**Cadre du projet**

Ce chapitre préliminaire est primordial et décisif car il facilite la mise en contexte du projet en mettant ce dernier dans son cadre générale.

Nous allons commencer en premier lieu par, la présentation du l’organisme d’accueil dans lequel mon stage a été effectué, ensuite on va spécifier une étude détaillée de l’existant, le contexte du projet suivi par la problématique, et a fini par citer les objectifs à atteindre à partir de la solution proposée.

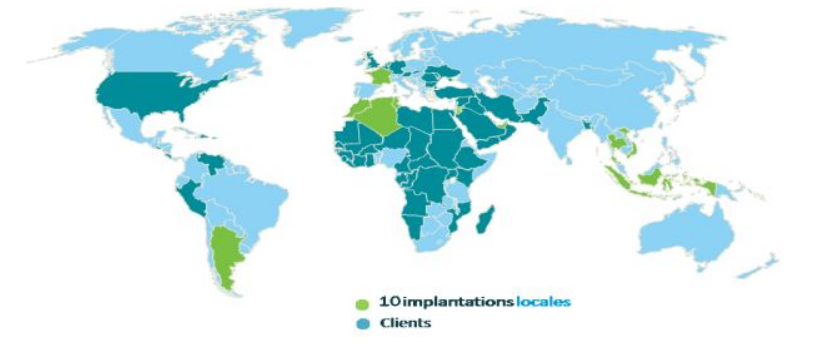
Pour clôturer, nous montrons un survol sur la méthodologie adoptée pour le développement tout au long du projet.

1. **Présentation du cadre de stage**

Dans la partie suivante on va présenter l’entreprise d’accueil pour la réalisation de ce projet.

**Groupe SOFRECOM**

  SOFRECOM est une filiale du groupe Orange, développé depuis 50 ans un savoir-faire unique dans les métiers de l’opérateur, ce qui en fait un leader mondial du conseil et de l’ingénierie de télécommunication. Le groupe accompagne ses clients (opérateurs, industriels, institutions, gouvernement) dans leurs projets de gestion, de création, de reprise ou de transformation en s’appuyant sur ses domaines d’expertise phares : les réseaux et services, les Solutions IT et le Conseil en entreprise, couvrant ainsi tous les métiers des opérateurs. Le groupe SOFRECOM compte aujourd’hui plus de 1500 experts répartis sur 10 implantations à travers le monde, ce qui a permis d’accompagner plus de 200 opérateurs, gouvernements et bailleurs de fonds, dans une centaine de pays (voir Figure 1.1).



**Figure 1. 1: SOFRECOM dans le monde [1]**

**Source :** <https://www.sofrecom.com/fr>

**Secteur d’activités**

SOFRECOM a pu confirmer son expérience dans plusieurs domaines d’expertise comme représenté dans la Figure 1.2.



**Figure 1. 2 : Secteurs d’activités de SOFRECOM**

**SOFRECOM Tunisie**

Inaugurée en octobre 2012, SOFRECOM Tunisie est un acteur majeur du conseil et d’ingénierie en télécommunications sur le marché local. C’est la plus jeune et la plus importante filiale du groupe SOFRECOM en zone Afrique et Moyen-Orient. En 8 ans seulement, SOFRECOM Tunisie compte déjà Plus de 650 Experts, elle a acquis sa force et sa crédibilité au travers de centaines de projets diversifiés à l’international, lui permettant d’avoir une longueur d’avance sur ses concurrents. Faisant partie du Groupe Orange, SOFRECOM Tunisie bénéficie d’une renommée internationale, elle travaille pour trois clients du groupe Orange: OLS, OAB, DSIF.

Elle connaît une très forte croissance depuis sa création en 2017 et en croissance continue en 2018.

1. **Cadre du projet :**

**Présentation du projet**

Le projet introduit par le présent rapport, rentre dans le cadre du projet de fin d’études qui vient de conclure notre formation d’ingénieur à l’Institut Supérieur d’Informatique.

Le travail demande c’est un générateur d’application web. Il a pour objectif de conception et développement d’un outil qui génère une application web normalisée avec la chaine devops.

**Etude de l’existant**

Dans cette partie, nous allons exposer les différentes solutions disponibles sur le marché. Nous passerons, par la suite à établir un bilan sur lequel nous nous sommes basés pour proposer notre solution.

