# 

[**PERSISTENCIA EN API REST**](#_7q4p9gt3rdeo) **2**

[**API REST Y MÉTODOS HTTP**](#_vt2sueq3le5c) **2**

[**CONOCIENDO ALGUNAS APIS**](#_m8aj65to9roc) **2**

[**CREACIÓN DEL PROYECTO Y API KEY**](#_8d2osx2mkdw6) **2**

[**LISTANDO DATOS DE APIS PÚBLICAS**](#_x4nqsf4fc5we) **2**

[**MOSTRAR DATOS DE LA API EN LA APLICACIÓN**](#_4zn7ft3040e) **2**

[**GUARDANDO DATOS EN APIS PÚBLICAS**](#_g10iwtfw9y4y) **2**

[**LISTAR DATOS GUARDADOS**](#_rmh1rvemoiyh) **2**

[**MOSTRAR FAVORITOS**](#_e1egjo6ua5rp) **2**

[**ELIMINAR DATOS DE LA API**](#_ktccvalyhg9u) **2**

[**BIBLIOGRAFÍA**](#_opkozhu5mhwa) **2**

# 

# **PERSISTENCIA EN API REST**

# **API REST Y MÉTODOS HTTP**

Una API, o Interfaz de Programación de Aplicaciones, es un método de comunicación que permite a tercero (nosotros) comunicarse entre sí para transferir información y realizar las 4 operaciones básicas usando métodos HTTP.

Las API funcionan bajo el protocolo HTTP que es con el que funcionan los sitios web, estos usan unos métodos o verbos HTTP que sirven para hacer las Cuatro Operaciones.

El método GET sirve para obtener información de una API

El método POST, envía información o la guarda

El método PUT o PATCH, actualiza o edita información

El método DELETE, borra información

Las API funcionan en diferentes tipos de formatos.

* JSON, es el estándar para las API RES
* XML se usaron en otros tipos de APIS en años anteriores.

Por su parte, API REST es una API diseñada bajo los principios de REST.

# **CONOCIENDO ALGUNAS APIS**

Herramientas:

A Continuación el enlace de para poder instalar postman junto con un video explicativo de la instalación.

[Postman | The Collaboration Platform for API Development](https://www.getpostman.com/)

<https://www.getpostman.com/>

Video:

<https://www.youtube.com/watch?v=laZv3I9XN0w>

APIS:

Facebook:

[API y SDK - Desarrollo de aplicaciones - Documentación - Facebook para desarrolladores](https://developers.facebook.com/docs/apis-and-sdks?locale=es_ES)

<https://developers.facebook.com/docs/apis-and-sdks?locale=es_ES>

Twitter:

<https://developer.twitter.com/>

<https://developer.twitter.com/>

TheCatAPI:

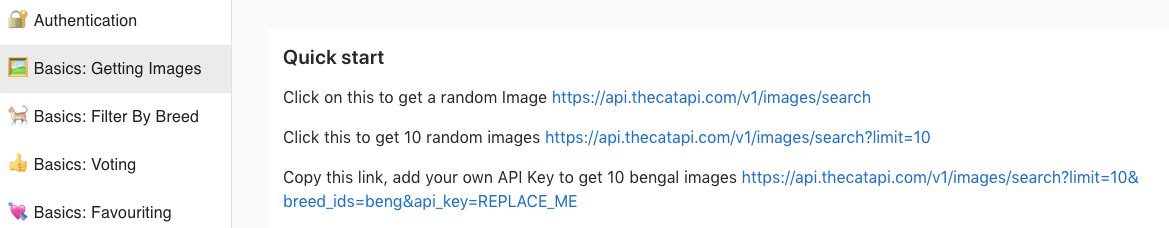
[TheCatAPI - Cats as a Service, Everyday is Caturday.](https://thecatapi.com/)

<https://thecatapi.com/>

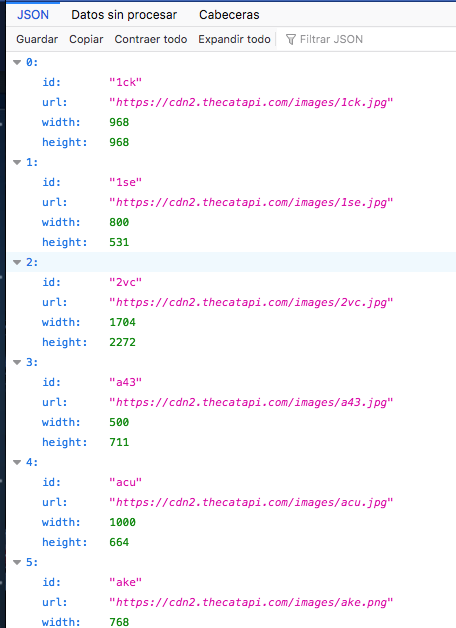
USANDO LA API CAT

Vamos a realizar el ejercicio con los 10 primero gatos de la AP, para esto hacemos los siguientes pasos:

1. En la página de la API
2. Ingresamos a la documentación
3. Clic en basic 10 random

[](https://www.getpostman.com/)

