**Aplicativo Móvil de Seguimiento de Mercado Movil**

**MANUAL**

**TÉCNICO**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **CONTROL DE VERSIONES** | | | | | | |
| **Versión** | **Hecha por** | **Revisada por** | **Aprobada por** | **Fecha** | **Motivo** | |
| 1.0 | LH,AB,OR | LH,AB,OR | LH, AB, OR | 24/06/2020 | Versión Original | |
| 1.1. | LH,AB,OR | LH,AB,OR | LH,AB,OR | 27/06/20 | Actualización de pruebas, programas y diagrama de clases |

**INDICE**

[1. INTRODUCCION 4](#_Toc44248671)

[2. DELIMITACION 4](#_Toc44248672)

[3. JUSTIFICACION 5](#_Toc44248673)

[4. OBJETIVOS 5](#_Toc44248674)

[5. ANALISIS 6](#_Toc44248675)

[5.1. ADMINISTRADOR 6](#_Toc44248676)

[5.2. CLIENTE 8](#_Toc44248677)

[6. DISEÑO 10](#_Toc44248678)

[6.1. Caso de uso general 10](#_Toc44248679)

[6.2. Vista de Casos de Uso de Gestionar Unidades 10](#_Toc44248680)

[6.3. Vista de Casos de Uso de Gestionar Personal 11](#_Toc44248681)

[6.4. Vista de Casos de Uso de Registrar Recorrido. 11](#_Toc44248682)

[6.5. Vista de Casos de Uso de Visualizar Unidades 12](#_Toc44248683)

[6.6. Diagrama de clases 12](#_Toc44248684)

[7. REQUERIMIENTOS TÉCNICOS. 13](#_Toc44248685)

[7.1. REQUERIMIENTOS MÍNIMOS DE HARDWARE 13](#_Toc44248686)

[7.2. REQUERIMIENTOS MÍNIMOS DE SOFTWARE 13](#_Toc44248687)

[8. HERRAMIENTAS UTILIZADAS PARA EL DESARROLLO 14](#_Toc44248688)

[8.1. ANDROID STUDIO PROJECTS 14](#_Toc44248694)

[8.2. JAVA 15](#_Toc44248695)

[8.3. FIREBASE DE GOOGLE 16](#_Toc44248696)

[9. PROGRAMAS 17](#_Toc44248697)

[10. PRUEBAS 17](#_Toc44248698)

[CAPTURA DE LAS PRUEBAS 17](#_Toc44248699)

1. INTRODUCCION

Este manual describe los pasos necesarios para cualquier persona que tenga ciertas bases de sistemas pueda realizar la configuración y ejecución del sistema. Para ello se establece es importante tener en cuenta que en el presente manual establecer las especificaciones mínimas de hardware y software para la correcta instalación de la aplicación. Además, se deberán establecer las pruebas que se deberán realizar para asegurar la correcta instalación del sistema.

1. DELIMITACION

El documento será elaborado para los Ingenieros de Sistemas o personas con conocimiento en desarrollo de software, orientado a guiar el proceso de dar soporte al Sistema de Información.

El proyecto está realizado de acuerdo con la metodología RUP con sus fases de Inicio, Elaboración, Construcción y Cierre.

El proyecto se realiza en base a un cronograma en las que se especifica las fechas de cada fase de la metodología y sus entregables (fechas de inicio y fecha final).

El presente proyecto cuenta con el presupuesto mínimo para la realización de la implementación y pruebas.

1. JUSTIFICACION

Debido a la falta de control en aglomeración de personas en los mercados y el buen estado de los productos, es por ello que se realizó el proyecto que está desarrollado para en entorno móvil en el que se podrá dar seguimiento el recorrido de los mercados móviles quien estará a cargo de los conductores de cada unidad, y la otra aplicación en donde el cliente podrá visualizar los mercados, dar seguimiento y ver el estado de cada unidad.

