DOCUMENTACIÓN "ASKFOR"

Lenguajes de programación

Descripción breve

El presente documento pretende ilustrar al lector con el proceso de desarrollo de la aplicación AskFor, que proviene de Ask Forum y pretende simular un Stack Overflow. Se mostrarán decisiones de diseño, algoritmos y librerías utilizadas entre otras cosas, además de un manual de usuario.

Fabián Fernández & Pablo Corrales

Tecnológico de Costa Rica

TABLA DE CONTENIDOS

DESCRIPCIÓN DEL PROBLEMA

El problema a resolver consta del diseño y creación de una aplicación para Android similar a la herramienta Stack Overflow en el lenguaje de programación Ruby para el backend y Android(java) para la GUI, para poder familiarizarse con los conceptos de la programación orientada a objetos y el desarrollo de aplicaciones móviles para la platadorma Android.

Mediante la utilización de la plataforma Heroku para poder "hostear" la aplicación, nos permite simular una base de datos que le brinda al usuario la posibilidad de registrarse, iniciar sesión y visualizar las diferentes preguntas de los demás usuarios en la plataforma.

Se debe conectar desde el apk al servidor en Heroku y desplegar las diferentes funciones del sistema.

DISEÑO DEL PROGRAMA

ESCOGENCIA DEL LENGUAJE DE PROGRAMACIÓN

Se decidió utilizar el lenguaje de programación Ruby para el backend en vez de Scala debido a las facilidades que brinda este lenguaje. Además la implementación de funciones, la manipulación de variables y cadenas es ciertas ocasiones es más simple. Dentro de los lenguajes propuestos para realizar esta tarea estaban Ruby y Scala por lo que se decidió descartar Scala lenguajes debido a la complejidad del proyecto y el poco tiempo con el que se disponía en un principio, además se decide usar java para el desarrollo de la aplicación para Android, con la herramienta de desarrollo Android Studio que debido a la experiencia previa con Eclipse no fue del todo satisfactoria y Android Studio nos brindaba todo lo que necesitábamos para hacer una interfaz agradable y agregar más opciones en el diseño.

ANDROID STUDIO

La razón de utilizar Android Studio es porque es el entorno en donde se van a desarrollar las aplicaciones de Android a futuro, está basado en IntelliJ IDEA, uno de los mejores IDE para java.

La forma de construir los apk, más versátil más potente y más similar a un proyecto en Java, el cual es un lenguaje en el que grupo ya está familiarizado y conoce muy bien. Facilita bastante al reusar código y recursos, facilita configurar, extender y personalizar el proceso.

Permite distribuir el código más fácilmente y por lo tanto el trabajo en equipo se hizo más sencillo. Se puede concluir que es una herramienta con potencial mayor, especialmente de cara a entornos empresariales.

CLASE USUARIO

La clase usuario permite almacenar todos los usuarios registrados, agregar usuarios nuevos, mostrarlos y validar su existencia en la base de datos.

```
class Usuario#########################################################################################################
    def initialize(user,pass) # Constructor de Usuario
        @user = user # Usuario
        @pass = pass # Contrasenha
    end

    def getUser() # Obtiene User de objeto Usuario
        @user
    end

    def getPass() # Obtiene Pass de objeto Usuario
        @pass
    end

    def self.getUsuarios()
        $users_registrados
    end

    def self.cargar_txt() # => Carga los usuarios a memoria en caso de cierre de programa
        archivo = File.new("users.txt","a+") #Abre el archivo "users", si no existe lo crea
        archivo.rewind #Vamos a la primera linea del archivo
        lista_users = archivo.readlines #Lee todas las lineas y las mete a la variable en forma de lista
        cont = 0 #contador para recorrer lista
        while cont < lista_users.length
            temp = lista_users[cont].split(",") #Los users y pass esta separados por "," (Comas) se crea una lista de esta forma [user,pass]
            $users_registrados[$users_registrados.length] = Usuario.new(temp[0],temp[1].chomp) #Agrega a final de lista a objeto tipo Usuario
            cont = cont + 1 #Aumenta contador
        end
        archivo.close # Cierra archivo y guarda los cambios
    end
```

