



Back End marketing conversationnel
Technologies web côté serveur

Paul Orhon

LP – MiAR – Université de Nantes

26 novembre 2017



Table des matières

1	Projet	3
2	Contexte du projet	3
2.1	Objectifs	3
2.2	Contraintes	3
3	Les vues	4
3.1	Vue globale du projet	4
3.2	Vue logique du Back End	4
4	Implémentation	5
4.1	Les Services	5
4.1.1	L'envoi de messages	5
4.1.2	L'envoi de réponse	5
4.1.3	Récupérer l'historique	5
4.2	Java	5
4.3	Librairie	6
4.3.1	Slf4j	6
4.3.2	Spring Boot	6
4.3.3	Spring Boot Test	6
4.3.4	Junit	6
4.3.5	Gson	6
4.4	Outils	6
4.4.1	IntelliJ IDEA	6
4.4.2	Maven	6
4.4.3	MongoDB	7
4.4.4	GitHub	7
4.4.5	Travis-ci	7
4.4.6	Codecov	7
5	Conclusion	7

Table des figures

1	Workflow classique d'un utilisateur	3
2	Diagramme de la séparation du Back / Front End	4
3	Diagramme de la vue logique du back end	4
4	Logo	7

1 Projet

Dans le cadre du module de technologies web côté serveur, il a été demandé de créer une partie du back end d'un marketing conversationnel.

Pour cela il va falloir développer différents services pour pouvoir mettre en place ce système.

2 Contexte du projet

2.1 Objectifs

L'objectif est donc de développer une partie du Back End d'un marketing conversationnel, en exposant les services avec une api REST.

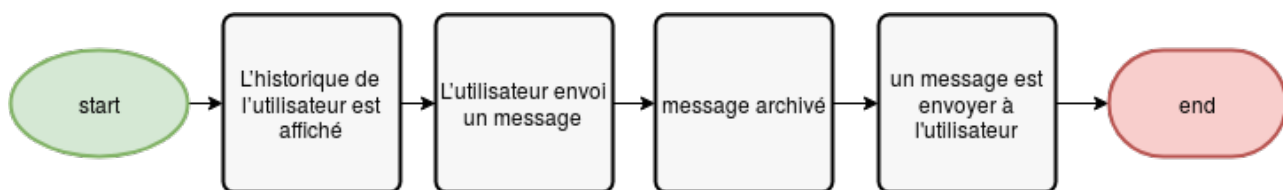


FIGURE 1 – Workflow classique d'un utilisateur

La partie à développer est l'envoi des messages et les sauvegardes des données. Ces sauvegardes sont utilisées pour conserver les messages, qui seront datés, afin d'afficher l'historique à l'utilisateur et pouvoir effectuer des statistiques anonymes sur toutes les conversations.

Le projet doit être séparé en sous-module, cela permettra de manipuler, modifier et diffuser plus facilement.

2.2 Contraintes

1. Le projet est à réaliser seul.
2. Le programme sera développé sur la plateforme Java.
 - Il sera programmé en Java pour des raisons de connaissance.
3. Le rapport est à rendre le 27 novembre 2017 à 08h00.
4. Le projet est à rendre le 5 décembre 2017 à 12h20.
5. Le système doit sauvegarder les messages pour réaliser des statistiques et pour l'historique des utilisateurs.
6. Le système doit être testé et testable sur le poste du développeur.
7. Le couplage doit être lâche.

3 Les vues

3.1 Vue globale du projet

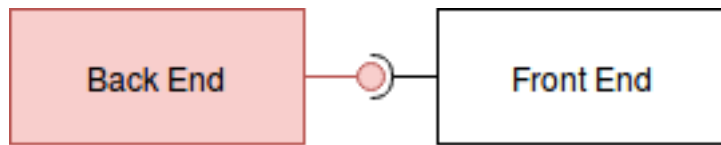


FIGURE 2 – Diagramme de la séparation du Back / Front End

Comme dit précédemment, seule le back end et son interface sera développé (partie en rouge de la figure 2).

3.2 Vue logique du Back End

Le back end sera divisé en plusieurs modules permettant ainsi de diviser les tâches, de réaliser un couplage lâche et de faciliter le maintien du système.

Cette représentation repose sur le *domain-driven design*¹. Le but du DDD est de former le code autour de notion métiers.

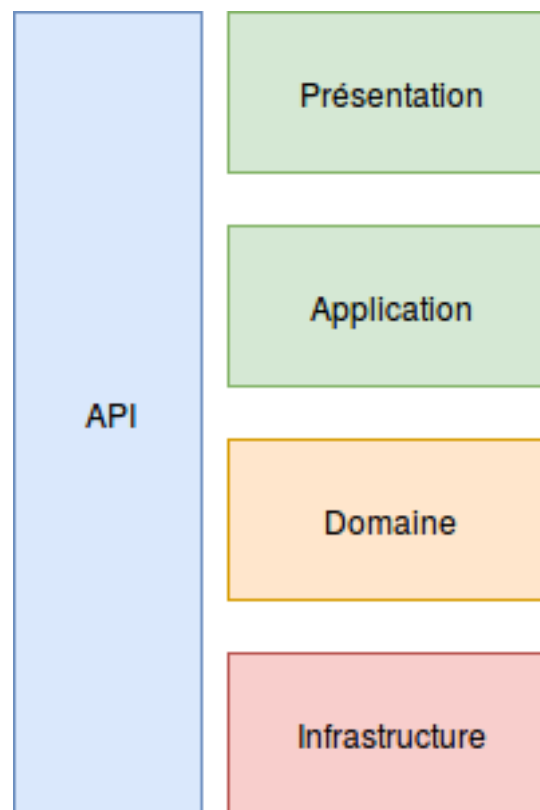


FIGURE 3 – Diagramme de la vue logique du back end

Ici chaque couleur correspond à un module.

