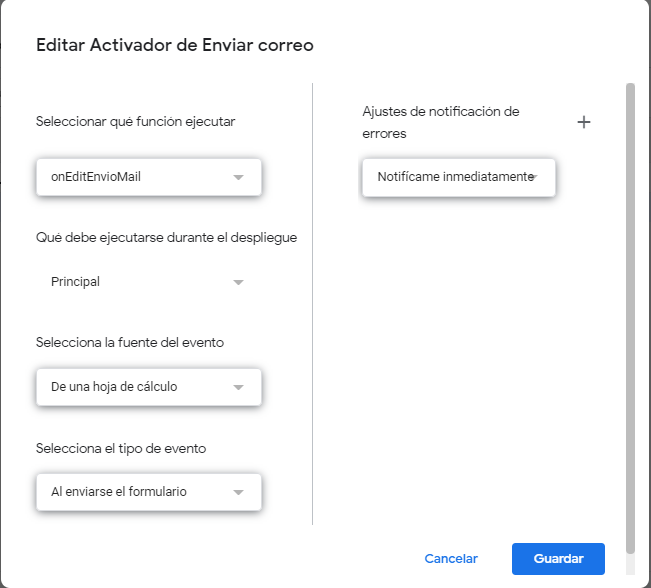
Uno de los objetivos de este trabajo es mantener conectados al usuario con la plataforma que se encargara de solucionar las incidencias. Por ello el usuario una vez registrado y mandado la incidencia recibirá un correo electrónico notificándole que la incidencia se a enviado correctamente. Adema en este correo irá el código de la incidencia, el usuario podrá introducir este código en la aplicación para poder dar por solucionada su incidencia.

Para poder realizar la función de mandar el correo automáticamente utilizaremos un archivo de Google Script. Para ello iremos a una hoja de Google Sheets y desde herramientas accedemos al editor de Script, crearemos el siguiente archivo.

|  |
| --- |
| Archivo mandar correo |

Para que la función de mandar el correo se realice automáticamente crearemos un trigger, desde la opción editar del menú accedemos a nuestros triggers y creamos uno nuevo.

Seleccionamos la función onEditEnvioMail, erigiremos una hoja de cálculo como fuente del evento, como evento elegimos al enviarse el formulario y le damos a guardar.



Trigger de función onEditEmvioMail

## -Enviar correo electrónico al cerrar incidencia.

Al igual que enviamos un correo al usuario cuando inicia la tramitación de una incidencia también se la enviaremos cuando esta se halla solucionado.

Cuando el operario de mantenimiento realice la puesta a punto de una determinada máquina, irá a la sección de mantenimiento de la página web de la plataforma y enviara el id de la maquina en el formulario.

Una vez mandado, este se registrará en una tabla. A partir de aquí, mediante un script de Google cogeremos los id de las maquinas, buscaremos las incidencias relacionadas con esta máquina, los usuarios de estas incidencias y con estos datos ya podremos mandar un correo a cada usuario notificando que su incidencia ya se ha solucionado.

|  |
| --- |
| Script de enviar correo o correo enviado |

-Cerrar incidencias desde la aplicación una vez solucionadas

Para que el usuario de la maquina pueda cerrar la incidencia cuando este vea que se a resuelto su problema habrá que poner a su disposición un mecanismo con el cual pueda realizar esta acción.

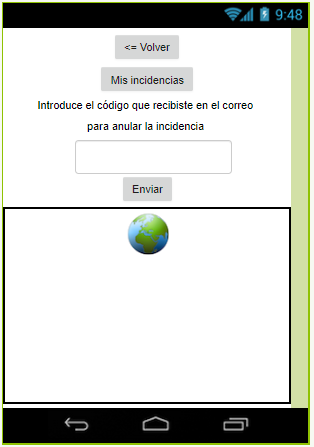
Iremos a Google Sheets desde donde crearemos una nueva hoja y desde esta un formulario con el campo código. Una vez hecho, iremos al código fuente de este formulario y copiaremos la etiqueta de acción de la pregunta código.

Ahora volveremos App Inventor y añadiremos los siguientes componentes a la pantalla de diseño “mis\_incidencias”:

-Una caja de texto que renombraremos “codigo”, este será el espacio en el que el usuario podrá introducir el código.

