

PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO  
FACULTAD DE INGENIERÍA  
ESCUELA DE INGENIERÍA INFORMÁTICA

# **PROYECTO**

## **Guía Turística**

**JAVIER ESTEBAN PEÑA REYES**  
**ALEN DAVID FIGUEROA MANDUJANO**  
**MARCELO ORLANDO BECERRA ROZAS**

Profesor Guía: Claudio Cubillos

Proyecto Entrega A.

Programación Avanzada, ICI3241-1

**Abril 2018**

# Índice

<b>Resumen.....</b>	<b>ii</b>
<b>Lista de Figuras.....</b>	<b>iii</b>
<b>1 Introducción .....</b>	<b>1</b>
<b>2 Descripción del Problema.....</b>	<b>2</b>
<b>3 Descripción de la Solución.....</b>	<b>3</b>
3.1 Análisis .....	3
3.2 Diseño .....	5
3.3 Base de Datos .....	7
3.3.1 Conexión a la Base de Datos a través de Java. ....	7
3.4 APIS de Google Maps .....	9
3.4.1 Geocoding API de Google .....	9
3.4.2 Static Maps API de Google .....	11
3.5 Reportes por PDF en Java. ....	12
<b>4 Diseño de la Interfaz .....</b>	<b>15</b>
4.1 Ventana Inicial.....	15
4.2 Ventana de Registro.....	16
4.3 Ventana de Categorías.....	17
4.4 Ventana de Lugares.....	17
4.5 Ventana de Lugar.....	18
4.6 Ventana de Comentarios.....	19
4.7 Ventana de Administrador .....	19
4.8 Ventana de Reporte.....	20
4.9 Ventana de Administrar Lugares .....	21
4.10 Ventana de Administrar Usuarios .....	22
<b>5 Planificación .....</b>	<b>23</b>
5.1 Carta Gantt Grupal.....	23
5.2 Carta Gantt Javier Esteban Peña Reyes .....	24
5.3 Carta Gantt Alen David Figueroa Mandujano.....	25
5.4 Carta Gantt Marcelo Orlando Becerra Rozas .....	26
<b>6 Conclusión .....</b>	<b>27</b>
<b>7. Referencias.....</b>	<b>28</b>

## **Resumen**

Este documento describe el proyecto título como “Guía Turística”, donde se da a conocer la descripción del problema y el contexto en el cual se encuentra este. También abarca lo que es el análisis, encerrando las clases e interfaces convenientemente elegidas para ser utilizadas, tales como el diseño del programa, diagramas referenciales, modelos explicativos y extractos de código para mejor comprensión.

Posteriormente se muestra la planificación del proyecto de manera gráfica a través de una Carta Gantt, pudiendo ser apreciados los plazos, tareas a desarrollar tanto a un nivel de grupo como individual de cada integrante del proyecto, y cada detalle de como se ha ido construyendo el proyecto.

Finalmente, se presentará una conclusión del trabajo englobando todo lo que ha sido el proyecto, enseñanzas, dificultades, así como también lo que se espera tener o cambiar en un futuro cercano.

## Lista de Figuras

Figura 3.2.1 Diagrama de Clases .....	5
Figura 3.2.2 Modelado del Sistema con sus Atributos y Métodos. ....	6
Figura 3.3.1 Modelo de la Base de Datos Relacional de la aplicación Guía Turística .....	7
Figura 4.1.1 Ventana Inicial de la aplicación Guía Turística .....	15
Figura 4.2.1 Ventana de Registro de la aplicación Guía Turística.....	16
Figura 4.3.1 Ventana de Categorías en la aplicación de Guía Turística.....	17
Figura 4.4.1 Ventana de Lugares en la aplicación de Guía Turística .....	17
Figura 4.5.1 Ventana de Lugar en la aplicación de Guía Turística.....	18
Figura 4.6.1 Ventana de Comentarios en la aplicación de Guía Turística .....	19
Figura 4.7.1 Ventana de Administrador en la aplicación Guía Turística.....	19
Figura 4.8.1 Ventana Generar Reporte en la aplicación Guía Turística .....	20
Figura 4.8.2 Ventana para guardar el reporte en la aplicación Guía Turística.....	20
Figura 4.9.1 Ventana de Administrar Lugares en la aplicación Guía Turística .....	21
Figura 4.10.1 Ventana de Administración de Usuarios en la aplicación de Guía Turística	22
Figura 5.1.1 Carta Gantt grupal parte 1.....	23
Figura 5.1.2 Carta Gantt grupal parte 2.....	23
Figura 5.2.1 Carta Gantt de Javier Esteban Peña Reyes parte 1 .....	24
Figura 5.2.2 Carta Gantt de Javier Esteban Peña Reyes parte 2 .....	24
Figura 5.3.1 Carta Gantt de Alen David Figueroa Mandujano parte 1 .....	25
Figura 5.3.2 Carta Gantt de Alen David Figueroa Mandujano parte 2 .....	25
Figura 5.4.1 Carta Gantt de Marcelo Orlando Becerra Rozas parte 1.....	26
Figura 5.4.2 Carta Gantt de Marcelo Orlando Becerra Rozas parte 2.....	26

# 1 Introducción

Desde los principios de los tiempos que el ser humano se deja fascinar por cosas desconocidas y novedosas, recordemos el descubrimiento del fuego, las bacterias, las microondas, pirámides, ciudades sumergidas, antiguas civilizaciones por distintas partes del mundo. Es común hacerse preguntas, el interesarse por lo desconocido, ¿por qué las nubes tienen esa forma?, ¿cómo surgió la vida?, ¿Qué habrá pasado con las antiguas civilizaciones?, ¿cómo sabían todo lo que sabían?

¿Se dan cuenta? Preguntas y más preguntas, no existen más opciones que la de ser curioso, el ser humano es así por naturaleza pues esa es la forma en la que funciona el cerebro. Le encanta saber las respuestas a las cosas que le intrigan. Es por esa necesidad que empuja a la persona hacer cosas como visitar y explorar lugares a los que es poco probable que vuelva alguna vez en nuestra vida.

Sin embargo, gracias a esta manía tan inquisitoria es también por lo que se ha conseguido grandes logros como llegar a la luna, dar la vuelta al planeta, a desarrollar antibióticos, a hallar la tumba de Tutankamón. Albert Einstein decía que él no poseía ningún talento especial, tan sólo una curiosidad insaciable y alentaba a todo el mundo a no dejar jamás de hacerse preguntas, debido a que ese es el motor del progreso de la humanidad y uno de los ingredientes principales que ha hecho que hoy se haya conseguido llegar hasta aquí.

Por lo cual actualmente muchas de estas personas tienen la necesidad de irse de su zona de confort y ver nuevos lugares, nuevas etnias, sensaciones, costumbres, comidas entre otros. ¿Existe una alternativa que permita conseguir todo lo descrito anteriormente?

Si, el turismo. Ir a “turistear” es una buena forma de ayudar a todos esos caprichos y no solo eso, también ayuda con el estrés y la ansiedad pues consigue despejar de toda obligación, eres solo tú y tu nueva aventura. Sin embargo, es bastante común que al realizar esta actividad se tengan problemas, muchas veces las personas se pueden sentir desorientadas, perderse por las calles no sería raro y no solo eso, también se pierden gran cantidad de acontecimientos y sensaciones nuevas.

## **2 Descripción del Problema**

Actualmente uno de los mayores problemas a la hora de realizar turismo es que las personas tienen inconvenientes con la ubicación de las zonas que desean visitar, o les cuesta encontrar establecimientos para poder comer o dormir.

Es común que realicen un viaje a un lugar, y que al llegar ahí se encuentren desorientados, a veces se extravían por las calles y no solo eso, también se pierden muchos acontecimientos o nuevas sensaciones que es por lo cual esa persona realizó el viaje, descubrir algo nuevo.

Por ello la aplicación de Guía Turística es la respuesta a este problema. Ya no importa si se pierde por las calles, o no sabe donde alojarse o tenga dificultades con buscar un establecimiento que le sirva comida, la aplicación es capaz de mostrar toda la información que necesita saber sobre zonas turísticas, hoteles y restaurantes sin perder tiempo.

## 3 Descripción de la Solución

### 3.1 Análisis

A continuación, se podrá apreciar todo lo necesario para poder llevar a cabo la aplicación de Guía Turística, desde lo más básico hasta lo más avanzado.

#### Clases Principales:

- **Lugar:** es una clase que contiene todos los datos básicos del lugar, estos son: el id, nombre del local, la latitud, longitud, puntuación, categoría y descripción de lugar, un Mapa con todos los comentarios que tiene el lugar respectivo, y tiene la función cargarComentario. Además de todo eso también contiene la clase **Dirección**, la cual tiene una dirección particular(String), comuna(String), región(String), país(String).
- **DbHandler:** es una clase creada para ejecutar todo lo que son las consultas con la base de dato, en palabras sencillas, es la que se encarga de hacer las queries de la base de datos.
- **CuentaUsuario:** es la clase abstracta, dado que es abstracta no se puede crear este tipo de clase, por lo tanto, se pueden crear subclases hijas, en nuestro caso **Usuario** y **Administrador**. Contiene los atributos nombreUsuario(String) y contraseña(String).
- **Comentario:** contiene los datos básicos de un comentario, el id del usuario, el id del lugar, el comentario en sí, y la puntuación.
- **MapaComentariosUsuario:** es herencia de **MapaComentarios**, esta clase realiza mapas de la Java Collection Framework de los comentarios de un usuario determinado
- **MapaZona:** es una clase que tiene una colección de tipo mapa que contiene todas las zonas disponibles de la base de datos, dentro de cada zona existe un mapa de categorías.
- **MapaCategorías:** es una clase que tiene una colección de tipo mapa con todas las categorías existentes en la base de datos, en cada categoría existe un mapa de lugares.
- **MapaLugares:** es una clase que tiene una colección de tipo mapa con todos los lugares que coinciden con las categorías y la zona específica.
- **MapaUsuarios:** es una clase que tiene una colección de tipo mapa con todos los usuarios, cada uno tiene un mapa de comentarios.
- **Db:** es una clase creada para generar la conexión con la base de datos.
- **Busqueda:** esta clase permite crear una búsqueda por parámetros. Permite generar reportes de todas las categorías existentes en una zona, de todos los usuarios, hace búsquedas de colecciones en general.

