

**DEPARTAMENTO DE**

**CIENCIAS DE LA COMPUTACIÓN**

**INGENIERÍA EN**

**SISTEMAS E INFORMÁTICA**

*QUINTO NIVEL*

*“SISTEMA DE ACCESO A INFORMACIÓN, UBICACIÓN Y NAVEGACIÓN MEDIANTE UNA APLICACIÓN MÓVIL PARA PUNTOS DE INTERÉS DENTRO DEL CAMPUS DE LA UNIVERSIDAD DE LA FUERZAS ARMADAS - ESPE”.*

**Integrantes:**

Maria Belen Ceron

Alex Chicaiza

Elian Llorente

**Tutor:**

Ing. Jenny Ruiz

Sangolquí, 2021

**Historial de Revisiones**

| **Fecha** | **Versión** | **Descripción** | **Autor** |
| --- | --- | --- | --- |
| 22/07/2016 | 1.0 | Revisión documental  Capítulo 1 | Maria Belen Ceron  Alex Chicaiza  Elian Llorente |

Contenido

[**RESUMEN EJECUTIVO** 4](#_heading=h.1fob9te)

[**INTRODUCCIÓN** 5](#_heading=h.2et92p0)

[**CAPÍTULO I**](#_heading=h.1t3h5sf) 7

[**1.1 Título del Proyecto** 7](#_heading=h.2s8eyo1)

[**1.2. Sistema de Objetivos** 7](#_heading=h.26in1rg)

[**1.2.1 Objetivo General** 7](#_heading=h.35nkun2)

[**1.2.2 Objetivos Específicos** 7](#_heading=h.44sinio)

[**1.3 Alcance** 8](#_heading=h.z337ya)

[**1.4 Definición y Justificación del Problema** 9](#_heading=h.1y810tw)

[**1.5 Presupuesto** 11](#_heading=h.2xcytpi)

[**CAPÍTULO (ANALSIS) MV3II**](#_heading=h.1ci93xb) 14

[**2.1 Modelamiento del Negocio y sus Entregables** 14](#_heading=h.3whwml4)

[**2.1.1 Documentos de Caso de Uso** 14](#_heading=h.2bn6wsx)

[**2.2 Definición de Requerimientos** 14](#_heading=h.qsh70q)

[**2.2.1 Especificación de Requerimientos de Software** 14](#_heading=h.3as4poj)

[**2.2.2 Modelos de Casos de Uso** 33](#_heading=h.1pxezwc)

[**2.2.3 Documento de Especificaciones suplementarias** 34](#_heading=h.49x2ik5)

[**Introducción** 38](#_heading=h.2p2csry)

[**Propósito** 38](#_heading=h.147n2zr)

[**Alcance** 38](#_heading=h.3o7alnk)

[**Definiciones, acrónimos y abreviaturas** 38](#_heading=h.ihv636)

[**Referencias** 39](#_heading=h.1hmsyys)

[**Suposiciones y dependencias** 39](#_heading=h.41mghml)

[**Usabilidad** 40](#_heading=h.vx1227)

[**El cumplimiento del Sistema Operativo** 40](#_heading=h.3fwokq0)

[**Diseño para la facilidad de uso** 40](#_heading=h.1v1yuxt)

[**Consistente** 40](#_heading=h.4f1mdlm)

[**Consistencia Interna** 40](#_heading=h.2u6wntf)

[**Completo** 40](#_heading=h.19c6y18)

[**Correcto** 40](#_heading=h.3tbugp1)

[**Verificable** 40](#_heading=h.28h4qwu)

[**Modificable** 41](#_heading=h.nmf14n)

[**Fiabilidad** 41](#_heading=h.1mrcu09)

[**El tiempo medio entre fallos** 41](#_heading=h.46r0co2)

[**Acceso Fiabilidad** 41](#_heading=h.2lwamvv)

[**Rendimiento** 41](#_heading=h.111kx3o)

[**Tiempo de respuesta** 41](#_heading=h.3l18frh)

[**Capacidad** 41](#_heading=h.206ipza)

[**Compatibilidad** 41](#_heading=h.4k668n3)

[**Diseño de restricciones** 41](#_heading=h.1egqt2p)

[**Interfaces** 42](#_heading=h.2dlolyb)

[**Interfaces de usuario** 42](#_heading=h.3cqmetx)

[**Interfaces de software** 42](#_heading=h.4bvk7pj)

[**Requisitos necesarios de diseño** 42](#_heading=h.1664s55)

[**2.3 Análisis y Diseño** 43](#_heading=h.kgcv8k)

[**2.3.1 Modelo Conceptual** 43](#_heading=h.34g0dwd)

[**2.3.2 Modelo Lógico** 44](#_heading=h.43ky6rz)

[**2.3.3 Modelo Físico** 45](#_heading=h.2iq8gzs)

[**2.3.4 Script de la Base de Datos** 45](#_heading=h.xvir7l)

[**2.4 Implementación** 48](#_heading=h.3hv69ve)

[**2.4.1 Documento de Arquitectura PDiseño** 48](#_heading=h.1x0gk37)

[**2.4.2 Código Fuente Desarrollado**](#_heading=h.4h042r0) 48

[**CAPÍTULO IV**](#_heading=h.2w5ecyt) 71

[**3.1 Pruebas** 72](#_heading=h.1baon6m)

[**3.1.1 Documentación de Plan de Pruebas** 72](#_heading=h.3vac5uf)

[**Control de versiones Plantilla** 72](#_heading=h.2afmg28)

[**3.1.2 Documentación de Casos de Prueba** 82](#_heading=h.pkwqa1)

[**3.1.3 Documentación de Informe de Errores** 86](#_heading=h.39kk8xu)

[**3.1.4 Documentación Técnicas Caja Blanca y Caja Negra** 89](#_heading=h.1opuj5n)

**RESUMEN EJECUTIVO**

El presente Proyecto tiene como finalidad desarrollar un Sistema de acceso a información, navegación y ubicación mediante una aplicación móvil para puntos de interés dentro del campus de la Universidad de las Fuerzas Armadas-ESPE.

El objetivo principal es desarrollar una aplicación móvil que brinde información básica de los departamentos u otras dependencias de la Universidad, facilitando a personas que visitan el campus datos valiosos y concretos como la ubicación geográfica de las distintas dependencias en el campus.

La aplicación móvil brindará la función de guía geográfica a través del campus, permitiendo así a los usuarios que la usen llegar de una manera rápida y por una óptima ruta a un punto específico dentro del campus, sin tener la necesidad de averiguar a terceros información, que muchas veces resulta no ser del todo confiable.

La implementación del proyecto proveerá a la Universidad de las Fuerzas Armadas ESPE una herramienta tecnológica que permita mejorar el rendimiento de nuevos empleados y reducir el tiempo que tardan visitantes en encontrar lo buscado, dejando una agradable experiencia a los usuarios y ofreciendo una ventaja respecto a otras instituciones donde los trámites y la dificultad de encontrar los lugares importantes causa malestar en la comunidad universitaria.

**INTRODUCCIÓN**

En el presente documento se lleva el seguimiento del proyecto, desde el levantamiento de requerimientos hasta la implementación de la aplicación, con todos los anexos usados en la metodología SCRUM. Además, se busca demostrar la efectividad de usar dicha metodología en este tipo de proyectos y las ventajas de usar una metodología tradicional.

En el Capítulo I se tiene los lineamientos generales del documento como ls objetivos, alcance, definición y justificación del proyecto y el presupuesto que se espera tener en todo el proceso de desarrollo.

En el capítulo II se tiene el Modelamiento del Negocio y ls documentos entregables que contiene, así como la Definición de Requerimientos con el documento Especificación de Requerimientos de Software donde se entra más a detalle en los requerimientos del proyecto. Además en este capítulo de tiene los modelos usados en la etapa de Análisis y Diseño como son modelo Conceptual, Lógico, Físico y el Script de la Base de Datos utilizada. Se posee también los documentos referentes a la implementación del proyecto entre ellos el Documento de Arquitectura, Código Fuente de la aplicación y la Carta de Aceptación de Usuario Final.

En el Capítulo III se tiene la documentación referente a las pruebas realizadas posterior a la implementación, aquí se tiene documentos como Plan de Pruebas, Casos de Pruebas, Informe de Errores y Técnicas de Caja Blanca y Caja Negra.

En el capítulo IV se encuentran las Conclusiones y Recomendaciones obtenidas en el proceso de desarrollo del proyecto por parte de todo el equipo de trabajo.

Finalmente se posee la Bibliografía usada para el desarrollo, así como los Anexos como Manual de Usuario, Manual de Instalación y un CD con información digital.

**CAPÍTULO I**

**1.1 Título del Proyecto**

* SISTEMA DE ACCESO A INFORMACIÓN, UBICACIÓN Y NAVEGACIÓN MEDIANTE UNA APLICACIÓN MÓVIL PARA PUNTOS DE INTERÉS DENTRO DEL CAMPUS DE LA UNIVERSIDAD DE LA FUERZAS ARMADAS - ESPE

**1.2. Sistema de Objetivos**

**1.2.1 Objetivo General**

* Desarrollar e implementar una aplicación móvil de guía y navegación dentro de la Universidad de las Fuerzas Armadas – ESPE a manera de servicio de información a estudiantes y visitantes de la Universidad.

**1.2.2 Objetivos Específicos**

* Utilizar la metodología ágil SCRUM en el proceso de desarrollo.
* Implementar la opción de descarga de la aplicación móvil en Android a los usuarios.
* Implementar un servidor de bases de datos local dentro de la Universidad para la recuperación de la información necesaria para la Guía, y un servidor externo para acceso a la información desde cualquier ubicación.
* Utilizar un sistema GPS para el mapa de navegación el cual permita llegar al lugar deseado dentro de la Universidad.

**1.3 Alcance**

El presente proyecto cubre el desarrollo e implementación de una aplicación móvil, que ofrece el servicio de información, ubicación y navegación dentro de la Universidad de las Fuerzas Armadas – ESPE.

El desarrollo está previsto a realizarse con el apoyo de herramientas y lenguajes de programación tales como Java versión 8 y Base de Datos SQL Server 2014, y como base principal, la herramienta Android Studio versión 2.0.

Esta aplicación permitirá a los usuarios obtener información relevante acerca de los departamentos de la Universidad mediante un menú desplegable clasificando así las unidades como los departamentos para obtener una mejor búsqueda de la información dentro de la aplicación, se ha tomado como categorías principales a las siguientes:

* Rectorado.
* Vicerrectorado Administrativo.
* Vicerrectorado de Docencia.
* Departamentos.
* Vicerrectorado de Investigación, Innovación y Transferencia de Tecnología.
* Vicerrectorado Académico General

Dentro de esta sección de información se mostrará una descripción acerca de los departamentos, así como su logo, el edificio al que pertenece y el piso en el que se ubica.

Se presentará un mapa de ubicación con la ruta indicada para llegar desde la ubicación de la terminal de información hasta el departamento o ubicación deseada para lo cual se usará los servicios de localización mediante GPS del dispositivo móvil.

Se permitirá la instalación de la aplicación móvil en sistema operativo Android desde la versión 2.3 hasta 6.0 para una navegación a lo largo de la ruta.

**Funcionalidad**

* Separar las distintas unidades y departamentos en la Universidad de las Fuerzas Armadas-ESPE en categorías organizadas de una mejor manera para su correcta búsqueda dentro de la aplicación.
* Mostrar la descripción, el logo, la ubicación exacta e instrucciones de cómo llegar al departamento o unidad seleccionada por el usuario.
* Usar los servicios de localización GPS del dispositivo móvil para obtener la ruta más óptima para llegar al destino deseado.

**1.4 Definición y Justificación del Problema**

Actualmente, existe una gran cantidad de personas que se acercan a la Universidad de las Fuerzas Armadas – ESPE por primera vez, además existe un gran número de estudiantes de la unidad de Posgrados y de MED (Modalidad de Educación a Distancia), que no conocen el lugar, y necesitan preguntar a estudiantes, docentes o trabajadores públicos que se encuentren en ese momento en la Universidad tanto información de la unidad que buscan, como la forma de llegar a la misma.

Este problema se ve incrementado debido a la organización geográfica de la Universidad, la cual cuenta con las siguientes construcciones en donde se ubican las distintas unidades y departamentos:

* Edificio Administrativo, ubicado en la entrada de la Universidad.
* Edificios de aulas:
  + Edificio Central (Bloques A y B)
  + Edificio MED (Bloques C, D, G, H)
* Laboratorios
  + Física
  + Biotecnología
  + Materiales
  + Mecánica
  + Electrónica
  + CICTE
  + FIGMA
* Biblioteca
* Coliseo
* Edificios de Residencias Estudiantiles
  + Hombres
  + Mujeres
  + Archivo General
* Edificios recreacionales
  + Bar politécnico
  + Comisariato
  + Comedor Politécnico
  + Zona Transportes
* Departamento Médico

Todos estos edificios se encuentran ubicados a lo largo de la Universidad de las Fuerzas Armadas – ESPE, sede Sangolquí, la cual cuenta con una extensión de aproximadamente 40 hectáreas o 439870 m2 [[1]](#footnote-0), lo cual también dificulta la ubicación de un determinado departamento.

Por tal motivo, se ha visto la necesidad de proveer a estudiantes, visitantes y postulantes que desean ingresar a los diferentes programas académicos en la universidad, un sistema de guía digital, el cual incluya una descripción general de cada departamento, el edificio y piso en el cual se encuentra, y un mapa de navegación que permita llegar al lugar deseado dentro de la Universidad.