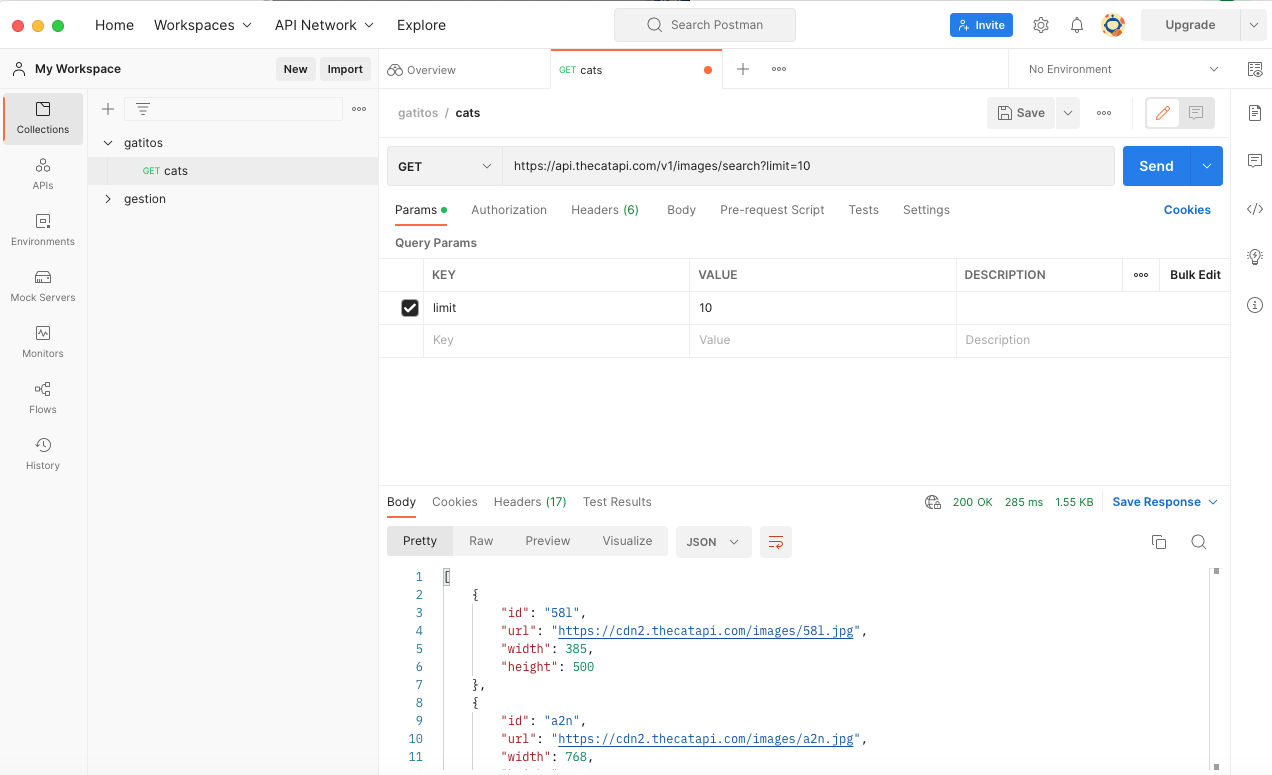
Si damos clic nos trae imágenes aleatorias.



Abrimos postman y pegamos el url de los gatos:

<https://api.thecatapi.com/v1/images/search?limit=10>

Después pasamos por postan la petición para ver si cargan en la página.



Luego nos dirigimos a code en postman y damos clic en el lenguaje desprogramación para realizar la conexión con la API.

| OkHttpClient client = **new** OkHttpClient().newBuilder()  .build();  MediaType mediaType = MediaType.parse("text/plain");  RequestBody body = RequestBody.create(mediaType, "");  Request request = **new** Request.Builder()  .url("https://api.thecatapi.com/v1/images/search?limit=10")  .method("GET", body)  .build();  Response response = client.newCall(request).execute(); |
| --- |

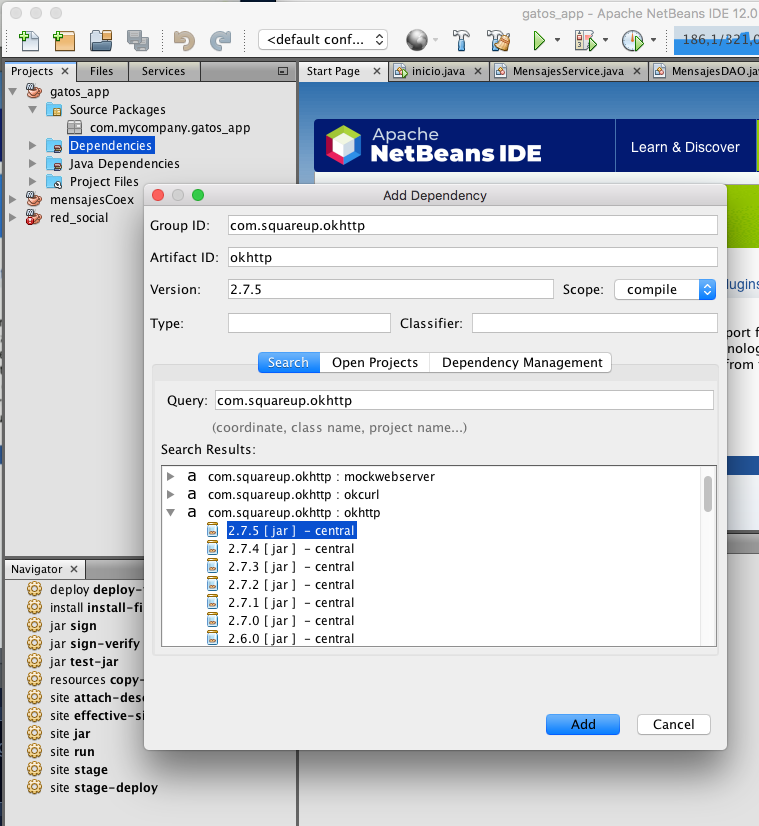
# **CREACIÓN DEL PROYECTO Y API KEY**

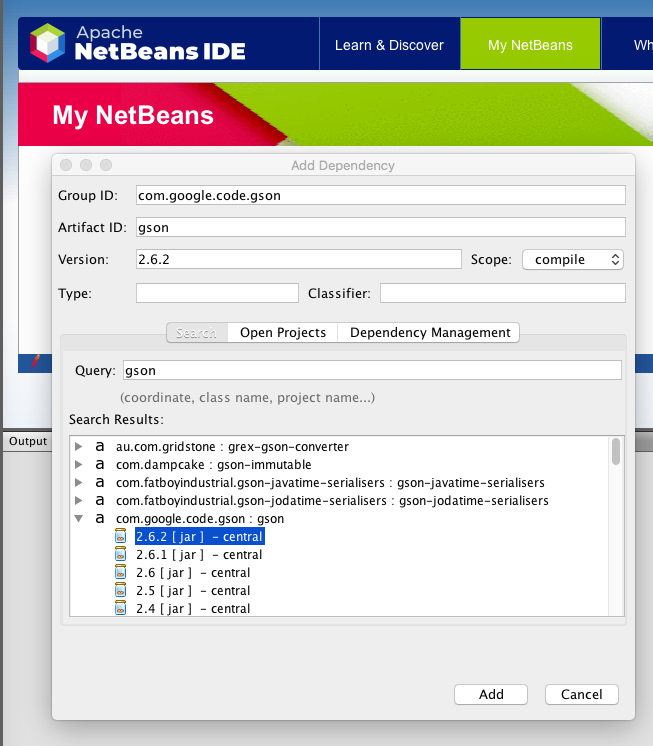
Debemos registrarnos por la página de la API , para que esta solo quede con nuestra información.

Al correo se enviará un código key con el que nos conectaremos directamente a esa API.