### Constructores:

- **ItemComentario:** es una clase que solo sirve para construir los elementos que tiene la ventanaComentarios. Crea etiquetas para todos los usuarios y áreas de textos que tiene el arraylist de comentarios, gracias a que están ingresados en la base de datos tanto el usuario como el comentario, las etiquetas y las áreas reciben estos datos y simplemente completa el campo.
- **ItemComentarioUsuarioActual:** es una clase creada para crear los elementos que contiene la ventanaComentarios a la hora de oprimir el botón de revisar los comentarios. Crea un par de botones y un área de texto para ingresar el comentario del usuario ingresado en la aplicación en ese momento, además también tiene dos botones para ingresar la puntuación.
- **ItemLugar:** es una clase creada para crear los elementos que contiene la ventanaLugares. Crea dos etiquetas, para el nombre del local o de la zona y otra para el rating que contenga este lugar, por otro lado, también se crea un botón para ingresar a este lugar y revisar la información más detallada. Los datos del nombre y del rating están previamente almacenados en la base de datos, por lo que esta clase lo único que debe hacer es recibir los datos y completar los campos.

### Interfaces:

- **Reportable:** es una interfaz utilizada por todas las colecciones, y permite generar reportes (por pantalla y archivo) de todas estas.

### Ventanas:

- VentanaAdmin, VentanaAdminLugares, VentanaCategorías, VentanaComentarios, VentanaInicioSesión, VentanaLugar, VentanaLugares, VentanaRegistro, VentanaUsuario, VentanaReporte, VentanaAdminUsuarios.

### Extras:

- **MapApi:** genera los datos de un lugar, hace la consulta a la API de Google sobre los datos específicos del lugar (datos de la clase Lugar) que se usan para construir los lugares.
- **MapHandler:** genera una URL y maneja la URL que genera para obtener la imagen del mapa que se va ingresar en la ventanaLugar.

### Excepciones:

- **PlaceAlreadyTakenException:** es una extensión de la excepción *PlaceException*, esta se lanza cuando el lugar ya está en el sistema.
- **PlaceException:** esta excepción se lanza cuando el lugar no puede ser ingresado a la base de datos o cuando no existe en la base de datos.



- **UserFindException:** esta excepción es lanzada cuando no se puede encontrar el usuario en el sistema.
- **UserCheckException:** esta excepción se lanza cuando el usuario no es válido a la hora de hacer una operación sobre él.
- **UserRegisterFailureException:** esta excepción es cuando falla el registro del usuario.

## 3.2 Diseño

Como solución a la problemática planteada anteriormente se ha creado una aplicación encargada de gestionar las consultas del usuario a la hora de buscar zonas de interés mientras turisteo, donde la mayor preocupación se basa en que esta aplicación sea simple y amigable con el usuario para que de esta manera sea más fácil de llegar a todas las personas.

Para hacerlo más comprensible y entender la arquitectura interna del programa se pueden ver los siguientes diagramas:

En esta figura se puede divisar la arquitectura de organización en las clases.

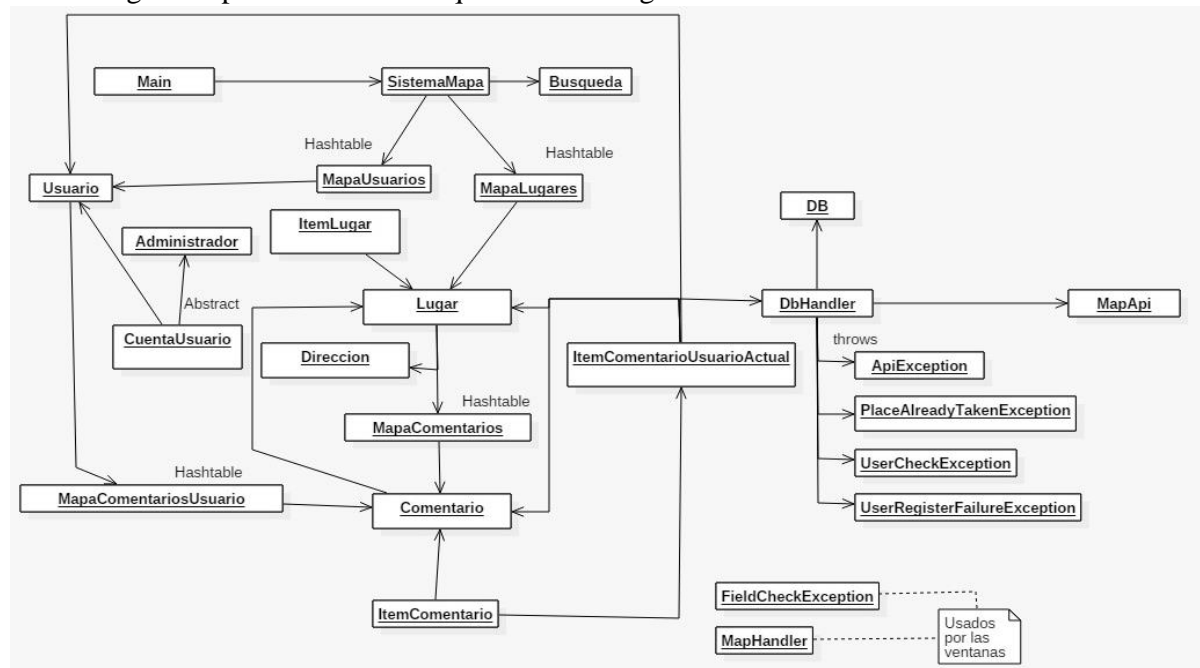


Figura 3.2.1 Diagrama de Clases

En la siguiente figura se podrán apreciar las clases mencionadas en el análisis de la solución con sus respectivos atributos y métodos<sup>1</sup>:

<sup>1</sup> Se ha decidido obviar los métodos getters() y setters() ya que de esa forma la imagen quedaría demasiado grande y como ya era difícil ingresarla, eso lo complicaba aún más.



### 3.3 Base de Datos

Para la elaboración del proyecto como grupo se decidió almacenar los datos en una base de datos relacional, esta se hospeda utilizando el *Amazon Relational Database Service (RDS)* – AWS.

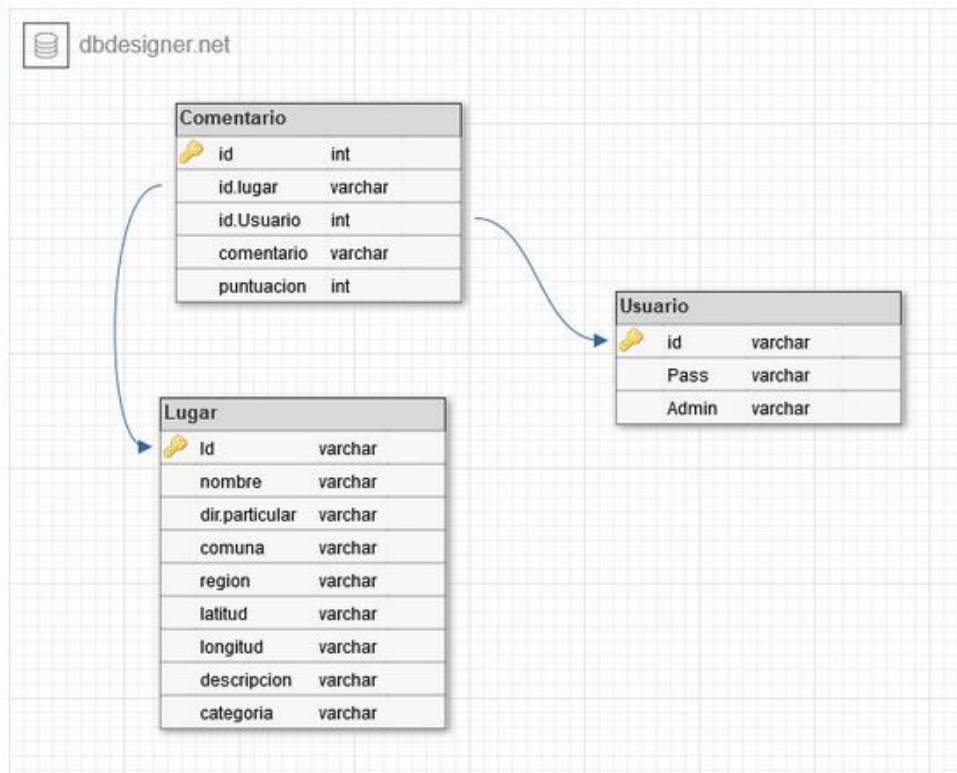


Figura 3.3.1 Modelo de la Base de Datos Relacional de la aplicación Guía Turística

La decisión de usar la base de datos de amazon, se debe a que es fácil de mantener debido a que esa mantención es realizada por amazon por lo que no hay que preocuparse de eso, la creación es sencilla, es accesible en todo lugar<sup>2</sup>, contiene una mayor portabilidad y finalmente porque este servicio otorga espacio suficiente para trabajar con los datos que requieren la aplicación.