**Présentation des solutions existantes sur le marché**

Parmi les solutions existantes sur le marché nous mentionnons :

* Jhipster : Son nom provient de Java Hipster est un générateur d’applications libre et Open Source, un Framework de développement composé de plusieurs technologies dont Angular et Spring Boot. Il vous permet de démarrer, développer et mettre en production votre projet en un temps record. Afin de pouvoir créer des applications web complètes, puissantes et polyvalentes, ou une architecture micro services, ce générateur intègre un certain nombre de détails importants participant à son efficacité.

La Figure 1.3 présente Jhipster Générateur



**Figure 1.3: Jhipster Générateur**

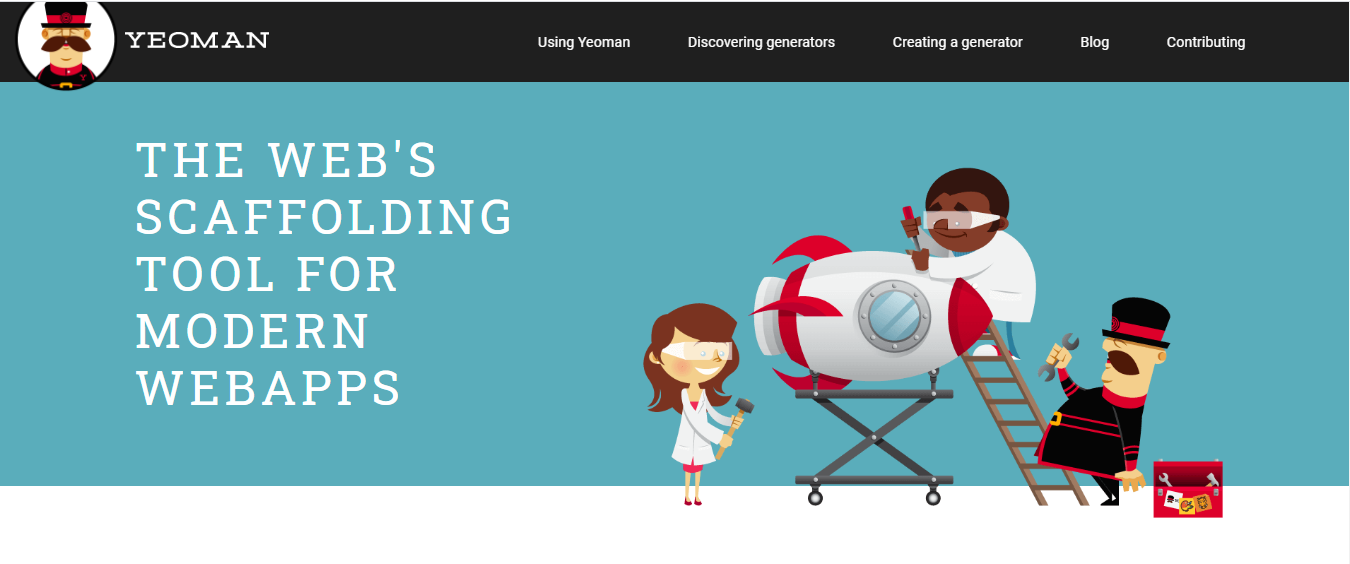
Source :<https://www.dtp-ag.com/dtp_index/jhipster-quest-ce-que-cest/>

* Yeoman : Yeoman est un générateur  générique permettant la création de tout type d'application. Il permet de démarrer rapidement de nouveaux projets et rationalise la maintenance des projets existants.

Yeoman est indépendant de la langue. Il peut générer des projets dans n'importe quel langage (Web, Java, Python, C #, etc.).

source : <https://yeoman.io/learning/>

La Figure 1.4 présente Yeoman Générateur



**Figure 1.3: Yeoman Générateur**

* Générateur Angular Fullstack : Le générateur Angular Fullstack développe une application complète utilisant Node, Express et Angular. Vous pouvez éventuellement choisir d'utiliser MongoDB, Bootstrap, ui-router, Compass et SASS.