El siguiente paso es ir a netbeans y crear el proyecto nuevo.

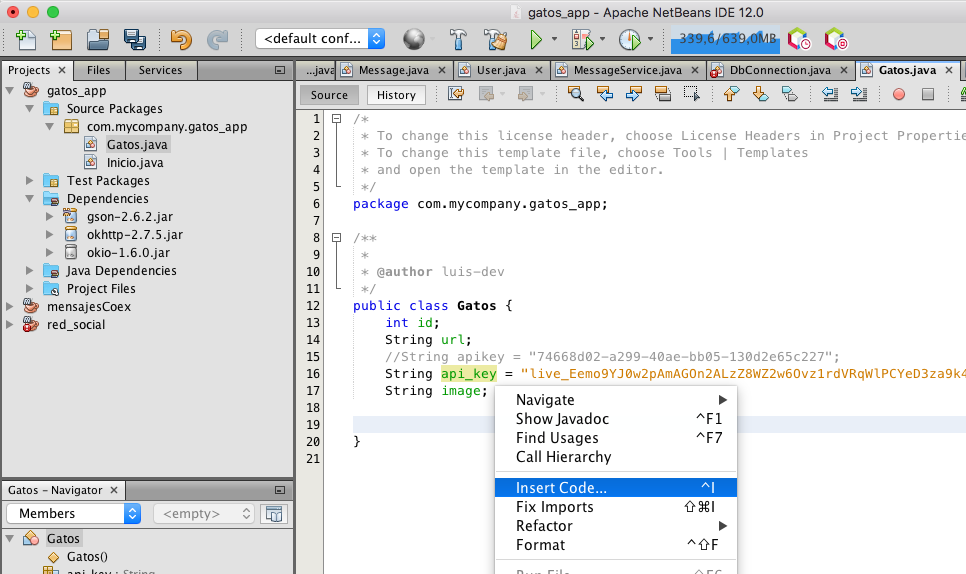
Luego importamos las dependencias okHTTP y la dependencia GSON.





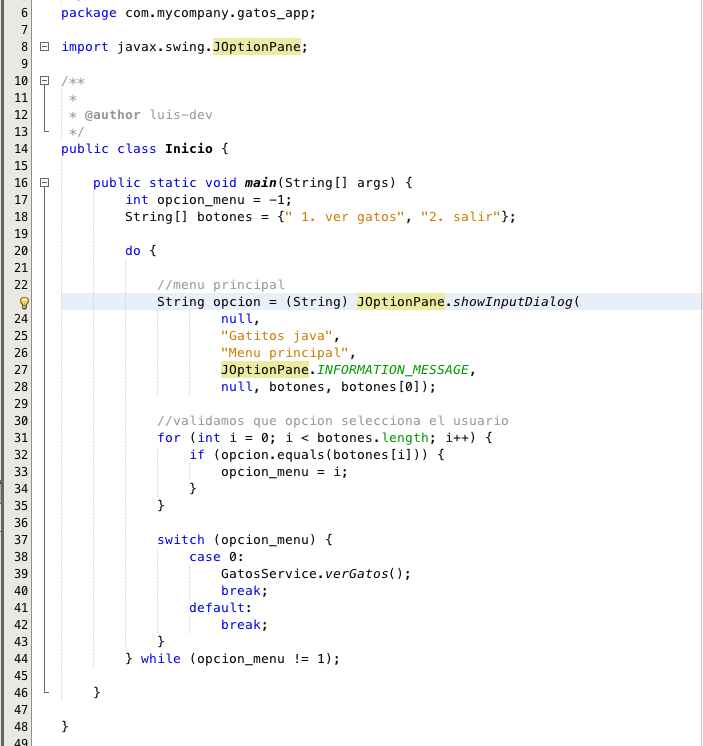
Luego de importar las dependencias vamos a crear las clases Gatos e Inicio , donde vamos a crear los atributos de los gatos y los comportamientos establecidos del mismo.

| package com.mycompany.gatos\_app;  /\*\*  \*  \* @author luis-dev  \*/  public class Gatos {  int id;  String url;  //String apikey = "74668d02-a299-40ae-bb05-130d2e65c227";  String api\_key = "live\_Eemo9YJ0w2pAmAGOn2ALzZ8WZ2w6Ovz1rdVRqWlPCYeD3za9k4iDOZKj1YyALZ1e";  String image;  public int getId() {  return id;  }  public void setId(int id) {  this.id = id;  }  public String getUrl() {  return url;  }  public void setUrl(String url) {  this.url = url;  }  public String getApi\_key() {  return api\_key;  }  public void setApi\_key(String api\_key) {  this.api\_key = api\_key;  }  public String getImage() {  return image;  }  public void setImage(String image) {  this.image = image;  }      } |
| --- |



Después de crear los getter y setter de nuestra clase gatos vamos a realizar el menu de nuestro programa .

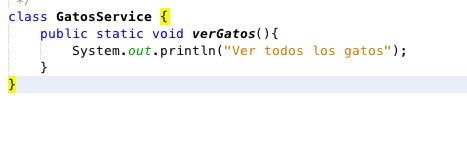
| package com.mycompany.gatos\_app;  import javax.swing.JOptionPane;  /\*\*  \*  \* @author luis-dev  \*/  public class Inicio {  public static void main(String[] args) {  int opcion\_menu = -1;  String[] botones = {" 1. ver gatos", "2. salir"};  do {  //menu principal  String opcion = (String) JOptionPane.showInputDialog(  null,  "Gatitos java",  "Menu principal",  JOptionPane.INFORMATION\_MESSAGE,  null, botones, botones[0]);  //validamos que opcion selecciona el usuario  for (int i = 0; i < botones.length; i++) {  if (opcion.equals(botones[i])) {  opcion\_menu = i;  }  }  switch (opcion\_menu) {  case 0:  GatosService.verGatos();  break;  default:  break;  }  } while (opcion\_menu != 1);  }  } |
| --- |