#### 3.3.1 Conexión a la Base de Datos a través de Java.

Para poder realizar la conexión a la base de datos mediante Java, primero que todo se debe importar las librerías correspondientes, estas son:

```
4 import java.sql.Connection;
5 import java.sql.ResultSet;
6 import java.sql.SQLException;
7 import java.sql.Statement;
8 import com.mysql.jdbc.jdbc2.optional.MysqlDataSource;
```

<sup>2</sup> Los puertos en el Wi Fi PUCV PRO están bloqueados por lo que no se puede realizar la conexión conectados a esta red.

Además de las librerías previamente importadas, se necesitan los siguientes datos:

```
21 //variables necesarias para realizar queries a la db
22 MysqlDataSource dbLugares;
23 Connection coneccion;
24 Statement stmt;
25 ResultSet rs;
```

*MysqlDataSource* **dbLugares** guarda los datos necesarios para generar la conexión a la base de datos.

*Conexión* **coneccion** guarda la conexión a la base de datos.

*Statement* **stmt** es usada para realizar las consultas.

*ResultSet* **rs** guarda los resultados de una consulta.

Para conectarse a la base de datos se debe crear una instancia de *MysqlDataSource*:

```
31 //se genera la conexión con una base de datos
32 dbLugares = new MysqlDataSource();
--
```

Se establece el usuario y la contraseña con la que se conectará a la base de datos.:

```
34 //establecemos el usuario y la contraseña para conectarse
35 dbLugares.setUser("cubiUsuario");
36 dbLugares.setPassword("c5536652c");
```

Se establece el hosting de la base de datos:

```
38 //se establece la url de la db
39 dbLugares.setServerName("guiaturistica.cxqdybqqakuce.sa-east-1.rds.amazonaws.com");
40
```

Se manda a conectar a la base de datos:

```
41 //se conecta a la base de datos
42 coneccion = dbLugares.getConnection();
43 stmt = coneccion.createStatement();
44
```

*ResultSet* es un tipo de dato que guarda los resultados de una consulta a la base de datos.

```
221 ResultSet rs = stmt.executeQuery("SELECT * FROM Lugar where id = '" + id + "'");
```

*Statement* **stmt** = *coneccion.createStatement()*; es para hacer un query que no afecte a las tablas, cabe mencionar que el *ResultSet* es como una lista enlazada, sin embargo este comienza en -1.

La función de recorrido del set de resultados que se va a utilizar será *next()*, esta se recorre como una lista. Como comienza en posición -1, si es que hay al menos 1 resultado, *rs.next()* retornará true la primera vez, por ese motivo se debe recorrer de la siguiente forma:

```
197 while(rs.next()) {
198
199     //se recibe el valor de la columna en la fila actual, y se retorna en forma de string.
200     String algo = rs.getString("nombre_columna");
201
202     //se recibe el valor de la columna en la fila actual, y se retorna en forma de int
203     int algo2 = rs.getInt("nombre_columna");
204
205 }
```

Para hacer queries que afecten a los datos, como eliminar, modificar y agregar se crea un *Statement* a partir de la conexión. Y a continuación se ejecuta la consulta, por ejemplo, esta:

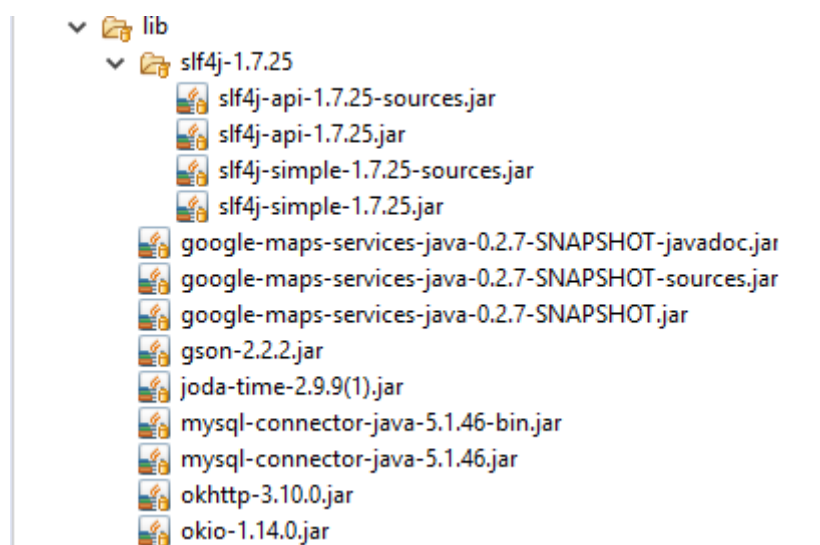
```
263 public void eliminar(Lugar l) throws SQLException{
264     Statement stmt = coneccion.createStatement();
265     stmt.executeUpdate("DELETE FROM Lugar WHERE id = '" + l.getId() + "'");
266 }
```

## 3.4 APIS de Google Maps

### 3.4.1 Geocoding API de Google

El geocoding de la API de Google es una aplicación que realiza el proceso de convertir direcciones (como “ibc, valparaiso, chile”) a coordenadas geográficas (latitud y longitud), además de proporcionar otra información como la dirección completa (calle y numero + comuna + región + país) y un ID por el que se puede identificar el lugar.

Para poder trabajar con con API en Java se debe utilizar los services de Google maps, un conjunto de apis de Google que incluye esta funcionalidad y para poder operar esto en Java se debe importar las librerías correspondientes.



Luego para la utilización de la API:

```
10 import com.google.gson.Gson;
11 import com.google.gson.GsonBuilder;
12 import com.google.maps.GeoApiContext;
13 import com.google.maps.GeocodingApi;
14 import com.google.maps.errors.ApiException;
15 import com.google.maps.model.GeocodingResult;
```

Se debe iniciar un constructor para incluir la librería en Java, en este caso:

```
36 //constructor por defecto, inicializa el contexto con la key de la api
37 //e inicializa el lugar con sus valores por defecto
38 public MapApi() {
39     contexto = new GeoApiContext.Builder()
40         .apiKey("AIzaSyAXUXkUsFImgP_B1pIMco8PrRhefS01oJ8")
41         .build();
42     lugar = new Lugar();
43 }
```

Luego, para poder buscar un Lugar se debe usar:

```
49 //se realiza la búsqueda del lugar a partir del string 'lugar' y se toma el primer resultado
50 GeocodingResult resultado = GeocodingApi.geocode(contexto, lugar).await()[0];
```

este comando genera un array de resultados, ordenados por relevancia, en este caso se asume que sólo se necesitará el resultado más relevante, el 0.

Se crea una instancia de *Gson* para manejar los distintos campos del resultado generado:

```
52 //se crea un Gson para manejar el resultado
53 Gson gson = new GsonBuilder().setPrettyPrinting().create();
54
55 //se crea un arreglo de strings con los parámetros del formattedAdres
56 //el formato es: "direccion, comuna, region, pais"
57 String[] direccion = gson.toJson(resultado.formattedAddress)
58     .replaceAll(", ", ",")
59     .replaceAll("\\\"", "\"")
60     .split(",") ;
```

**resultado.formattedAdres** es un String de la forma “dirección, comuna, región, país” así que se le sacan los espacios, las “” y se separa por comas.

Se obtienen los valores de la latitud, longitud y el ID de la ubicación, para su inserción con **resultado.geometry.location.lat**, **resultado.geometry.location.lng**, **resultado.placeID**, como se observa en la siguiente figura:

```
62 //se cambian los valores de 'lugar' por los del resultado de búsqueda y se retorna el objeto lugar
63 this.lugar.setId(resultado.placeId);
64 this.lugar.setDir(direccion);
65 this.lugar.setLat( resultado.geometry.location.lat );
66 this.lugar.setLng( resultado.geometry.location.lng );
```

### 3.4.2 Static Maps API de Google

Static Maps API devuelve una imagen (gif, png o jpg) en respuesta a una solicitud HTTP a través de una dirección URL.

La siguiente figura son las importaciones necesarias para trabajar con Static Maps API:

```
3 import java.awt.image.BufferedImage;
4 import java.io.IOException;
5 import java.net.MalformedURLException;
6 import java.net.URL;
7
8 import javax.imageio.ImageIO;
9 import javax.swing.ImageIcon;
```

La URL especificará el tamaño de la imagen, zoom, dónde estará centrado el mapa, el tipo de mapa que muestra (mapa común, que resalte las calles o vista satelital) e incluso se pueden agregar marcadores opcionales con etiquetas personalizadas.

La construcción de la URL es de la siguiente forma:

```
22 //key de api, necesaria para poder usar la api de mapas estáticos de google
23 private static String apiKey = "AIzaSyC4ClGpPgudrfyIWr0528j9DxXLilQ3K1U";
24
25 //el tamaño de la imagen generada
26 private static String size = "410x297" ;
27
28 //genera la url inicial, con la key de la api y el tamaño
29 private String dirIn = "https://maps.googleapis.com/maps/api/staticmap?key="
30                       + apiKey
31                       + "&size="
32                       + size
33                       + "center=IBC,+valparaiso"
34                       + "&zoom=18&maptype=roadmap";
```

Los distintos parámetros pueden estar en cualquier orden, los que están en la URL de arriba son los obligatorios para que tenga sentido, la **apiKey** es una clave de acceso que ofrece Google a los desarrolladores para ocupar las distintas APIS que posee. Se pueden obtener estas keys desde *console.developers.google.com*.

