

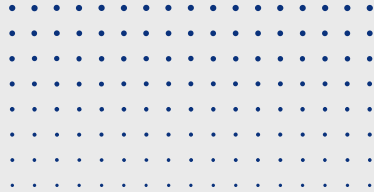


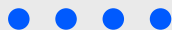
# BonitaSoft

Compte rendu des TPs | SOA

Encadré par: Pr. FISSAA

Travail de: Anas HANNOUR, AMOA INE2,  
2023/2024

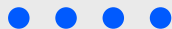




Bienvenue à cette présentation détaillant le projet que j'ai mené à bien en utilisant Bonitasoft, une plateforme de gestion de processus métier (BPM). Au cours de ce projet, j'ai suivi attentivement la documentation de Bonitasoft pour comprendre en profondeur son fonctionnement et ses fonctionnalités. Mon objectif était de créer des diagrammes de processus, d'élaborer des modèles de données métier, de concevoir des formulaires intuitifs, et enfin, de connecter l'ensemble de ces éléments de manière cohérente pour automatiser des processus métier complexes.

Dans cette présentation, je vais vous guider à travers les différentes étapes de mon projet. Tout d'abord, nous allons explorer le processus de démarrage avec Bonitasoft, en examinant comment j'ai suivi le tutoriel de prise en main et comment j'ai dessiné des diagrammes BPMN (Business Process Model and Notation). Ensuite, nous allons plonger dans la création du modèle de données métier, en décrivant comment j'ai structuré les informations essentielles pour notre application. Par la suite, nous aborderons la création de formulaires conviviaux pour les utilisateurs, en mettant en lumière les choix de conception et les fonctionnalités intégrées pour une expérience utilisateur optimale.

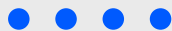




De plus, je vais partager avec vous la mise en œuvre d'un connecteur de processus, détaillant comment j'ai intégré des actions automatisées dans nos processus métier. Enfin, nous allons explorer la phase de clôture du projet, où j'ai réalisé une application REST utilisant Maven, et j'ai utilisé Postman pour tester et valider notre solution.

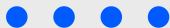
Cette présentation sera accompagnée d'un fichier de documentation détaillé, où vous pourrez trouver des informations supplémentaires sur chaque étape de ce projet. Sans plus tarder, plongeons dans l'univers passionnant de Bonitasoft et découvrons comment j'ai utilisé cette puissante plateforme pour transformer des idées en processus métier automatisés et efficaces.





**Voici le lien git pour la  
documentation de BonitaSoft  
<https://github.com/anasgg/soa>**

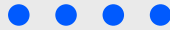




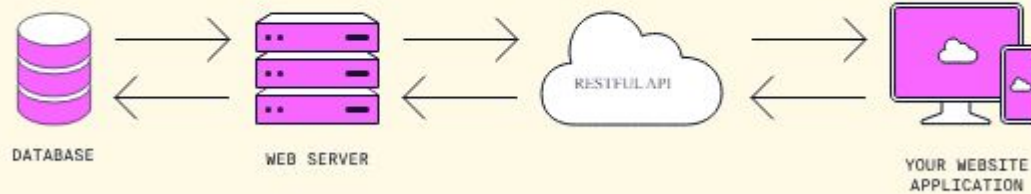
# Bonita Rest API Extension archetype

REST (REpresentational State Transfer) est un style d'architecture logicielle définissant un ensemble de contraintes à utiliser pour créer des services web. Les services web conformes au style d'architecture REST, aussi appelés services web RESTful, établissent une interopérabilité entre les ordinateurs sur Internet.



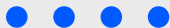


## What is Rest API?



Intégrer un service web REST à votre application web révolutionne l'interaction avec votre base de données. Ces services permettent des requêtes HTTP simples pour ajouter, modifier ou récupérer des données, offrant une connectivité fluide et efficace. Avec une architecture flexible, ils assurent une extensibilité et une scalabilité sans pareil. Chaque interaction devient harmonieuse, offrant une expérience utilisateur optimale. Le service web REST devient le pont solide entre votre frontend et votre backend, permettant une navigation fluide à travers les vastes océans de données de votre base de données.