# 

| **LÍNEA** | **EXPLICACIÓN** |
| --- | --- |
| **8** | Importamos el módulo de jOptionPane para mostrar de forma gráfica el menú |
| **17/18** | Se crean las variables opción menú y botones en forma de arreglo para luego ser solicitados por el usuario |
| **20** | iniciamos el ciclo do while , mientras que la opción menú sea diferente de 1 el sistema seguirá corriendo. |
| **23** | Se crea la opción opción que crea un objeto string que al momento de ser elegido traiga un mensaje como nombre menú principal gatitos java. |
| **24** | Se utiliza el joptionpane mensaje donde se mostrará el arreglo del menú de los botones . |
| **28** | bucle for para recorrer el arreglo según la longitud del mismo. |
| **29** | si la opción es igual a botones este será es que el comparativo y guardará la información en la variable opcion-menu. |
| **34** | Se crea un switch para verificar lo que está almacenado en la variable opción menú y entrar en el condicional switch. |
| **36** | Si el usuario digita la opción 1 se carga el método de la clase gatos service que este llamara el comportamiento de ver los gatos de la API. |
| **38** | Si no entra al condicional se rompe y sale . |

# **Se crea la clase Gato Service y en esta se crea un método para ver los gatos.**

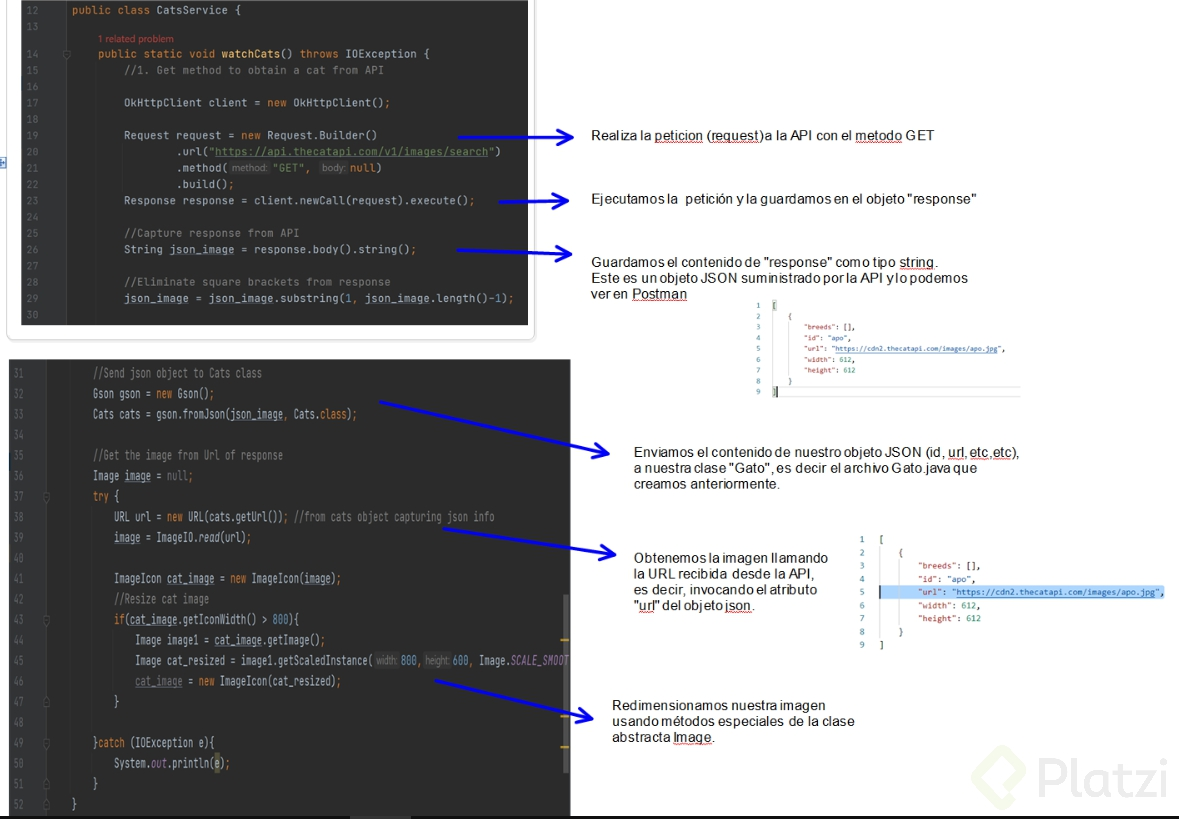


# **LISTANDO DATOS DE APIS PÚBLICAS**

Recuerden que lo que hacemos es enviar una petición a la API mediante el método GET.

La API nos da una respuesta en formato JSON, y de esta respuesta sacamos la URL de la imagen, para llamarla y guardarla en una variable.

Luego, redimensionar la imagen obtenida, usando métodos especiales de la clase abstracta “Image”.

