



REALISATION D'UNE PLATEFORME INFORMATIQUE POUR LA GESTION D'UNE BIBLIOTHEQUE

Réalisé par : Chebbi Amira/3LFIG3
Projet : Architecture logicielle





Sommaire

- I. Remerciement
- II. Introduction
- III. Environnement
- IV. Conception du projet
- V. L'implémentation des différentes couches
- VI. Couche DAO
- VII. Couche Métier
- VIII. Couche Service (REST)
- IX. Couche Présentation



I- Remerciement :

Il apparaît opportun de commencer ce rapport par des remerciements, à ceux qui m'ont beaucoup appris au cours de cette semestre, et même à ceux qui ont eu la gentillesse de m'aider à atteindre ce résultat.

Ainsi, je remercie Monsieur Ahmed Badredine, Madame Ilhem Souissi et Monsieur Ezzedine Fatnassi, mes enseignants de cours qui m'ont formé et accompagné avec beaucoup de patience et de pédagogie.



II-Introduction :

L'objectif de ce projet est de proposer une plateforme informatique pour la gestion d'une bibliothèque.

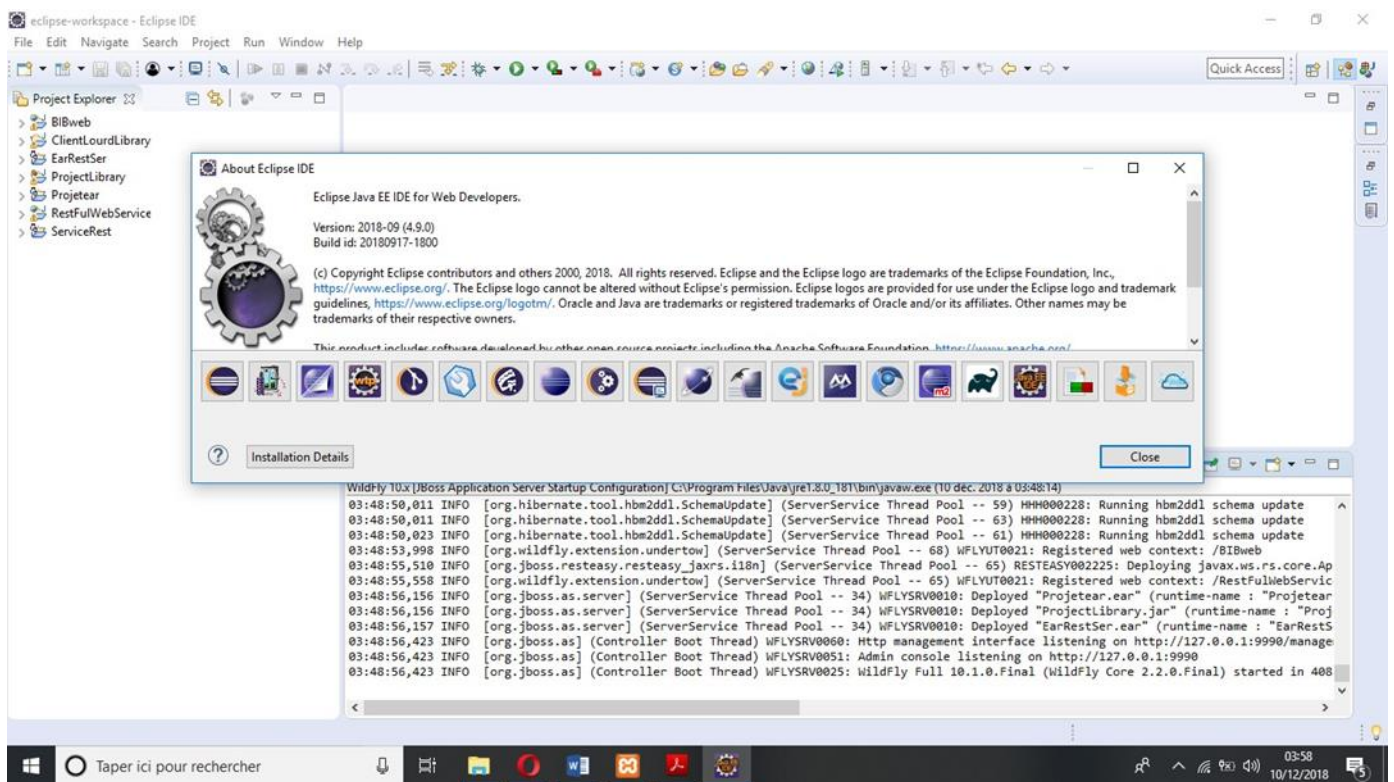
Cette plateforme doit être capable de gérer 3 types de client :