1. OBJETIVOS

* **General:**

Brindar la información necesaria para poder realizar la configuración y ejecución de la aplicación, a través de los artefactos de ingeniería de software generados en el desarrollo del aplicativo.

* **Específicos**
* Representar la funcionalidad técnica de la estructura, diseño y definición de la aplicación.
* Definir claramente el procedimiento de configuración de la aplicación.
* Detallar la especificación de los requerimientos de Hardware y Software necesarios para la ejecución de la aplicación.

1. ANALISIS
   1. ADMINISTRADOR
2. **RF001 Gestionar Unidades (ALTA)**

Se requiere que el administrador de la aplicación pueda registrar, editar y eliminar las unidades en la aplicación.

Para que se registre una unidad, la información que debe contener es la siguiente:

* + - Nombre de la Placa: Nombre breve que identifica a la unidad.
    - Nombre de la Unidad: Nombre breve (Mercado1, Mercado2, Mercado3)
    - Una foto referencial de la unidad.
    - Seleccionar la zona de la unidad (Cono sur, alto de la alianza, Pocollay, Centro de Tacna)

Una vez que se haya registrado la unidad se mostrara en la pantalla principal el estado de la unidad, la siguiente información que debe contener es la siguiente:

* + - Verde: El mercado está vacío (1 a 10 personas).
    - Amarillo: El mercado está a punto de llenarse (10 a 20 personas).
    - Rojo: El mercado está lleno (20 a 30 personas).

1. **RF002 Gestionar Productos**(**MEDIA)**

Se requiere que el administrador de la aplicación pueda registrar, editar y eliminar los productos en la pantalla principal de la aplicación.

Para que se registre un producto, la información que debe contener es la siguiente:

* + - Nombre de la unidad
    - Nombre del producto: Nombre breve que identifica al producto.
    - Una foto referencial del producto.
    - Precio del producto.
    - Categoría del producto: (Verduras, Abarrotes, Carnes)

1. **RF003 Gestionar Horario** **(MEDIA)**

Se requiere que el administrador de la aplicación pueda registrar, editar y eliminar el horario de las unidades.

Para que se registre el horario, la información que debe contener es la siguiente:

* + - Nombre de la unidad
    - Hora de inicio
    - Hora fin
    - Los días de atención de la unidad

1. **RF004 Enviar MENSAJE Y NOTIFICACION (MEDIA)**

Se requiere que el administrador de la aplicación pueda mandar alerta de llegada por zonas.

1. **RF005 Gestionar personal (ALTA)**

Se requiere que el administrador de la aplicación pueda ver, agregar, editar y eliminar el personal que estará atendiendo el mercado móvil, para que agregue un personal el administrador tiene que completar los siguientes datos:

* + - Seleccionar la unidad (Mercado móvil).
    - Nombres y apellidos completos
    - DNI
    - Domicilio
    - Celular
    - Tipo de personal (vendedor, chofer, copiloto, otro)

1. **RF006 Autenticación de usuario** **(BAJA)**

Se requiere que el administrador de la aplicación pueda iniciar sesión con su cuenta de usuario y contraseña.

* + Una cuenta de email.
  + Número telefónico.

1. **RF007 Registrar Recorrido (ALTA)**
   1. Se requiere que el administrador de la aplicación pueda registrar las rutas de las unidades.

Para que se registre una ruta, el administrador tendrá que seleccionar la unidad correspondiente y podrá agregar marcadores en el mapa según el recorrido que seguirá la unidad.

* 1. CLIENTE

1. **RF008 Registrar Usuario** **(BAJA)**

Se requiere que el cliente de la aplicación pueda registrar y editar sus datos en la pantalla principal de la aplicación.

Para que se registre un usuario, la información que debe contener es la siguiente:

* + - Nombre del usuario: Nombre que identifica al usuario.
    - Apellido del usuario: Apellido que identifica al usuario.
    - Zona del usuario: Nombre de la zona del usuario.
    - Dirección del usuario. Descripción breve de la dirección del usuario.
    - Contraseña del usuario: Contraseña de usuario para el acceso del usuario.