**1.5 Presupuesto**

| HARDWARE | | | |
| --- | --- | --- | --- |
| Cantidad | Descripción | Precio Unitario | V. Total |
| 1 | Computadora Portátil Toshiba Satellite P75-A7200 | 900 | 900 |
| 1 | Computadora Portátil Lenovo G-400s | 725 | 725 |
| 1 | Computadora Portátil Macbook Pro 15” | 2000 | 2000 |
| 1 | Computadora Portátil Sony VAIO, SVE1513KSXS | 1000 | 1000 |
| 1 | Computadora DELL Optiplex 990 (Servidor local) | 500 | 500 |
|  | | TOTAL | 5125 |

| SOFTWARE | | | |
| --- | --- | --- | --- |
| Cantidad | Descripción | Precio Unitario | V. Total |
| 1 | Sistema Operativo Ubuntu 14.04 | Gratuito | Gratuito |
| 2 | Sistema Operativo Windows 10 | 119.99 | 239.98 |
| 1 | Sistema Operativo OSX El Capitan | Incluido en el computador | Incluido en el computador |
| 1 | Windows Server 2012 R2 | Licencia Estudiante | Gratuito |
| 3 | Android Studio | Gratuito | Gratuito |
| 1 | Microsoft SQL Server 2014 | Licencia Estudiante | Gratuito |
|  | | TOTAL | 239.98 |

| ADMINISTRATIVO | | | |
| --- | --- | --- | --- |
| Cantidad | Descripción | Precio Unitario | V. Total |
| 1 | Resma de Papel | 3.70 | 3.70 |
| 4 | Recarga de tinta | 10 | 40 |
| 200 | Horas de Internet | 0.60 | 120 |
| 1 | Extra | 30 | 30 |
|  | | TOTAL | 193.70 |

| PRESUPUESTO TOTAL | |
| --- | --- |
| Hardware | 5125 |
| Software | 239.98 |
| Administrativo | 193.70 |
| Total | 5558.68 |

**CAPÍTULO II**

**2.1 Modelamiento del Negocio y sus Entregables**

**2.1.1 Documentos de Caso de Uso**

**2.2 Definición de Requerimientos**

**2.2.1 Especificación de Requerimientos de Software**

**Historial de Revisiones**

| **Fecha** | **Versión** | **Descripción** | **Autor** |
| --- | --- | --- | --- |
| 08/11/2016 | 1.0 | Desarrollo preliminar del documento: Especificación de requerimientos de software | * Daniel Marín * Adrián Mena * Joel Rivera * Diego Terán |
| 13/06/2016 | 1.1 | Revisión 1 del documento: Especificación de requerimientos de software | * Daniel Marín * Adrián Mena * Joel Rivera * Diego Terán |
| 16/06/2016 | 1.2 | Revisión 2 del documento: Especificación de requerimientos de software | * Adrián Mena |
| 29/06/2016 | 1.3 | Revisión 3 del documento: Especificación de requerimientos de software | * Daniel Marín * Adrián Mena * Joel Rivera * Diego Terán |

**Contenido**

**1.      Introducción**

1.1      Propósito

1.2      Alcance

1.3      Definiciones, Acrónimos, y Abreviaciones

1.3.1   Del negocio

1.3.2   Del sistema

1.3.3   De tecnología

1.4      Referencias

1.5      Visión General

**2.      Descripción General**

2.1      Perspectiva del Producto

2.1.1   Interfaces del sistema

2.1.2   Interfaces de Hardware

Requerimientos Mínimos

Requerimientos óptimos

2.1.4   Interfaces de Software

2.1.5   Interfaces de Comunicación

2.2      Funciones del Producto

2.3      Características del Usuario

2.4      Restricciones

2.5      Asunciones y dependencias

**3.      Requerimientos Específicos**

**3.1      Requisitos de las interfaces externas.**

3.1.1   Interfaz con el usuario.

3.1.2   Interfaz con el hardware.

3.1.3   Interfaz de comunicacione**s.**

**3.2 Requerimientos Funcionales**

**3.3      Especificación de Casos de Uso**

**3.4      Requerimientos No Funcionales**

**1. Introducción**

**1.1 Propósito**

Este documento tiene como propósito detallar la especificación de requerimientos funcionales y no funcionales de la página web que permitirá gestionar la información de una base de conocimiento con soluciones a problemas comunes surgidos en la UTIC, cumpliendo así con las expectativas del cliente.

Este documento está sujeto a constantes revisiones por parte del cliente para llegar a una aprobación final. Una vez aprobado servirá de base para el desarrollo del software y para posibles requerimientos futuros.

**1.2 Alcance**

| **Funcionalidad del software** |
| --- |
| 1. Separar las distintas unidades y departamentos en la Universidad de las Fuerzas Armadas-ESPE en categorías organizadas de una mejor manera para su correcta búsqueda dentro de la aplicación. |
| 1. Mostrar la descripción, el logo, la ubicación exacta e instrucciones de cómo llegar al departamento o unidad seleccionada por el usuario. |
| 1. Usar los servicios de localización GPS del dispositivo móvil para obtener la ruta más óptima para llegar al destino deseado. |

**1.3 Definiciones, Acrónimos y Abreviaciones**

***1.3.1    Del sistema***

1. *Metodología Scrum***:** Metodología ágil para el desarrollo de Software.
2. *Pruebas***:** Proceso mediante el cual se realizan actividades para verificar la óptima función del sistema.
3. *Entorno móvil***:** Hace referencia a un ambiente de desarrollo y/o ejecución programas o servicios en dispositivos móviles.
   * 1. ***De negocio***
4. *Aplicación móvil. -* *Producto software desarrollado para dispositivos móviles con Sistema Operativo Android.*
5. *Categoría. -* Agrupación general de las dependencias pertenecientes a la universidad.
6. *Dependencia.* - Departamento o unidad perteneciente a la universidad.

***1.3.3 De tecnología***

1. *Sistema de gestión de Base de Datos***:** Son un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan.
2. *Base de Datos***:** Es un conjunto de datos que pertenecen al mismo contexto almacenados sistemáticamente para su posterior uso.
3. *SQL Server 2014***:** Sistema de gestión de base de datos relacional.
4. *Android Studio 2.1:* IDE para el desarrollo Java para dispositivos móviles.

**1.4 Referencias**

* Historias de usuario.
* SCRUM
* Diagrama de casos de uso.
* Diagrama de clases.
* Modelos de bases de datos.

**1.5. Visión General**

El presente documento contiene una descripción general y detalle de características sobre el aplicativo móvil para la obtención de información y guía de ubicación dentro de la Universidad de las Fuerzas Armadas-ESPE

* + Perspectiva del producto
  + Interfaces del Sistema
  + Interfaces del Usuario
  + Interfaces del Hardware
  + Interfaces del Software
  + Interfaces de Comunicaciones
  + Restricciones de memoria
  + Funcionamientos
  + Funciones del producto
  + Características del usuario
  + Restricciones
  + Atención y dependencias
  + Los requisitos específicos
  + Requisitos de las interfaces externas
  + Interfaz con el usuario
  + Interfaz con el hardware
  + Interfaz con el software
  + Interfaces de comunicaciones
  + Requisitos funcionales
  + Requisitos del desarrollo
  + Restricciones del diseño
  + Atributos de sistema de software

**2.      Descripción General**

**2.1      Perspectiva del Producto**

El sistema a desarrollar tendrá como plataforma el entorno móvil, además de conectarse a la base de datos generada con información sobre las dependencias de la Universidad de las Fuerzas Armadas-ESPE.

**2.1.1   Interfaces del sistema**

El sistema tendrá interfaces de fácil uso que proveerán información obtenida de la base de datos generada y organizada de la mejor manera, así como un mapa de guía de la universidad.

**2.1.2 Interfaces de Usuario**

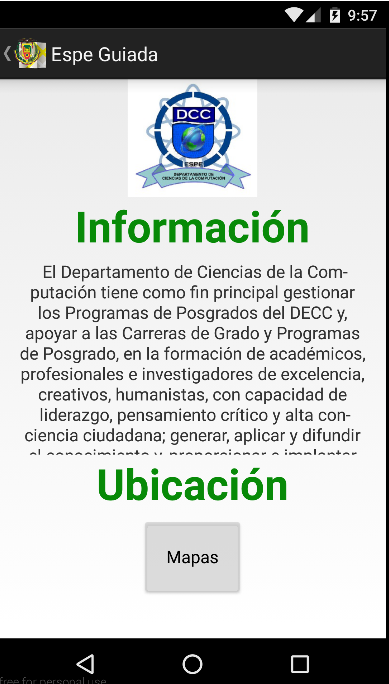
* **Pantalla inicial donde se muestran las categorías**

****

* **Listas desplegables para cada categoría que muestra sus subcategorías o dependencias.**

****

* **Pantalla que muestra la información de la dependencia.**

****

**2.1.3   Interfaces de Hardware**

**Requerimientos Mínimos**

* Sistema operativo Android v2.3.3 API 15
* Sistema GPS

**Requerimientos óptimos**

* Sistema Operativo Android v6.0 API 23
* Sistema GPS

**2.1.4    Interfaces de Software**

Para el buen funcionamiento del aplicativo web se aconseja que el Smartphone cumpla con las siguientes características:

* Sistema operativo Android v2.3.3 API 15 o superior

**2.1.5    Interfaces de Comunicación**

* Conexión a internet.

**2.2**    **Funciones del Producto**

La aplicación web debe tener las siguientes funcionalidades:

* Separar las distintas unidades y departamentos en la Universidad de las Fuerzas Armadas-ESPE en categorías organizadas de una mejor manera para su correcta búsqueda dentro de la aplicación.
* Mostrar la descripción, el logo, la ubicación exacta e instrucciones de cómo llegar al departamento o unidad seleccionada por el usuario.
* Usar los servicios de localización GPS del dispositivo móvil para obtener la ruta más óptima para llegar al destino deseado.

**2.3       Características del Usuario**

**Miembros y visitantes de la Universidad de las Fuerzas Armadas-ESPE:** consultar información de las diferentes dependencias dentro de la universidad.

**2.4       Restricciones**

* El sistema no permitirá la edición de datos de las dependencias.
* El acceso al servidor de base de datos se realizará en modo de solo lectura.
* El sistema no almacenará ningún dato de la base de datos de forma local en el dispositivo. Todo el almacenamiento de datos se realizará en el servidor remoto.
* El sistema no proveerá al usuario de una guía de forma hablada al usuario.

**2.5       Suposiciones y dependencias**

Para el uso de la aplicación móvil será necesario un dispositivo móvil con sistema operativo Android.

**3.Requerimientos Específicos**

El sistema estará sujeto a la plataforma en el que sea desarrollado.

**3.1. Requisitos de las interfaces externas.**

**3.1.1 Interfaz con el usuario.**

El sistema debe ser amigable y predictivo con el usuario ya que lo podrá utilizar cualquier persona que sepa operar un dispositivo móvil.

**3.1.2 Interfaz con el hardware.**

El sistema requiere componentes de hardware como GPS.

**3.1.3 Interfaz de comunicaciones.**

Adaptador wifi o de red móvil.

**3.2 Requisitos Funcionales**

**3.2.1 Requisito funcional 1**

| Id. Requerimiento | EG-RF001 |
| --- | --- |
| Nombre | Mostrar categorías |
| Actor | Usuario |
| Descripción | La aplicación móvil debe mostrar una lista de las categorías existentes en la universidad. |
| Entradas |  |
| Salidas | Lista de categorías |
| Proceso | 1. La primera pantalla de la aplicación muestra una lista de las categorías principales dentro de la universidad. |
| Precondiciones | Haber instalado la aplicación en el dispositivo. |
| Post condiciones | Se permitirá desplegar las dependencias pertenecientes a cada categoría. |
| Efectos Colaterales | Si no se posee una conexión a internet no se podrá acceder a la información a la base de datos y por lo tanto no se desplegará la información. |
| Prioridad | Media |

**3.2.2 Requisito funcional 2**

| Id. Requerimiento | EG-RF002 |
| --- | --- |
| Nombre | Mostrar dependencias |
| Actor | Usuario |
| Descripción | La aplicación móvil debe desplegar sub listas con las dependencias pertenecientes a cada categoría. |
| Entradas | -Seleccionar una categoría |
| Salidas | -Sub listas de dependencias para cada categoría. |
| Proceso | 1. Seleccionar la categoría 2. Visualizar la lista de dependencias pertenecientes a la categoría |
| Precondiciones | * Haber ingresado a la aplicación ESPE móvil * Haber seleccionado una categoría |
| Post condiciones | Se permitirá seleccionar una dependencia para mostrar su información. |
| Efectos Colaterales |  |
| Prioridad | Media |

**3.2.3 Requisito funcional 3**

| Id. Requerimiento | EG-RF003 |
| --- | --- |
| Nombre | Mostrar información dependencia |
| Actor | Usuario |
| Descripción | La aplicación móvil debe mostrar logo, descripción, información de contacto, micro-sitio. |
| Entradas | * Seleccionar una dependencia en la lista de dependencias. |
| Salidas | -Información desplegada en una pantalla. |
| Proceso | 1. Seleccionar una dependencia de la lista. 2. Visualizar la información detallada: Descripción breve de la dependencia, nombre y contacto (e-mail, teléfono) del director de la dependencia, micro-sitio de la dependencia. |
| Precondiciones | Haber seleccionado una dependencia |
| Post condiciones | Se permitirá el acceso a la opción de visualizar el mapa de guía con la ruta hacia el lugar buscado. |
| Efectos Colaterales |  |
| Prioridad | Baja |