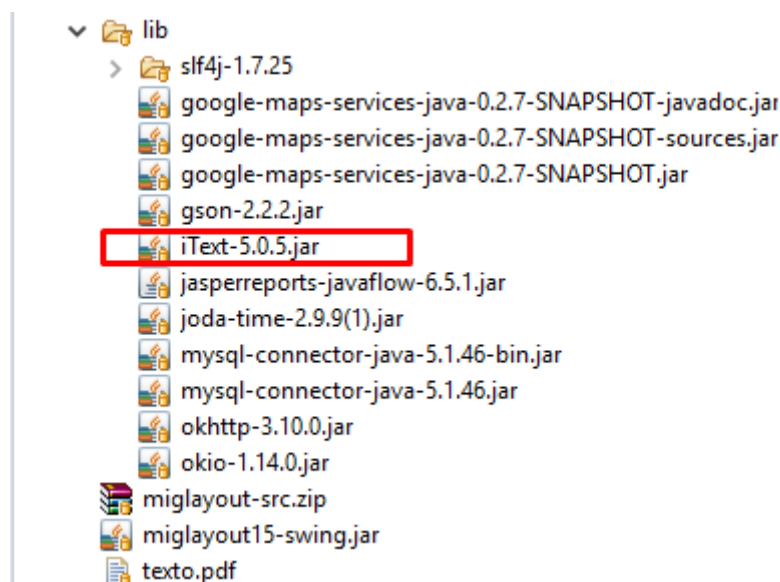
Para manejar la URL en Java se debe generar un icono manejable, y que este almacene el mapa obtenido por la petición a la API. Para esto se debe crear un *ImageIcon* con la respuesta de la URL.

```
//se lee la imagen que entrega la url y se crea el icono con esta
map = new ImageIcon( ImageIO.read(url) );
}
```



### 3.5 Reportes por PDF en Java.

Para generar un archivo pdf básico fácilmente en java se utiliza la librería **iText**, en el proyecto se encuentra en: */Guia\_Turistica/lib/Text-5.0.5.jar*.



Para poder generar reportes ya sean de pantalla o por PDF se necesita importar primero las librerías, estas son:

Las librerías para hacer reportes por PDF:

```
19 import com.itextpdf.text.Document;
20 import com.itextpdf.text.DocumentException;
21 import com.itextpdf.text.Paragraph;
22 import com.itextpdf.text.pdf.PdfWriter;
23
```

Para poder hacer reportes por pantalla:

```
12 import javax.swing.JFrame;
13 import javax.swing.JPanel;
14 import javax.swing.JTextPane;
15 import javax.swing.border.EmptyBorder;
16 import javax.swing.text.BadLocationException;
17 import javax.swing.text.StyledDocument;
18
```

Y finalmente las librerías necesarias para mostrar el menú de guardado del archivo:

```
27 import javax.swing.JFileChooser;
```

Una vez terminado las importaciones de las librerías, se debe generar el reporte por pantalla, así que hay que proceder a llenar el panel de texto:



Primero se crea el área de texto:

```
55     textPane = new JTextPane();
56     textPane.setEditable(false);
57     textPane.setBounds(5, 5, 549, 409);
58
```

Se crea una instancia del formato de fecha deseado para guardar en el reporte:

```
59     //Se establece el formato de la fecha que se escribirá en el documento
60     DateFormat format = new SimpleDateFormat("dd-MM-yyyy HH:mm:ss");
```

Se obtiene el usuario que se agregará al documento:

```
65     //se obtiene el StyledDocument del textPane para agregar cosas
66     StyledDocument document = (StyledDocument) textPane.getDocument();
```

Para agregar un String al *TextPane* se debe hacer como en la siguiente figura:

```
try {
    //se inserta la primera línea del reporte
    document.insertString(document.getLength(), nombre, null);

    //se inserta la segunda línea del reporte
    document.insertString(document.getLength(), "\n\n Fecha de creación:\n\t " + format.format(new Date()), null);

    //se inserta la categoría buscada
    document.insertString(document.getLength(), "\nCategoría:\n " + l.get(0).getCategoria(), null);

    document.insertString(document.getLength(), "\nLugares Buscados:", null);

    //se insertan todos los lugares encontrados en ese parámetro de búsqueda
    for(Lugar i : l) {
        document.insertString(document.getLength(), "\n\t" + i.getNombreLocal() + ", " + i.getComuna() + ", "
            + " " + i.getRegion() + ", " + i.getPais(), null);
    }
}
```

Luego para guardar el texto generado, se debe usar la librería **iText**, para eso primero se pregunta al usuario dónde quiere guardar el documento y el nombre por el que lo guardará, esto es simple de implementar gracias a la clase **JFileChooser**.

Se crea y muestra la ventana donde el usuario podrá seleccionar dónde guardar, esto retorna un **int** el cual depende si es una ruta válida.

```
//se crea y manda la pantalla de elección de la ruta dónde se guardará el archivo
JFileChooser path = new JFileChooser();
int option = path.showSaveDialog(estaVentana);
```

En caso de que la ruta sea válida, entonces se abre el archivo y se guarda la ruta de este:

```
//en caso de que se pueda crear el archivo, se crea el archivo
//se agrega la extensión .pdf en caso de ser necesario y se manda a escribir el contenido
//del textPane en el archivo de la ruta seleccionada
if(option == JFileChooser.APPROVE_OPTION) {
    File f = path.getSelectedFile();
    documentPath = f.toString();
    if(!documentPath.endsWith(".pdf"))
        documentPath = documentPath.concat(".pdf");
    escribeDocumento();
}
```

Una vez seleccionado dónde guardarlo, podemos escribir el documento:

```
public void escribeDocumento() {
    //se crea una instancia de Document
    Document pdf = new Document();

    //se crea una instancia de escritura de archivo
    FileOutputStream archivo;
    try {

        //se abre el archivo en modo escritura
        archivo = new FileOutputStream(documentPath);

        //se crea una instancia de escritura del pdf
        PdfWriter.getInstance(pdf, archivo);

        //se abre el archivo
        pdf.open();

        //se agrega un párrafo con el contenido del textPane
        pdf.add(new Paragraph(textPane.getText()));

        //se agrega el autor y la fecha de creación al documento
        pdf.addAuthor("Aplicación guía turística");
        pdf.addCreationDate();

        //se cierra el documento
        pdf.close();

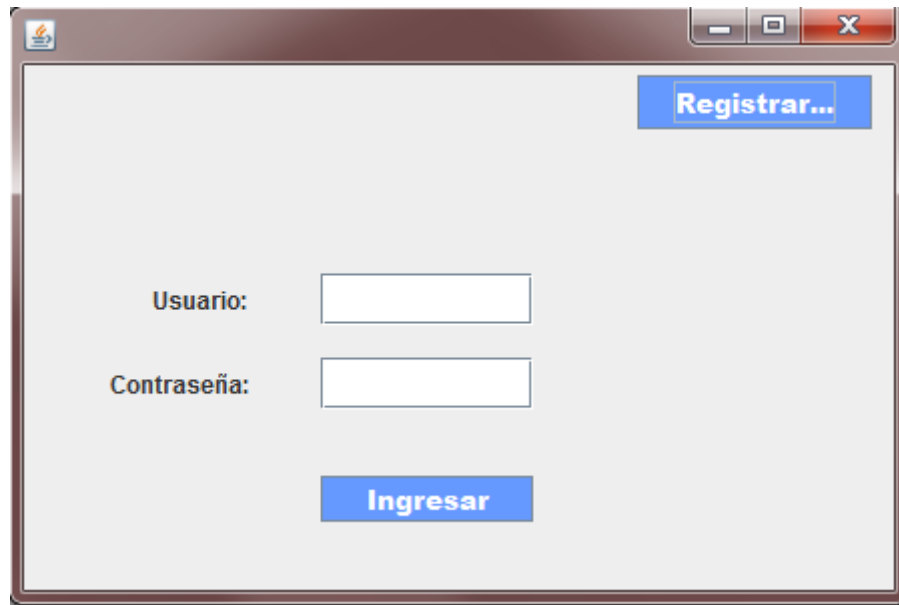
    } catch (FileNotFoundException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    } catch (DocumentException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
}
```

## 4 Diseño de la Interfaz

### 4.1 Ventana Inicial

Este es el menú de Inicio de Sesión, en caso de la primera vez de un usuario, este puede registrarse fácilmente en el botón de Registrarse.

Para poder utilizar esta ventana el programa previamente a través de la base de datos carga los nombres de los usuarios ya sean clientes o administradores, con su respectiva contraseña, además de otra información necesaria para otras funciones del programa.