1. **RF009 autenticación de usuario** **(BAJA)**

Se requiere que el cliente pueda iniciar sesión con su cuenta de usuario y contraseña.

1. **RF010 Visualizar Unidades (ALTA)**

Se requiere que el cliente de la aplicación pueda visualizar y dar seguimiento las unidades en la pantalla principal de la aplicación.

La información que se va a visualizar es la siguiente:

* + - Nombre de la unidad: Nombre breve que identifica a la unidad.
    - Ubicación de la unidad: El punto de la ubicación de la unidad en el mapa en tiempo real.

Adicionalmente se diferenciarán las unidades por colores:

* + - Verde: Significara que la unidad esta libre, quiere decir que hay un mínimo de personas esperando a ser atendidas.
    - Amarillo: Significara que la unidad está a media capacidad, quiere decir que hay un promedio de personas esperando a ser atendidas.
    - Rojo: Significara que hay un tope de personas a ser atendidas, quiere decir que será mejor que el cliente espere en casa hasta que el estado de la unidad cambie a amarillo o verde para que no se exponga a la aglomeración.

1. **RF011 Visualizar lista de Unidades (MEDIA)**

El cliente podrá visualizar la lista de unidades con sus detalles correspondientes (personal de la unidad, placa de la unidad, la zona a la que esta pertenece y adicionalmente su ruta completa).

1. **RF012 Visualizar Productos (MEDIA)**

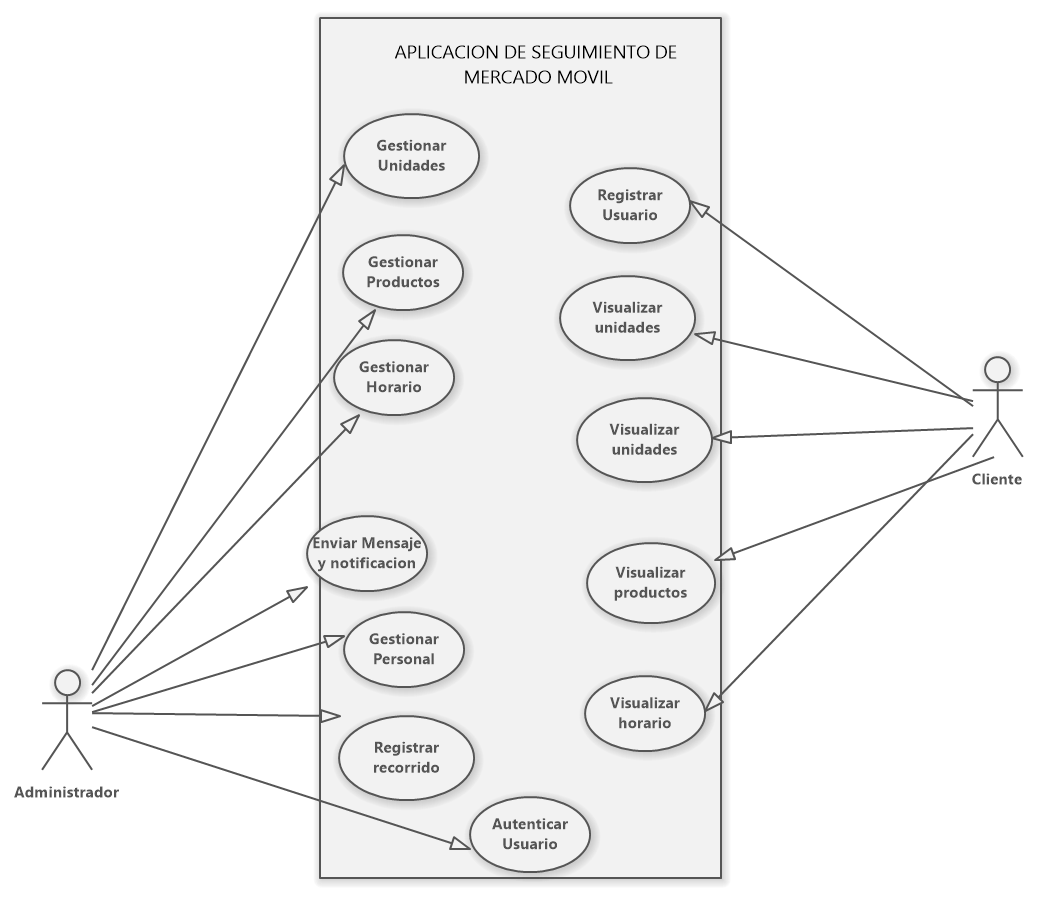
El cliente podrá ver los nombres de los insumos (según las categorías: “verduras”,” abarrotes”,” carnes”) disponibles en la unidad que haya seleccionado previamente, también podrá ver el precio promedio de cada insumo.