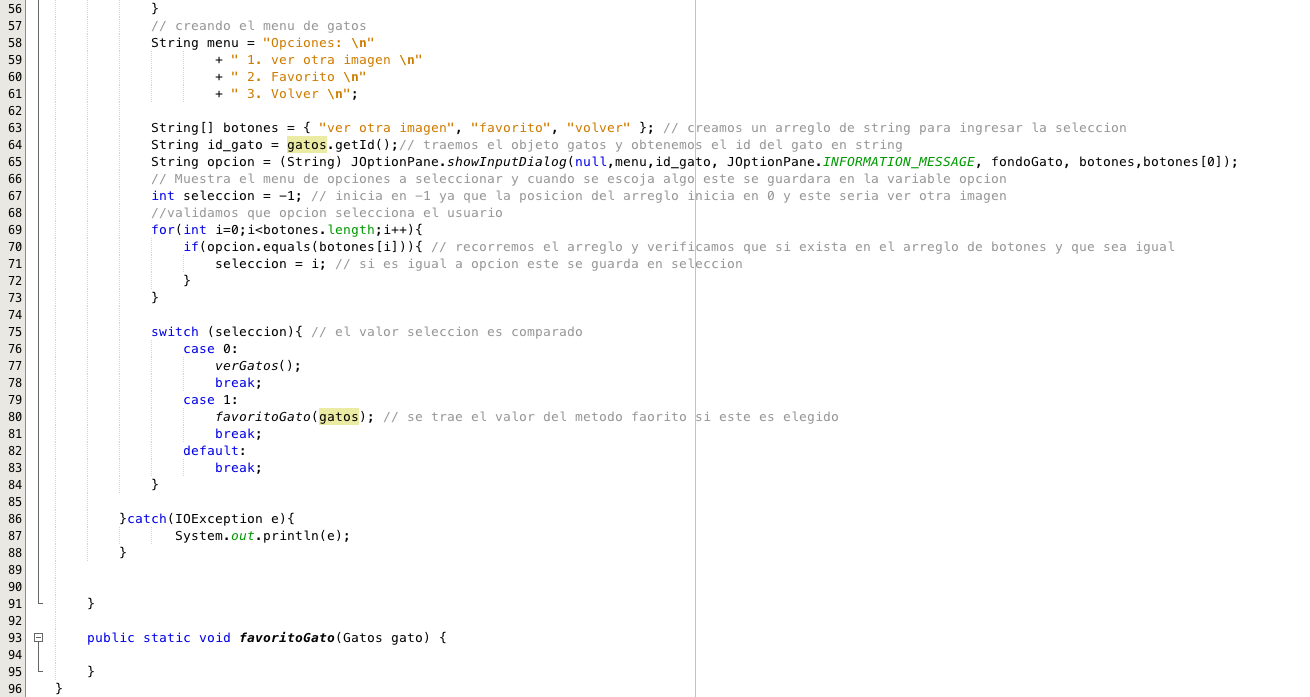


| package com.mycompany.gatos\_app;  import com.google.gson.Gson;  import com.squareup.okhttp.OkHttpClient;  import com.squareup.okhttp.Request;  import com.squareup.okhttp.Response;  import java.awt.Image;  import java.io.IOException;  import java.net.URL;  import javax.imageio.ImageIO;  import javax.swing.ImageIcon;  /\*\*  \*  \* @author luis-dev  \*/  class GatosService {      public static void verGatos() throws IOException{  //1. vamos a traer los datos de la API  OkHttpClient client = new OkHttpClient();  Request request = new Request.Builder().url("https://api.thecatapi.com/v1/images/search").get().build();  Response response = client.newCall(request).execute();    String elJson = response.body().string();    //cortar los corchetes  elJson = elJson.substring(1, elJson.length());  elJson = elJson.substring(0, elJson.length()-1);    //crear u objeto de la clase Gson  Gson gson = new Gson();  Gatos gatos = gson.fromJson(elJson, Gatos.class);    //redimensionar en caso de necesitar  Image image = null;  try{  URL url = new URL(gatos.getUrl());  image = ImageIO.read(url);    ImageIcon fondoGato = new ImageIcon(image);    if(fondoGato.getIconWidth() > 800){  //redimensionamos  Image fondo = fondoGato.getImage();  Image modificada = fondo.getScaledInstance(800, 600, java.awt.Image.SCALE\_SMOOTH);  fondoGato = new ImageIcon(modificada);  }    }catch(IOException e){  System.out.println(e);  }      }  } |
| --- |

# **MOSTRAR DATOS DE LA API EN LA APLICACIÓN**

Ahora vamos a ver cómo podemos crear un menú para nuestra aplicación de gatos.

| /\*  \* To change this license header, choose License Headers in Project Properties.  \* To change this template file, choose Tools | Templates  \* and open the template in the editor.  \*/  package com.mycompany.gatos\_app;  import com.google.gson.Gson;  import com.squareup.okhttp.OkHttpClient;  import com.squareup.okhttp.Request;  import com.squareup.okhttp.Response;  import java.awt.Image;  import java.io.IOException;  import java.net.URL;  import javax.imageio.ImageIO;  import javax.swing.ImageIcon;  import javax.swing.JOptionPane;  /\*\*  \*  \* @author luis-dev  \*/  public class GatosService {    public static void verGatos() throws IOException{  //1. vamos a traer los datos de la API  OkHttpClient client = new OkHttpClient();  Request request = new Request.Builder().url("https://api.thecatapi.com/v1/images/search").get().build();  Response response = client.newCall(request).execute();    String elJson = response.body().string();    //cortar los corchetes  elJson = elJson.substring(1, elJson.length());  elJson = elJson.substring(0, elJson.length()-1);    //crear u objeto de la clase Gson  Gson gson = new Gson();  Gatos gatos = gson.fromJson(elJson, Gatos.class);    //redimensionar en caso de necesitar  Image image = null;  try{  URL url = new URL(gatos.getUrl());  image = ImageIO.read(url);    ImageIcon fondoGato = new ImageIcon(image);    if(fondoGato.getIconWidth() > 800){  //redimensionamos  Image fondo = fondoGato.getImage();  Image modificada = fondo.getScaledInstance(800, 600, java.awt.Image.SCALE\_SMOOTH);  fondoGato = new ImageIcon(modificada);  }  // creando el menu de gatos  String menu = "Opciones: \n"  + " 1. ver otra imagen \n"  + " 2. Favorito \n"  + " 3. Volver \n";    String[] botones = { "ver otra imagen", "favorito", "volver" }; // creamos un arreglo de string para ingresar la seleccion  String id\_gato = gatos.getId();// traemos el objeto gatos y obtenemos el id del gato en string  String opcion = (String) JOptionPane.showInputDialog(null,menu,id\_gato, JOptionPane.INFORMATION\_MESSAGE, fondoGato, botones,botones[0]);  // Muestra el menu de opciones a seleccionar y cuando se escoja algo este se guardara en la variable opcion  int seleccion = -1; // inicia en -1 ya que la posicion del arreglo inicia en 0 y este seria ver otra imagen  //validamos que opcion selecciona el usuario  for(int i=0;i<botones.length;i++){  if(opcion.equals(botones[i])){ // recorremos el arreglo y verificamos que si exista en el arreglo de botones y que sea igual  seleccion = i; // si es igual a opcion este se guarda en seleccion  }  }    switch (seleccion){ // el valor seleccion es comparado  case 0:  verGatos();  break;  case 1:  favoritoGato(gatos); // se trae el valor del metodo faorito si este es elegido  break;  default:  break;  }    }catch(IOException e){  System.out.println(e);  }    }    public static void favoritoGato(Gatos gato) {    }  } |
| --- |