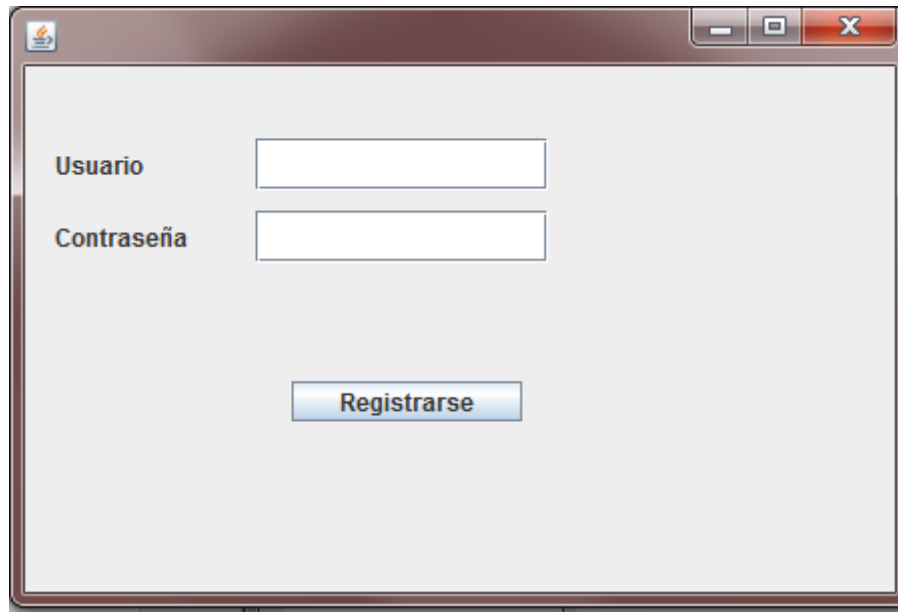


The image shows a graphical user interface window for a login system. The window has a title bar with a small icon on the left and standard Windows window controls (minimize, maximize, close) on the right. The main area of the window is light gray. In the top right corner, there is a blue button with white text that says "Registrar...". Below this, there are two input fields. The first is labeled "Usuario:" and the second is labeled "Contraseña:". Below these fields, there is a blue button with white text that says "Ingresar".

Figura 4.1.1 Ventana Inicial de la aplicación Guía Turística

## 4.2 Ventana de Registro

En caso de que el usuario entre por primera vez a la aplicación lo primero que deberá hacer es registrarse con un Nombre de Usuario y una Contraseña pues de otra forma este no podrá acceder a la aplicación, una vez haya apretado el botón de Registrarse le aparecerá algo como esto:



**Figura 4.2.1 Ventana de Registro de la aplicación Guía Turística.**

El usuario lo único que deberá hacer es ingresar el nombre de usuario y la contraseña que desee, una vez haya apretado el botón Registrarse, los datos que ingreso en los campos en blanco serán almacenados en la base de datos para una vez de vuelta en la ventana inicial pueda ingresar sin problema alguno.

### 4.3 Ventana de Categorías

La ventana que visualizará tendrá una barra con opciones de búsqueda, en esta opción habrá nombres de ciudades del país, cuando haya seleccionado la zona en la que desea ubicarse deberá seleccionar la categoría, estas serán: Atracciones, Hoteles, Restaurantes y Vida Nocturna. Dependiendo de cual elija le saldrá más adelante otra ventana con lugares de esa índole.



Figura 4.3.1 Ventana de Categorías en la aplicación de Guía Turística.

### 4.4 Ventana de Lugares

En cuanto el usuario oprima sobre una de las cuatro categorías nombradas anteriormente (4.3 Ventana de Categorías), se abrirá una ventana de lugares que contiene una lista de todos los locales o lugares que se encuentran en la zona específica con la categoría señalada. Esta lista contiene a primera vista el nombre del local o la zona, una puntuación que es dependiente de la puntuación que le dan los mismos usuarios y un botón para ver más en detalle ese lugar o local.

Aparte de la lista que contiene también hay un botón para generar reportes.

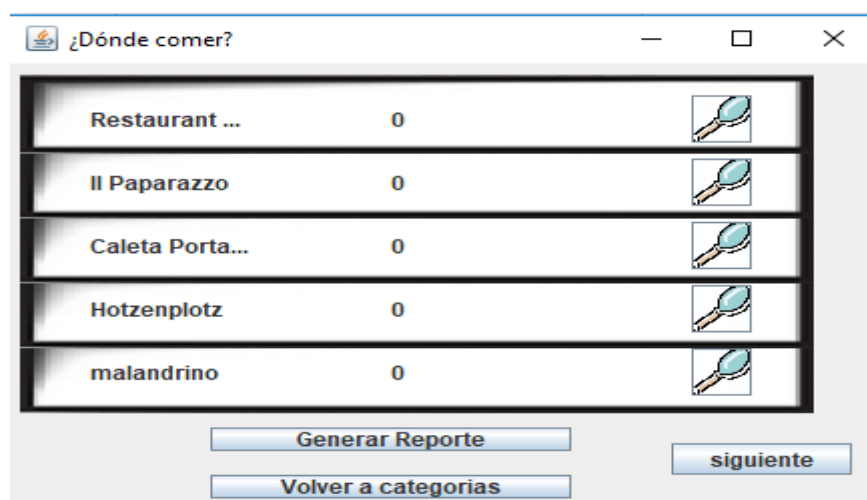


Figura 4.4.1 Ventana de Lugares en la aplicación de Guía Turística

## 4.5 Ventana de Lugar

Si se oprime en el botón para ver más en detalle del local o el lugar, dará paso a una ventana que contiene información detallada de este sitio, esta información es: el nombre del lugar, una dirección, una puntuación total, una descripción, un mapa gráfico para poder llegar al destino correspondiente y finalmente un botón que contiene todos los comentarios de los usuarios.

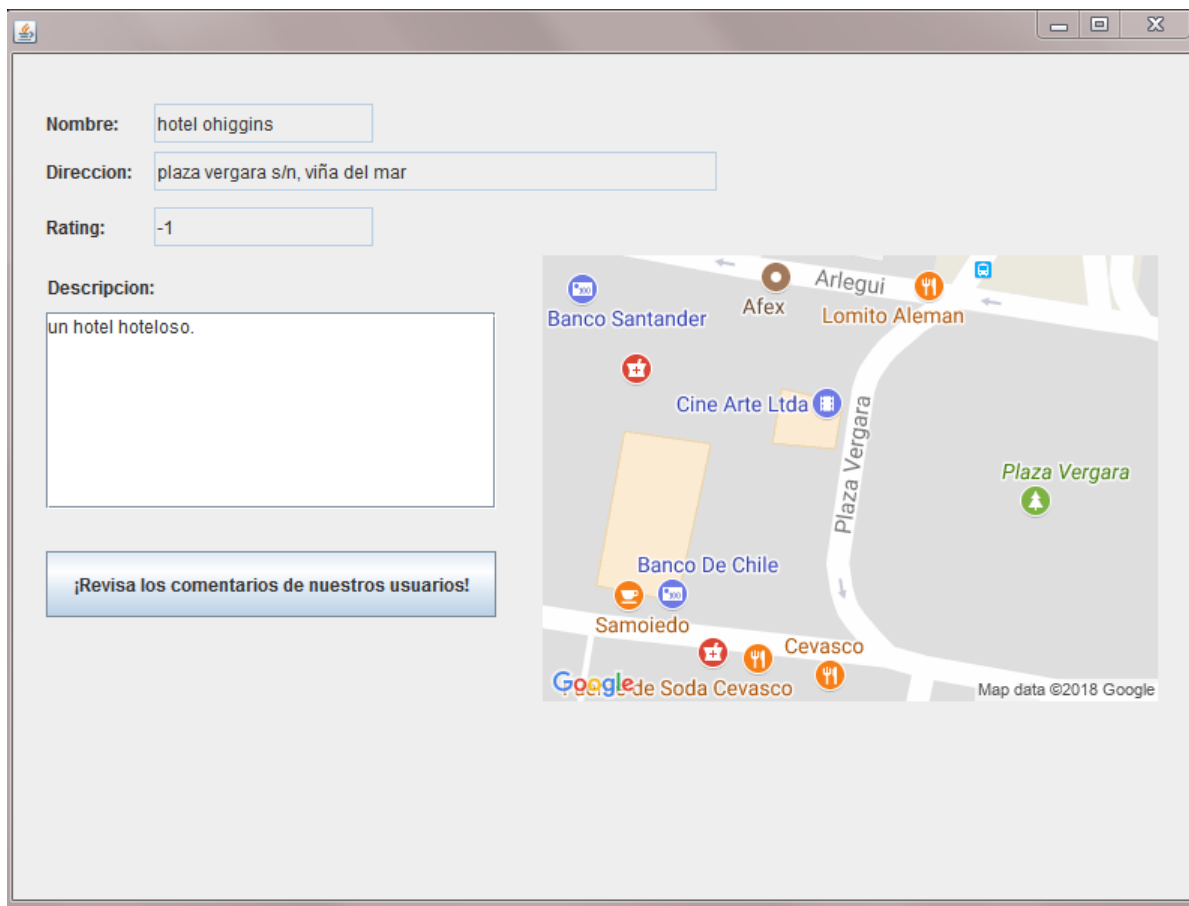


Figura 4.5.1 Ventana de Lugar en la aplicación de Guía Turística

## 4.6 Ventana de Comentarios

En la ventana de comentarios se tiene una lista de todos los comentarios con los nombres de sus respectivos usuarios además de su puntuación, también esta ventana te permite realizar tu propio comentario al lugar y dejarle una puntuación positiva o negativa. Una vez terminado tu propio comentario solo queda presionar el botón de actualizar y estará tanto el comentario como la puntuación registrados.