  Source : <https://parsed.io/intro-to-yeoman-angular-fullstack/>

La Figure 1.5 présente Angular Full-Stack Générateur

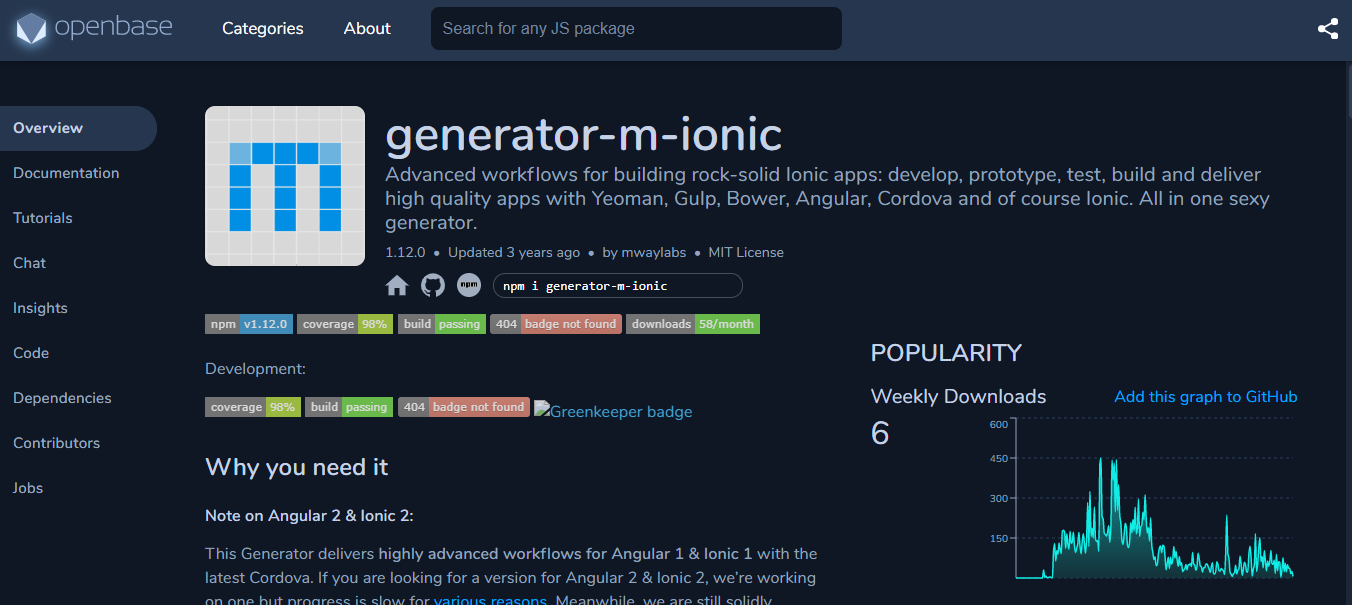


**Figure 1.5: Angular Full-Stack Générateur**

* Générateur  M-Ionic : M-ionic c’est une collection open source de workflows avancés pour la création d'applications mobiles HTML5 multiplateforme avec Ionic. Ce générateur fournit des flux de travail avancés pour créer des applications ioniques solides  comme le roc : développer, prototyper, tester, construire et fournir des applications de haute qualité avec Yeoman, Gulp, Bower, Angularjs, Cordova et bien sur Ionic.

Source : <https://yeoman.io/blog/generator-m-ionic-js-fatigue-and-transitioning-to-typescript-angular-2-and-ionic-2.html>

La Figure 1.6 présente M-Ionic Générateur



**Figure 1.6 : M-Ionic Générateur**

Afin d’étudier de plus près les solutions présentées, nous évaluons, dans le tableau 1.1, ces applications selon un nombre de critères :

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Générateur** | **Open Source** | **Langage et Framework** | **Application Front** | **Application Back** | **Application Full Stack** |
| **Jhipster** | Oui | Spring boot,  Angular,  React,  Vue. |  |  |  |
| **Yeoman** | Oui | Angular, Ionic, React, Java, Python, C # |  |  |  |
| **Angular Full-Stack** | Oui | Angular, React |  |  |  |
| **M-Ionic** | Oui | Ionic |  |  |  |

**Tableau 1.1 : Tableau comparatif des solutions existantes sur le marché**

Certes ces solutions répondent aux besoins de leurs utilisateurs et assurent la création et la génération des applications web, cependant nous devons signaler que ces solutions cachent derrière plusieurs limites.