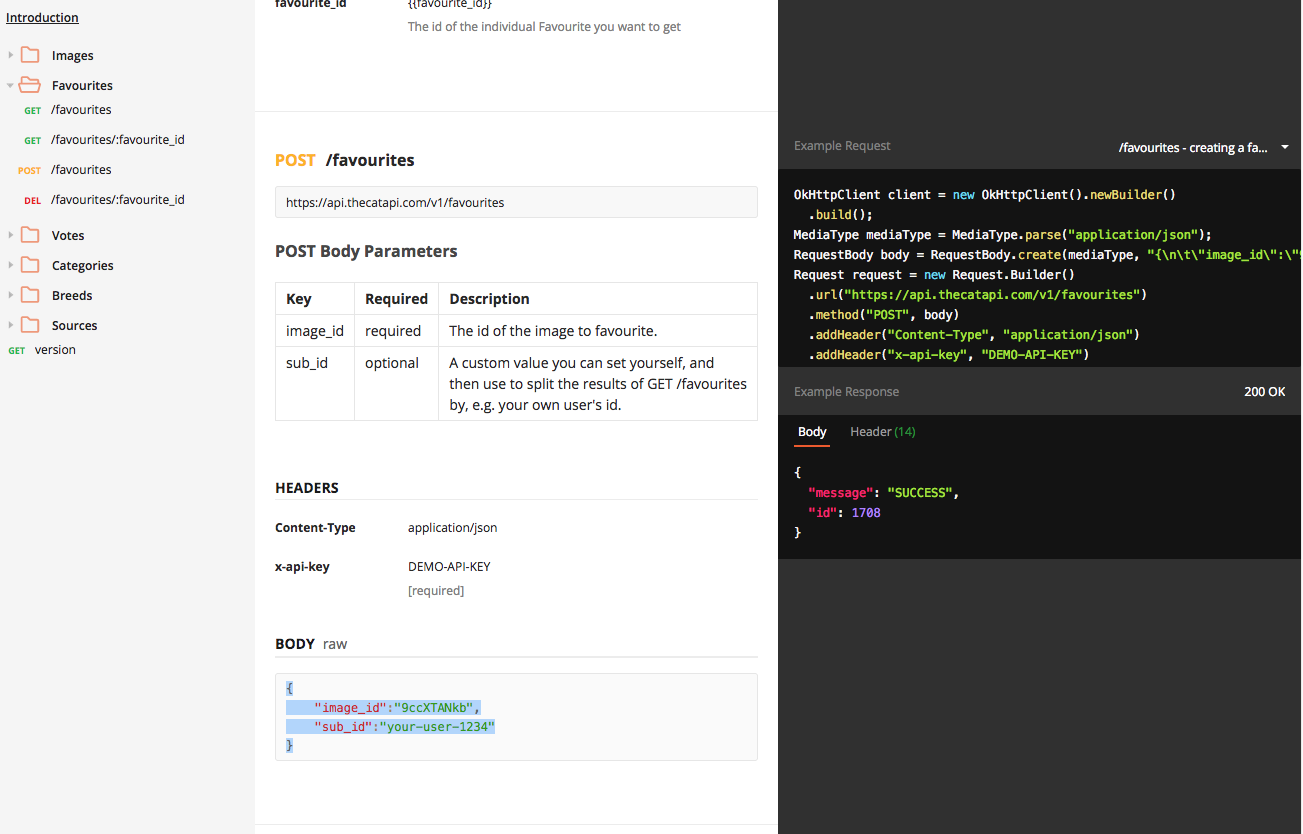


# **GUARDANDO DATOS EN APIS PÚBLICAS**

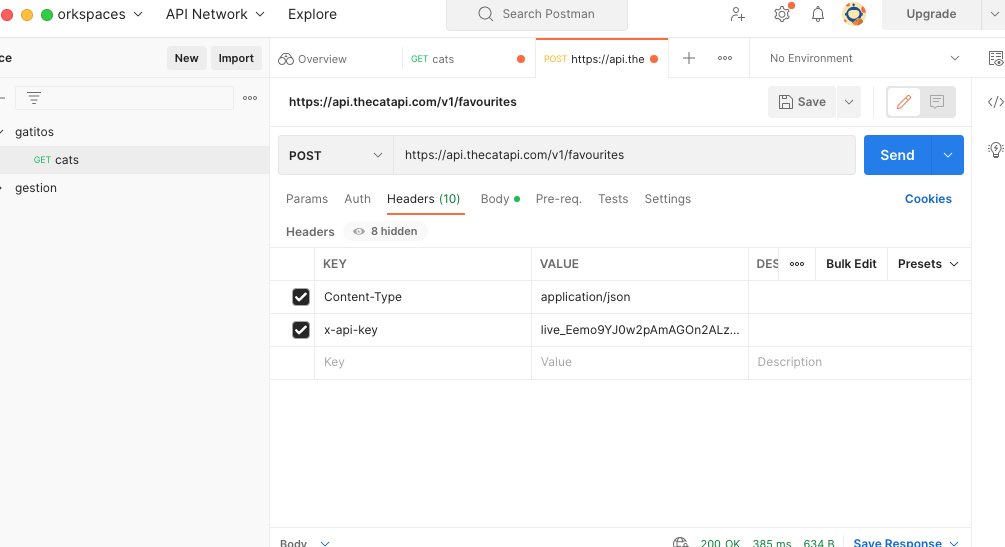
Enviar datos a una API ,

Realizaremos una petición POST para marcar como favorita una de las fotografías que se encuentran en la API de los gatos.

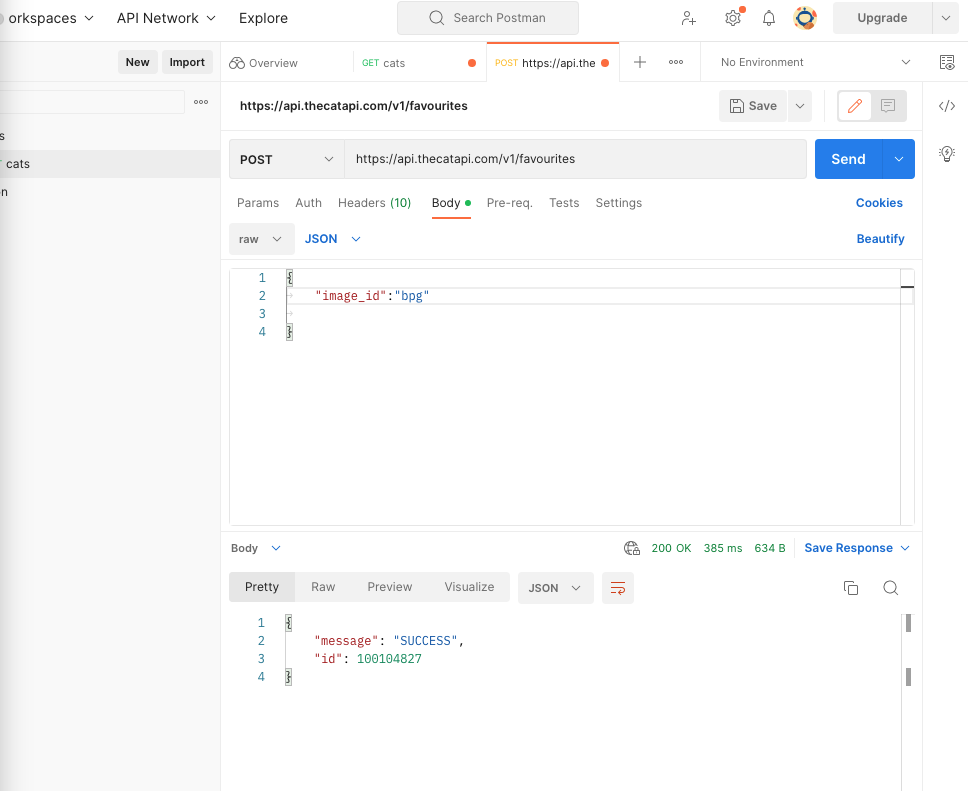
Primero debemos ingresar a nuestra pagina de api gatos y seguir los pasos uno a uno del ejemplo que se está solicitando .



luego en postman hacemos los ejemplos.