1. *domain-driven design*(DDD) : en français *conception pilotée par le domaine* est une approche de la conception de logiciel. Plus d'information : https://en.wikipedia.org/wiki/Domain-driven_design

Application contient la **présentation** du fait de l'utilisation du framework *Spring* qui permet une mise en place simple d'une api REST et donc de l'interface du back end. Son rôle est de récupérer les requête arrivant à l'interface, de les envoyer au bon service et de répondre.

Domaine correspond aux *services* de l'application. Contient aussi les *interfaces*, pressant dans l'**API** et implémenté dans l'**infrastructure**, utile à ces services.

Infrastructure correspond à l'implémentation des différentes *interfaces* et *classe abstraites* qui sont les factory. Il gère le stockage persistant des données.

API contient les *classe* communes aux modules. Cela permet notamment au **domaine** d'utiliser une classe et de laisser l'implémentation à l'**Infrastructure**.

4 Implémentation

4.1 Les Services

Le back end va donc proposer différents services.

4.1.1 L'envoi de messages

Un service proposera au client d'envoyer un message vers le serveur. Ce message sera transmis aux personnes concernées. Il sera aussi sauvegardé. Cette sauvegarde est utile pour l'historique de conversation de l'utilisateur (cf. 4.1.3) et pour réaliser des statistiques sur l'ensemble des demandes.

4.1.2 L'envoi de réponse

Suite à l'envoi d'un message par un utilisateur, s'il le peut, le serveur envoie un message sinon la personne concernée répond à la demande.

Pour le moment le serveur répondra avec un message générique, pour cause d'un manque de temps et de moyen.

4.1.3 Récupérer l'historique

Lors de sa connexion l'utilisateur peut récupérer son historique de conversation afin de reprendre le fil de sa conversation.

Si l'utilisateur n'a pas d'historique une liste vide lui est retournée.

4.2 Java

Comme dit précédemment le programme sera développé pour la plateforme java et développé en java et plus précisément dans la version 8. Cela permet de faciliter le déploiement du programme vu que java est multi-plateforme.

4.3 Librairie

4.3.1 Slf4j

*Slf4j*¹ est une api de logging. Elle permet de logger les informations souhaiter à l'endroit souhaiter. Par exemple dans un fichier. Cela va permettre de pouvoir avoir les logs même après l'arrêt du programme, car les messages ne sont plus directement dans le terminal.

4.3.2 Spring Boot

Nous allons aussi utiliser *Spring Boot*² en version 1.5.8 qui est un framework permettant de développer une application web avec une gestion simplifiée des différents services mis en place et de leur présentation sous forme de service REST.

4.3.3 Spring Boot Test

*Spring Boot Test*³ est utilisé pour réaliser les tests de l'api REST. Nous l'utiliserons dans la même version que Spring Boot (cf. 4.3.2).

4.3.4 Junit

Pour les tests plus générales, nous allons utiliser *junit*⁴ en version 4.12, qui est un framework de test unitaire.

4.3.5 Gson

*Gson*⁵ est une librairie qui permet de convertir des objets Java en JSON et inversement. Il nous sera utile pour comprendre les informations reçu par les clients et de pouvoir envoyer des informations.

4.4 Outils

Pour ce projet nous utiliserons différents outils qui vont simplifier le développement.

4.4.1 IntelliJ IDEA

*IntelliJ IDEA*⁶ est l'IDE java qui est utilisé pour ce projet. Il va permettre de faciliter le développement grâce à ces différentes options.

4.4.2 Maven

*Maven*⁷ est un outil de gestion et d'automatisation de production des projets Java. Il va être utiles pour la séparation du programme en module, la gestion des dépendances et les build.

-
1. <http://www.slf4j.org>
 2. <http://projects.spring.io/spring-boot>
 3. https://spring.io/guides/gs/spring-boot/#_add_unit_tests
 4. <http://junit.org/junit4/>
 5. <https://github.com/google/gson>
 6. <https://www.jetbrains.com/idea/>
 7. <http://maven.apache.org/>

4.4.3 MongoDB

*MongoDB*¹ est un système de gestion de base de données orientée documents, il est simple d'utilisation. Il va être utilisé pour sauvegarder les messages des utilisateurs afin de leur retourner leur historique(cf. 4.1.3).

4.4.4 GitHub

*GitHub*² est un service de gestion de développement. Il va être utilisé pour gérer les versions du projet. Cela va aussi permettre le partage des sources sous licence MIT afin de récupérer le retour des utilisateurs et de leur amélioration.

4.4.5 Travis-ci

*Travis-ci*³ est un outil d'intégration continue, qui permet d'effectuer les build et les tests sous différent environnement UNIX. Il permet aussi de déployer les build. Il est aussi possible d'utiliser *AppVeyor* pour un environnement Windows.

4.4.6 Codecov

*Codecov*⁴ est un outil permettant de regrouper, fusionner, archiver et comparer des rapports de couverture de code. Cela permet, notamment, de trouver les codes morts.

5 Conclusion

Ce projet a donc pour but de développer le back end et l'interface d'un marketing conversationnel. Pour des raisons de temps seuls l'échange de messages et l'historique sera développé. Le projet sera réalisé en java. Le développement se fera autour des tests afin de valider le fonctionnement des différentes méthodes. Les différentes tâches doivent être séparées pour avoir un couplage lâche. Différent outil seront utilisés pour faciliter le développement.



FIGURE 4 – Logo

1. <https://www.mongodb.com/what-is-mongodb>

2. <https://github.com/>

3. <https://travis-ci.org/>

4. <https://codecov.io/gh>