FUNCION DE CARGA DE USUARIOS

Básicamente carga los usuarios en memoria mediante un .txt, que le permite al programa almacenar los datos de los usuarios.

```
def self.cargar_txt() # => Carga los usuarios a memoria en caso de cierre de programa
    archivo = File.new("users.txt","a+") #Abre el archivo "users", si no existe lo crea
    archivo.rewind #Vamos a la primera linea del archivo
    lista_users = archivo.readlines #Lee todas las lineas y las mete a la variable en forma de lista
    cont = 0 #contador para recorrer lista
    while cont < lista_users.length
        temp = lista_users[cont].split(",") #Los users y pass esta separados por "," (Comas) se crea una lista de esta forma [user,pass]
        $users_registrados[$users_registrados.length] = Usuario.new(temp[0],temp[1].chomp) #Agrega a final de lista a objeto tipo Usuario
        cont = cont + 1 #Aumenta contador
    end
    archivo.close # Cierra archivo y guarda los cambios
end
```

FUNCION DE REGISTRO DE NUEVOS USUARIOS

Esta función permite en caso de que haya un usuario nuevo, registrarlo en el sistema para que pueda acceder y pueda utilizar la aplicación desde su móvil.

```
def self.registra_user(texto) # => Agrega el usuario al txt y a la lista de users_registrados. Recibe un string con el formato user-pass
    texto = texto.gsub(" ","") #Quita todos los espacios " "
    texto = texto.gsub("-",",") #Sustituye "-" por ","
    u_p = texto.split(",") #Crea una lista con el formato [user,pass]
    if not Usuario.user_existe(u_p[0])
        $users_registrados[$users_registrados.length] = Usuario.new(u_p[0],u_p[1]) #Agrega a final de lista global a objeto tipo Usuario
        archivo = File.new("users.txt","a+") #Abre el archivo "users", si no existe lo crea
        archivo.puts texto
        archivo.close
        return true
    else
        return false
    end
end
```

FUNCIÓN DE LOGIN

Para esta función es necesario tener una función auxiliar que permita saber si existe el usuario en el sistema, de ser verdadera valida la contraseña que coincida con el usuario y le permite entrar.

```
    def self.user_existe(user) # Verifica que un usuario exista
        cont = 0 #Contador para recorrer lista de usuarios registrados
        while cont < $users_registrados.length #Verifica que el contador sea menor a largo de la lista de usuarios registrados
            if $users_registrados[cont].getUser == user  #Verifica que usuario sea igual al usuario del sistema
                return true #En caso de serlo, retorna true
            end
            cont = cont + 1 #Aumenta contador
        end
        false #Retorna falso en caso de no existir ningun usuario igual en el sistema
    end

    def self.LogUser(texto) # Verifica que el usuario y contrasenha sean validos
        texto = texto.gsub(" ","") #Quita todos los espacios " "
        texto = texto.gsub("-",",") #Sustituye "-" por ","
        u_p = texto.split(",") #Crea una lista con el formato [user,pass]
        e_user = u_p[0] #Agarra usuario
        e_pass = u_p[1] #Agarra Pass
        cont = 0 #Contador para recorrer lista de usuarios registrados
        while cont < $users_registrados.length #Verifica que el contador sea menor a largo de la lista de usuarios registrados
            if $users_registrados[cont].getUser == e_user #Verifica que usuario sea igual al usuario del sistema
                if $users_registrados[cont].getPass == e_pass #Verifica que la contrasenha sea igual a la del usuarios del sistema
                    return true #En caso de serlo, retorna true
                end
                return false #En caso de no ser la contrasenha, devuelve false sin seguir verificando mas usuarios
            end
            cont = cont + 1 #Aumenta contador
        end
        false #Retorna falso en caso de no existir ningun usuario igual en el sistema
    end
end
```