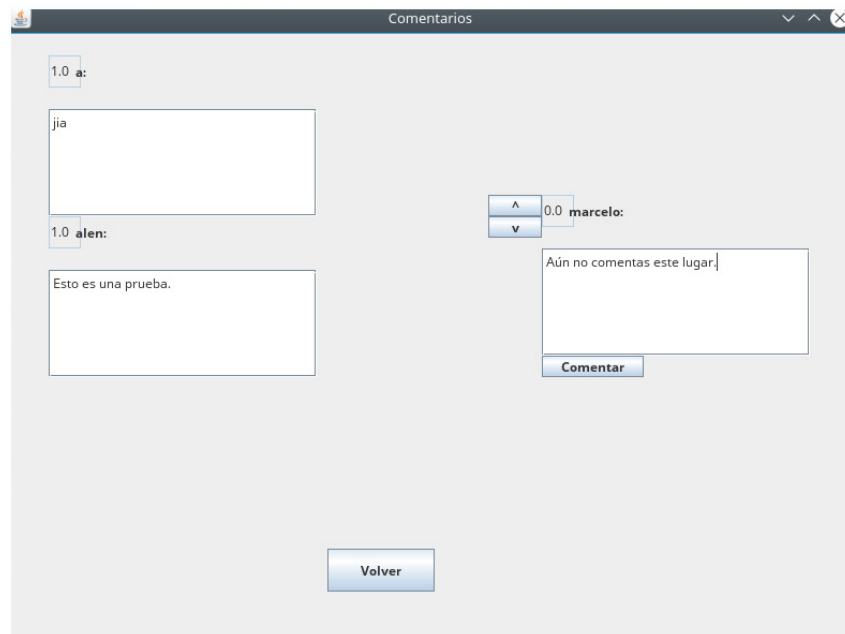


Figura 4.6.1 Ventana de Comentarios en la aplicación de Guía Turística

## 4.7 Ventana de Administrador

Cuando se ingresa con una cuenta de tipo administrador lo manda a una ventana que contiene las siguientes opciones: Administrar Lugares y Administrar usuarios.



Figura 4.7.1 Ventana de Administrador en la aplicación Guía Turística

## 4.8 Ventana de Reporte

En caso de oprimir el botón de *Generar Reporte*, se abrirá una ventana que indica el nombre de usuario, la fecha de creación del reporte, y el tipo de lista del reporte, es decir, si el reporte que se oprimió es ***generar reporte por zonas*** entonces se obtendrá una lista de las zonas donde residen esos lugares en la base de dato.

Por otro lado, también se encuentra un botón con la opción de *Guardar Reporte*, el cual guardará en la ubicación y con el nombre que el usuario desee.

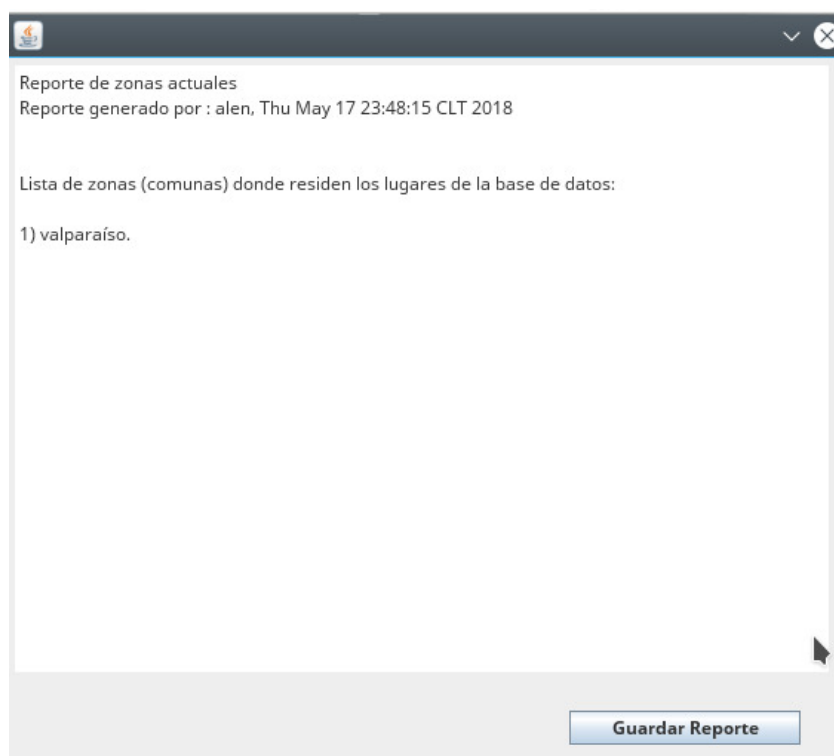


Figura 4.8.1 Ventana Generar Reporte en la aplicación Guía Turística

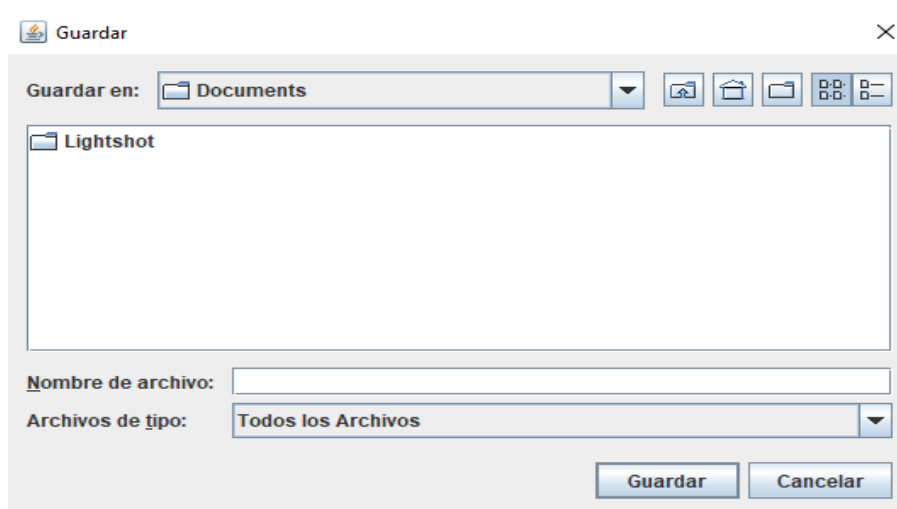


Figura 4.8.2 Ventana para guardar el reporte en la aplicación Guía Turística



## 4.9 Ventana de Administrar Lugares

Si la opción que se elige como administrador es administrar un lugar, entonces aparecerá una ventana donde se tendrá una barra para ingresar una zona, se podrá ingresar su nombre, su dirección, la comuna, la región y el país de donde se encuentra, la latitud y la longitud que tiene, se le asigna un ID que es entregada por la API de Google Maps y seleccionar la categoría a la que pertenece.

Por otro lado, hay un mensaje donde indica si esa zona está registrada en la base de datos, en caso de no ser así tenemos la opción de ingresarlo donde se guardarán todos los datos previamente mencionados, una vez realizado todas las modificaciones que se desean al lugar le aplicamos los cambios, en caso contrario se puede eliminar el lugar. También existe la opción de dirigirse a ese lugar oprimiendo *Ver Lugar*, al oprimir esta llevará al administrador a su respectiva ventana. (Figura 4.5 Ventana de Lugar).

Administración de lugares.

IBC, valparaíso

En base de datos:

Nombre:

Dirección:

Comuna:

Región:

País:  Descripción:

Latitud:

Longitud:

ID:

Figura 4.9.1 Ventana de Administrar Lugares en la aplicación Guía Turística

## 4.10 Ventana de Administrar Usuarios

En caso de elegir la opción de administrar usuarios al ser administrador se abrirá una ventana donde le permitirá buscar un usuario específico, en caso de encontrarlo indicará su nombre de usuario y contraseña correspondientes. Por otro lado, se puede modificar la contraseña o bien si se quiere el estatus de “Administrador” o “Usuario Normal” marcando o desmarcando la casilla de *Administrador* oprimiendo el botón de *Modificar*. En caso de querer eliminar el usuario puede oprimir el botón de *Eliminar*.

También existe el botón de *Reporte* que le permite al Administrador generar una lista de todos los usuarios que existen en la base de datos.



Figura 4.10.1 Ventana de Administración de Usuarios en la aplicación de Guía Turística

## 5 Planificación

### 5.1 Carta Gantt Grupal

En la siguiente figura se procederá a mostrar todos los avances *grupales* realizados en su tiempo determinado con el fin de poder entregar satisfactoriamente la parte asignada del proyecto.