Pasamos el ejemplo :



vamos al código de nuestro netbeans y creamos el try catch para poder hacer la verificación de la petición Post .

para esto es necesario tener en cuenta la conexión anterior del endpoint de esta forma estaremos verificando que los datos por el método post estén funcionando correctamente.

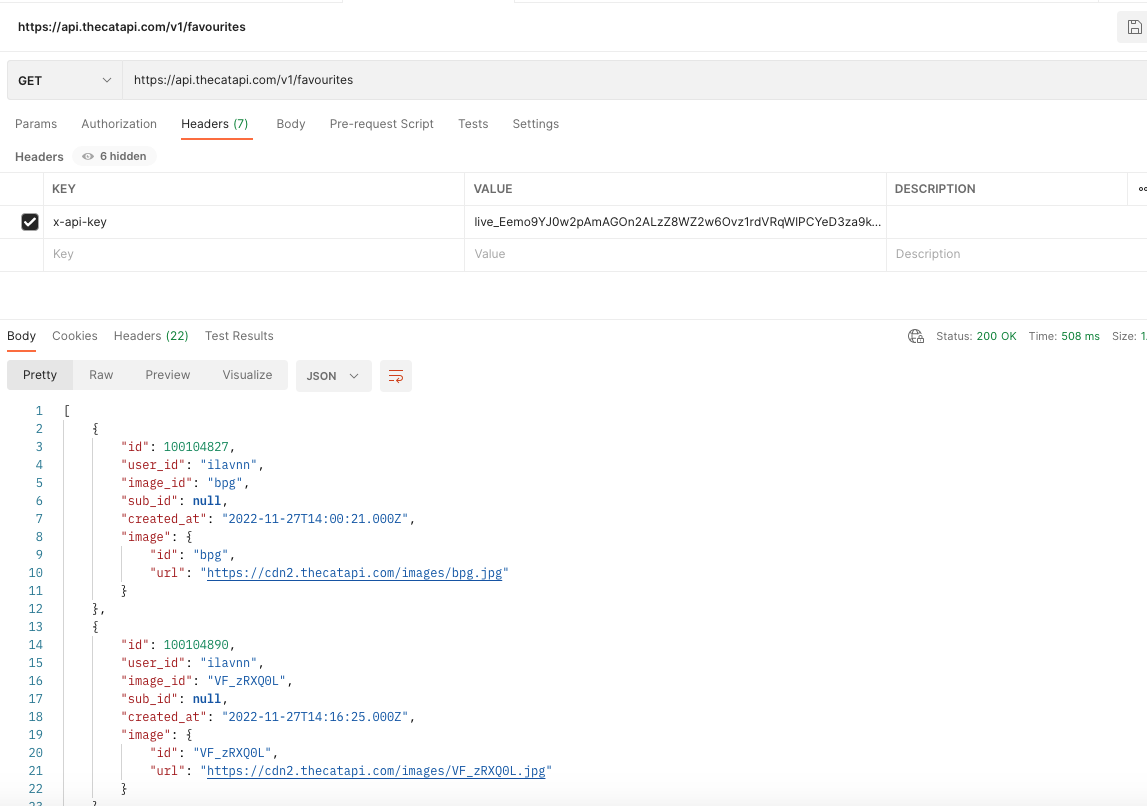
| public static void favoritoGato(Gatos gato) {  try {    OkHttpClient client = new OkHttpClient();  MediaType mediaType = MediaType.parse("application/json");  RequestBody body = RequestBody.create(mediaType, "{\n\t\"image\_id\":\"" + gato.getId() + "\"\n}");  Request request = new Request.Builder()  .url("https://api.thecatapi.com/v1/favourites")  .method("POST", body)  .addHeader("Content-Type", "application/json")  .addHeader("x-api-key", "live\_Eemo9YJ0w2pAmAGOn2ALzZ8WZ2w6Ovz1rdVRqWlPCYeD3za9k4iDOZKj1YyALZ1e")  .build();  Response response = client.newCall(request).execute();    } catch (IOException e) {  System.out.println(e);  }  }  } |
| --- |



De la línea 96 hasta la 105 pegamos los datos que viene del método POST que nos envía postman esto con el fin de realizar una conexión estandarizada.

en la línea 98 modificamos el valor de la api que se busco anteriormente con el fin de traer la que tenemos con el objeto gato.id().

realizamos las pruebas enviando la misma api de favoritos pero con el método GET y de esta forma nos mostrará los favoritos que tenemos almacenados en nuestra API.



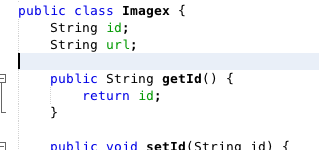
# **LISTAR DATOS GUARDADOS**

Ahora se va a crear un método para poder visualizar los datos de los gatos que tenemos como favoritos, para esto debemos realizar la modificación de los elementos del menú y crear un método para visualizar los datos.

Vamos a realizar la creación de las clases de GatosFav, imageX .



Creamos los métodos de la imagen para la manipulación de esta.



Edición del menú.



| LÍNEA | EXPLICACIÓN |
| --- | --- |
| 19 | Se agrega en el arreglo el número 2 de ver favoritos. |
| 38 | En el condicional se agrega un objeto nuevo gato , este se pasa como parámetro validando el api Key para ver los favoritos. |

Agregamos en la clase GatosService , un método verFavorito en este método pasamos como parámetros el string de la apiKey y en los comportamientos del método traemos la información que nos arroja el codigo de postdam donde visualizamos la información de todos los favoritos.

Para poder manipular la respuesta , la almacenamos en la variable de elJson .