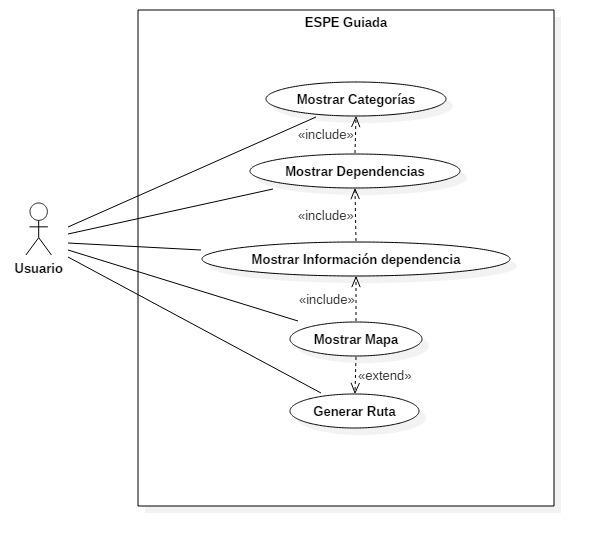
**3.2.4 Requisito funcional 4**

| Id. Requerimiento | EG-RF004 |
| --- | --- |
| Nombre | Mostrar mapa |
| Actor | Usuario |
| Descripción | La aplicación móvil mostrará un mapa de la universidad con la ubicación de la dependencia elegida. |
| Entradas | * Seleccionar la opción de mapas dentro de la información de la dependencia. |
| Salidas | -Se mostrará el mapa correspondiente a la universidad con la ruta a la dependencia en una nueva pantalla. |
| Proceso | 1. Seleccionar la opción de mapa dentro de la información de la dependencia. 2. se visualizará el mapa. |
| Precondiciones | 1. Haber seleccionado una dependencia y la opción de mapa. |
| Post condiciones | Se permite al usuario seguir la ruta que se trazará posteriormente en el mapa, o regresar a la pantalla anterior donde se despliega la información de la dependencia. |
| Efectos Colaterales |  |
| Prioridad | Alta |

**3.2.5 Requisito funcional 5**

| Id. Requerimiento | EG-RF005 |
| --- | --- |
| Nombre | Generar Ruta |
| Actor | Usuario |
| Descripción | Se generará la ruta trazada hacia la dependencia elegida. |
| Entradas | * Seleccionar la opción de mapas dentro de la información de la dependencia. |
| Salidas | -Se cargará la ruta al mapa. |
| Proceso | 1) Seleccionar la opción de mapa dentro de la información de la dependencia.  2) Se cargará la ruta especificada al lugar de destino.  3) Se mostrará la ruta sobre el mapa de la universidad. |
| Precondiciones | Haber seleccionado una dependencia y la opción de mapa. |
| Post condiciones | Se permite al usuario seguir la ruta trazada en el mapa, o regresar a la pantalla anterior donde se despliega la información de la dependencia. |
| Efectos Colaterales |  |
| Prioridad | Alta |

**3.2  Especificación de Casos de Uso**



**3.3 Requerimientos No Funcionales**

**3.4.1 Requisitos de Interfaz**

* El aplicativo móvil Android tendrá una interfaz accesible e intuitiva; los procedimientos que debe llevar el usuario serán muy claros.
* La aplicación mostrará la información en idioma español.

**3.4.2. Requisitos de Navegación**

* El aplicativo web tendrá consistencia entre los diferentes dispositivos que pueden funcionar con sistema operativo Android como un Smartphone o una Tablet en diferentes modelos, tamaños, capacidad de hardware, resolución de pantalla.
* El aplicativo móvil tendrá medios de navegación estándar (Regresar, salir).

**3.4.3. Requisitos de usabilidad.**

* El aplicativo móvil contará con una interfaz atractiva, simple y de fácil uso.
* El aplicativo móvil mostrará mensajes de error alertando que un procedimiento no se ha realizado correctamente.

**3.4.4. Requisitos de escalabilidad.**

* El aplicativo móvil estará en capacidad de realizar nuevos cambios, con nuevas funcionalidades en cuanto al volumen de datos y clasificación de las dependencias.

**3.4.5. Requisitos operacionales**

* El aplicativo móvil contará con listas desplegables y botones para cumplir las funciones de acceso a pantallas de datos y de mapas de navegación de la Universidad de las Fuerzas Armadas ESPE.

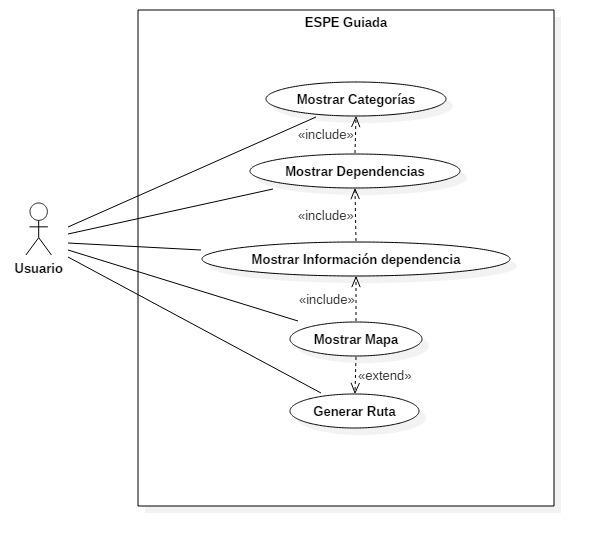
**3.4.6. Requisitos de seguridad**

* El aplicativo móvil no permitirá la edición de datos almacenados en la base de datos.
* El aplicativo móvil manejará los datos de cada dependencia de la Universidad y de sus directores de forma apropiada.

**Clasificación de Requisitos Funcionales**

| **Funcionalidad** | **Tipo** | **Responsable** |
| --- | --- | --- |
| Mostrar categorías | Alta | Adrián Mena |
| Mostrar dependencias | Alta | Adrián Mena, Joel Rivera |
| Mostrar información dependencia | Alta | Adrián Mena, Joel Rivera |
| Mostrar mapa | Media | Diego Terán, Daniel Marín |
| Generar ruta | Media | Diego Terán, Daniel Marín |

**2.2.2 Modelos de Casos de Uso**



**2.2.3 Documento de Especificaciones suplementarias**

**Historial de Revisiones**

| **Fecha** | **Versión** | **Descripción** | **Autor** |
| --- | --- | --- | --- |
| 29/07/2016 | 1.1 | Propuestas de Requerimientos Suplementarios | Marín Daniel  Mena Adrian  Rivera Joel  Terán Diego |

**Introducción**

**Propósito**

El presente documento ha sido redactado con el objetivo de definir las necesidades generales y especificas, conocidas como especificación de requerimientos de una correspondiente aplicación VISTA-CONTROLADOR, la cuál permitirá realizar una gestión tanto de socios como unidades dentro de la Compañía de Transportes Vingala, para poder satisfacer las necesidades que presenta actualmente dicha compañía.

Este documento estará al alcance de nuestros usuarios para las respectivas revisiones y posibles cambios que se presenten durante el desarrollo, esperando de esta manera poder llenar las expectativas y recibir una aprobación directa. Una vez que esto suceda podremos tenerlo de base para trabajar en el equipo de desarrollo con la elaboración del software.

**Alcance**

| **Funcionalidad del software** |
| --- |
| 1. Separar las distintas unidades y departamentos en la Universidad de las Fuerzas Armadas-ESPE en categorías organizadas de una mejor manera para su correcta búsqueda dentro de la aplicación. |
| 1. Mostrar la descripción, el logo, la ubicación exacta e instrucciones de cómo llegar al departamento o unidad seleccionada por el usuario. |
| 1. Usar los servicios de localización GPS del dispositivo móvil para obtener la ruta más óptima para llegar al destino deseado. |

**Definiciones, acrónimos y abreviaturas**

***1.3.1    Del sistema***

1. *Metodología Scrum***:** Metodología ágil para el desarrollo de Software.
2. *Pruebas***:** Proceso mediante el cual se realizan actividades para verificar la óptima función del sistema.
3. *Entorno móvil***:** Hace referencia a un ambiente de desarrollo y/o ejecución programas o servicios en dispositivos móviles.
   * 1. ***De negocio***
4. *Aplicación móvil. -* *Producto software desarrollado para dispositivos móviles con Sistema Operativo Android.*
5. *Categoría. -* Agrupación general de las dependencias pertenecientes a la universidad.
6. *Subcategoría. -* Subagrupación de las categorías.
7. *Elemento. -* Clasificación de las subcategorías (dependencia).
8. *Dependencia.* - Departamento o unidad perteneciente a la universidad.

***1.3.3 De tecnología***

1. *Sistema de gestión de Base de Datos***:** Son un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan.
2. *Base de Datos***:** Es un conjunto de datos que pertenecen al mismo contexto almacenados sistemáticamente para su posterior uso.
3. *SQL Server 2014***:** Sistema de gestión de base de datos relacional.
4. *Android Studio 2.0:* IDE para el desarrollo Java para dispositivos móviles.

**Referencias**

* Historias de usuario.
* SCRUM
* Diagrama de casos de uso.
* Diagrama de clases.
* Modelos de bases de datos.

**Suposiciones y dependencias**

Para el uso de la aplicación móvil será necesario un dispositivo móvil con sistema operativo Android.

**Usabilidad**

**El cumplimiento del Sistema Operativo**

La interfaz para el usuario móvil será compatible con cualquier dispositivo que cuente con un sistema operativo Android 2.3 en adelante.

**Diseño para la facilidad de uso**

La interfaz de la aplicación móvil “ESPE GUIADA” estará diseñada con estilos de letras legibles y colores llamativos para el usuario. En caso de error se mostrará una ventana emergente que indique el error.

**Consistente**

**Consistencia Interna**

La consistencia interna de la aplicación depende de la consistencia de la base de datos.

**Completo**

El sistema completo deberá cumplir con requerimientos de escalabilidad, disponibilidad, desempeño y portabilidad. Además, se realizará mantenimiento y actualización de datos después del despliegue de la aplicación. Se tomará en cuenta el manual de usuario para evitar errores de los mismos.

**Correcto**

El aplicativo debe inicializarse y ejecutarse sin errores siempre y cuando el usuario tenga acceso a internet lo cual permitirá el acceso a los datos.

**Verificable**

Despliegue de datos en la aplicación: es un requisito verificable puesto que se comprueba la conexión a la base de datos, de lo contrario no se desplegará la información.

**Modificable**

La aplicación móvil, al ser un asistente que muestra información y un mapa de ubicación, no será modificable para el usuario.

**Fiabilidad**

**El tiempo medio entre fallos**

Los requisitos de tiempo medio entre fallos se definirán en la siguiente iteración.

**Acceso Fiabilidad**

El sistema deberá proporcionar el 100% de fiabilidad de acceso con cualquier tipo de conexión a internet.

**Rendimiento**

**Tiempo de respuesta**

La pantalla de inicio de la aplicación se cargará en aproximadamente 5s, así mismo el despliegue de la información. Una causa de retardo en el inicio de la aplicación dependeré de la velocidad de conexión internet.

**Capacidad**

1. Número de usuarios simultáneos que soporte el sistema.

Entre 1 y 50.

1. Cantidad y tipo de información que el sistema esté en la capacidad de sobrellevar.

Tipo de información: datos numéricos, texto, visuales.

**Compatibilidad**

El sistema será compatible con todo tipo de dispositivo móvil tanto gama baja, media y alta con sistema operativo Android 2.3 en adelante.

**Diseño de restricciones**

En esta sección se enumera las restricciones de diseño en la aplicación que se desea desarrollar.

**Requisitos de la plataforma**

El sistema será compatible con todo tipo de dispositivo móvil tanto gama baja, media y alta con sistema operativo Android 2.3 en adelante, conexión a internet y antena GPS.

**Interfaces**

Esta sección define las interfaces que deben ser apoyadas por la aplicación. Debe contener una adecuada especificidad, de modo que el software puede ser desarrollado y verificado en contra de los requisitos de interfaz.

**Interfaces de usuario**

El sistema debe ser amigable y predictivo con el usuario ya que lo podrá utilizar cualquier persona que sepa operar un dispositivo móvil.

**Interfaces de software**

Se muestra la información de las diferentes dependencias de la Universidad de las Fuerzas Armadas ESPE

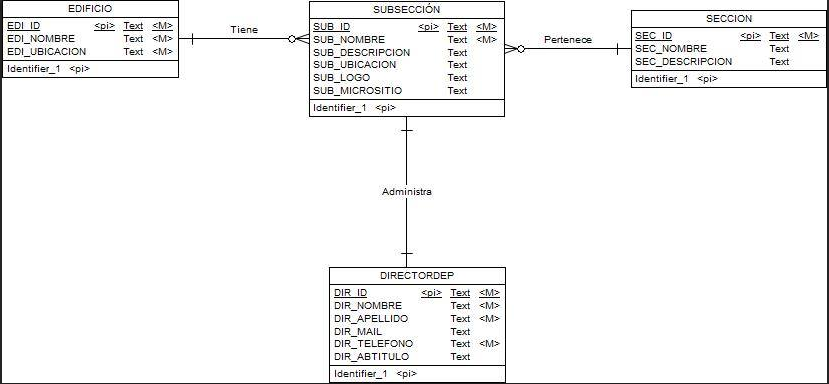
**Requisitos necesarios de diseño**

La aplicación mostrara tres ventanas de forma sucesiva:

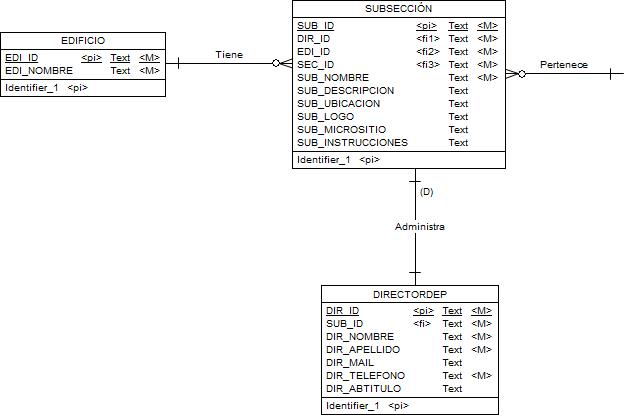
* Página inicial con el logotipo de la universidad, y una lista desplegable con las categorías y dependencias pertenecientes a cada una.
* Ventana con la información general de dependencia seleccionada, información de contacto, enlace al micro sitio, acceso al mapa de ubicación.
* Venta de mapa con un marcador en la ubicación seleccionada.