- ✓ Un Client Léger (Web)
- ✓ Un Client Mobile
- ✓ Un Client Lourd



III-Environnement :

Eclipse est un [projet](#), décliné et organisé en un ensemble de sous-projets de développements logiciels, de la [fondation Eclipse](#) visant à développer un environnement de production de logiciels [libre](#) qui soit extensible, universel et polyvalent, en s'appuyant principalement sur [Java](#).



Wild Fly, anciennement JBoss Application Server ou JBoss, est un serveur d'applications Java EE Libre écrit en Java, publié sous licence GNU LGPL. ... Le nom JBoss est aujourd'hui utilisé pour JBoss EAP, produit dérivé Wild Fly et faisant l'objet d'un support commercial.





Xampp, est une plateforme de développement Web de type Xampp, permettant de faire fonctionner localement (sans avoir à se connecter à un serveur externe) des scripts PHP.

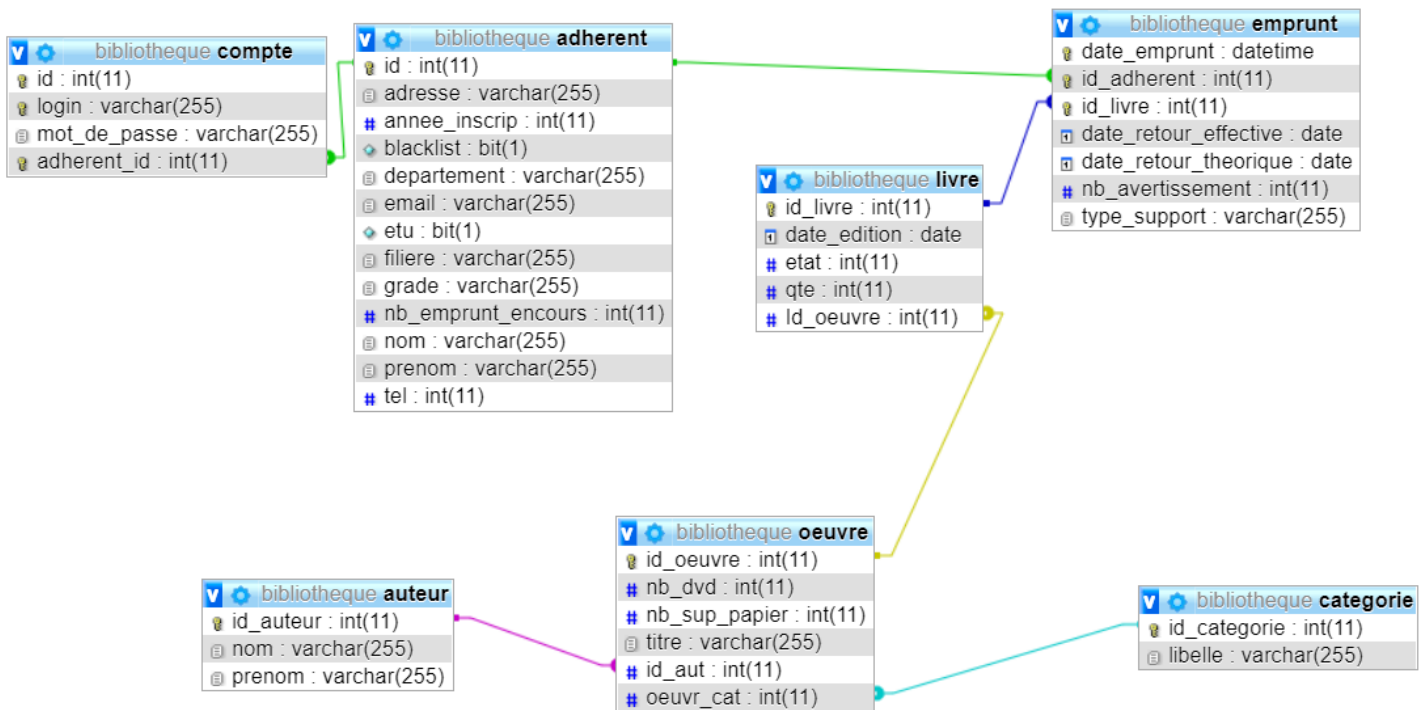


Android Studio, est un environnement de développement pour développer des applications mobiles Android