**Problématiques**

La bonne gestion du temps dans l’entreprise est une notion primordiale, elle a un impact positif sur la productivité des employées et augmente la satisfaction du travail comme il a dit Benjamin Franklin « Time is money ».

C’est pour cette raison il faut que les employées avoir bien organisé leurs temps de travail. Néanmoins, dans chaque entreprise il y a des taches qui sont répétitives et on ne peut plus les ignorer sur tous dans notre domaine d’informatique et spécifiquement dans le développement d’application.

Le développeur est chargé au début de chaque lancer d’un nouveau projet de créer et configurer le socle de développement d’application de la partie cliente à la partie serveur en passant par l’outillage, la réalisation du développement d’un projet Web aujourd’hui est un vrai casse-tête et on constate qu’il est coûteux en matière de temps dans le traitement des tâches répétitives qui deviennent rapidement fastidieuses et engendrent une certaine monotonie dans l’équipe.

D’un autre côté, le développeur SOFRECOM n’a pas l’autorisation d’utiliser les solutions existantes sur le marché pour gagner du temps lors de développement et création du projet ainsi que ces derniers ne répondent pas aux certains besoins spécifiques d’équipe développement SOFRECOM.

Au final, nous nous trouvons face à un modèle de travail très long et non stable. Notre objectif est de concevoir et mettre en place un système permettant de minimiser le cycle de travail tout en diminuant la probabilité d’erreur. Ce système doit automatiser les tâches répétitives.

C’est ici que s’inscrit notre projet de fin d’études qui consiste à la mise en place d’un générateur d’application web normalisée avec la culture devops.

**Solution proposée**

Après avoir établi une étude approfondie des solutions existantes sur le marché, il est donc primordial au regard des limites de ces systèmes présents, de proposer une solution qui pourra répondre aux exigences d’équipe SOFRECOM.

Le but ultime du notre projet c’est de développer un générateur qui permet de générer et construire d’autres applications web qui sont personnalisés et répondent aux normes, critères et besoins d’équipe SOFRECOM dans de brefs délais.

Il est important de préciser qu’il est un générateur d’applications Angular et Spring Boot répondant aux besoins d’utilisateurs. Avec notre générateur, le développeur peut créer soit une application Front End, Back End ou bien Full Stack. Une série de choix sera proposé au développeur, lui permettant de sélectionner les outils dont son application aura besoin. Notre-solution-offre-des applications qui regroupent un-Back solide- et-performant avec java-et le framework-spring boot, un front moderne et élégant avec angular et boosted. La performance et la puissance de ce générateur se montre dans la combinaison de ces derniers frameworks.

Ainsi que notre générateur fournit une application complète et moderne, avec différentes options (IHM, la couche service et le domaine métier, la couche de persistance et le schéma de base de données), en plus de crud d’entités, d’autres part il support des bases de données No SQL et SQL et finalement on ne peut pas négliger que notre générateur support toute la culture devops dont la containerisation, l’orchestration d’application, l’intégration et déploiement continue.

**III. Méthodologie adoptée**

Le succès d’un projet dépend de son adéquation au processus de développement. Le choix d’une méthodologie constitue une étape décisive pour l’élaboration d’une application indépendante de toute plateforme d’exécution et de tout langage de programmation.

**Méthodes agiles**

Les approches traditionnelles de gestion de projet reposent sur un processus séquentiel et prédictif de développement. Ces approches attendent du client une expression détaillée des besoins dès le début du projet. Suite à l’établissement du cahier des charges, le processus de développement commence et le client n’intervient qu’à la fin pour vérifier la conformité du projet délivré avec le cahier des charges. Ceci réduit énormément l’implication du client dans le projet et génère un effet tunnel qui peut être néfaste et conflictuel, ainsi, on constate souvent un déphasage entre les besoins exprimés et l’application réalisée. Pour résoudre ce problème, il faut adapter une approche plus flexible ; L’approche « Agile » définit un groupe de pratiques de pilotage et de réalisation de projets reposant sur un modèle semi-itératif incrémental. Cette méthode permet une implication maximale du client et une grande réactivité à ses demandes. Pour chaque itération, il faut choisir premièrement un sous-ensemble des besoins puis concevoir, implémenter et tester.