**2.3 Análisis y Diseño**

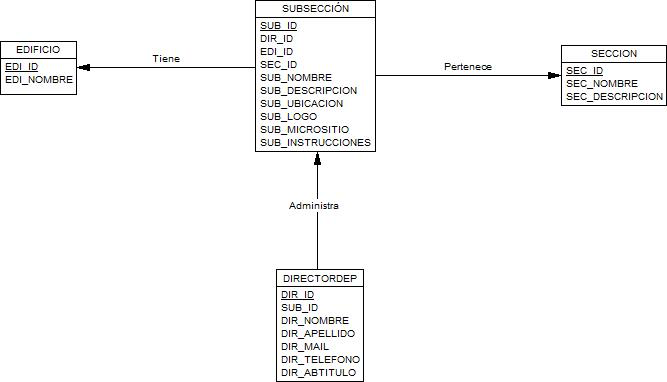
**2.3.1 Modelo Conceptual**



**2.3.2 Modelo Lógico**



**2.3.3 Modelo Físico**



**2.3.4 Script de la Base de Datos**

CREATE TABLE [dbo].[DIRECTORDEP](

[id] [nvarchar](255) NOT NULL,

[SUB\_ID] [nvarchar](max) NULL,

[DIR\_NOMBRE] [nvarchar](max) NOT NULL,

[DIR\_APELLIDO] [nvarchar](max) NOT NULL,

[DIR\_MAIL] [nvarchar](max) NOT NULL,

[DIR\_TELEFONO] [nvarchar](max) NULL,

[DIR\_ABTITULO] [nvarchar](50) NOT NULL

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[EDIFICIO](

[id] [nvarchar](255) NOT NULL CONSTRAINT [DF\_EDIFICIO\_id] DEFAULT (CONVERT([nvarchar](255),newid(),(0))),

[EDI\_NOMBRE] [nvarchar](max) NULL,

[EDI\_UBICACION] [nvarchar](max) NULL

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[SECCION](

[id] [nvarchar](255) NOT NULL CONSTRAINT [DF\_SECCION\_id] DEFAULT (CONVERT([nvarchar](255),newid(),(0))),

[SEC\_NOMBRE] [nvarchar](max) NULL,

[SEC\_DESCRIPCION] [nvarchar](max) NULL

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[SUBSECCION](

[id] [nvarchar](255) NOT NULL CONSTRAINT [DF\_SUBSECCION\_id] DEFAULT (CONVERT([nvarchar](255),newid(),(0))),

[SEC\_ID] [nvarchar](max) NULL,

[EDI\_ID] [nvarchar](max) NULL,

[SUB\_NOMBRE] [nvarchar](max) NULL,

[SUB\_DESCRIPCION] [nvarchar](max) NULL,

[SUB\_INSTRUCCIONES] [nvarchar](max) NULL,

[SUB\_LOGO] [nvarchar](max) NULL,

[SUB\_MICROSITIO] [nvarchar](max) NULL,

[SUB\_UBICACION] [nvarchar](max) NULL

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

**2.4 Implementación**

**2.4.1 Documento de Arquitectura (Diagrama de Despliegue)**

**2.4.2 Código Fuente Desarrollado**

**package** com.example.espeguiada;  
  
**import** android.content.Intent;  
**import** android.graphics.Bitmap;  
**import** android.graphics.drawable.Drawable;  
**import** android.os.Bundle;  
**import** android.app.Activity;  
**import** android.view.Gravity;  
**import** android.view.View;  
**import** android.widget.Button;  
**import** android.widget.ImageView;  
**import** android.widget.TextView;  
  
**import** com.example.espeguiada.R;  
  
**public class** ActivityInfo **extends** Activity {  
  
 TextView **txtInfo**;  
 TextView **txtTitle**;  
 TextView **txtNombreDir**;  
 TextView **txtMailDir**;  
 TextView **txtTelfDir**;  
 TextView **txtMicrositio**;  
 String[] **info**;  
 ImageView **imgMap**;  
 **int mapResource**;  
 Button **btnMaps**;  
 View.OnClickListener **btnMapsListener**=**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 NextMapsActivity();  
 }  
 };  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***info\_subseccion***);  
  
 **txtTitle**=(TextView) findViewById(R.id.***txtTitleInfo***);  
 **txtInfo**=(TextView)findViewById(R.id.***txtInfoSubseccion***);  
 **txtNombreDir**=(TextView) findViewById(R.id.***txtNombreDirector***);  
 **txtMailDir**=(TextView) findViewById(R.id.***txtMailDirector***);  
 **txtTelfDir**=(TextView) findViewById(R.id.***txtTelfDirector***);  
 **txtMicrositio**=(TextView) findViewById(R.id.***txtMicrositio***);  
  
 **imgMap**=(ImageView)findViewById(R.id.***imageViewMap***);  
  
 Intent intent = getIntent();  
 **info**=intent.getStringArrayExtra(MainActivity.***ACT\_INFO***);  
 *//mapResource=intent.getIntExtra("Map",1);* **txtTitle**.setText(**info**[0]);  
 **txtTitle**.setGravity(Gravity.***CENTER\_HORIZONTAL***);  
  
 **txtInfo**.setText(**info**[1]);  
 **txtInfo**.setPadding(40,5,40,5);  
 **txtInfo**.setGravity(Gravity.***CENTER\_HORIZONTAL***);  
 **txtInfo**.setTextSize(16);  
  
 **txtNombreDir**.setText(**info**[3]);  
 **txtMailDir**.setText(**info**[4]);  
 **txtTelfDir**.setText(**info**[5]);  
 **txtMicrositio**.setText(**info**[6]);  
  
 *//mapResource=0x7f020002;  
 //imgMap.setImageResource(mapResource);  
 //String name="logo\_bio";* String name=**info**[2];  
 String resource=**"drawable"**;  
 **int** res\_imagen = getResources().getIdentifier(name, resource,getPackageName());  
 **imgMap**.setImageResource(res\_imagen);  
 *//getActionBar().setDisplayHomeAsUpEnabled(true);* **btnMaps**=(Button)findViewById(R.id.***btnMaps***);  
 **btnMaps**.setOnClickListener(**btnMapsListener**);  
  
  
 }  
 **final static** String ***ACT\_INFO***=**"com.example.espeguiada.MapsActivity"**;  
  
 **public void** NextMapsActivity()  
 {  
 Intent maps=**new** Intent(**this**,MapsActivity.**class**);  
 maps.putExtra(***ACT\_INFO***,**info**);  
 startActivity(maps);  
 }  
  
}

**ConexionSQL**

**package** com.example.espeguiada;  
  
**import** android.os.StrictMode;  
**import** android.util.Log;  
  
**import** java.sql.Connection;  
**import** java.sql.DriverManager;  
**import** java.sql.SQLException;  
**public class** ConexionSQL {  
 **public static** Connection ConnectionHelper() {  
 StrictMode.ThreadPolicy policy = **new** StrictMode.ThreadPolicy.Builder().permitAll().build();  
 StrictMode.*setThreadPolicy*(policy);  
 Connection connection = **null**;  
 String ConnectionURL = **null**;  
  
 **try** {  
 Class.*forName*(**"net.sourceforge.jtds.jdbc.Driver"**);  
  
 ConnectionURL = **"jdbc:jtds:sqlserver://10.9.9.175:1433;database=LugaresEspe;user=usguia;password=Admin112358.;loginTimeout=1;"**;  
 connection = DriverManager.*getConnection*(ConnectionURL);  
  
 } **catch** (SQLException se) {  
 *//Log.e("ERROR", se.getMessage());* } **catch** (ClassNotFoundException e) {  
 *//Log.e("ERROR", e.getMessage());* } **catch** (Exception e) {  
 *//Log.e("ERROR", e.getMessage());* }  
  
 **if** (connection == **null**) {  
 **try** {  
  
 ConnectionURL = **"jdbc:jtds:sqlserver://186.69.217.61:1433;database=LugaresEspe;user=sa;password=Joelram5635726.;loginTimeout=5;"**;  
 connection = DriverManager.*getConnection*(ConnectionURL);  
  
 } **catch** (SQLException se) {  
 Log.*e*(**"ERROR"**, **"No se pudo conectar a la base de datos. Revise su conexión a internet."**);  
  
 }**catch** (Exception e) {  
 Log.*e*(**"ERROR"**, **"No se pudo conectar."**);  
 }  
  
 }  
  
 **return** connection;  
 }  
}

**DIRECTORDEP**

**package** com.example.espeguiada;  
**public class** DIRECTORDEP {  
 **private** String **id**;  
 **private** String **SUB\_ID**;  
 **private** String **DIR\_NOMBRE**;  
 **private** String **DIR\_APELLIDO**;  
 **private** String **DIR\_MAIL**;  
 **private** String **DIR\_TELEFONO**;  
 **private** String **DIR\_ABTITULO**;  
  
 **public** DIRECTORDEP(){}  
  
 **public** DIRECTORDEP(String id, String SUB\_ID, String DIR\_NOMBRE, String DIR\_APELLIDO, String DIR\_MAIL, String DIR\_TELEFONO, String DIR\_ABTITULO){  
 **this**.setId(id);  
 **this**.setSUB\_ID(SUB\_ID);  
 **this**.setDIR\_NOMBRE(DIR\_NOMBRE);  
 **this**.setDIR\_APELLIDO(DIR\_APELLIDO);  
 **this**.setDIR\_MAIL(DIR\_MAIL);  
 **this**.setDIR\_TELEFONO(DIR\_TELEFONO);  
 **this**.setDIR\_ABTITULO(DIR\_ABTITULO);  
 }  
  
 **public** String getId() {  
 **return id**;  
 }  
  
 **public void** setId(String id) {  
 **this**.**id** = id;  
 }  
  
 **public** String getSUB\_ID() {  
 **return SUB\_ID**;  
 }  
  
 **public void** setSUB\_ID(String SUB\_ID) {  
 **this**.**SUB\_ID** = SUB\_ID;  
 }  
  
 **public** String getDIR\_NOMBRE() {  
 **return DIR\_NOMBRE**;  
 }  
  
 **public void** setDIR\_NOMBRE(String DIR\_NOMBRE) {  
 **this**.**DIR\_NOMBRE** = DIR\_NOMBRE;  
 }  
  
 **public** String getDIR\_APELLIDO() {  
 **return DIR\_APELLIDO**;  
 }  
  
 **public void** setDIR\_APELLIDO(String DIR\_APELLIDO) {  
 **this**.**DIR\_APELLIDO** = DIR\_APELLIDO;  
 }  
  
 **public** String getDIR\_MAIL() {  
 **return DIR\_MAIL**;  
 }  
  
 **public void** setDIR\_MAIL(String DIR\_MAIL) {  
 **this**.**DIR\_MAIL** = DIR\_MAIL;  
 }  
  
 **public** String getDIR\_TELEFONO() {  
 **return DIR\_TELEFONO**;  
 }  
  
 **public void** setDIR\_TELEFONO(String DIR\_TELEFONO) {  
 **this**.**DIR\_TELEFONO** = DIR\_TELEFONO;  
 }  
  
 **public** String getDIR\_ABTITULO() {  
 **return DIR\_ABTITULO**;  
 }  
  
 **public void** setDIR\_ABTITULO(String DIR\_ABTITULO) {  
 **this**.**DIR\_ABTITULO** = DIR\_ABTITULO;  
 }  
}

**EDIFICIO**

**package** com.example.espeguiada;  
  
*/\*\*  
 \* Created by adrian on 28/03/16.  
 \*/***public class** EDIFICIO {  
  
 @com.google.gson.annotations.SerializedName(**"id"**)  
 **private** String **id**;  
 @com.google.gson.annotations.SerializedName(**"EDI\_NOMBRE"**)  
 **private** String **EDI\_NOMBRE**;  
 @com.google.gson.annotations.SerializedName(**"EDI\_UBICACION"**)  
 **private** String **EDI\_UBICACION**;  
 @com.google.gson.annotations.SerializedName(**"deleted"**)  
 **private boolean deleted**;  
  
  
 **public** EDIFICIO(){}  
  
 **public** String getid() {  
 **return id**;  
 }  
  
 **public void** setid(String id) {  
 **this**.**id** = id;  
 }  
  
 **public** String getEDI\_NOMBRE() {  
 **return EDI\_NOMBRE**;  
 }  
  
 **public void** setEDI\_NOMBRE(String EDI\_NOMBRE) {  
 **this**.**EDI\_NOMBRE** = EDI\_NOMBRE;  
 }  
  
 **public** String getEDI\_UBICACION() {  
 **return EDI\_UBICACION**;  
 }  
  
 **public void** setEDI\_UBICACION(String EDI\_UBICACION) {  
 **this**.**EDI\_UBICACION** = EDI\_UBICACION;  
 }  
  
 **public boolean** isDeleted() {  
 **return deleted**;  
 }  
  
 **public void** setDeleted(**boolean** deleted) {  
 **this**.**deleted** = deleted;  
 }  
}

**MainActivity**

**package** com.example.espeguiada;  
  
**import** java.net.MalformedURLException;  
**import** java.sql.Connection;  
**import** java.sql.DriverManager;  
**import** java.sql.ResultSet;  
**import** java.sql.SQLException;  
**import** java.sql.Statement;  
**import** java.util.ArrayList;  
**import** java.util.HashMap;  
**import** java.util.List;  
**import** java.util.Map;  
**import** java.util.concurrent.ExecutionException;  
  