IV-Conception du projet :



- Le client léger doit être capable de :
- ❖ S'authentifier.
 - ❖ Rechercher des œuvres selon un ou plusieurs critères.
 - ❖ Gérer ses préférences (Gestion du panier).
 - ❖ Passer des commandes.
- Le client mobile doit être capable de :
- ❖ S'authentifier.
 - ❖ Rechercher des œuvres selon un ou plusieurs critères.



❖ Passer des commandes.

→ Le client lourd (Administrateur) doit être capable de :

- ❖ S'authentifier.
- ❖ Gérer tous les utilisateurs (Enseignant, Etudiant).
- ❖ Gérer la bibliothèque (gestion des œuvres).



V- L'implémentation des différentes couches :

V.1-Couche DAO :

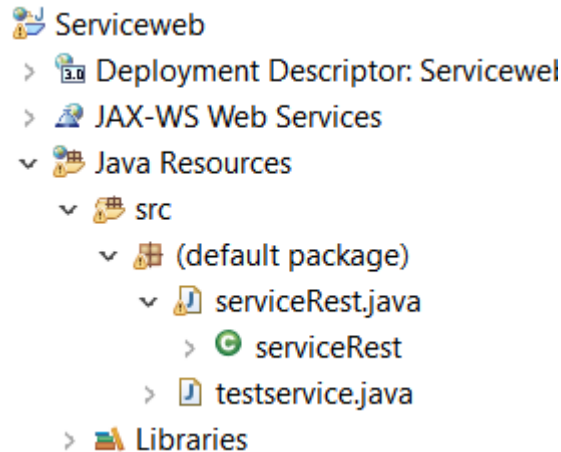
- metier.entities
 - > Adherent.java
 - > Auteur.java
 - > Categorie.java
 - > Compte.java
 - > Emprunt.java
 - > Livre.java
 - > Oeuvre.java
 - > Panier.java
 - > Pk_emprunt.java

V.2-Couche Session :

- metier.sessions.service
 - > AdherentLocal.java
 - > AdherentRemote.java
 - > AuteurLocal.java
 - > AuteurRemote.java
 - > BibliothequeLocale.java
 - > CategorieLocal.java
 - > CategorieRemote.java
 - > EmpruntLocal.java
 - > LivreLocal.java
 - > OeuvreLocal.java
 - > OeuvreRemote.java
- metier.sessions.service.impl
 - > AdherentService.java
 - > AuteurService.java
 - > BibliothequeImpl.java
 - > CategorieService.java
 - > EmpruntService.java
 - > LivreService.java
 - > OeuvreService.java



V.3-Couche Service (Rest) :

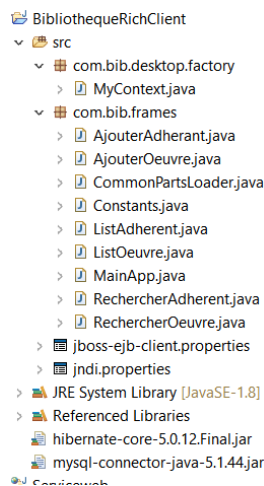


```
1* import java.text.DateFormat;
23
24
25
26 @Stateless
27 @Path("/bibliotheque")
28 public class serviceRest {
29     @EJB
30     private BibliothequeLocale metier;
31
32     @GET
33     @Path("/listecategories")
34     @Produces(MediaType.APPLICATION_JSON)
35     public List<metier.entities.Categorie> consulterCategorie_()
36     {
37         return metier.consulterCategories();
38     }
39     @GET
40     @Path("/oeuvres")
41     @Produces(MediaType.APPLICATION_JSON)
42
43     public List<metier.entities.Oeuvre> consulterOeuvres()
44     {return metier.consulterOeuvres();
45     }
46     @GET
47     @Path("/auteurs")
48     @Produces(MediaType.APPLICATION_JSON)
49     public List<metier.entities.Auteur> consulterAuteur()
50     {return metier.consulterAuteurs();
51     }
52
53     @GET
54     @Path("/categories/{idCat}/oeuvres")
55     @Produces(MediaType.APPLICATION_JSON)
56     public List<metier.entities.Oeuvre> consulterOeuvreParCat(@PathParam(value="idCat")int idCat)
57     {return metier.consulterOeuvreParCategorie(idCat);
58     }
59     @GET
```