source : <https://www.fynedc.com/journal/approche-agile-vers-une-gestion-de-projet-plus-collaborative-et-transparente>

**Scrum**

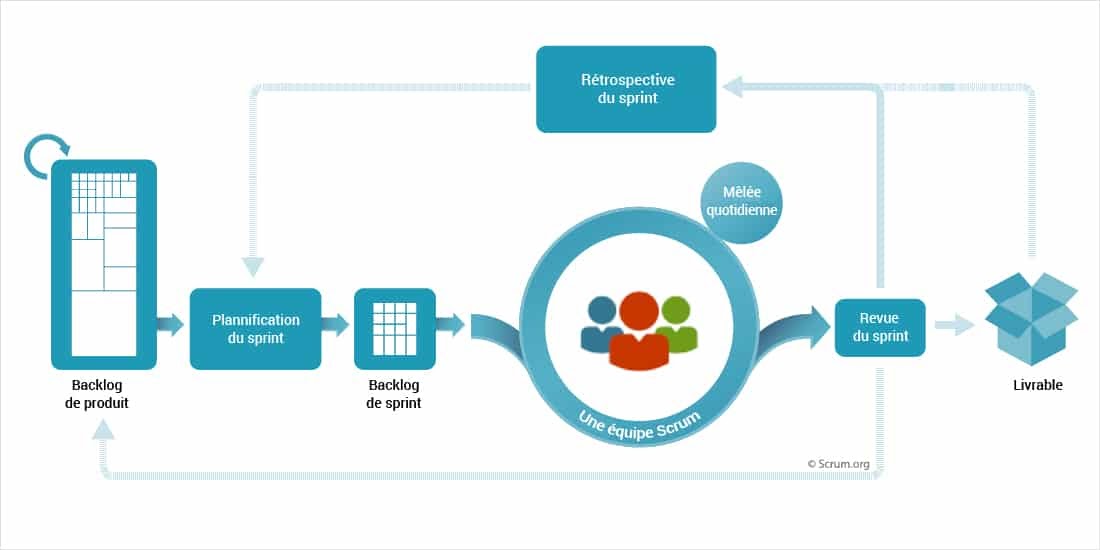
Les méthodes de développement agile SCRUM, aussi appelés « développement adaptatif » signifie en anglais « mêlée », est une méthode itérative et inductible qui permet de changer les objectifs du projet à n’importe quelle phase d’avancement tout au long de sa fabrication. Le SCRUM est fondé sur la motivation, la cohésion et la rivalité de l’équipe afin d’améliorer la productivité pour atteindre un objectif commun.

Avant toute chose, SCRUM est une méthode où certains rôles sont à attribués à des membres du projet qui vont permettre de superviser des parties précises. On distingue donc :

* Le product Owner : il s'agit du gestionnaire du produit. C'est le propriétaire du produit, celui qui va définir les caractéristiques finales souhaitées pour le produit.
* Le SCRUM master : il s'agit du "maître SCRUM". C'est lui qui aura pour but de superviser les membres en les aidant, en les encourageant, en facilitant le partage, la communication des informations essentielles. Il se rapproche du chef de projet mais cela n'est pas exactement son rôle puisqu'il va insister sur le côté de gestion spécifique à SCRUM. Grossièrement, il est le représentant et le premier élément qui illustre SCRUM au sein de l'entreprise.
* L'équipe de développement : elle recouvre l'ensemble des ingénieurs, techniciens etc... En charge de la réalisation du produit.

Source : <https://www.supinfo.com/articles/single/2912-%20pourquoiutiliser-methode-scrum-votre-projet>

La Figure 1.3 présente le principe du développement itératif



**Figure 1.3: Principe du développement itératif**

**source :** [**https://www.nutcache.com/fr/blog/nutcache-ajoute-le-cadre-scrum-son-application/**](https://www.nutcache.com/fr/blog/nutcache-ajoute-le-cadre-scrum-son-application/)

**Méthode de conception :**

* **UML**

Nous avons utilisé, pour la spécification et la conception de ce travail, le formalisme UML (Unified Modeling Language) est une méthode de modélisation orientée objet développée en réponse à l’appel à propositions lancé par l’OMG (Object Management Group) dans le but de définir la notation standard pour la modélisation des applications construites à l’aide d’objets.