**import** android.app.Activity;  
**import** android.app.AlertDialog;  
**import** android.content.Intent;  
**import** android.database.sqlite.SQLiteDatabase;  
**import** android.graphics.Bitmap;  
**import** android.os.AsyncTask;  
**import** android.os.Build;  
**import** android.os.Bundle;  
**import** android.os.StrictMode;  
**import** android.util.Log;  
**import** android.view.Menu;  
**import** android.view.MenuItem;  
**import** android.view.View;  
**import** android.widget.EditText;  
**import** android.widget.ExpandableListAdapter;  
**import** android.widget.ExpandableListView;  
**import** android.widget.ListView;  
**import** android.widget.ProgressBar;  
  
**import** com.google.common.util.concurrent.FutureCallback;  
**import** com.google.common.util.concurrent.Futures;  
**import** com.google.common.util.concurrent.ListenableFuture;  
**import** com.google.common.util.concurrent.SettableFuture;  
**import** com.microsoft.windowsazure.mobileservices.MobileServiceClient;  
**import** com.microsoft.windowsazure.mobileservices.MobileServiceList;  
**import** com.microsoft.windowsazure.mobileservices.http.NextServiceFilterCallback;  
**import** com.microsoft.windowsazure.mobileservices.http.ServiceFilter;  
**import** com.microsoft.windowsazure.mobileservices.http.ServiceFilterRequest;  
**import** com.microsoft.windowsazure.mobileservices.http.ServiceFilterResponse;  
**import** com.microsoft.windowsazure.mobileservices.table.MobileServiceTable;  
**import** com.microsoft.windowsazure.mobileservices.table.query.Query;  
**import** com.microsoft.windowsazure.mobileservices.table.query.QueryOperations;  
**import** com.microsoft.windowsazure.mobileservices.table.sync.MobileServiceSyncContext;  
**import** com.microsoft.windowsazure.mobileservices.table.sync.MobileServiceSyncTable;  
**import** com.microsoft.windowsazure.mobileservices.table.sync.localstore.ColumnDataType;  
**import** com.microsoft.windowsazure.mobileservices.table.sync.localstore.MobileServiceLocalStoreException;  
**import** com.microsoft.windowsazure.mobileservices.table.sync.localstore.SQLiteLocalStore;  
**import** com.microsoft.windowsazure.mobileservices.table.sync.synchandler.SimpleSyncHandler;  
**import** android.content.Context;  
  
  
**import static** com.microsoft.windowsazure.mobileservices.table.query.QueryOperations.\*;  
  
  
**public class** MainActivity **extends** Activity {  
  
  
 HashMap<String,List<String>> **subSections**;  
 List<String> **sections**;  
 ExpandableListView **exp\_sections**;  
 ExpandableListAdapter **listAdapter**;  
 List<String>**rectorado**;  
 List<String>**viceAdm**;  
 List<String>**viceDoc**;  
 List<String>**viceInv**;  
 List<String>**viceAcad**;  
 List<String>**departamentos**;  
 List<String>**otros**;  
  
 */\*Variable para la conexion sql\*/* Connection **conexion**;  
 Connection **conexion1**;  
  
 */\*\*  
 \* Adapter to sync the items list with the view  
 \*/* **private** SECCION\_adapter **seccionAdapter**;  
  
  
 */\*\*  
 \* Progress spinner to use for table operations  
 \*/* **private** ProgressBar **mProgressBar**;  
  
  
  
  
 */\*\*  
 \* Initializes the activity  
 \*/* @Override  
 **public void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_to\_do***);  
  
  
 *//mProgressBar = (ProgressBar) findViewById(R.id.loadingProgressBar);  
  
 // Initialize the progress bar  
 //mProgressBar.setVisibility(ProgressBar.GONE);* **rectorado** = **new** ArrayList<String>();  
 **viceAcad** = **new** ArrayList<String>();  
 **viceAdm** = **new** ArrayList<String>();  
 **viceDoc** = **new** ArrayList<String>();  
 **viceInv** = **new** ArrayList<String>();  
 **departamentos** = **new** ArrayList<String>();  
 **otros** = **new** ArrayList<String>();  
  
 **try** {  
  
 **conexion** = ConexionSQL.*ConnectionHelper*();  
  
 **exp\_sections**=(ExpandableListView)findViewById(R.id.***expandableListView***);  
 SeparateLists();  
 prepareListData();  
 **listAdapter**=**new** SECCION\_adapter(**sections**, **subSections**, **this**);  
 **exp\_sections**.setAdapter(**listAdapter**);  
 *// coneccion.close();* } **catch** (Exception e) {  
 createAndShowDialog(**"No se pudo conectar al servidor. Revise su conexión a Internet."**, **"Error"**);  
 }  
  
 *//exp\_sections=(ExpandableListView)findViewById(R.id.expandableListView);  
 //SeparateLists();  
 //prepareListData();  
 //listAdapter=new SECCION\_adapter(sections, subSections, this);  
 // exp\_sections.setAdapter(listAdapter);* **exp\_sections**.setOnChildClickListener(**new** ExpandableListView.OnChildClickListener() {  
 @Override  
 **public boolean** onChildClick(ExpandableListView parent, View v, **int** groupPosition, **int** childPosition, **long** id) {  
 SUBSECCION subseccion;  
  
 String subsecName = **subSections**.get(**sections**.get(groupPosition)).get(childPosition);  
  
 subseccion=getSubseccionItem(subsecName);  
  
  
  
 nextActivity(v,subseccion);  
 **return false**;  
 }  
 });  
 }  
  
 */\*\*  
 \* Initializes the activity menu  
 \*/* @Override  
 **public boolean** onCreateOptionsMenu(Menu menu) {  
 getMenuInflater().inflate(R.menu.***activity\_main***, menu);  
 **return true**;  
 }  
  
  
 **private** SUBSECCION getSubseccionItem(String name){  
 ResultSet itemSubseccion;  
 SUBSECCION subseccion;  
 String sql=**"use LugaresEspe; select \* from subseccion where SUB\_NOMBRE='"**+name+**"'"**;  
 **try**{  
 Statement statement = **conexion**.createStatement();  
 itemSubseccion = statement.executeQuery(sql);  
 itemSubseccion.next();  
 subseccion = **new** SUBSECCION(itemSubseccion.getString(**"id"**), itemSubseccion.getString(**"SEC\_ID"**), itemSubseccion.getString(**"EDI\_ID"**), itemSubseccion.getString(**"SUB\_NOMBRE"**), itemSubseccion.getString(**"SUB\_DESCRIPCION"**), itemSubseccion.getString(**"SUB\_UBICACION"**), itemSubseccion.getString(**"SUB\_LOGO"**), itemSubseccion.getString(**"SUB\_MICROSITIO"**), Boolean.*parseBoolean*(itemSubseccion.getString(**"deleted"**)));  
 **return** subseccion;  
  
  
  
 }**catch**(SQLException se){  
 System.***out***.println(**"Error: "** + **"No se pudo conectar al servidor. Revise su conexión a Internet."**);  
 }  
  
  
  
 **return null**;  
  
 }  
  
 **private** String getCoord(String edi\_id){  
 ResultSet Coord;  
 String coordinates;  
 String sql1=**"use LugaresEspe; select EDI\_UBICACION from EDIFICIO where id='"**+edi\_id+**"'"**;  
 **try**{  
 Statement statement = **conexion**.createStatement();  
 Coord = statement.executeQuery(sql1);  
 Coord.next();  
 coordinates=Coord.getString(**"EDI\_UBICACION"**);  
 **return** coordinates;  
  
  
  
 }**catch**(SQLException se){  
 System.***out***.println(**"Error: "** + **"No se pudo conectar al servidor. Revise su conexión a Internet."**);  
 }  
  
  
  
 **return null**;  
  
 }  
  
  
  
 **private** DIRECTORDEP getDirectorItem(String Subseccionid){  
 ResultSet itemDirector;  
 DIRECTORDEP director;  
 String sub;  
  
 String sql=**"use LugaresEspe; select \* from DIRECTORDEP where SUB\_ID='"**+Subseccionid+**"'"**;  
 **try**{  
 Statement statement = **conexion**.createStatement();  
 itemDirector = statement.executeQuery(sql);  
 itemDirector.next();  
  
 director = **new** DIRECTORDEP();  
 director.setId(itemDirector.getString(**"id"**));  
 director.setSUB\_ID(itemDirector.getString(**"SUB\_ID"**));  
 director.setDIR\_NOMBRE(itemDirector.getString(**"DIR\_NOMBRE"**));  
 director.setDIR\_APELLIDO(itemDirector.getString(**"DIR\_APELLIDO"**));  
 director.setDIR\_MAIL(itemDirector.getString(**"DIR\_MAIL"**));  
 director.setDIR\_TELEFONO(itemDirector.getString(**"DIR\_TELEFONO"**));  
 director.setDIR\_ABTITULO(itemDirector.getString(**"DIR\_ABTITULO"**));  
  
 **return** director;  
  
 }**catch**(SQLException se){  
 System.***out***.println(**"Error: "** + **"No se pudo conectar al servidor. Revise su conexión a Internet."**);  
 }  
  
 **return null**;  
  
 }  
  
  
 **private void** SeparateLists() {  
  
 ResultSet itemsSubseccion;  
 String sql = **"use LugaresEspe; select \* from subseccion"**;  
  
 **try** {  
  
 Statement statement = **conexion**.createStatement();  
 itemsSubseccion = statement.executeQuery(sql);  
 *// Ajuste nuestro SimpleAdapter* **int** id;  
 id = 0;  
  
 **while** (itemsSubseccion.next()) {  
 *// navegar por nuestro ResultSet en cada registro, siempre y cuando exista un prox.* id = Integer.*parseInt*(itemsSubseccion.getString(**"SEC\_ID"**));  
  
 **switch** (id) {  
 **case** 1:  
 **rectorado**.add(itemsSubseccion.getString(**"SUB\_NOMBRE"**));  
 **break**;  
 **case** 2:  
 **viceAdm**.add(itemsSubseccion.getString(**"SUB\_NOMBRE"**));  
 **break**;  
 **case** 3:  
 **viceDoc**.add(itemsSubseccion.getString(**"SUB\_NOMBRE"**));  
 **break**;  
 **case** 4:  
 **departamentos**.add(itemsSubseccion.getString(**"SUB\_NOMBRE"**));  
 **break**;  
 **case** 5:  
 **viceInv**.add(itemsSubseccion.getString(**"SUB\_NOMBRE"**));  
 **break**;  
 **case** 6:  
 **viceAcad**.add(itemsSubseccion.getString(**"SUB\_NOMBRE"**));  
 **break**;  
 **case** 7:  
 **otros**.add(itemsSubseccion.getString(**"SUB\_NOMBRE"**));  
 **break**;  
 }  
 }  
  
 } **catch** (SQLException se) {  
 System.***out***.println(**"Error: "** + se.toString());  
 }  
 }  
  
 **private void** prepareListData() {  
 **sections** = **new** ArrayList<String>();  
 **subSections** = **new** HashMap<String, List<String>>();  
 ResultSet itemsSeccion;  
 String sql = **"use LugaresEspe; select \* from seccion"**;  
  
 **try** {  
 Statement statement = **conexion**.createStatement();  
 itemsSeccion = statement.executeQuery(sql);  
 **int** id=0;  
  
 **while** (itemsSeccion.next())  
 {  
 **sections**.add(itemsSeccion.getString(**"SEC\_NOMBRE"**));  
 }  
  
 } **catch** (SQLException e) {  
 e.printStackTrace();  
 }  
  
 **subSections**.put(**sections**.get(0),**rectorado**);  
 **subSections**.put(**sections**.get(1),**viceAdm**);  
 **subSections**.put(**sections**.get(2),**viceDoc**);  
 **subSections**.put(**sections**.get(3),**departamentos**);  
 **subSections**.put(**sections**.get(4),**viceInv**);  
 **subSections**.put(**sections**.get(5),**viceAcad**);  
 **subSections**.put(**sections**.get(6),**otros**);  
  
 }  
  
  
  
  
  
 **private void** createAndShowDialogFromTask(**final** Exception exception, String title) {  
 runOnUiThread(**new** Runnable() {  
 @Override  
 **public void** run() {  
 createAndShowDialog(exception, **"Error"**);  
 }  
 });  
 }  
  
  
 */\*\*  
 \* Creates a dialog and shows it  
 \*  
 \** ***@param exception*** *\* The exception to show in the dialog  
 \** ***@param title*** *\* The dialog title  
 \*/* **private void** createAndShowDialog(Exception exception, String title) {  
 Throwable ex = exception;  
 **if**(exception.getCause() != **null**){  
 ex = exception.getCause();  
 }  
 createAndShowDialog(ex.getMessage(), title);  
 }  
  
 */\*\*  
 \* Creates a dialog and shows it  
 \*  
 \** ***@param message*** *\* The dialog message  
 \** ***@param title*** *\* The dialog title  
 \*/* **private void** createAndShowDialog(**final** String message, **final** String title) {  
 **final** AlertDialog.Builder builder = **new** AlertDialog.Builder(**this**);  
  
 builder.setMessage(message);  
 builder.setTitle(title);  
 builder.create().show();  
 }  
  