V.4-Couche Présentation :

V.4.1- Client Desktop :



```
1 package com.bib.desktop.factory;
2
3 import java.util.Properties;
4
5 public class MyContext {
6
7     private static Context context;
8
9     private static void init() throws NamingException{
10         Properties clientProp = new Properties();
11         clientProp.put("remote.connectionprovider.create.options.org.xnio.Options.SSL_ENABLED", "false");
12         clientProp.put("remote.connections", "default");
13         clientProp.put("remote.connection.default.host", "localhost"); // comes from JVM argument
14         clientProp.put("remote.connection.default.port", "8080"); // comes from JVM argument
15         clientProp.put("remote.connection.default.connect.options.org.xnio.Options.SASL_POLICY_NOANONYMOUS", "false");
16
17         EJBClientConfiguration cc = new PropertiesBasedEJBClientConfiguration(clientProp);
18         ContextSelector<EJBClientContext> selector = new ConfigBasedEJBClientContextSelector(cc);
19         EJBClientContext.setSelector(selector);
20
21         Properties props = new Properties();
22         props.put(Context.URL_PKG_PREFIXES, "org.jboss.ejb.client.naming");
23
24         // props.put("jboss.naming.client.ejb.context"/**/, true); // check if needed...
25
26         context = new InitialContext(props);
27     }
28
29     public static Context getInstance() throws NamingException{
30         if(context==null)
31             init();
32         return context;
33     }
34 }
```



Bibliothèque

Nom utilisateur

Mot de passe

Connecter

Bibliothèque

Gestion Ouvre Gestion Adherent

Title :

Nb DVD :

nb Supérieur de Papier :

Auteur :

Categorie :

nombre : Date Edition : **Ajouter Edition**

0 editions ajoutées

Enregister



Bibliothèque

Gestion Oeuvre Gestion Adherent

Identifiant:

Title:

Nb DVD:

nb Supérieur de Papier:

Auteur:

Categorie:

nombre: Date Edition:

1 éditions ajoutées

Bibliothèque

Gestion Oeuvre Gestion Adherent

Identifiant	Titre	Auteur	Nb Sup Pages	Nb DVD
1	et si c'était vrai	jean paul	3	2
2	les misérables	victor hugo	2	0
3	Jamais sans ma fille	marc levy	6	3
4	vendredi ou la vie sauvage	marc levy	2	0
5	Les Fleurs du mal	marc levy	2	2
6	l'amour au temps de colera	marc levy	1	2
7	les misérables	jean paul	5	2
8	les misérables	jean paul	5	2
9	zola	marc levy	2	0
15	mille nuit et une nuit	jean paul	5	2



Bibliothèque

Gestion Oeuvre Gestion Adherent

Nom :

Prenom :

Email :

Adresse :

Telephone :

Type : **Etudiant** ▼

Department :

Filiere :

Grade :

Enregistrer

Bibliothèque

Gestion Oeuvre Gestion Adherent

Recherche : **Rechercher**

Nom :

Prenom :

Email :

Adresse :

Telephone :

Type : **Enseignant** ▼

Department :

Filiere :

Grade :

Enregistrer



Bibliotheque				
Gestion Ouvre		Gestion Adherent		
Nom	Prenom	Email	Telephone	Adresse
chebbi	amira	amira.chebbi@outlook.com	94596863	23 rue oued el koura enna...
mohamed	ben mohamed	mohammed.benmohame...	22252119	ariana

V.4.2- Client Web :