# Configuration de l'extension REST API Bonita

Dans le cadre de notre documentation pour BonitaSoft, nous avons configuré une extension REST API selon deux approches : avec Maven et avec Postman.

Avec Maven:

Version Bonita: 1.3.1

Langage de programmation: Java

Nom de l'API: monApiExtension

Nom d'affichage de l'API: Mon Extension API

Description de l'API: Description de mon extension API REST

Verbe HTTP: GET

Modèle de chemin: /my-api

Autorisations: myRestAPIPermission

Paramètres d'URL: Non spécifiés

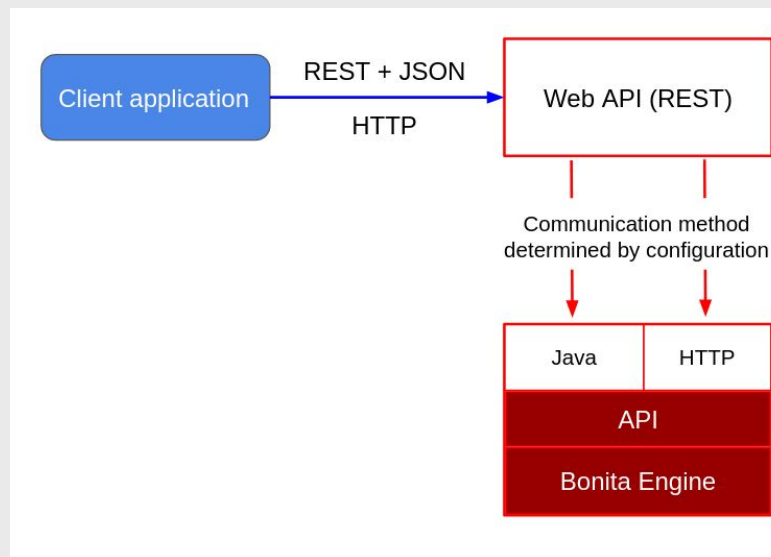
Wrapper: Activé

Groupld Maven: com.example

ArtifactId Maven: my-rest-api

Version Maven: 1.0-SNAPSHOT

Package: com.example



Install the latest PowerShell for new features and improvements! <https://aka.ms/PSWindows>

```
PS C:\Users\grzeje> mvn archetype:generate "-DarchetypeGroupId=org.bonitasoft.archetypes" "-DarchetypeArtifactId=bonita-rest-api-extension-archetype"
```

```
[INFO] Scanning for projects...
```

```
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-install-plugin/3.1.1/maven-install-plugin-3.1.1.pom
```

```
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-install-plugin/3.1.1/maven-install-plugin-3.1.1.pom (7.8 kB at 9.4 kB/s)
```

```
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-plugins/39/maven-plugins-39.pom
```

```
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-plugins/39/maven-plugins-39.pom (8.1 kB at 147 kB/s)
```

```
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-parent/39/maven-parent-39.pom
```

```
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-parent/39/maven-parent-39.pom (48 kB at 470 kB/s)
```

```
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/apache/29/apache-29.pom
```

```
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/apache/29/apache-29.pom (21 kB at 406 kB/s)
```

```
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-install-plugin/3.1.1/maven-install-plugin-3.1.1.jar
```

```
Define value for property 'language' (should match expression 'java|groovy|kotlin'): Define value for property 'language' (should match expression '^java$|^groovy$|^kotlin$'): java
```

```
Value does not match the expression, please try again: java
```

```
Define value for property 'apiName' (should match expression '^[a-zA-Z0-9]+$'): monApiExtension
```

```
Value does not match the expression, please try again: monApiExtension
```

```
Define value for property 'apiDisplayName': Mon Extension API
```

```
[INFO] Using property: apiDesc = My Rest API extension description
```

```
Define value for property 'httpVerb' (should match expression '^GET$|^POST$|^PUT$|^PATCH$|^DELETE$|^HEAD$|^OPTIONS$|^TRACE$'): get
```

```
Value does not match the expression, please try again: GET
```

```
Define value for property 'pathTemplate': /my-api
```

```
[INFO] Using property: permissionNames = myRestAPIPermission
```