CLASE PREGUNTA

Esta clase permite al usuario realizar preguntas y desplegar la clase con todas las preguntas realizadas.

```
class Pregunta##########################################################################
    def initialize(preg,user)
        @IDPregunta = $ID + 1
        @ObjPregunta = preg
        @ObjRes = []
        @tags = []
        @user = user
    end

    def getID()
        @IDPregunta
    end

    def getPregunta()
        @ObjPregunta
    end

    def getRes()
        @ObjRes
    end

    def getUser()
        @user
    end

    def setRes(res)
        @ObjRes[@ObjRes.length] = res
    end

    def agregaTag(tag)
        @tags[@tags.length] = tag
    end
```

FUNCIÓN CREAR NUEVA PREGUNTA

Básicamente lo que realiza este método lo que dice en el título de la función, crea una nueva pregunta creada por el usuario

```
def self.crearPregunta(preg,user) #crea una nueva pregunta y asi
    res = Pregunta.new(preg,user)
    $all_preguntas[$all_preguntas.length] = res
end
```

FUNCIÓN QUE AGREGA ETIQUETA

Busca el ID de la pregunta y obtiene las etiquetas de la pregunta y las agrega.

```
    def self.agregaTagPregunta(tag,id)
        cont = 0
        while $all_preguntas.length > cont
            if $all_preguntas[cont].getID = id

                res = "#{res}/%/#{$all_preguntas[cont].getID()}/%/#{$all_preguntas[cont].getPregunta()}"
            end
            cont+=1
        end
    end
end
```

CLASE RESPUESTA

Permite a los usuarios publicar respuestas para las preguntas que ya han sido realizadas

```
class Respuesta#############################################################################################
    def initialize(res,user)
        @respuesta = res
        @user = user
    end

    def getRes()
        @respuesta
    end

    def getUser()
        @user
    end

    def self.responde_pregunta(res,id,user)
        cont = 0
        while cont < $all_preguntas.length
            if id = $all_preguntas[cont].getID
                $all_preguntas[cont].setRes(Respuesta.new(res,user))
            end
            cont+=1
        end
    end

    def self.listaRespuestas(id)
        cont = 0
        str = "%%%" #separador de preguntas
        while cont < $all_preguntas.length
            if id = $all_preguntas[cont].getID
                l_res = $all_preguntas[cont].getRes
                cont = 0
                while l_res.length > cont
                    str = "#{str}%%%#{l_res[cont].getUser}/%/#{l_res[cont].getRes()}"
                    cont+=1
                end
                return str
            end
            cont+=1
        end
        str
    end
```

CLASE PRUEBA

Es donde está Sinatra, carga el .txt También es donde está almacenado todo lo del servidor y esto permite hacer consultas.

```
require 'sinatra'
require "./Logica.rb"
require 'json'

Usuario.cargar_txt

get '/' do
    "Foro FdzCM!"
end

####################Modulo de Registro y Login##############################

get '/registrar/:name' do #recibe usuario-contrasenna. Ejm: fdz-123
    res = Usuario.registra_user(params[:name])
    "#{res}"
end

get '/existe/:name' do #recibe usuario. Ejm: fdz link/existe/pacm
    # if Usuario.user_existe(params[:name]) == true
    #   return "Usuario si existe!"
    # end
    # if Usuario.user_existe(params[:name]) == false
    #   return "Usuario no existe!"
    # end
    return "#{Usuario.user_existe(params[:name])}"
end

get '/login/:user_pass' do #recibe usuario-contrasenna. Ejm: fdz-123
    return "#{Usuario.LogUser(params[:user_pass])}"
end

get '/users' do
    response = {}
    cont = 0
    while cont < $users_registrados.length
        response[:user] = $users_registrados[cont].getUser
        response[:pass] = $users_registrados[cont].getPass
        cont = cont + 1
    end
    return $users_registrados.to_json
    #return response.to_json
end
```

