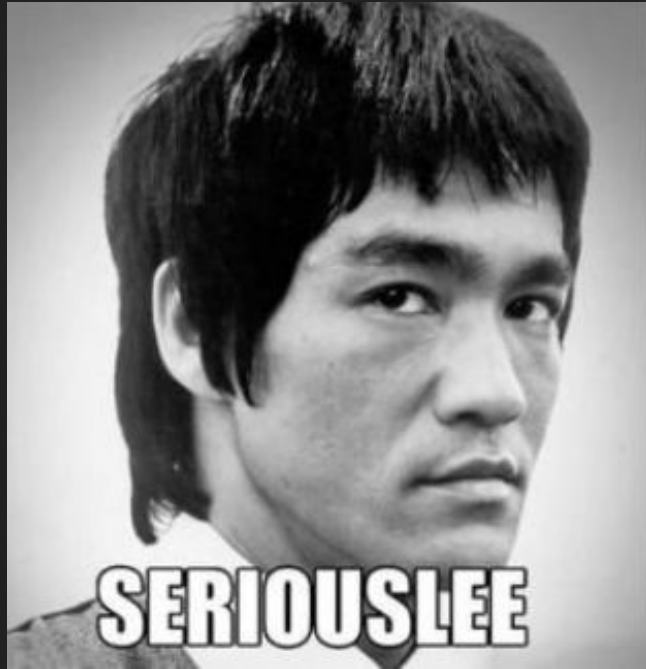
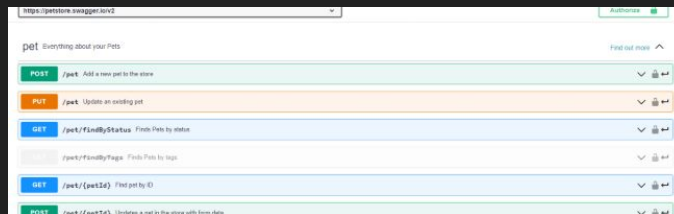
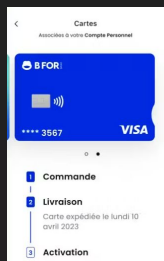


Rappel des fondamentaux



Application Programming Interface

nous swagger backend



```
protected void detectHandlerMethods(Object handler) {
    Class<?> handlerType = (handler instanceof String ?
        obtainApplicationContext().getBean(String) : hand

    if (handlerType != null) {
        Class<?> userType = ClassUtils.getUserClass(handlerType);
        Map<Method, T> methods = MethodIntrospector.selectMethods(userType,
            (MethodIntrospector.MetadataLookup<T>) method -> {
                try {
                    return getMappingForMethod(method, userType);
                }
                catch (Throwable ex) {
                    throw new IllegalArgumentException("Invalid mapping
                        userType.getName() + ": " + method, ex
                );
            });
        if (logger.isDebugEnabled()) {
            logger.trace(formatMappings(userType, methods));
        }
        else if (mappingsLogger.isDebugEnabled()) {
            mappingsLogger.debug(formatMappings(userType, methods));
        }
    }
}
```

Avantage

Sécurité : On n'expose pas sa base de données

Modularité : On peut changer le code interne sans casser l'appli du client.

Évolutivité : On peut ajouter des fonctionnalités sans tout refaire.

Centralisation : Une seule règle métier pour le Web, le Mobile et les partenaires.