1. **RF014 Visualizar Horario (MEDIA)**

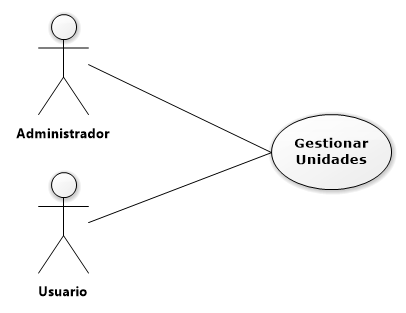
El cliente podrá visualizar el horario de recorrido de la unidad que seleccione:

* + - Podrá ver la hora de inicio y fin de su trayectoria y los días que trabajará dicha unidad

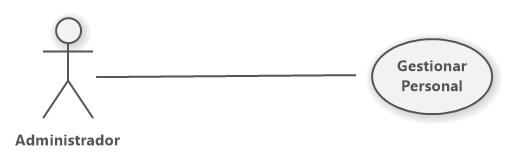
1. DISEÑO
   1. Caso de uso general



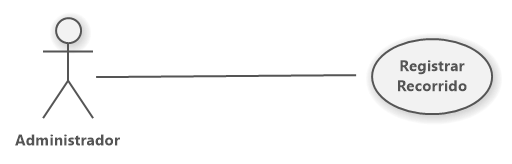
* 1. Vista de Casos de Uso de Gestionar Unidades



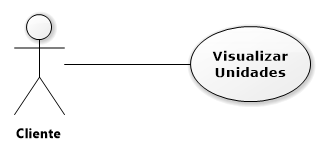
* 1. Vista de Casos de Uso de Gestionar Personal



* 1. Vista de Casos de Uso de Registrar Recorrido.



* 1. Vista de Casos de Uso de Visualizar Unidades



* 1. Diagrama de clases



1. REQUERIMIENTOS TÉCNICOS.
   1. REQUERIMIENTOS MÍNIMOS DE HARDWARE

* **Procesador :** Intel (R) Core (TM) i7-8550U CPU @ 1.80 GHz 1.99 GHz
* **Memoria RAM: Mínimo :** 8.00 Gigabytes (GB)
* **Disco Duro :** 500Gb.
  1. REQUERIMIENTOS MÍNIMOS DE SOFTWARE
* Privilegios de administrador
* Acceso a Internet
* Sistema Operativo: : Windows 10
* Programas:
* Android Studio Project
* Herramienta de Firebase de Google

1. HERRAMIENTAS UTILIZADAS PARA EL DESARROLLO
   1. ANDROID STUDIO PROJECTS

Es un IDE, una interfaz de desarrollo. En realidad, es una especie de escritorio de trabajo para un desarrollador. Allí se encuentra nuestro proyecto, las carpetas del mismo, los archivos que hay en él, y todo lo necesario para acabar creando la aplicación. Lo mejor de Android Studio es que ha sido creado por Google y fue presentado hace tan solo unos meses, por lo que no hablamos de una herramienta antigua y nada depurada, sino de un programa muy moderno que encima ha sido creado por los mismos que han creado el sistema operativo.