ANDROID

CLASE LOGIN

Esta clase es el primer activity que muestra la aplicación, permite al usuaro iniciar sesión o registrar un usuario nuevo, es una interfaz sencilla que permite al usuario realizar estas funciones sin errores.

```
package com.example.pablo.askforf;
//Librerías importadas y asi
import android.app.Activity;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.AsyncTask;
import android.os.Bundle;
import android.util.Log;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.content.Intent;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import org.apache.http.HttpResponse;
import org.apache.http.client.HttpClient;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.impl.client.DefaultHttpClient;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;


public class Login extends Activity {
    static String res = "";
    Button   LogInButton; //crea variable para buton de loggeo qye permite entrar
    EditText userT; //variable que permite tomar el texto en el campo de usuario
    EditText passT;    //variable que permite tomar el texto en el campo de contraseña
```

MÉTODO CARGAR

Este método permite validar los datos que ingresa el usuario, en los espacios usuario y contraseña y de ser true, le permite ingresar a la aplicación, tal y como muestra en la documentación interna.

```
//aquí empieza la magia
    //toma los datos ingresados en usuario y contraseño y los compara en heroku, si es true se loggea sino da error
    public void cargar(View view) throws IOException {
        EditText cajaUser = (EditText) findViewById(R.id.userText);
        EditText cajaPass = (EditText) findViewById(R.id.passText);
        String usuario = cajaUser.getText().toString();
        String passw = cajaPass.getText().toString();
        new HttpAsyncTask().execute("https://foro-fdz-cm.herokuapp.com/login/" + usuario + "-" + passw);
        System.out.println("Resultado:"+res);
        if(res.equals("true")){ //compara el res para conocer si es true
            res = "";
            System.out.println("Entro al TRUE");
            Intent intent = new Intent(this, MainActivity.class);
            Button MainAct = (Button) findViewById(R.id.login);
            startActivity(intent);
        }else{
            res = "";
            Toast.makeText(this,"Usuario o contraseña incorrecta",Toast.LENGTH_LONG).show();
        }
    }
```

MÉTODO GET

Permite establecer la conexión entre la aplicación y Heroku.

```
public static String GET(String url){ //establece la conexión entre Heroku y la aplicación
    InputStream inputStream = null;
    String result = "";
    try {

        // crea el HttpClient
        HttpClient httpclient = new DefaultHttpClient();

        // hace GET request a la dirección
        HttpResponse httpResponse = httpclient.execute(new HttpGet(url));

        // receive response as inputStream
        inputStream = httpResponse.getEntity().getContent();

        // convert inputstream to string
        if(inputStream != null)
            result = convertInputStreamToString(inputStream);
        else
            result = "Did not work!";

    } catch (Exception e) {
        Log.d("InputStream", e.getLocalizedMessage());
    }

    return result;
}
```

MÉTODO REGISTROUSER

Llama al activity de registro de usuario.

```
    //llama a la ventana de registro de usuario
    public void registroUser(View v) throws IOException {
        Intent intent = new Intent(this, Registro.class);
        startActivity(intent);
    }
}
```

CLASE REGISTRO

Permite registrar usuario, se utiliza una reutilización de código(valga la redundancia) de la clase Login, para establecer conexión e intercambiar datos.

```
package com.example.pablo.askforf;

import ...


public class Registro extends Activity {
    static String res = "";
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_registro);
    }


    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.registro, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action bar will
        // automatically handle clicks on the Home/Up button, so long
        // as you specify a parent activity in AndroidManifest.xml.
        int id = item.getItemId();
        if (id == R.id.action_settings) {
            return true;
        }
        return super.onOptionsItemSelected(item);
    }
```