- BibEJB_Web
 - Deployment Descriptor: BibEJB_Web
 - JAX-WS Web Services
 - Java Resources
 - src
 - com.bib.filters
 - UserFilter.java
 - UserFilter
 - com.bib.servlets
 - AfficherPanier.java
 - AjouterAuPanier.java
 - Deconnecter.java
 - DeleteFromPanier.java
 - HomePage.java
 - LoginServlet.java
 - Rechercher.java
 - Libraries
 - JavaScript Resources
 - Referenced Libraries
 - build
 - WebContent
 - assets
 - css
 - js
 - META-INF
 - WEB-INF
 - Images
 - lib
 - AfficherPanier.jsp
 - Home.jsp
 - Login.jsp
 - web.xml

```

3* import java.io.IOException;
16 @WebFilter(dispatcherTypes = {
17     DispatcherType.REQUEST,
18     DispatcherType.FORWARD
19 }, urlPatterns = { "/user/*" })
20 public class UserFilter implements Filter {
21     public UserFilter() {
22     }
23 }
24 public void destroy() {
25 }
26 }
27 public void doFilter(ServletRequest req, ServletResponse res, FilterChain chain) throws IOException, ServletException {
28     HttpServletRequest request = (HttpServletRequest) req;
29     HttpServletResponse response = (HttpServletResponse) res;
30     HttpSession session = request.getSession();
31     if(session.getAttribute("user")!=null){
32         chain.doFilter(request, response);
33     }else{
34         response.sendRedirect(request.getContextPath()+"/login");
35     }
36 }
37 }
38
39 /**
40  * @see Filter#init(FilterConfig)
41  */
42 public void init(FilterConfig fConfig) throws ServletException {

```



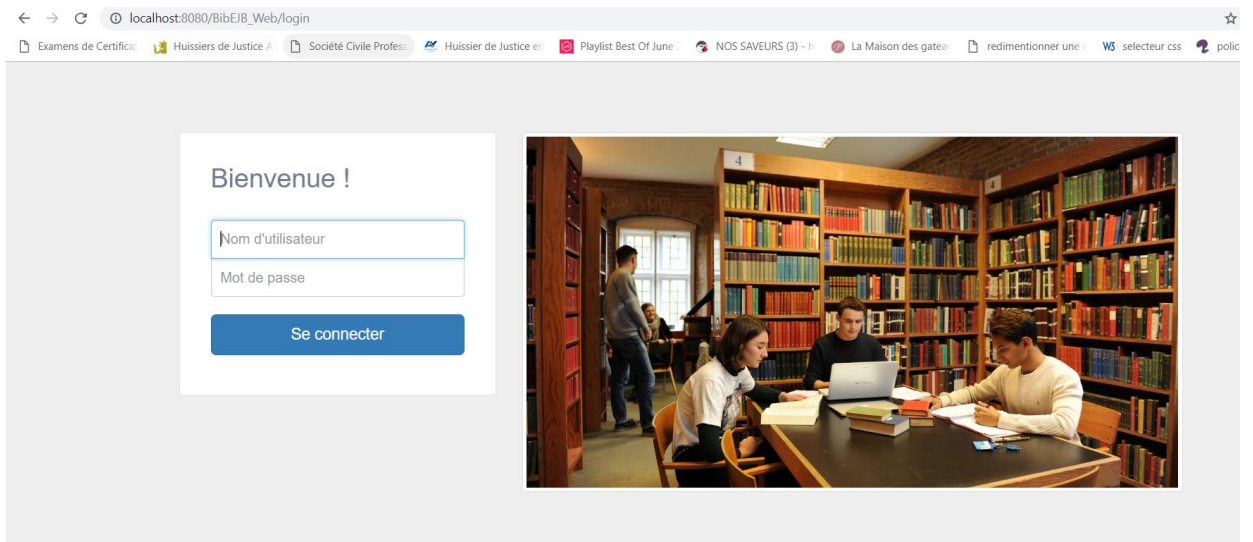
```
@WebServlet("/login")
public class LoginServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

    @EJB
    private AdherentLocal adherentLocal;

    public LoginServlet() {
        super();
    }

    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        getServletContext().getRequestDispatcher("/WEB-INF/Login.jsp").forward(request, response);
    }

    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        String login = request.getParameter("login");
        String pass = request.getParameter("pass");
        Adherent ad = adherentLocal.findByLoginAndPassword(login, pass);
        if(ad == null){
            request.setAttribute("error", true);
            getServletContext().getRequestDispatcher("/WEB-INF/Login.jsp").forward(request, response);
        }else{
            HttpSession session = request.getSession(true);
            session.setAttribute("user", ad.getId());
            response.sendRedirect(request.getContextPath()+"/user/");
        }
    }
}
```