-Un botón, al que nombraremos “BotonAnular” que será el encargado de enviar el código.

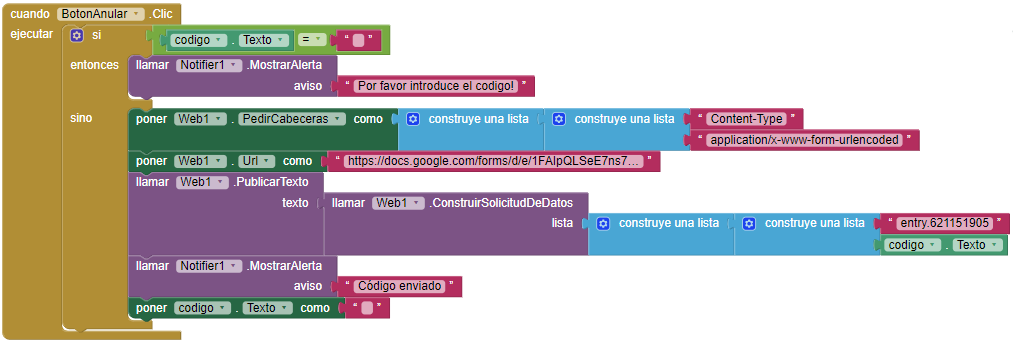
-El elemento de notificación, que nos ayudara a notificar cualquier fallo que se produzca.



Pantalla mis\_incidencias con campo de texto y botón para anular incidencia

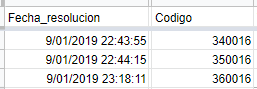
Una vez añadidos los componentes pasaremos a editar los bloques para poder realizar el trabajo descrito.

Crearemos un bloque que al hacer clic sobre el “BotonAnular” enviara mediante un post el código a la base de datos. Lo haremos de la misma forma que para registrarnos o enviar la incidencia, la diferencia es que para este bloque utilizaremos la etiqueta de código que hemos copiado anteriormente.



Bloque BotonAnular, pantalla mis\_incidencias

Después de realizar estos cambios, al escribir el código en la caja de texto y pulsar el “BotonAnular” se enviará junto con la fecha a la tabla elegida de nuestra base de datos.



*Tabla de código recibido*

## -Ver estadísticas desde la pagina web

Para llevar un control de lo que ocurre con las maquinas y llevar un seguimiento de los problemas más habituales, o conocer en que maquinas se ocasionan el mayor número de incidencias, crearemos una sección a la que llamaremos estadísticas.

En esta sección publicaremos gráficos actualizados con los de las máquinas e incidencias.

## -Crear código QR para descargar aplicación

Para facilitar a los usuarios la descarga de la aplicación crearemos un código QR. Una vez creado, lo colocaremos en todas las maquinas de modo que este accesible a los clientes.

## Fin. Resumen

Una vez terminado el trabajo se habrá obtenido una solución al problema. El resultado estará compuesto por una app que podrá instalarse en el móvil de forma sencilla, una página web y una base de datos.

La aplicación se podrá descargar mediante el sistema QR, en la maquina habrá una pegatina a la vista que contenga un código QR con el que se podrá descargar la aplicación a cualquier dispositivo móvil. Esta aplicación contendrá un formulario con el que se podrá introducir cualquier incidencia ocurrida con la máquina, además el cliente podrá crear un usuario y poder ver sus incidencias y el estado en el que se encuentran. Estas incidencias la página las mandará directamente a la base de datos. Una vez que el cliente vea su problema subsanado podrá anular la incidencia introduciendo un código en la aplicación, este código lo recibirá el usuario en el correo una vez creada la incidencia.

La página web será una plataforma virtual en la que se podrán ver las incidencias, todas y las de la última semana. También habrá una sección donde se podrán ver estadísticas como maquinas que causan más problemas o problemas más comunes. La página web contendrá también un espacio donde se podrán terminar las incidencias, las incidencias provocadas por una devolución errónea se finalizarán una vez se a devuelto el dinero y el resto de las incidencias se darán por terminadas una vez se ha pasado el técnico para realizar el mantenimiento de la máquina.

La base de datos almacenará todos los datos necesarios para la gestión de las incidencias, manejará también los datos para el estudio de estadísticas, además, se encargará de tramitar las incidencias con la ayuda de programas en Google Apps Script.