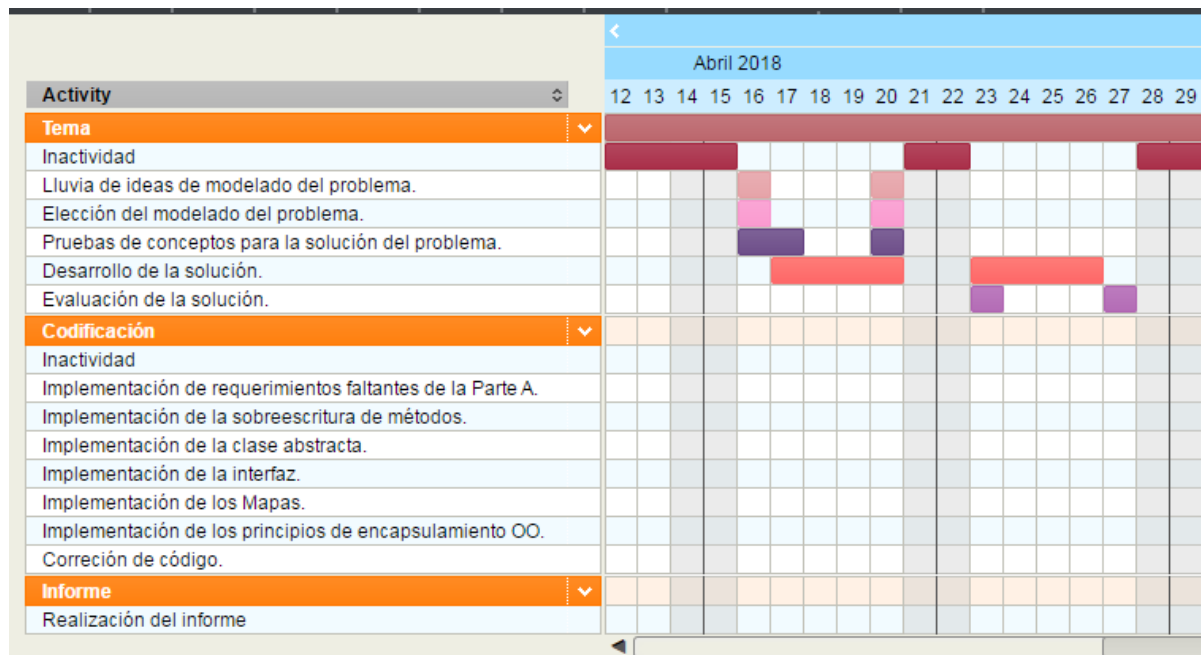


Figura 5.1.1 Carta Gantt grupal parte 1

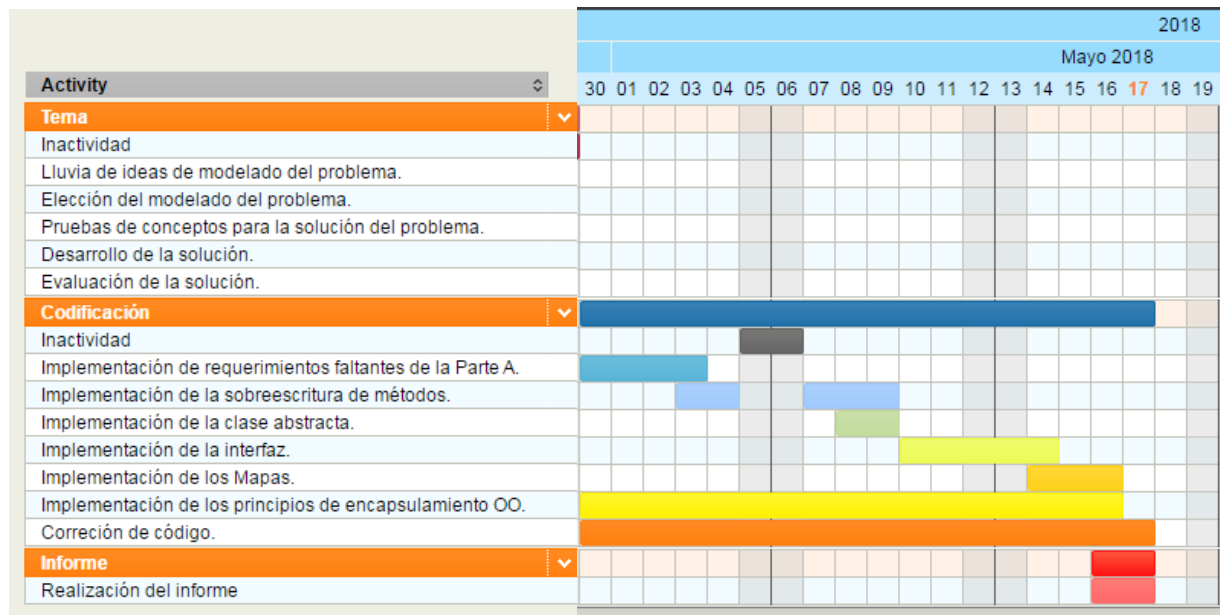


Figura 5.1.2 Carta Gantt grupal parte 2

En las figuras que vienen a continuación se presentarán todos los procesos de forma individual realizados en su tiempo determinado para llevar entregar satisfactoriamente la parte asignada del proyecto.