Source : <http://helios.mi.parisdescartes.fr/~bouzy/Doc/UML-NotesCours.pdf>

**Conclusion**

 Dans ce chapitre introductif nous avons présenté l’entreprise au sein de laquelle nous avons réalisé notre travail. Nous nous sommes basées sur une étude de l’existant pour nous lancer dans ce projet. Nous avons mis en évidence les modules qui représentent le cœur de notre projet. Ensuite nous avons fini par illustrer la méthodologie que nous avons optée. Le chapitre suivant sera consacré à l’analyse et la spécification des besoins.

**Chapitre 2 : Analyse et spécification des** **besoins**

**Introduction**

Après avoir présenté le cadre général de notre projet, la problématique et les objectifs, nous entamons dans ce chapitre une phase cruciale dans le cycle de développement de notre portail qui va nous permettre d’identifier les acteurs, de formaliser les besoins escomptés des utilisateurs, et de déterminer les fonctionnalités du système.

Ainsi, nous présentons dans ce qui suit une étude des besoins fonctionnels et non fonctionnels.

Pour ce faire, nous exploitons les diagrammes UML comme une modélisation des cas d’utilisation du système ainsi que les scénarios d’utilisation et les séquences d’interaction entre acteurs et système.

**Spécification des besoins**

Dans cette partie, nous entamons la spécification des exigences à laquelle doit répondre notre système. Cette phase a pour but d’identifier les acteurs et de décrire les différents besoins fonctionnels et non fonctionnels.

**Les besoins Fonctionnels**

Cette partie est consacrée à la définition des besoins fonctionnels de notre application. Les besoins fonctionnels dégagés se résument dans les points suivants :

* Authentification
* La génération d’application Front, Back et Full Stack moderne et complète avec les options et les choix d’utilisateur
* L’automatisation des tâches routinières et répétitives : notre générateur s’occupe de processus d’authentification avec JWT, si l’utilisateur a choisi de généré une application full stack notre solution prend en charge la communication entre la partie front et la partie back, ainsi que la génération de crud d’entités, la création des components de base login et home, la couche service et le domaine métier, la couche de persistance et le schéma de base de données.
* L’implémentation de la culture Devops : de plus notre générateur chargé de la containerisation, orchestration, l’intégration continue et déploiement continue de l’application à générer.

**Les besoins non Fonctionnels**

Outre les besoins fondamentaux, notre application doit assurer les besoins non fonctionnels pour valoriser et améliorer la qualité de notre générateur. Donc, il faut que notre générateur assure et répond aux critères suivants :

* Ergonomie : Les interfaces fournit par notre système doivent être compréhensibles, simples, conviviales et sur tout répondre à la charte graphique d’Orange pour garantir la facilité d’utilisation des différents fonctionnalités offert par notre solution.
* Fiabilité et extensibilité: Notre générateur doit assurer la sûreté de fonctionnement de diverses fonctionnalités sans défaillance ainsi qu’il doit envisager des possibilités d’ajout de nouvelles fonctionnalités pour enrichir notre application.
* Maintenabilité : Le code source de notre application doit être maintenue, commenté et documenté pour assurer la facilité de mise à jour et réutilisation.
* Rapidité : Il est obligatoirement nécessaire que notre système génère et traite une application dans des brefs délais.

**Identification des acteurs et des cas d’utilisations :**

Les cas d’utilisations servent à modéliser les besoins et les fonctionnalités nécessaires des clients d’un système dont le but de structurer, filtrer et clarifier les besoins d’utilisateur. Les principaux composants des cas d’utilisation sont :

* Acteur : C’est une entité externe qui interagit avec le système (opérateur, autre système. . .), il peut consulter ou modifier l’état du système, ce dernier fournit un service qui correspond à son besoin.
* Cas d’utilisation : Peut-être définit comme l’ensemble des actions réalisées par le système, en réponse à une action d’u acteur. L’ensemble des cas d’utilisations décrit les objectifs (le but) du système