MÉTODO REGISTRAR

Realiza el registro de un nuevo usuario, en caso de que exista le notifica que ese nombre de usuario ya existe. Y de tener éxito vuelve al activity de Login

```
public void registrar(View v) throws IOException { //permite registrar el nuevo usuario en el backend de Heroku
    EditText cajaUser = (EditText) findViewById(R.id.userRText);
    EditText cajaPass = (EditText) findViewById(R.id.passRText);
    String usuario = cajaUser.getText().toString();
    String passw = cajaPass.getText().toString();
    new HttpAsyncTask().execute("https://foro-fdz-cm.herokuapp.com/registrar/"+usuario+"-"+passw);
    if (res.equals("true")){
        Toast.makeText(this, "Registro exitoso", Toast.LENGTH_LONG).show();
        Intent intent = new Intent(this, Login.class);
        startActivity(intent);
    }else{
        Toast.makeText(this,"Usuario ya existe",Toast.LENGTH_LONG).show();
    }
}
```

LIBRERÍAS USADAS

Sinatra para la parte del backend en Ruby, lo cual permite el manejo de la parte web de la aplicación.

```
require 'sinatra'
require "./Logica.rb"

Usuario.cargar_txt

get '/' do
    "Foro FdzCM!"
end

#####################Modulo de Registro y Login##########
```

LIBRERÍAS ANDROID

Estas librerías se utilizaron para el manejo de activities, conectividad, información de red, para el activity de loggeo, desplegar el menú, crear botones entre otras funciones.

```
package com.example.pablo.askforf;
//Librerías importadas y así
import android.app.Activity;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.AsyncTask;
import android.os.Bundle;
import android.util.Log;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.content.Intent;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
```

LIBRERÍAS HTTP

Básicamente permitió generar la conexión entre Heruku y la aplicación.

```
import org.apache.http.HttpResponse;
import org.apache.http.client.HttpClient;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.impl.client.DefaultHttpClient;
```

LIBRERÍAS JAVA

Usadas para leer inputs y pasarlos a strings, entre otras cosas.