 */\*\*  
 \* Run an ASync task on the corresponding executor  
 \** ***@param task*** *\** ***@return*** *\*/* **private** AsyncTask<Void, Void, Void> runAsyncTask(AsyncTask<Void, Void, Void> task) {  
 **if** (Build.VERSION.***SDK\_INT*** >= Build.VERSION\_CODES.***HONEYCOMB***) {  
 **return** task.executeOnExecutor(AsyncTask.***THREAD\_POOL\_EXECUTOR***);  
 } **else** {  
 **return** task.execute();  
 }  
 }  
  
  
  
  
 **final static** String ***ACT\_INFO***=**"com.example.espeguiada.ActivityInfo"**;  
  
 **public void** nextActivity(View view, SUBSECCION subseccion) {  
  
 DIRECTORDEP director = getDirectorItem(subseccion.getId().toString());  
  
 **if**(director==**null**)  
 {  
 director = **new** DIRECTORDEP();  
 }  
  
 String[] info = **new** String[8];  
  
 **if** (subseccion.getSUB\_NOMBRE() == **null**)  
 {  
 subseccion.setSUB\_NOMBRE(**" "**);  
 }  
  
 **if**(subseccion.getSUB\_DESCRIPCION()==**null**)  
 {  
 subseccion.setSUB\_DESCRIPCION(**" "**);  
 }  
  
 **if**(subseccion.getSUB\_LOGO()==**null**)  
 {  
 subseccion.setSUB\_LOGO(**" "**);  
 }  
  
 **if**(director.getDIR\_ABTITULO()==**null**)  
 {  
 director.setDIR\_ABTITULO(**" "**);  
 }  
  
 **if**(director.getDIR\_APELLIDO()==**null**)  
 {  
 director.setDIR\_APELLIDO(**" "**);  
 }  
  
 **if**(director.getDIR\_NOMBRE()==**null**)  
 {  
 director.setDIR\_NOMBRE(**" "**);  
 }  
  
 **if**(director.getDIR\_MAIL()==**null**)  
 {  
 director.setDIR\_MAIL(**" "**);  
 }  
  
 **if**(director.getDIR\_TELEFONO()==**null**)  
 {  
 director.setDIR\_TELEFONO(**" "**);  
 }  
  
 **if**(subseccion.getSUB\_MICROSITIO()==**null**)  
 {  
 subseccion.setSUB\_MICROSITIO(**" "**);  
 }  
  
 info[0]=subseccion.getSUB\_NOMBRE().toString();  
 info[1]=subseccion.getSUB\_DESCRIPCION().toString();  
 info[2]=subseccion.getSUB\_LOGO();  
 info[3] = director.getDIR\_ABTITULO().toString() + **". "** + director.getDIR\_NOMBRE().toString() + **" "** + director.getDIR\_APELLIDO().toString();  
 info[4] = director.getDIR\_MAIL().toString();  
 info[5] = director.getDIR\_TELEFONO().toString();  
 info[6]=subseccion.getSUB\_MICROSITIO().toString();  
 info[7]= **this**.getCoord(subseccion.getEDI\_ID());  
  
 Intent act=**new** Intent(**this**,ActivityInfo.**class**);  
 act.putExtra(***ACT\_INFO***, info);  
 *//act.putExtra("Map",subseccion.getMAP\_RESOURCE());* startActivity(act);  
 }  
  
}

**MapsActivity**

**package** com.example.espeguiada;  
  
**import** android.content.Intent;  
**import** android.content.pm.PackageManager;  
**import** android.support.v4.app.ActivityCompat;  
**import** android.support.v4.app.FragmentActivity;  
**import** android.os.Bundle;  
  
**import** com.google.android.gms.maps.CameraUpdateFactory;  
**import** com.google.android.gms.maps.GoogleMap;  
**import** com.google.android.gms.maps.OnMapReadyCallback;  
**import** com.google.android.gms.maps.SupportMapFragment;  
**import** com.google.android.gms.maps.model.LatLng;  
**import** com.google.android.gms.maps.model.MarkerOptions;  
  
**public class** MapsActivity **extends** FragmentActivity **implements** OnMapReadyCallback {  
  
 **private** GoogleMap **mMap**;  
 **private** Ubicacion **ub**;  
 **private** String[] **info**;  
 **private double lat**;  
 **private double len**;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_maps***);  
 *// Obtain the SupportMapFragment and get notified when the map is ready to be used.* SupportMapFragment mapFragment = (SupportMapFragment) getSupportFragmentManager()  
 .findFragmentById(R.id.***map***);  
 mapFragment.getMapAsync(**this**);  
 **ub** = **new** Ubicacion(**this**);  
 Intent intent=getIntent();  
 **info**=intent.getStringArrayExtra(ActivityInfo.***ACT\_INFO***);  
 }  
  
  
 */\*\*  
 \* Manipulates the map once available.  
 \* This callback is triggered when the map is ready to be used.  
 \* This is where we can add markers or lines, add listeners or move the camera. In this case,  
 \* we just add a marker near Sydney, Australia.  
 \* If Google Play services is not installed on the device, the user will be prompted to install  
 \* it inside the SupportMapFragment. This method will only be triggered once the user has  
 \* installed Google Play services and returned to the app.  
 \*/* @Override  
 **public void** onMapReady(GoogleMap googleMap) {  
 **mMap** = googleMap;  
 coordenadas();  
 LatLng decc = **new** LatLng(**lat**, **len**);  
 **mMap**.addMarker(**new** MarkerOptions().position(decc).title(**"Decc"**));  
 **mMap**.moveCamera(CameraUpdateFactory.*newLatLng*(decc));  
 **mMap**.moveCamera(CameraUpdateFactory.*zoomTo*(17));  
 **if** (ActivityCompat.*checkSelfPermission*(**this**, android.Manifest.permission.***ACCESS\_FINE\_LOCATION***) != PackageManager.***PERMISSION\_GRANTED*** && ActivityCompat.*checkSelfPermission*(**this**, android.Manifest.permission.***ACCESS\_COARSE\_LOCATION***) != PackageManager.***PERMISSION\_GRANTED***) {  
 *//* ***TODO: Consider calling*** *// ActivityCompat#requestPermissions  
 // here to request the missing permissions, and then overriding  
 // public void onRequestPermissionsResult(int requestCode, String[] permissions,  
 // int[] grantResults)  
 // to handle the case where the user grants the permission. See the documentation  
 // for ActivityCompat#requestPermissions for more details.* **mMap**.setMyLocationEnabled(**true**);  
  
 }**else** {  
 **mMap**.setMyLocationEnabled(**true**);  
  
 }  
 }  
  
 **private void** coordenadas()  
 {  
 String aux=**info**[7];  
 String[] latlen;  
 latlen=aux.split(**","**);  
 **lat**=Double.*parseDouble*(latlen[0]);  
 **len**=Double.*parseDouble*(latlen[1]);  
 }  
  
}

**SECCION**

**package** com.example.espeguiada;  
**public class** SECCION {  
  
 @com.google.gson.annotations.SerializedName(**"id"**)  
 **private** String **id**;  
  
 @com.google.gson.annotations.SerializedName(**"SEC\_NOMBRE"**)  
 **private** String **SEC\_NOMBRE**;  
  
 @com.google.gson.annotations.SerializedName(**"SEC\_DESCRIPCION"**)  
 **private** String **SEC\_DESCRIPCION**;  
  
 @com.google.gson.annotations.SerializedName(**"deleted"**)  
 **private boolean deleted**;  
  
 **public** SECCION(){}  
  
  
  
 **public** SECCION(String id, String nombre, String descripcion) {  
 **this**.**id**=id;  
 **this**.**SEC\_NOMBRE**=nombre;  
 **this**.**SEC\_DESCRIPCION**=descripcion;  
 }  
  
 **public** String getid() {  
 **return id**;  
 }  
  
 **public void** setid(String id) {  
 **this**.**id** = id;  
 }  
  
 **public** String getSEC\_NOMBRE() {  
 **return SEC\_NOMBRE**;  
 }  
  
 **public void** setSEC\_NOMBRE(String SEC\_NOMBRE) {  
 **this**.**SEC\_NOMBRE** = SEC\_NOMBRE;  
 }  
  
 **public** String getSEC\_DESCRIPCION() {  
 **return SEC\_DESCRIPCION**;  
 }  
  
 **public void** setSEC\_DESCRIPCION(String SEC\_DESCRIPCION) {  
 **this**.**SEC\_DESCRIPCION** = SEC\_DESCRIPCION;  
 }  
  
 **public boolean** isDeleted() {  
 **return deleted**;  
 }  
  
 **public void** setDeleted(**boolean** deleted) {  
 **this**.**deleted** = deleted;  
 }  
}

**SECCION\_adapter**

**package** com.example.espeguiada;  
  
**import** android.app.Activity;  
**import** android.content.Context;  
**import** android.graphics.Typeface;  
**import** android.view.LayoutInflater;  
**import** android.view.View;  
**import** android.view.ViewGroup;  
**import** android.widget.ArrayAdapter;  
**import** android.widget.BaseExpandableListAdapter;  
**import** android.widget.CheckBox;  
**import** android.widget.TextView;  
  
**import** java.util.ArrayList;  
**import** java.util.HashMap;  
**import** java.util.List;  
**public class** SECCION\_adapter **extends** BaseExpandableListAdapter {  
  
  
 **private** List<String> **header\_titles**;  
 **private** HashMap<String,List<String>> **child\_titles**;  
 **private** Context **context**;  
  
 **public** SECCION\_adapter(List<String> header\_titles, HashMap<String,List<String>> child\_titles, Context context){  
 **this**.**header\_titles**=header\_titles;  
 **this**.**child\_titles**=child\_titles;  
 **this**.**context**=context;  
 }  
  
 @Override  
 **public int** getGroupCount() {  
 **return header\_titles**.size();  
 }  
  
 @Override  
 **public int** getChildrenCount(**int** groupPosition) {  
 **return child\_titles**.get(**header\_titles**.get(groupPosition)).size();  
 }  
  
 @Override  
 **public** Object getGroup(**int** groupPosition) {  
 **return header\_titles**.get(groupPosition);  
 }  
  
 @Override  
 **public** Object getChild(**int** groupPosition, **int** childPosition) {  
 **return child\_titles**.get(**header\_titles**.get(groupPosition)).get(childPosition);  
 }  
  
 @Override  
 **public long** getGroupId(**int** groupPosition) {  
 **return** groupPosition;  
 }  
  
 @Override  
 **public long** getChildId(**int** groupPosition, **int** childPosition) {  
 **return** childPosition;  
 }  
  
 @Override  
 **public boolean** hasStableIds() {  
 **return false**;  
 }  
  
 @Override  
 **public** View getGroupView(**int** groupPosition, **boolean** isExpanded, View convertView, ViewGroup parent) {  
 String title = (String)**this**.getGroup(groupPosition);  
 **if**(convertView == **null**){  
 LayoutInflater layoutInflater= (LayoutInflater)**this**.**context**.getSystemService(Context.***LAYOUT\_INFLATER\_SERVICE***);  
 convertView=layoutInflater.inflate(R.layout.***parent\_exp***,**null**);  
 }  
 TextView textView = (TextView)convertView.findViewById(R.id.***heading\_item***);  
 textView.setTypeface(**null**, Typeface.***BOLD***);  
 textView.setText(title);  
  
 **return** convertView;  
 }  
  
 @Override  
 **public** View getChildView(**int** groupPosition, **int** childPosition, **boolean** isLastChild, View convertView, ViewGroup parent) {  
 String title=(String)**this**.getChild(groupPosition,childPosition);  
 **if**(convertView==**null**) {  
 LayoutInflater layoutInflater=(LayoutInflater)**this**.**context**.getSystemService(Context.***LAYOUT\_INFLATER\_SERVICE***);  
 convertView=layoutInflater.inflate(R.layout.***child\_exp***,**null**);  
 }  
  
 TextView textView=(TextView)convertView.findViewById(R.id.***child\_item***);  
 textView.setText(title);  
  
 **return** convertView;  
 }  
  
 @Override  
 **public boolean** isChildSelectable(**int** groupPosition, **int** childPosition) {  
 **return true**;  
 }  
}

**SUBSECCION**

**package** com.example.espeguiada;  
**public class** SUBSECCION {  
  
 @com.google.gson.annotations.SerializedName(**"id"**)  
 **private** String **id**;  
 @com.google.gson.annotations.SerializedName(**"SEC\_ID"**)  
 **private** String **SEC\_ID**;  
 @com.google.gson.annotations.SerializedName(**"EDI\_ID"**)  
 **private** String **EDI\_ID**;  
 @com.google.gson.annotations.SerializedName(**"SUB\_NOMBRE"**)  
 **private** String **SUB\_NOMBRE**;  
 @com.google.gson.annotations.SerializedName(**"SUB\_DESCRIPCION"**)  
 **private** String **SUB\_DESCRIPCION**;  
 @com.google.gson.annotations.SerializedName(**"SUB\_UBICACION"**)  
 **private** String **SUB\_UBICACION**;  
 *//@com.google.gson.annotations.SerializedName("SUB\_LOGO")  
 //private Bitmap SUB\_LOGO;* @com.google.gson.annotations.SerializedName(**"deleted"**)  
 **private boolean deleted**;  
  