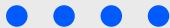
```
[INFO] Using property: urlParameters = !
```



```
[INFO] Project created from Archetype in dir: C:\Users\grzeje\my-rest-api
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 23:20 min
[INFO] Finished at: 2024-05-03T22:21:24+01:00
[INFO] -----
PS C:\Users\grzeje> |
```

Avec Maven, j'ai initié la création d'une extension REST API pour BonitaSoft. J'ai choisi d'utiliser Java comme langage de programmation pour ce projet. En définissant le modèle de chemin d'URL, j'ai précisé comment les utilisateurs pourront accéder à mon API. J'ai également configuré les autorisations requises pour sécuriser l'accès à mon API. Pour faciliter le déploiement et l'intégration, j'ai activé le wrapper pour mon API. Enfin, j'ai organisé la structure du projet en spécifiant le package dans lequel seront créés les fichiers sources de mon extension. En somme, avec Maven, j'ai mis en place les bases de mon projet d'extension REST API pour BonitaSoft.

+ +  
+ +



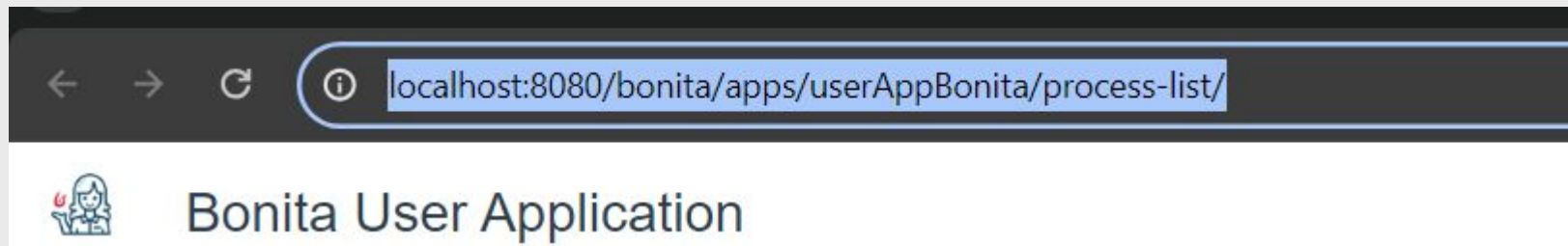
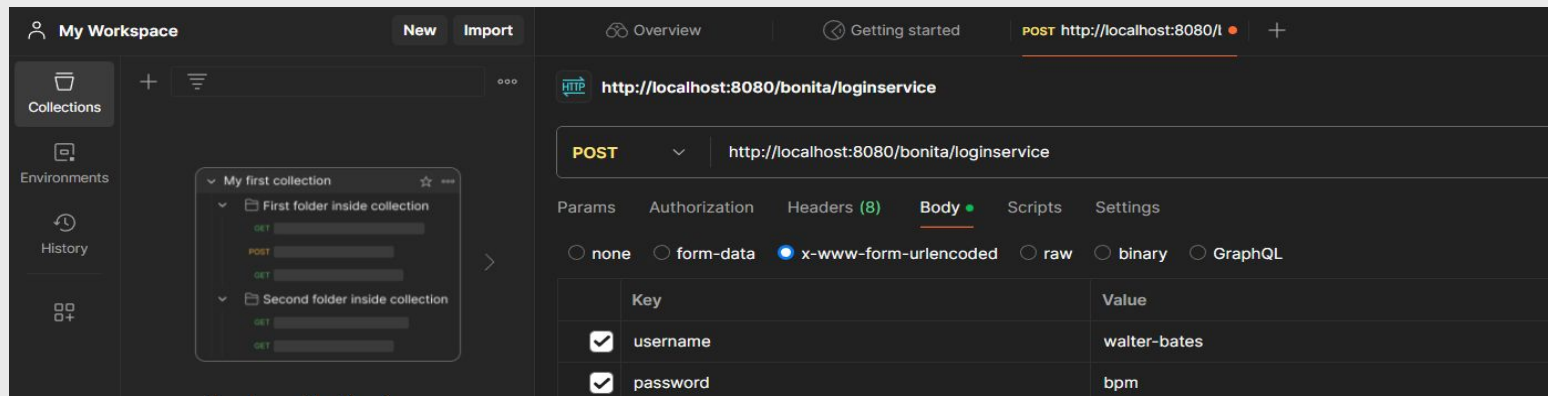
### Pour connecter Postman à Bonita :