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
```

ANÁLISIS DE RESULTADOS

- La utilización del lenguaje de programación Ruby fue una decisión acertada ya que nos permitió agilizar el proceso de trabajo.
- Los requisitos solicitados por el profesor se fueron abarcando por partes y distribuyendo entre los integrantes del grupo con el fin de poder interiorizar el problema y la posible solución a desarrollar.
- El trabajo grupal gestionado por GitHub fue un aspecto positivo que facilitó el manejo de los archivos y documentos necesarios para el desarrollo del trabajo.
- La distribución total del trabajo en el equipo tuvo aspectos positivos y negativos ya que se realizaron funciones individualmente que agilizaron el proceso de construcción de la tarea pero que dificultaron la unión y complementación de unas partes con otras.
- El funcionamiento general de la aplicación no es en su totalidad satisfactorio ya que no logramos pulir el código fuente de manera tal que se mejorara su efectividad y rendimiento en ejecución así como la robustez y estabilidad del mismo. Además no se lograron concluir con todas las funcionalidades del programa, sin embargo se logró lo más importante.
- Haber aprendido a programar aplicaciones para Android, nos permitió adquirir conocimientos que ayudan al desarrollo como profesionales de cada miembro del equipo.

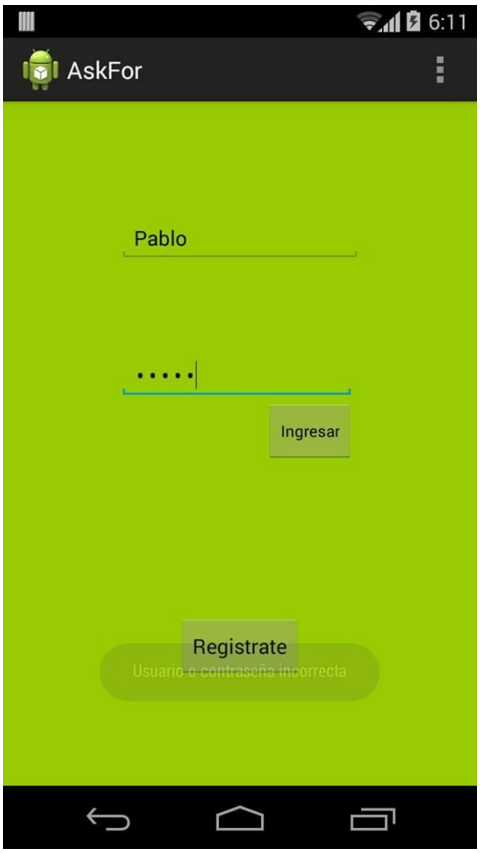
MANUAL DE USUARIO

Para ejecutar la aplicación se requiere tener un teléfono móvil con sistema operativo Android 4.0(Ice Cream Sandwich) o superior, en caso de no poseer, se le recomienda al usuario instalar Genymotion que le permitirá desde su computadora virtualizar un celular con el sistema operativo.

A continuación se mostrarán una serie de imágenes que permitirá guiar al usuario a través de la aplicación utilizando sus diferentes funcionalidades.

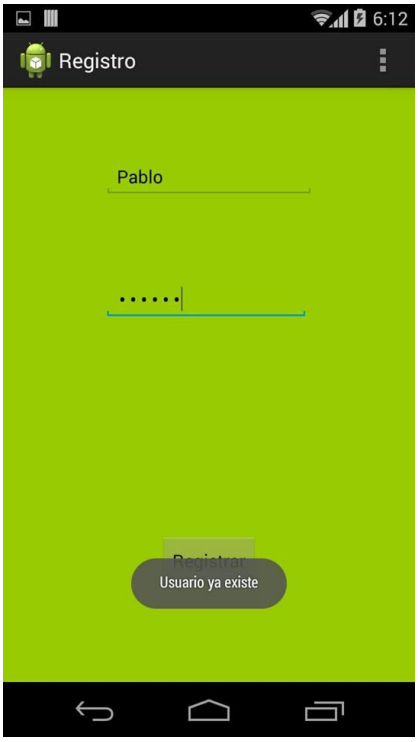
LOGIN

Es la primer ventana de la aplicación, en esta el usuario debe ingresar su nombre de usuario y contraseña tal y como se muestra a continuación.



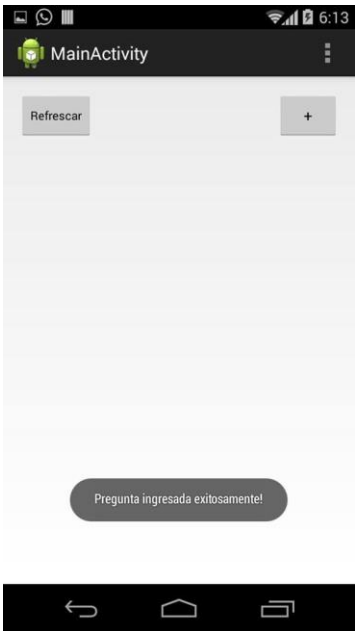
REGISTRO

El usuario si no tiene un nombre de usuario, deberá registrarse en el sistema.



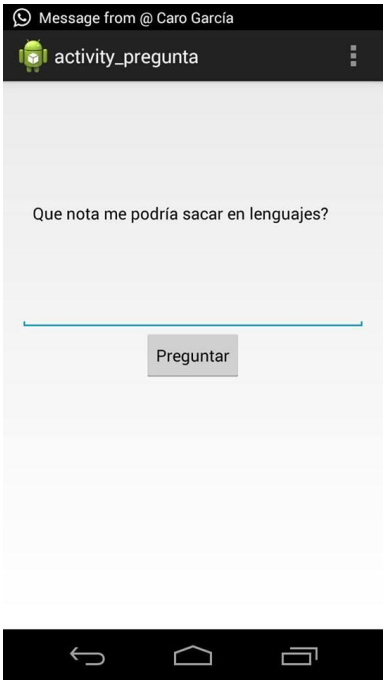
FORO

En esta ventana se debería de mostrar las preguntas realizadas por todos los usuarios.



NUEVA PREGUNTA

Aquí el usuario agrega una nueva pregunta al foro.



CONCLUSIÓN PERSONAL

El trabajo realizado nos permitió expandir nuestros conocimientos en el área de los lenguajes de la programación ya que para poder desarrollar un lenguaje de programación utilizando otro similar es necesario conocer aspectos relevantes sobre ambos, es decir se debe manejar el funcionamiento del lenguaje a desarrollar, en nuestro caso Prolog y de igual manera el lenguaje en el cual se desarrolló (Python).

Nos permitió conocer y familiarizarnos con los diferentes tipos de paradigmas de programación, lo cual es de suma importancia para el desarrollo de la lógica del programador así como de las habilidades y aptitudes requeridas para implementar una solución computacional