Cada proyecto de Android Studio incluye uno o más módulos con archivos de código fuente y archivos de recursos. Entre los tipos de módulos se incluyen los siguientes:

* Módulos de apps para Android
* Módulos de biblioteca
* Módulos de Google App Engine

De manera predeterminada, Android Studio muestra los archivos de tu proyecto en la vista de proyecto de Android

* 1. JAVA

Java es un lenguaje de programación orientado a objetos que se incorporó al ámbito de la informática en los años noventa. La idea de Java es que pueda realizarse programas con la posibilidad de ejecutarse en cualquier contexto, en cualquier ambiente, siendo así su portabilidad uno de sus principales logros. Fue desarrollado por Sun Microsystems, posteriormente adquirido por Oracle. En la actualidad puede utilizarse de modo gratuito, pudiéndose conseguir sin problemas un paquete para desarrolladores que oriente la actividad de programar en este lenguaje. Puede ser modificado por cualquiera, circunstancia que lo convierte en lo que comúnmente se denomina “código abierto”.

Cuando se programa con un lenguaje, se utiliza un tipo de código que la máquina es incapaz de reconocer. Es por ello que se requiere de un proceso de compilación, esto es, de “traducir” el lenguaje utilizado para que la máquina sea capaz de reconocerlo y procesarlo. Cada vez que un código requiere ser ejecutado en un ambiente distinto, deberá compilarse nuevamente. Con el desarrollo de Java se intentó evitar este tipo de circunstancia, haciendo que el código corriera en cualquier ambiente. El modo en que se logró este objetivo es desarrollando una máquina virtual que ejecutase el código compilado; esto significa que la “traducción” ya no se hace para que sea comprendida por un determinado sistema operativo; se hace para que la interprete la “máquina virtual”, que puede correr en cualquier sistema operativo. De esta manera, solo es necesario realizar una compilación, que podrá ser utilizada en cualquier entorno que tenga instalada la máquina virtual java.

* 1. FIREBASE DE GOOGLE

Firebase es una plataforma para el desarrollo de [aplicaciones web](https://es.wikipedia.org/wiki/Aplicaci%C3%B3n_web) y aplicaciones móviles desarrollada por [Google](https://es.wikipedia.org/wiki/Google) en 2014.[1](https://es.wikipedia.org/wiki/Firebase#cite_note-1)​

Es una plataforma ubicada en la nube, integrada con [Google Cloud Platform](https://es.wikipedia.org/wiki/Google_Cloud), que usa un conjunto de herramientas para la creación y sincronización de proyectos que serán dotados de alta calidad, haciendo posible el crecimiento del número de usuarios y dando resultado también a la obtención de una mayor monetización.

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

* Sincronizar fácilmente los datos de sus proyectos sin tener que administrar conexiones o escribir lógica de sincronización compleja.
* Usa un conjunto de herramientas multiplataforma: se integra fácilmente para plataformas web como en aplicaciones móviles. Es compatible con grandes plataformas, como [IOS](https://es.wikipedia.org/wiki/IOS), [Android](https://es.wikipedia.org/wiki/Android), [aplicaciones web](https://es.wikipedia.org/wiki/Aplicaci%C3%B3n_web), [Unity](https://es.wikipedia.org/wiki/Unity_(entorno_de_escritorio)) y [C++](https://es.wikipedia.org/wiki/C%2B%2B).
* Usa la infraestructura de Google y escala automáticamente para cualquier tipo de aplicación, desde las más pequeñas hasta las más potentes.
* Crea proyectos sin necesidad de un servidor: Las herramientas se incluyen en los [SDK](https://es.wikipedia.org/wiki/SDK) para los dispositivos móviles y web, por lo que no es necesario la creación de un servidor para el proyecto.