1. Obtenez l'URL de l'API Bonita que vous souhaitez utiliser.
2. Configurez Postman en créant une nouvelle requête et en ajoutant l'URL de l'API, la méthode HTTP appropriée, les en-têtes nécessaires (comme l'authentification), et les données requises dans le corps de la requête si nécessaire.
3. Envoyez la requête depuis Postman.
4. Traitez la réponse de l'API Bonita dans Postman pour vérifier le succès de l'opération et traiter les données retournées selon vos besoins.



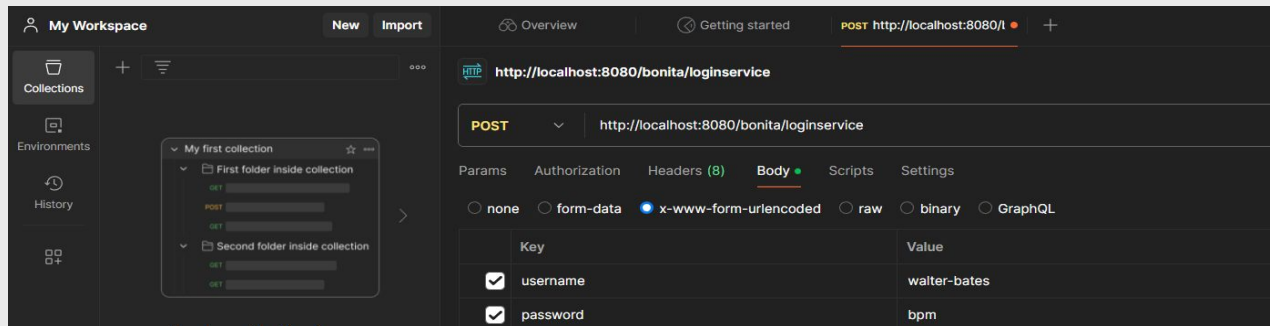


Obtenir l'URL de l'API Bonita : Vous devrez d'abord obtenir l'URL de l'API Bonita à laquelle vous souhaitez accéder.





**2. Authentification : Bonita**  
nécessite généralement une  
authentification pour accéder à ses  
API. Vous devrez fournir les  
informations d'identification  
appropriées dans vos requêtes API  
depuis Postman.



Body Cookies (3) Headers (8) Test Results

Status: 204 No Content Time: 90 ms Size: 403 B Save as example

Name	Value	Domain	Path	Expires	HttpOnly	Secure
JSESSIONID	8A8F059483 ...	localhost	/	Session	true	false
X-Bonita-API...	d64266e2-5f...	localhost	/	Session	false	false

Si vous effectuez une requête  
POST, PUT ou DELETE, vous  
devrez également fournir les  
données requises dans le corps de  
la requête, au format JSON ou  
autre, selon les spécifications de  
l'API Bonita.



<input checked="" type="checkbox"/>	Key	Value
<input checked="" type="checkbox"/>	X-Bonita-API-Token	d64266e2-5519-4182-a6a2-ace4fb5838cc
	Key	Value

**Envoyer la requête :** Une fois que vous avez configuré tous les paramètres nécessaires dans Postman, vous pouvez envoyer la requête à l'API Bonita.

**Traiter la réponse :** Postman affichera la réponse de l'API Bonita. Assurez-vous de vérifier la réponse pour vous assurer que l'opération a réussi et de traiter les données retournées selon vos besoins.

```
{
  "displayDescription": "",
  "deploymentDate": "2024-05-03 18:50:34.385",
  "displayName": "Pool",
  "name": "Pool",
  "description": "",
  "deployedBy": "4",
  "id": "5181362391406334035",
  "activationState": "ENABLED",
  "version": "1.0",
  "configurationState": "RESOLVED",
}
```