Liste des Oeuvres

[Deconnecter](#)

Livre	Auteur	Categorie	Nb DVD	Nb livres	
et si c'était vrai	jean paul	historique	2	1	Edition : 1998-05-23
les misérables	victor hugo	Biographie	0	1	Edition : 2007-03-06
Jamais sans ma fille	marc levy	historique	3	1	
vendredi ou la vie sauvage	marc levy	Biographie	0	0	
Les Fleurs du mal ola	marc levy	historique	2	1	Edition : 2013-03-01
l'amour au temps de colera	marc levy	historique	2	0	
les miserables	jean paul	scientifique	2	0	
les miserables	jean paul	scientifique	2	1	Edition : 2001-05-07
zola	marc levy	Science-fiction	0	0	
mille nuit et une nuit	jean paul	Science-fiction	2	1	Edition : 2007-02-02

[Mon Panier](#)[Activer](#)
[Accédez à](#)

Liste des Oeuvres

[Deconnecter](#)



Livre	Auteur	Categorie	Nb DVD	Nb livres
l'amour au temps de colera	marc levy	historique	2	0
Les Fleurs du mal ola	marc levy	historique	2	0
Jamais sans ma fille	marc levy	historique	3	1
les misérables	victor hugo	Biographie	0	1
et si c'était vrai	jean paul	historique	2	1

[Mon Panier](#)



Mon Panier

[Deconnecter](#)

Livre	Edition	Categorie	
les misérables	2007-03-06	Biographie	
Les Fleurs du mal ola	2013-03-01	historique	
			 Valider

V.4.3- Client Mobile :

```
app
├── manifests
├── java
│   └── com.example.asus.bibliomobile
│       ├── Accueil
│       ├── Adherent
│       ├── API
│       ├── APIUTILS
│       ├── Auteur
│       ├── Categorie
│       ├── Compte
│       ├── Details_livre
│       ├── Emprunt
│       ├── Emprunter
│       ├── Id
│       ├── Livre
│       ├── MainActivity
│       ├── mes_Emprunts
│       ├── Oeuvre
│       ├── Profil
│       ├── Recherche
│       ├── Rechercheresult
│       └── RetrofitClient
```



```
package com.example.asus.bibliomobile;

import ...

public interface API {

    @GET("comptes/{login}/{motdepasse}")
    Call<List<Compte>> getComptes(@Path("login")String login, @Path("motdepasse")String motdepasse);
    @GET("listecategories")
    Call<List<Categorie>> getCategories();
    @GET("auteurs")
    Call<List<Auteur>> getAuteurs();
    @GET("oeuvres/all/{idAut}/{idCat}/{mc}")
    Call<List<Oeuvre>> getOeuvreparautcatmc(@Path("idAut")int idAut,@Path("idCat") int idCat , @Path("mc") String mc);
    @GET("oeuvres")
    Call<List<Oeuvre>> getOeuvres();
    @GET("oeuvre/{idOeuv}")
    Call<Oeuvre> getOeuvre(@Path("idOeuv")int idOeuv);
    @GET("categories/{idCat}/oeuvres")
    Call<List<Oeuvre>> getOeuvresparcat(@Path("idCat") int idCat);
    @GET("auteurs/oeuvres/{idAut}")
    Call<List<Oeuvre>> getOeuvresparaut(@Path("idAut")int idAut);
    @GET("oeuvres/{mc}")
    Call<List<Oeuvre>> getOeuvresparmc(@Path("mc") String mc);
    @GET("oeuvres/{idAut}/{idCat}")
    Call<List<Oeuvre>> getOeuvreparautcat(@Path("idAut")int idAut,@Path("idCat") int idCat );
    @GET("oeuvres/{idAut}/{mc}")
    Call<List<Oeuvre>> getOeuvreparautmc(@Path("idAut")int idAut, @Path("mc") String mc);
    @GET("oeuvres/{idCat}/{mc}")
    Call<List<Oeuvre>> getOeuvreparcatmc(@Path("idCat") int idCat , @Path("mc") String mc);
}
```