1. PROGRAMAS

(REFERENCIA DEL CODIGO FUENTE)

@Override  
**public void** mtdDoLogin(String email, String password) {  
 **mAuth**.signInWithEmailAndPassword(email, password).addOnCompleteListener(**new** OnCompleteListener<AuthResult>() {  
 @Override  
 **public void** onComplete(@NonNull Task<AuthResult> task) {  
 **if**(task.isSuccessful()){  
 **listener**.mtdOnSuccess();  
 }**else**{  
 **if**(task.getException()!=**null**)  
 **listener**.mtdOnError(task.getException().getMessage());  
 }  
 }  
 });  
}

@Override  
**public void** mtdDesactivarInputs() {  
 mtdSetEnable(**false**);  
}  
**private void** mtdSetEnable(**boolean** b) {  
 **miEditTextNombre**.setEnabled(b);  
 **miEditTextApellido**.setEnabled(b);  
 **zona**.setEnabled(b);  
 **miEditTextDireccion**.setEnabled(b);  
 **miEditTextEmail**.setEnabled(b);  
 **miEditTextPassword**.setEnabled(b);  
 **miButtonRegister**.setEnabled(b);  
}

@Override  
**public void** mtdHandleRegistro() {  
 **if**(!mtdValidarViews()){  
 **presenter**.mtdDoRegistro(**miEditTextNombre**.getText().toString(), **miEditTextApellido**.getText().toString(), **zona**.getSelectedItem().toString(), **miEditTextDireccion**.getText().toString(), **miEditTextEmail**.getText().toString().trim(), **miEditTextPassword**.getText().toString().trim());  
 }  
}

@Override  
**public boolean** mtdValidarViews() {  
 **boolean** retornarValor = **true**;  
 *//TextUtils.isEmpty: si esta vacio y es menor que 4, no es valido* **if** (TextUtils.*isEmpty*(**miEditTextNombre**.getText())) {  
 **miEditTextNombre**.setError(**"Este campo es obligatorio"**);  
 retornarValor = **false**;  
 }**else if**(**miEditTextNombre**.getText().toString().length() <2){  
 **miEditTextNombre**.setError(**"Debes escribir al menos 2 caracteres"**);  
 retornarValor = **false**;  
 }  
  
 **if** (TextUtils.*isEmpty*(**miEditTextApellido**.getText())) {  
 **miEditTextApellido**.setError(**"Este campo es obligatorio"**);  
 retornarValor = **false**;  
 }**else if**(**miEditTextApellido**.getText().toString().length() <2){  
 **miEditTextApellido**.setError(**"Debes escribir al menos 2 caracteres"**);  
 retornarValor = **false**;  
 }  
  
 **if** (TextUtils.*isEmpty*(**miEditTextDireccion**.getText())) {  
 **miEditTextDireccion**.setError(**"Este campo es obligatorio"**);  
 retornarValor = **false**;  
 }**else if**(**miEditTextDireccion**.getText().toString().length() <2){  
 **miEditTextDireccion**.setError(**"Debes escribir al menos 2 caracteres"**);  
 retornarValor = **false**;  
 }  
  
 **if** (TextUtils.*isEmpty*(**miEditTextEmail**.getText())) {  
 **miEditTextEmail**.setError(**"Este campo es obligatorio"**);  
 retornarValor = **false**;  
 }**else if**(!Patterns.***EMAIL\_ADDRESS***.matcher(**miEditTextEmail**.getText().toString().trim()).matches()){  
 **miEditTextEmail**.setError(**"No es un email valido"**);  
 retornarValor = **false**;  
 }  
  
 **if** (TextUtils.*isEmpty*(**miEditTextPassword**.getText())) {  
 **miEditTextPassword**.setError(**"Este campo es obligatorio"**);  
 retornarValor = **false**;  
 }**else if**(**miEditTextPassword**.getText().toString().length() <6){  
 **miEditTextPassword**.setError(**"Debes escribir al menos 6 caracteres"**);  
 retornarValor = **false**;  
 }  
  
 **return** retornarValor;  
}

1. PRUEBAS

CAPTURA DE LAS PRUEBAS