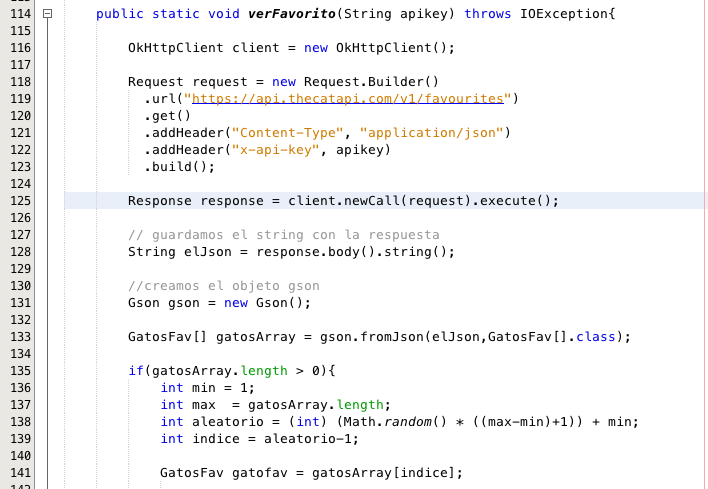


# **MOSTRAR FAVORITOS**

Ya teniendo nuestros gatos listados , vamos a mostrar por pantalla los datos de los favoritos que tenemos marcados y que se encuentran guardados.

para esto copiamos el método de ver gatos y los pegamos .

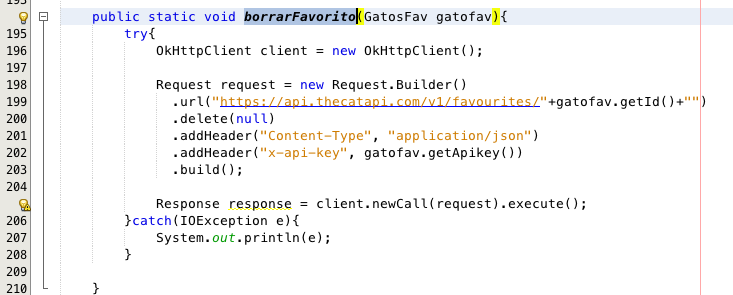
| public static void verFavorito(String apikey) throws IOException{    OkHttpClient client = new OkHttpClient();  Request request = new Request.Builder()  .url("https://api.thecatapi.com/v1/favourites")  .get()  .addHeader("Content-Type", "application/json")  .addHeader("x-api-key", apikey)  .build();  Response response = client.newCall(request).execute();    // guardamos el string con la respuesta  String elJson = response.body().string();    //creamos el objeto gson  Gson gson = new Gson();    GatosFav[] gatosArray = gson.fromJson(elJson,GatosFav[].class);    if(gatosArray.length > 0){  int min = 1;  int max = gatosArray.length;  int aleatorio = (int) (Math.random() \* ((max-min)+1)) + min;  int indice = aleatorio-1;    GatosFav gatofav = gatosArray[indice];    //redimensionar en caso de necesitar  Image image = null;  try{  URL url = new URL(gatofav.image.getUrl());  image = ImageIO.read(url);  ImageIcon fondoGato = new ImageIcon(image);  if(fondoGato.getIconWidth() > 800){  //redimensionamos  Image fondo = fondoGato.getImage();  Image modificada = fondo.getScaledInstance(800, 600, java.awt.Image.SCALE\_SMOOTH);  fondoGato = new ImageIcon(modificada);  }  String menu = "Opciones de favoritos: \n"  + " 1. ver otra imagen \n"  + " 2. Eliminar Favorito \n"  + " 3. Volver \n";  String[] botones = { "ver otra imagen", "eliminar favorito", "volver" };  String id\_gato = gatofav.getId();  String opcion = (String) JOptionPane.showInputDialog(null,menu,id\_gato, JOptionPane.INFORMATION\_MESSAGE, fondoGato, botones,botones[0]);  int seleccion = -1;  //validamos que opcion selecciona el usuario  for(int i=0;i<botones.length;i++){  if(opcion.equals(botones[i])){  seleccion = i;  }  }  switch (seleccion){  case 0:  verFavorito(apikey);  break;  case 1:  borrarFavorito(gatofav);  break;  default:  break;  }  }catch(IOException e){  System.out.println(e);  }    }    } |
| --- |





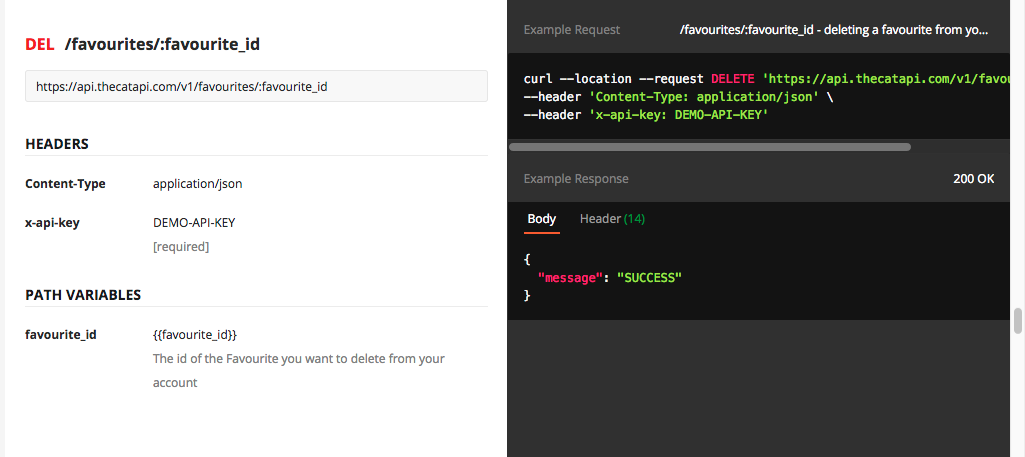
Como podemos ver el código de la línea 144 es el mismo de ver gatos con esto estamos utilizando la interfaz gráfica para poder visualizar y modificamos los valores de eliminar gatos para este menú.

# **ELIMINAR DATOS DE LA API**



Este método es el encargado de borrar datos de los favoritos.

en la página de apikey podemos encontrar el ejemplo de cómo eliminar un favorito, esto teniendo como apoyo el documento.



Como podemos visualizar , realizamos un método similar a los anteriores donde identificamos el id que queremos eliminar y este lo tomará como parámetro para eliminarlo de los favoritos.

# **BIBLIOGRAFÍA**