Esta tarea además de fomentar el trabajo grupal, permitió conocer aspectos claves, de gran importancia y utilidad sobre el lenguaje Prolog, permitiéndonos como estudiantes ampliar nuestros conocimientos en el área informática y mantener un punto de comparación más elaborado respecto a otros lenguajes. El conocimiento de este lenguaje es de suma importancia, ya que nos permite tener una noción de lo que es el comportamiento Prolog, su funcionalidad, características, limitaciones, usos, facilidades y demás aspectos negativos como positivos que poseen todos los lenguajes de programación utilizados en alrededor del mundo.

BIBLIOGRAFÍA

A manera de referencia se utilizaron diferentes páginas web en las cuales se explicaba el uso y aplicación tanto de Ruby con Sinatra, como con Android(java) con Android Studio.

Android - Get Value of a Edit Text field - Stack Overflow (s. f.). Recuperado de http://stackoverflow.com/questions/4531396/get-value-of-a-edit-text-field

Android - Menú de opciones y ActionBar - YouTube (s. f.). Recuperado de https://www.youtube.com/watch?v=3CPCI4boc8o

Android Internet Connection Using HTTP GET (HttpClient) | HMKCode (s. f.). Recuperado de http://hmkcode.com/android-internet-connection-using-http-get-httpclient/

Android Studio [webview] - Mostrar pagina web (HTML5, PHP, ETC) dentro de aplicacion Android APK - YouTube (s. f.). Recuperado de https://www.youtube.com/watch?v=kzLmjGBFkcY

Building a Simple User Interface | Android Developers (s. f.). Recuperado de http://developer.android.com/training/basics/firstapp/building-ui.html

Design | Android Developers (s. f.). Recuperado de http://developer.android.com/design/index.html

Getting Started: WebView-based Applications for Web Developers - Google Chrome (s. f.). Recuperado de https://developer.chrome.com/multidevice/webview/gettingstarted

How to get a web page's source code from Java - Stack Overflow (s. f.). Recuperado de http://stackoverflow.com/questions/8616781/how-to-get-a-web-pages-source-code-from-java

Java equivalent of Pythons urllib.urlencode(HashMap based UrlEncode) - Stack Overflow (s. f.). Recuperado de http://stackoverflow.com/questions/2088502/java-equivalent-of-pythons-urllib-urlencodehashmap-based-urlencode

Javascript - Get the HTML code from loaded WebView - Stack Overflow (s. f.). Recuperado de http://stackoverflow.com/questions/19518950/get-the-html-code-from-loaded-webview

Pérez, B. (2014). Tupera (Versión Alnitak) [Software]. San José, Costa Rica.

Simple android app + database: what do I need in my android application? (PostgreSQL, REST API) - Stack Overflow (s. f.). Recuperado de http://stackoverflow.com/questions/17243633/simple-android-app-database-what-do-i-need-in-my-android-application-postgr

Tutorial 04 Programación Android: Conectar nuestro teléfono a Android Studio - YouTube (s. f.). Recuperado de https://www.youtube.com/watch?v=u3Zgf9dTY24

Urllib for java (s. f.). Recuperado de https://gist.github.com/tonetheman/85815

Using an ArrayAdapter with ListView · codepath/android_guides Wiki · GitHub (s. f.). Recuperado de https://github.com/codepath/android_guides/wiki/Using-an-ArrayAdapter-with-ListView