HTTP Status Codes



1XX
INFORMATIONAL

2XX
SUCCESS

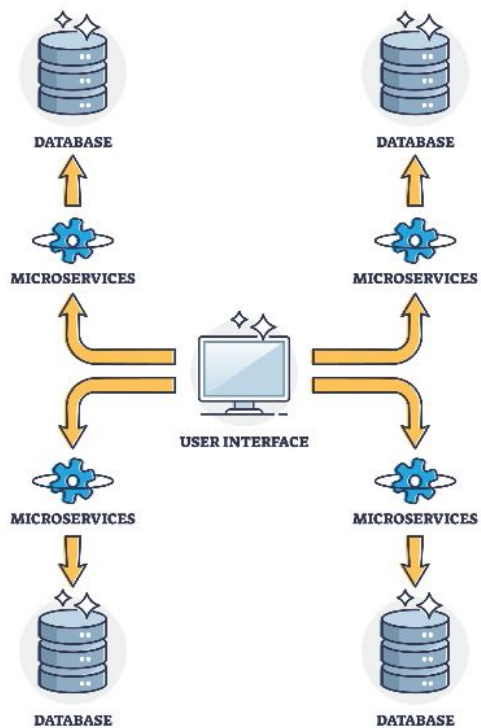
3XX
REDIRECTION

4XX
CLIENT ERROR

5XX
SERVER ERROR

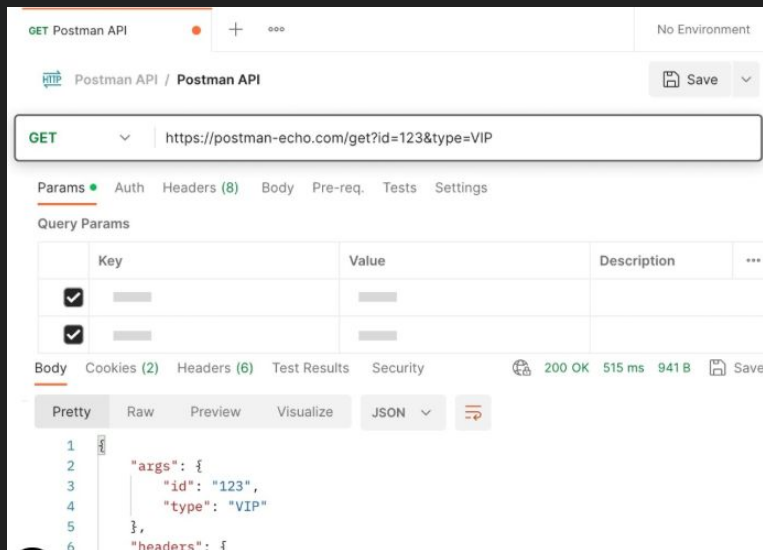
- 200 : succès de la requête ;
- 301 et 302 : redirection
- 401 : utilisateur non authentifié ;
- 403 : accès refusé ;
- 404 : ressource non trouvée ;
- 500, 502 et 503 : erreurs serveur ;
- 504 : le serveur n'a pas répondu.

MICROSERVICES



KarateDSL

Postman



restAssured

```
> import ...

public class NewProjectAction extends AnAction implements DumbAware, NewProjectOrModuleAction {
    @Override
    public void actionPerformed(@NotNull AnActionEvent e) {
        NewProjectWizard wizard = new NewProjectWizard(null, ModulesProvider.EMPTY_MODULES_PROVIDER,
        NewProjectUtil.createNewProject(wizard);
    }

    @Override
    public void update(@NotNull AnActionEvent e) {
        updateActionIcon(e);
        updateActionText(this, e);
    }

    @Override
    public @NotNull ActionUpdateThread getActionUpdateThread() { return ActionUpdateThread.BGT; }

    private static void updateActionIcon(@NotNull AnActionEvent e) {
        if (NewWelcomeScreen.isNewWelcomeScreen(e)) {
            NewWelcomeScreen.updateNewProjectIconIfWelcomeScreen(e);
        }
        else if (ExperimentalUI.isNewUI() && ActionPlaces.PROJECT_WIDGET_POPUP.equals(e.getPlace())) {
            e.getPresentation().setIcon(ExpUuiIcons.General.Add);
        }
    }
}
```

KarateDSL

```
@user_list
Scenario: Get list of all the Users
Given path '/some/path/' + id + '/something/else'
And param role = 'admin'
And header Authorization = token
When method get
Then status 200
And match each response == schema
And assert responseTime < 3000
And print response
```

Native JSON

Assertions "Fuzzy"

"id": "#number",

"name": "#string",

"tags": "#array"

Zéro dépendance

Configuration

KarateDSL

Mocks

Performance (gatling)

UI Testing (navigateur)



Basique

MonSuperTest.feature

Feature: Gestion des commandes

Background :

***URL** "MON_URL"

Scenario: Vérifier une commande existante

Given path "MON_PATH"

And param MON_PARAM = PARAM_VALEUR

When method **get**

Then status 200

And match == "#present"

Match

Match partiel (contient ces champs)

And match response **contains** { name: 'John', age: 30 }

Match exact (tous les champs)

And match response == { id: 1, name: 'John', age: 30 }

Match avec types flexibles

And match response == { id: '#number', name: '#string', age: '#number' }

Match

Feature: Récupérer les animaux disponibles dans le Petstore

Background:

```
* url 'https://petstore.swagger.io/v2'
```

Scenario: Récupérer tous les animaux disponibles

```
# TODO: Ajouter le path 'pet/findByStatus'
```

```
# TODO: Ajouter le paramètre status=available
```

```
# TODO: Définir la méthode HTTP GET
```

```
# TODO: Vérifier que le status est 200
```

```
# TODO: Vérifier que la réponse est un tableau (array)
```

```
# TODO: Vérifier que le tableau n'est pas vide ->#[_ > 0]
```