 **private** String **SUB\_LOGO**;  
 **private** String **SUB\_MICROSITIO**;  
  
 **public** SUBSECCION(){}  
  
 **public** SUBSECCION(String id, String SEC\_ID, String EDI\_ID, String SUB\_NOMBRE, String SUB\_DESCRIPCION, String SUB\_UBICACION, String SUB\_LOGO, String SUB\_MICROSITIO, **boolean** deleted) {  
 **this**.setId(id);  
 **this**.setSEC\_ID(SEC\_ID);  
 **this**.setEDI\_ID(EDI\_ID);  
 **this**.**SUB\_NOMBRE** = SUB\_NOMBRE;  
 **this**.**SUB\_DESCRIPCION** = SUB\_DESCRIPCION;  
 **this**.**SUB\_UBICACION** = SUB\_UBICACION;  
 **this**.**SUB\_LOGO**=SUB\_LOGO;  
 **this**.setSUB\_MICROSITIO(SUB\_MICROSITIO);  
  
 *//this.SUB\_LOGO = SUB\_LOGO;* **this**.**deleted** = deleted;  
 }  
  
  
  
 **public** String getSUB\_NOMBRE() {  
 **return SUB\_NOMBRE**;  
 }  
  
 **public void** setSUB\_NOMBRE(String SUB\_NOMBRE) {  
 **this**.**SUB\_NOMBRE** = SUB\_NOMBRE;  
 }  
  
 **public** String getSUB\_DESCRIPCION() {  
 **return SUB\_DESCRIPCION**;  
 }  
  
 **public void** setSUB\_DESCRIPCION(String SUB\_DESCRIPCION) {  
 **this**.**SUB\_DESCRIPCION** = SUB\_DESCRIPCION;  
 }  
  
 **public** String getSUB\_UBICACION() {  
 **return SUB\_UBICACION**;  
 }  
  
 **public void** setSUB\_UBICACION(String SUB\_UBICACION) {  
 **this**.**SUB\_UBICACION** = SUB\_UBICACION;  
 }**public boolean** isDeleted() {  
 **return deleted**;  
 }  
  
 **public void** setDeleted(**boolean** deleted) {  
 **this**.**deleted** = deleted;  
 }  
  
 **public** String getId() {  
 **return id**;  
 }  
  
 **public void** setId(String id) {  
 **this**.**id** = id;  
 }  
  
 **public** String getSEC\_ID() {  
 **return SEC\_ID**;  
 }  
  
 **public void** setSEC\_ID(String SEC\_ID) {  
 **this**.**SEC\_ID** = SEC\_ID;  
 }  
  
 **public** String getEDI\_ID() {  
 **return EDI\_ID**;  
 }  
  
 **public void** setEDI\_ID(String EDI\_ID) {  
 **this**.**EDI\_ID** = EDI\_ID;  
 }  
  
 **public** String getSUB\_LOGO() {  
 **return SUB\_LOGO**;  
 }  
  
 **public void** setSUB\_LOGO(String SUB\_LOGO) {  
 **this**.**SUB\_LOGO** = SUB\_LOGO;  
 }  
  
 **public** String getSUB\_MICROSITIO() {  
 **return SUB\_MICROSITIO**;  
 }  
  
 **public void** setSUB\_MICROSITIO(String SUB\_MICROSITIO) {  
 **this**.**SUB\_MICROSITIO** = SUB\_MICROSITIO;  
 }  
}

**Ubicación**

**package** com.example.espeguiada;  
  
**import** android.content.Context;  
**import** android.content.pm.PackageManager;  
**import** android.location.Location;  
**import** android.location.LocationListener;  
**import** android.location.LocationManager;  
**import** android.os.Build;  
**import** android.os.Bundle;  
**import** android.widget.Toast;  
**public class** Ubicacion **implements** LocationListener {  
 **private** Context **ctx**;  
 LocationManager **locationMan**;  
 **private** Location **lc**;  
 **private** String **proveedor**;  
 **private boolean networkon**;  
 **private double latitud**;  
 **private double longitud**;  
  
  
 **public** Ubicacion(Context ctx) {  
 **this**.**ctx** = ctx;  
 **locationMan** = (LocationManager) ctx.getSystemService(Context.***LOCATION\_SERVICE***);  
 **proveedor** = LocationManager.***GPS\_PROVIDER***;  
 **networkon** = **locationMan**.isProviderEnabled(**proveedor**);  
 **if** (Build.VERSION.***SDK\_INT*** >= Build.VERSION\_CODES.***M***) {  
 **if** (ctx.checkSelfPermission(android.Manifest.permission.***ACCESS\_FINE\_LOCATION***) != PackageManager.***PERMISSION\_GRANTED*** && ctx.checkSelfPermission(android.Manifest.permission.***ACCESS\_COARSE\_LOCATION***) != PackageManager.***PERMISSION\_GRANTED***)  
 {  
 **return**;  
 }**else** {  
 **locationMan**.requestLocationUpdates(**proveedor**, 1000, 1, **this**);  
 }  
 } **else** {  
 **locationMan**.requestLocationUpdates(**proveedor**, 1000, 1, **this**);  
 }  
 getLocation();  
 }  
  
 **public void** getLocation()  
 {  
 **if**(**networkon**)  
 {  
 **if** (Build.VERSION.***SDK\_INT*** >= Build.VERSION\_CODES.***M***) {  
 **if** (**ctx**.checkSelfPermission(android.Manifest.permission.***ACCESS\_FINE\_LOCATION***) != PackageManager.***PERMISSION\_GRANTED*** && **ctx**.checkSelfPermission(android.Manifest.permission.***ACCESS\_COARSE\_LOCATION***) != PackageManager.***PERMISSION\_GRANTED***)  
 {  
 **return** ;  
 }  
 }  
 **lc**=**locationMan**.getLastKnownLocation(**proveedor**);  
 **if**(**lc**!=**null**)  
 {  
 **latitud**=**lc**.getLatitude();  
 **longitud**=**lc**.getLongitude();  
 StringBuilder sbuilder= **new** StringBuilder();  
 sbuilder.append(**"Latitud: "**).append(**lc**.getLatitude()).append(**" Longitud:"**).append(**lc**.getLongitude());  
 Toast.*makeText*(**ctx**, sbuilder.toString(), Toast.***LENGTH\_LONG***).show();  
 }  
 }  
 }  
  
 @Override  
 **public void** onLocationChanged(Location location) {  
 getLocation();  
 }  
  
 @Override  
 **public void** onStatusChanged(String provider, **int** status, Bundle extras) {  
 getLocation();  
  
 }  
  
 @Override  
 **public void** onProviderEnabled(String provider) {  
  
 }  
  
 @Override  
 **public void** onProviderDisabled(String provider) {  
  
 }  
  
 **public double** getLatitud() {  
 **return latitud**;  
 }  
  
 **public double** getLongitud() {  
 **return longitud**;  
 }  
}

**CAPÍTULO III**

**3.1 Pruebas**

**3.1.1 Documentación de Plan de Pruebas**

**Revisiones de Plantilla**

| **Versión** | **Fecha de Entrega** | **Revisado por** |
| --- | --- | --- |
| 1.0 | 2016-06-29 | Adrián Mena |

**Control de versiones Plantilla**

| **Versión** | **Encargado** | **Descripción** |
| --- | --- | --- |
| 1.0 | Daniel Marín | Creación de plantilla |

**Revisiones**

| **Versión** | **Fecha de Entrega** | **Revisado por** |
| --- | --- | --- |
| 1.0 | 2016-06-29 | Adrián Mena |

**Control de versiones**

| **Versión** | **Encargado** | **Descripción** |
| --- | --- | --- |
| 1.0 | Adrián Mena  Daniel Marín | Creación de documento |

**Carta de Aprobación**

Por medio de la firma de esta carta de aceptación los lineamientos aquí descritos se aprueban y serán desarrollados.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Adrián Mena**

Jefe de Proyecto

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Daniel Marín**

Desarrollador

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Joel Rivera**

Desarrollador

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Diego Terán**

Desarrollador

**Introducción**

Se pretende describir las diferentes pruebas que se llevarán a cabo sobre el aplicativo móvil (SISTEMA DE ACCESO A INFORMACIÓN, UBICACIÓN Y NAVEGACIÓN MEDIANTE UNA APLICACIÓN MÓVIL PARA PUNTOS DE INTERÉS DENTRO DEL CAMPUS DE LA UNIVERSIDAD DE LA FUERZAS ARMADAS - ESPE), cuyos resultados serán entregados al director del grupo de Investigación Ing. Freddy Tapia a fin de validar su cumplimiento.

**Propósito**

Este documento pretende establecer un estándar de pruebas que se van aplicar al aplicativo móvil el cual está dirigido al equipo de desarrollo y grupo de investigación dirigido por el Ing. Freddy Tapia en el departamento de Ciencias de la Computación de la Universidad de las Fuerzas Armadas-ESPE, con el propósito de dar a conocer la exactitud y el nivel de cumplimiento de los requerimientos planteados al inicio de este proyecto.

**Alcance**

Se validarán y verificarán los procesos para mostrar a información requerida y las rutas de navegación hacia las diferentes dependencias de la Universidad de las Fuerzas Armadas ESPE, buscando solucionar el problema de la ubicación para miembros de la universidad, personal nuevo, y visitantes. Para ello se llevarán a cabo diferentes pruebas para validar lo antes propuesto.

| N° | Nombre | Objetivo | Actor | Responsable |
| --- | --- | --- | --- | --- |
| 01 | Mostrar categorías | Mostrar una lista de las categorías existentes en la universidad. | Usuario. | Adrián Mena |
| 02 | Mostrar dependencias | Desplegar sub listas con las dependencias pertenecientes a cada categoría. | Usuario. | Adrián Mena  Joel Rivera |
| 03 | Mostrar información dependencia | Mostrar logo, descripción, información de contacto, micro-sitio. | Usuario. | Adrián Mena  Joel Rivera |
| 04 | Mostrar mapa | Mostrar un mapa de la universidad con la ubicación de la dependencia elegida. | Usuario. | Diego Terán  Daniel Marín |
| 05 | Generar Ruta | Modificar la información a soluciones de problemas existentes. | Usuario. | Diego Terán  Daniel Marín |

**Definición de términos y acrónimos.**

| Término | Descripción / Significado |
| --- | --- |
| Pruebas | Proceso mediante el cual se realizan actividades para verificar la óptima función del sistema. |
| Usuario | Miembro o visitante de la Universidad de las Fuerzas Armadas ESPE. |
| Entorno móvil | Hace referencia a un ambiente de desarrollo y/o ejecución programas o servicios en dispositivos móviles. |
| Aplicación móvil | Producto software desarrollado para dispositivos móviles con sistema operativo Android (Smartphone, Tablet) |
| Categoría | Agrupación general de las dependencias pertenecientes a la universidad. |
| Dependencia | Departamento o unidad perteneciente a la universidad. |

**Visión General del Documento**

En las siguientes secciones se describirán las herramientas, documentos y plantillas que se utilizarán en las pruebas para el aplicativo móvil.

**Ítems a probar**

**Funciones**: Las funciones que se van a probar del sistema son las siguientes:

* Mostrar categorías
* Mostrar dependencias
* Mostrar información dependencia
* Mostrar mapa
* Generar Ruta

**Módulos**: Los módulos que se van a probar son los siguientes:

**Interfaces**: Las interfaces que se probarán son las siguientes:

* Interfaz inicial de presentación de lista desplegable de categorías y dependencias.
* Interfaz de presentación de información de dependencia.
* Interfaz de presentación de mapa y ruta de llegada al destino.

**Estrategia**

A fin de dar cumplimiento a este plan, se procederá a realizar la navegación a través de las opciones buscando una dependencia específica, con el fin comprobar el correcto funcionamiento del sistema con la base de datos y que todos los datos mostrados sean correctos.

**Condiciones especiales**

Se describen las condiciones bajo las cuales el plan debe ser suspendido, repetido o culminado, en caso de que estas existan, se usa la siguiente plantilla:

| **Estado** | **Acontecimientos** |
| --- | --- |
| Suspendido | * Si el usuario realiza ninguna acción de selección |
| Repetido | <<No existen estado de repetido en la aplicación>> |
| Culminado | * La selección de la dependencia fluye sin problemas. * Se accede a la base de datos correctamente. * La información desplegada es correcta. |

**Documentos representativos**

Los documentos involucrados en la creación del plan de pruebas son: los documentos generados en base a la información proporcionados por el cliente, el documento de Especificación de Requerimientos del proyecto y los casos de prueba generados. Los documentos que se generarán a partir de la aplicación de este plan serán el resultado de pruebas y el reporte de errores.

Los documentos involucrados son:

* Descripción de Casos de Uso
* Especificación de requerimientos del proyecto Base de Conocimiento
* Plantilla para la creación de casos de prueba para el aplicativo móvil
* Plantilla para el reporte de errores ocurridos en el aplicativo móvil

Los documentos a generar son los siguientes:

* EspeGuiada\_Reporte\_de\_errores\_v1.0.docx
* EspeGuiada\_Casos\_de\_prueba\_v1.0.docx

**Recursos**

Para la aplicación de pruebas se necesitará de varios dispositivos móviles de diferentes características de hardware y con sistema operativo Android superior a la versión 2.3.3.

**Calendarización**

Las actividades a realizarse en el proceso de aplicación de pruebas de funcionalidad, están evidenciadas en el archivo “Cronograma\_EspeGuiada\_v1.1.mpp”.

**Responsables**

| **Responsable** | **Tareas o Actividades** |
| --- | --- |
| Adrián Mena  Joel Rivera | Casos de Prueba CP001, CP002, CP003.  Pruebas de funcionamiento de Interfaces y funcionalidad CP001, CP002, CP003. |
| Diego Terán  Daniel Marín | Casos de Prueba CP004, CP005.  Pruebas de funcionamiento de Interfaces y funcionalidad CP004, CP005. |
| Adrián Mena | Plan de Pruebas |

**Herramientas de prueba**

Servidor externo a la Universidad con Sistema Operativo Windows Server 2012 y Base de Datos SQL Server 2014 para almacenamiento y acceso a los datos desde la aplicación.