Source : <http://uml.free.fr/cours/p10.html>

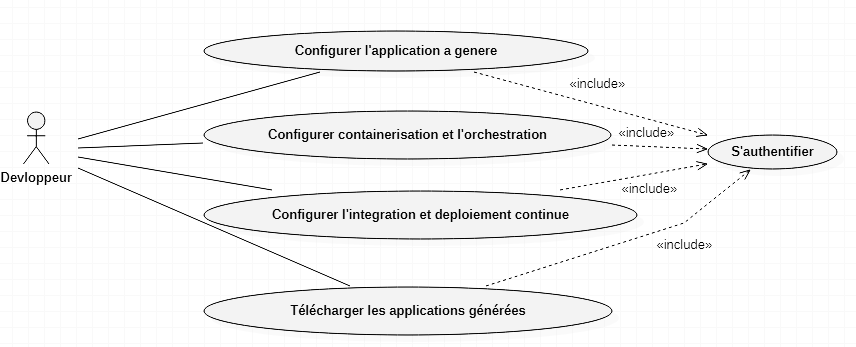
**Description détaillée d’acteur :**

La détermination des acteurs d’un système permet en premier temps d’analyser, délimiter et de comprendre l’interaction du système avec les acteurs en précisant le rôle de chaque acteur et de ses exigences. Dans notre cas, il existe un seul acteur principal et fondamental dans notre application.

* Développeur : L’acteur principal de notre application c’est le développeur d’équipe SOFRECOM. Ce dernier aura l’accès au générateur pour créer et construire une application répondre aux besoins spécifiques d’utilisateur. Il s’occupe de choisir, remplir et sélectionner les informations et les outils nécessaires son application aura besoin.

**Diagramme de cas d’utilisation globale :**





**Organisation des cas d’utilisations :**

Le Tableau 2.1 présente la description textuelle des acteurs et des cas d’utilisations.

**Tableau 2.1:**Identification d’acteur et des cas d’utilisation

|  |  |
| --- | --- |
| **Acteur** | **Cas d’utilisation** |
| **Développeur** | * S’authentifier * Configurer l’application à générer : * Choisir nom d’application et types d’application (application front end, back end, full stack) * Remplir les informations nécessaires pour générer l’application (language, framework, version...) * Choisir type de base de données et remplir les informations nécessaires de cette base (nom de base, serveur, port, nom d’utilisateur, mot de passe…). * Configurer containerisation et l'orchestration d’application générée * Configurer l'intégration et déploiement continue * Télécharger les applications générées. |

**Modélisation du processus métier**

La partie suivante illustre les-principales-solutions-actuelles-pour-la-gestion-des- processus-contribuent-dans-notre- générateur.

**Les BPMN**

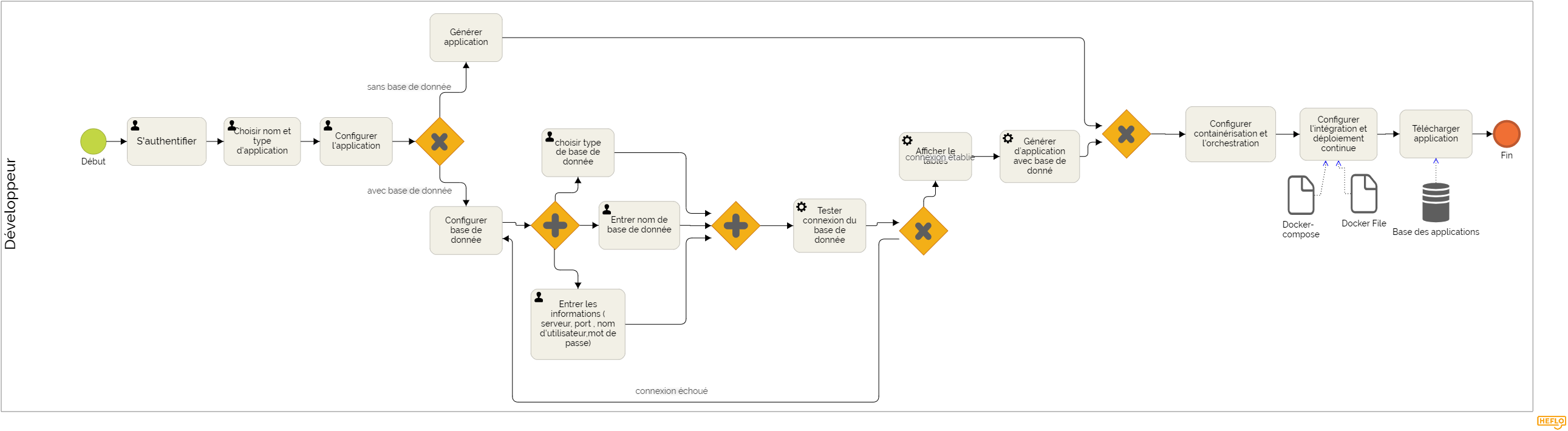
La norme de modélisation des processus métier (BPMN) est une méthode d'organigramme qui modélise de A à Z les étapes d'un processus métier planifié. Élément clé de la gestion d'un processus métier, elle permet de représenter visuellement une séquence détaillée des activités commerciales et des flux d'informations nécessaires à la réalisation d'un processus.