## 5.2 Carta Gantt Javier Esteban Peña Reyes

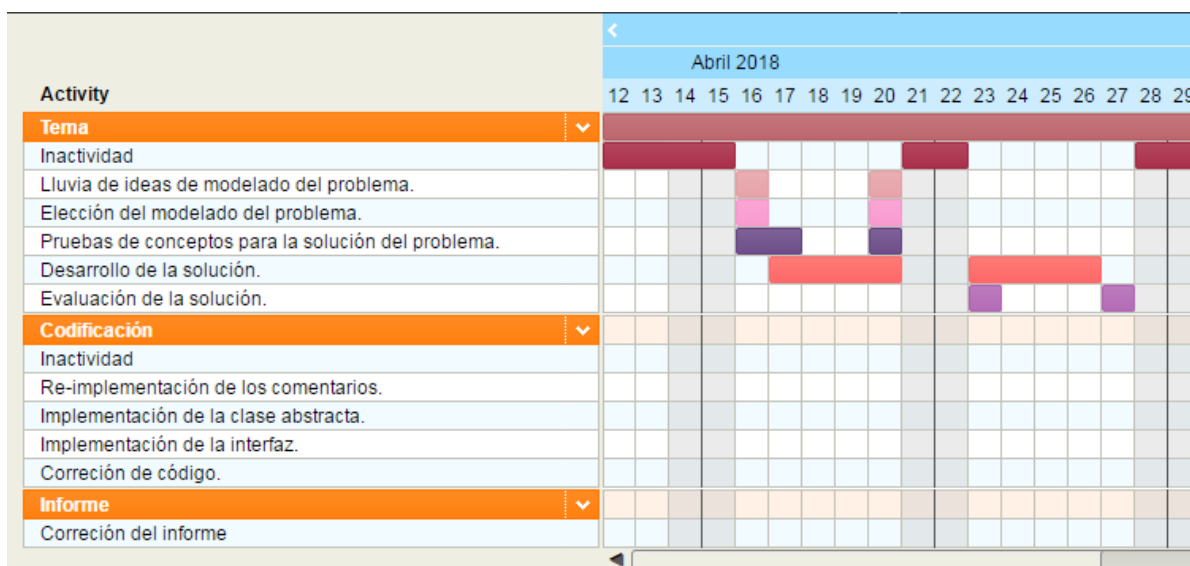


Figura 5.2.1 Carta Gantt de Javier Esteban Peña Reyes parte 1



Figura 5.2.2 Carta Gantt de Javier Esteban Peña Reyes parte 2

### 5.3 Carta Gantt Alen David Figueroa Mandujano

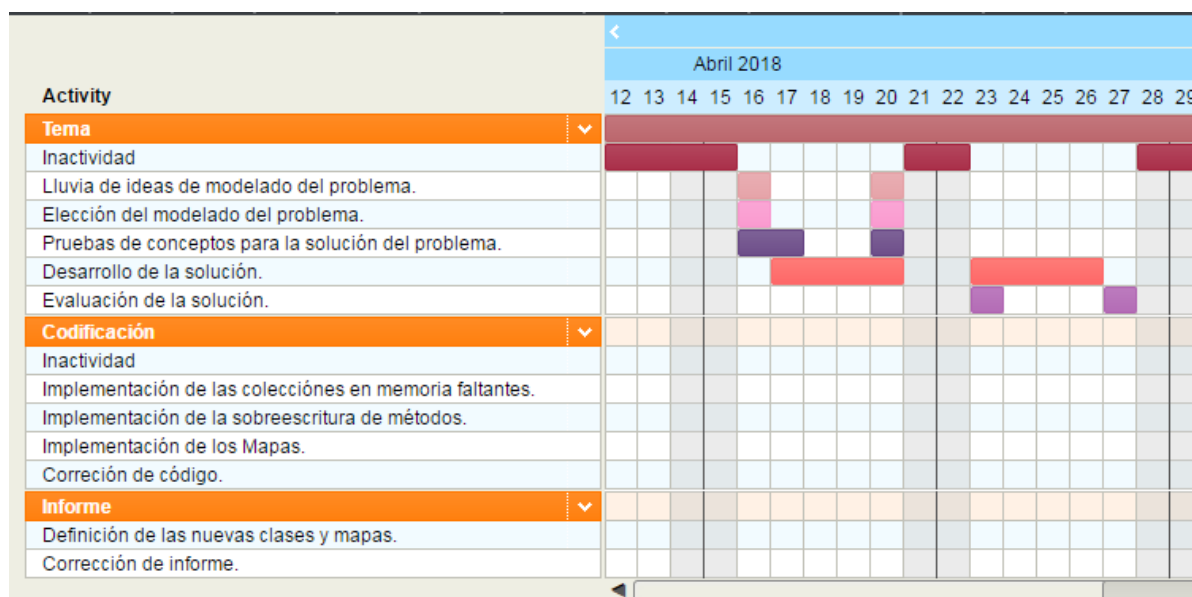


Figura 5.3.1 Carta Gantt de Alen David Figueroa Mandujano parte 1

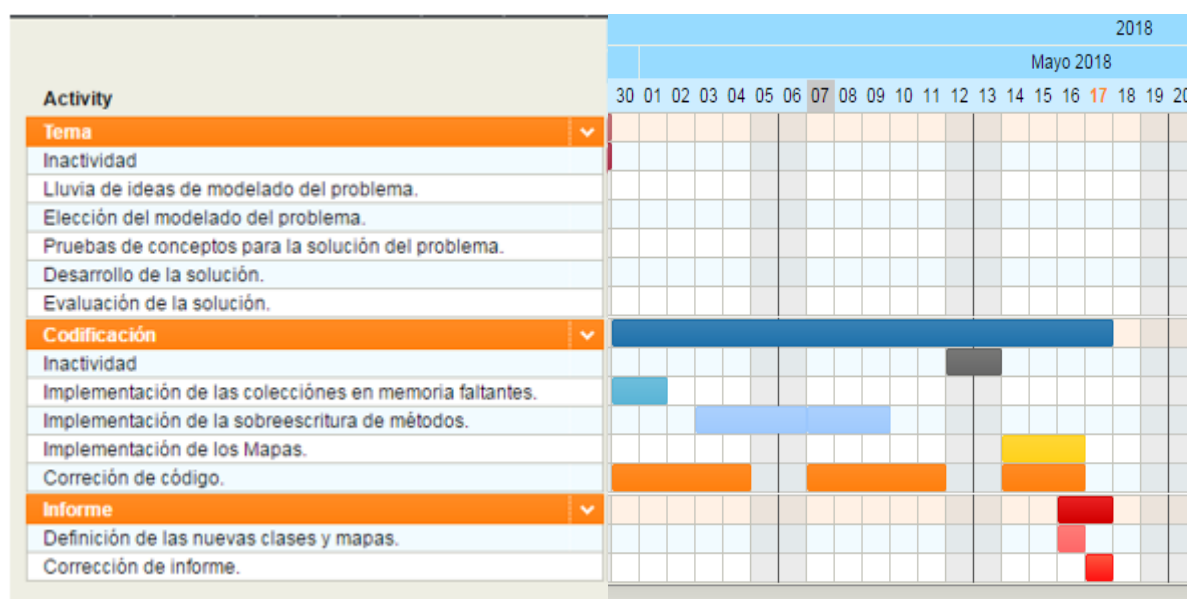


Figura 5.3.2 Carta Gantt de Alen David Figueroa Mandujano parte 2

## 5.4 Carta Gantt Marcelo Orlando Becerra Rozas

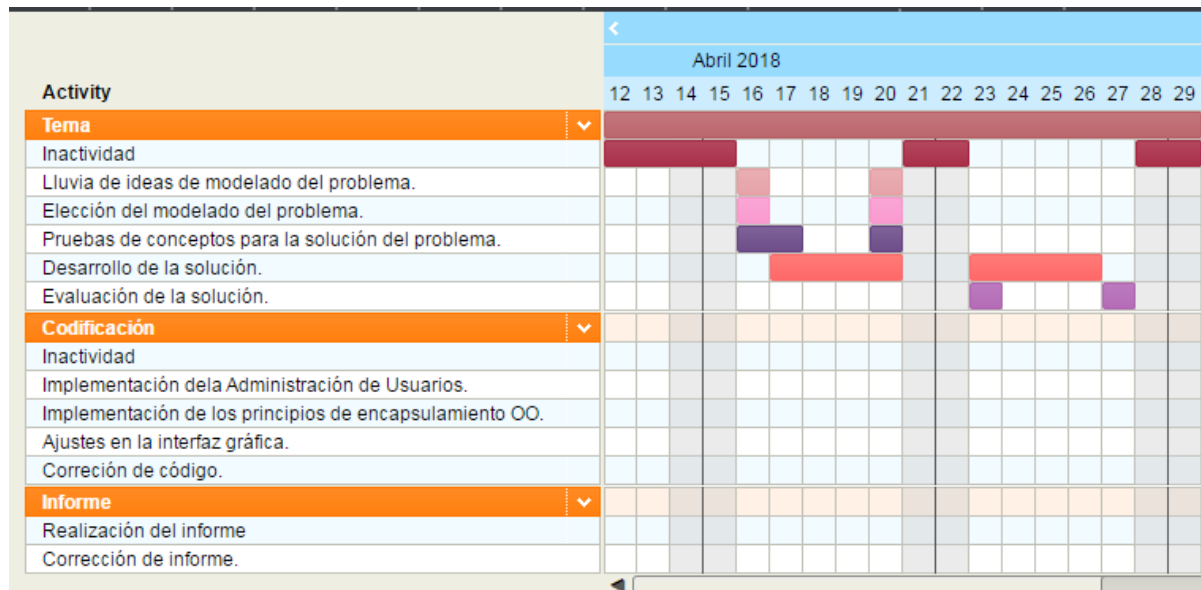


Figura 5.4.1 Carta Gantt de Marcelo Orlando Becerra Rozas parte 1

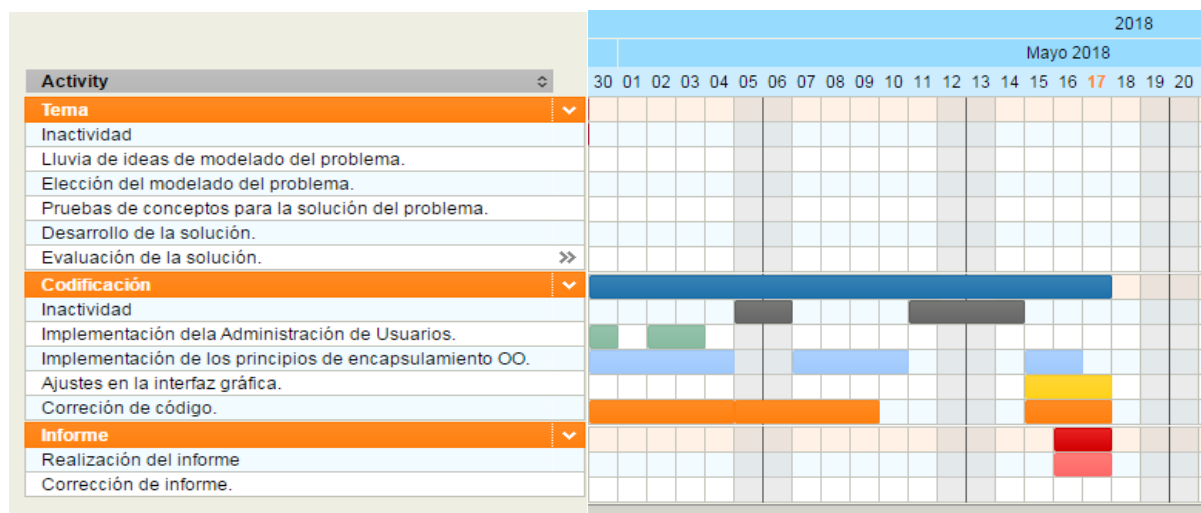


Figura 5.4.2 Carta Gantt de Marcelo Orlando Becerra Rozas parte 2

## 6 Conclusión

Durante el transcurso del proyecto se han presentado distintas dificultades, principalmente en lo que respecta a la programación de éste, dado que a pesar de poseer conocimientos teóricos enseñados en clase, los conocimientos prácticos aplicando a lo aprendido eran muy vagos o casi inexistentes, esto no quiere decir que no se entendiera el lenguaje, por el contrario lo se comprendía sin embargo, había que profundizar en su aplicación saliendo de esa zona ideal donde todo por teoría debía funcionar y no generar ningún error.

Una de las partes más complicadas del proyecto fue hacer la conexión de la base de datos ya que no se sabía cómo hacer las mismas de manera correcta, otro problema también fue la API de Google Maps, debido a casi lo mismo que la base de datos, habíanputa varios acontecimientos en los que no había conocimiento, también estaban los conceptos de encapsulamiento, las clases abstractas e interfaces. Resumiendo, gran parte de todos estos problemas eran prácticamente por la falta de práctica. Sin embargo, ahora mirando hacia el pasado, la mayoría de esos problemas y conceptos no son tan grandes como parecían en su momento, incluso para algunos es sorprendente como costó comprenderlo, pero ahora parece algo casi obvio.

Respecto a la creación de la aplicación y al futuro de esta hay muchos cambios que se desean hacer, por ejemplo, en la gran mayoría de las ventanas deseamos dejarlo estéticamente hablando “más presentable” para que cuando el usuario vaya a hacer uso de él, sea agradable a la vista y fácil de manejar, pues tal y como está ahora, es tosco de usar y a veces hasta desagradable. Por otro lado, hay grandes deseos de agregar algunas funcionalidades, así como de modificar otras, también arreglar el tema del orden del programa, se tiene intenciones de dejarlo más legible y entendible.

Se siente que todo esto fue una buena experiencia, desde no comprender nada y no poder dormir algunos días para poder finalizar los plazos en los tiempos estimados hasta complacer al cliente que pide especificaciones que a nuestra opinión no parecen útiles para el programa, todo esto y más, seguramente serán experiencias que otras personas nunca habrán vivido y que servirán para poder superar los problemas que surjan en un futuro.

Para finalizar, dar las gracias a el profesor y las ayudantes, debido a que respondieron los problemas que surgían y a la vez nos guiaban, con lo cual pudimos finalizar nuestra parte del proyecto satisfactoriamente.

## 7. Referencias

- 1.- Google Developers. (abril 6, 2017). La geocodificación, de Google Sitio web:  
<https://developers.google.com/maps/documentation/geocoding/intro>
- 2.- GitHub, Inc. Google Maps Services, de GitHub Sitio web:  
<https://github.com/googlemaps/google-maps-services-java>
- 3.- Google Developers. (abril 6, 2017). Google Maps Services, de Google Sitio web:  
<https://developers.google.com/maps/documentation/static-maps/intro>
- 4.- Yañez, J. (febrero 24, 2016). Creando un pdf en Java con iText, de codigoxules.org Sitio web:  
<http://codigoxules.org/java-itext-pdf-creando-pdf-java-itext/>