Dispositivos móviles Android con la aplicación Espe Guiada instalada mediante un archivo apk y acceso a internet.

**Casos de prueba**

Los casos de pruebas creados se ven reflejados en el documento adjunto “EspeGuiada\_Casos de prueba V.1.0.docx”.

**Documentación de los resultados de pruebas**

El resultado de las pruebas realizadas al aplicativo Web se refleja en los archivos de imagen “EspeGuiada\_CP001\_1”, “EspeGuiada\_CP001\_2”, “EspeGuiada\_CP002\_1”, “EspeGuiada\_CP002\_2”, “EspeGuiada\_CP003\_1”, “EspeGuiada\_CP003\_2”.

**Reporte de errores encontrados**

El reporte de errores de las pruebas realizadas al aplicativo Web se refleja en el documento “EspeGuiada\_Reporte\_de\_errores\_v1.0.docx”.

**Prueba de Caja Negra**

El resultado de las pruebas de Caja Negra realizadas al módulo de cobranzas se refleja en el documento adjunto “EspeGuiada\_Prueba\_Caja\_negra\_v1.0.docx”.

**Prueba de Caja Blanca**

El resultado de las pruebas de Caja Blanca realizadas al módulo de cobranzas se refleja en el documento adjunto “EspeGuiada\_Prueba\_Caja\_blanca\_v1.0.docx”.

**3.1.2 Documentación de Casos de Prueba**

| **Caso de Prueba** 001  **Elaborado por:**           Daniel Marín  Adrián Mena  Joel Rivera  Diego Terán | |
| --- | --- |
| **Código de Identificación:** | MC. Mostrar Categoría CP01 |
| **Nombre del Proyecto:** | ”Sistema de acceso a información, ubicación, y navegación mediante una aplicación móvil para puntos de interés dentro del Campus de la Universidad de las Fuerzas Armadas – ESPE” |
| **Descripción (Alcance y Objetivos):** | La aplicación móvil debe mostrar una lista de las categorías existentes en la universidad. |
| **Requisitos asociados** | EG-RF001 |
| **Variables de Entrada (Inputs):** |  |
| **Flujo normal del evento** | 1. La aplicación accede a la base de datos y recupera la lista de categorías. 2. La primera pantalla de la aplicación muestra una lista de las categorías principales dentro de la universidad. |
| **Resultado esperado:** | Se mostrará al usuario un listado de categorías que organizan a los departamentos y dependencias dentro de la universidad. |
| **Flujo alterno** | Si el dispositivo móvil no cuenta con acceso a internet, no se podrá acceder a los datos y se mostrará un mensaje de error solicitando al usuario verificar su conexión a la red de la universidad. |
| **Resultado alternativo esperado:** | -Mensaje de error  -El usuario no podrá visualizar el listado de categorías si no dispone de acceso a la red. |
| **Evaluación de prueba** | |
| **Fecha de ejecución:** |  |
| **Ejecutado por:** |  |
| **Lugar de ejecución:** |  |
| **Resultados obtenidos:** |  |
| **Observaciones:** |  |
| **Gravedad del error** |  |
| **Notas del programador** | |
| **Estado:** |  |
| **Acciones de corrección:** |  |
| **Corregido por:** |  |
| **Caso de Prueba** 002  **Elaborado por:**           Daniel Marín  Adrián Mena  Joel Rivera  Diego Terán | |
| **Código de Identificación:** | MD. Mostrar Dependencias CP02 |
| **Nombre del Proyecto:** | ”Sistema de acceso a información, ubicación, y navegación mediante una aplicación móvil para puntos de interés dentro del Campus de la Universidad de las Fuerzas Armadas – ESPE” |
| **Descripción (Alcance y Objetivos):** | La aplicación móvil debe desplegar sub listas con las dependencias pertenecientes a cada categoría. |
| **Requisitos asociados** | EG-RF002 |
| **Variables de Entrada (Inputs):** | * Seleccionar una categoría mediante la pantalla táctil haciendo clic en el ítem deseado desplegará la sub lista con las dependencias pertenecientes a dicha categoría. |
| **Flujo normal del evento** | 1. La aplicación muestra la lista de categorías. 2. Seleccionar la categoría 3. Visualizar la lista de dependencias pertenecientes a la categoría |
| **Resultado esperado:** | Se mostrará al usuario un listado de dependencias que pertenecen a la categoría seleccionada. |
| **Flujo alterno** | Si el dispositivo móvil no cuenta con acceso a internet, no se podrá acceder a los datos y se mostrará un mensaje de error solicitando al usuario verificar su conexión a la red de la universidad. |
| **Resultado alternativo esperado:** | -Mensaje de error  -El usuario no podrá visualizar las categorías ni dependencias si no se dispone de acceso a la red. |
| **Evaluación de prueba** | |
| **Fecha de ejecución:** |  |
| **Ejecutado por:** |  |
| **Lugar de ejecución:** |  |
| **Resultados obtenidos:** |  |
| **Observaciones:** |  |
| **Gravedad del error** |  |
| **Notas del programador** | |
| **Estado:** |  |
| **Acciones de corrección:** |  |
| **Corregido por:** |  |
| **Caso de Prueba** 003  **Elaborado por:**           Daniel Marín  Adrián Mena  Joel Rivera  Diego Terán | |
| **Código de Identificación:** | MID.  Mostrar Información Dependencia  CP03 |
| **Nombre del Proyecto:** | ”Sistema de acceso a información, ubicación, y navegación mediante una aplicación móvil para puntos de interés dentro del Campus de la Universidad de las Fuerzas Armadas – ESPE” |
| **Descripción (Alcance y Objetivos):** | La aplicación móvil debe mostrar logo, descripción, instrucciones de llegada. |
| **Requisitos asociados** | EG-RF003 |
| **Variables de Entrada (Inputs):** | * Seleccionar una dependencia de la sub lista mostrada mediante la acción de clic en la pantalla táctil del dispositivo móvil. |
| **Flujo normal del evento** | 1. Seleccionar una dependencia de la lista. 2. Visualizar la información: Descripción de la dependencia, Nombre del director de la dependencia, y contacto del director. |
| **Resultado esperado:** | Se mostrará al usuario una descripción breve de la dependencia, el nombre y contacto (e-mail, número telefónico) del director de la dependencia. |
| **Flujo alterno** | Si el dispositivo móvil no cuenta con acceso a internet, no se podrá acceder a los datos y se mostrará un mensaje de error solicitando al usuario verificar su conexión a la red de la universidad. |
| **Resultado alternativo esperado:** | -Mensaje de error  -El usuario no podrá visualizar la información de la dependencia. |
| **Evaluación de prueba** | |
| **Fecha de ejecución:** |  |
| **Ejecutado por:** |  |
| **Lugar de ejecución:** |  |
| **Resultados obtenidos:** |  |
| **Observaciones:** |  |
| **Gravedad del error** |  |
| **Notas del programador** | |
| **Estado:** |  |
| **Acciones de corrección:** |  |
| **Corregido por:** |  |

**3.1.3 Documentación de Informe de Errores**

**Función de la Plantilla**

La función de esta plantilla es estandarizar y formalizar todos los aspectos que conformarán un reporte de errores de las pruebas que se aplicaron al aplicativo móvil: “Sistema de acceso a información, navegación y ubicación mediante una aplicación móvil para puntos de interés dentro del campus de la Universidad de las Fuerzas Armadas-ESPE”

**Objetivos de la Plantilla**

Crear una estructura para la creación de un reporte de errores.

**Alcance de la Plantilla**

Esta plantilla sólo se usará para elaborar los reportes de errores según los  casos de prueba que se aplicaron al proyecto.

**Revisiones de Plantilla**

| **Versión** | **Fecha de Entrega** | **Revisado por** |
| --- | --- | --- |
| 1.0 | 30/06/2016 | Adrián Mena |

**Control de versiones Plantilla**

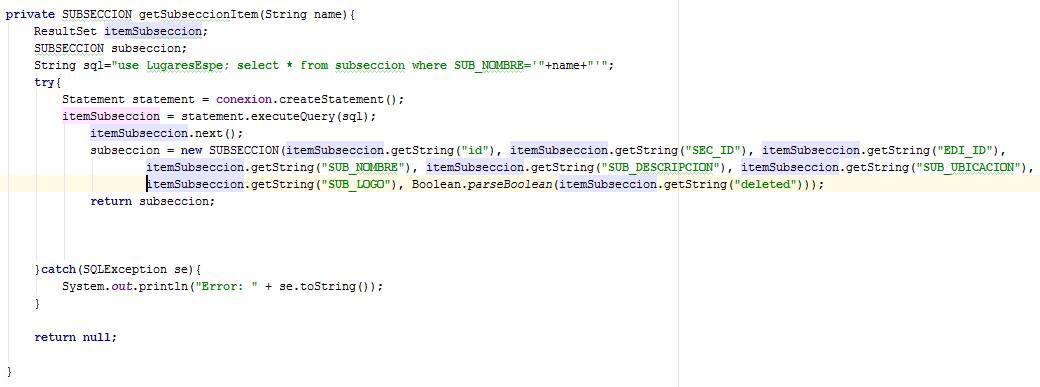
| **Versión** | **Encargado** | **Descripción** |
| --- | --- | --- |
| 1.0 | Daniel Marin | Revisión plantilla |

| **Reporte de Errores e Inconsistencias** | | | | |
| --- | --- | --- | --- | --- |
| **Nombre del Proyecto:** | | Sistema de acceso a información, navegación y ubicación mediante una aplicación móvil para puntos de interés dentro del campus de la Universidad de las Fuerzas Armadas-ESPE | | |
| **Fecha de pruebas:** | | 17-06-2016 | | |
| **Módulos:** | | Mostrar categorías, Mostrar dependencias, Mostrar información de dependencias. | | |
| **Analista:** | | Ing. Jenny Ruiz | | |
| **Responsable:** | | Daniel Marin, Adrián Mena, Joel Rivera, Diego Terán. | | |
| **Fecha de revisión:** | | 17-06-2016 | | |
| **Identificación Caso Prueba** | **Descripción de prueba.** | | **Descripción del error.** | **Acciones de corrección** |
| CP-001 | Se despliega el listado de categorías clasificadas. | | El logotipo de la ESPE está muy claro. | N/A |
| CP-002 | Se despliega correctamente las sublistas correspondientes a las dependencias. | | Cuando no hay internet la aplicación se queda colgada. | Mensaje al desconectarse en cada acción, no solo al inicio de la aplicación. |
| CP-003 | Muestra la información de las dependencias como el nombre del director de carrera. | | Demora en cargar los datos. | N/A |

**3.1.4 Documentación Técnicas Caja Blanca y Caja Negra**

**BUSCAR PROBLEMAS**

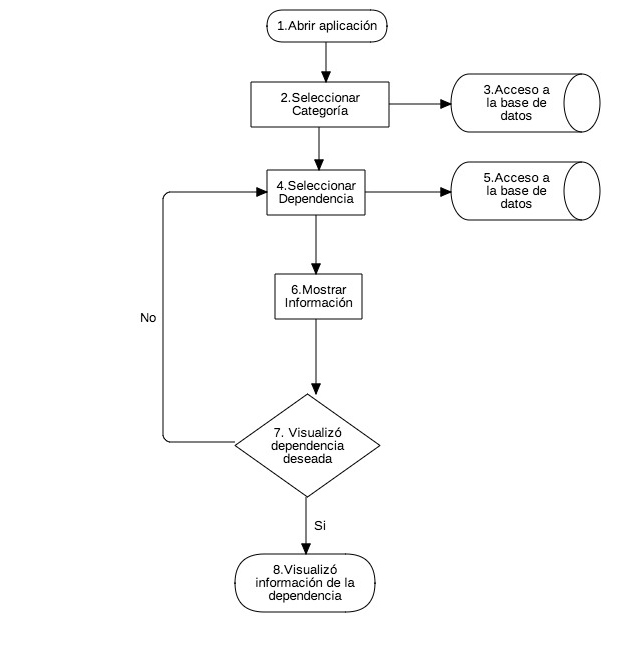
**CÓDIGO FUENTE**

****

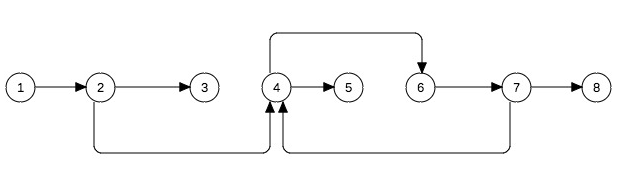


****

**DIAGRAMA DE FLUJO**



**GRAFO**

****

**RUTAS**

**R1: 1,2,3,2,4,5,4,6,7,8**

**R2: 1,2,3,2,4,5,4,6,7,4,5,4,6,7,8**

**COMPLEJIDAD CICLOMATICA**

Se puede calcular de las siguientes formas:

* V(G) = número de nodos predicados(decisiones)+1 = 2
* V(G) = A – N + 2 = 8 – 8 + 2 = 2

DONDE:

**P:** Número de nodos predicado

**A:** Número de aristas

**N:** Número de nodos

**Requisito tomado para las pruebas de caja negra:** Requisito Funcional 3 (EG-RF003)

Ingresar a la aplicación en el dispositivo móvil.



Seleccionar una categoría.



Seleccionar una dependencia y se muestra la información de la misma.



Notas:

* La aplicación móvil no admite el ingreso de información por parte del usuario.
* El usuario interactúa con la aplicación realizando operaciones de navegación y selección de las diferentes opciones que muestra la interfaz al acceder a la base de datos y mostrar la información correspondiente a cada pantalla.

1. Según datos calculados en la página web Daftlogic.com [↑](#footnote-ref-0)