Source : <https://www.lucidchart.com/pages/fr/bpmn>

**Le processus métier**

Ce processus illustre l’enchaînement des activités du notre générateur.

La Figure 2.2 montre le processus de fonctionnement du générateur

****

**Figure 2.2:** Processus de parcours du générateur

Ce-processus-définit-l’ensemble-des-étapes-qui-présentent-le-parcours-d’un-développeur. Pour-ce-faire, ce dernier-doit-suivre-l’enchaînement-suivant :

* S’authentifier
* Choisir nom et type d’application
* Configurer la partie de base de données d’application :
* Si le développeur aura besoin d’une base de données il peut l’implémenter et continuer la configuration de la partie devops.
* Si le développeur n’a pas besoin d’une base de données il continue le processus de génération d’applications avec la partie devops.
* Configurer containerisation et l’orchestration d’application.
* Configurer l’intégration et déploiement continue.
* Télécharger l’application générée.

**Backlog produit et planification des Sprints**

Dans la partir suivante on va présenter tout d’abord le backlog produit qui est le base de la méthodologie scrum ainsi que la planification des sprints.

**Le Backlog de produit**

Un backlog produit représente une liste priorisée des fonctionnalités à développer ou améliorer dans le cadre d’un produit informatique (logiciel, application mobile, etc.). Dans un backlog, chaque user story constitue un item ou une ligne. Ces derniers se présentent par ordre de priorité, et pour chaque histoire, on décrit généralement dans le backlog :

• ID : un identifiant unique, qui s’incrémente à chaque nouvelle histoire, comme une clé primaire dans une BDD. Cet identifiant a pour but de garder la trace des histoires même en cas de renommage.

• User Story : une description de l’histoire prend une forme déterminée, dont une bonne façon de procéder est d’utiliser le Template suivant : « En tant que X, je veux Y, afin de Z ».

• Priorité : l’importance de la story est un nombre qu’attribue le product owner à l’histoire. Plus l’histoire doit être traitée en priorité, plus son importance est élevée.

• Story Point : l’estimation initiale de l’équipe pour cette histoire. Généralement cette case est vide au moment où le product owner insère l’histoire dans le backlog et elle est complétée par la suite avec l’équipe.

Source : <https://www.unow.fr/blog/le-coin-des-experts/product-backlog-projet-scrum/>

Le Tableau 2.2 illustre le backlog produit de l’application.

**Tableau 2.2:** Backlog de produit

|  |  |  |
| --- | --- | --- |
| **Nom** | **Id user story** | **User Story** |
| S’authentifier | 1 | En tant qu’utilisateur je dois être capable de m’authentifier pour m’accéder aux différents services fournis par l’application. |
| Configuration de l’application à générer | 2.1 | En tant qu’utilisateur je veux gérer des applications sous format .Zip |
| 2.2 | En tant qu’utilisateur je veux choisir le type d’application Front End, Back End ou Full Stack. |
| 2.3 | En tant qu’utilisateur je veux choisir le nom d’application |
| 2.4 | En tant qu’utilisateur je veux choisir le nom du package |
| 2.5 | En tant qu’utilisateur je veux choisir le type d’authentification |
| 2.6 | En tant qu’utilisateur je veux choisir le type de base de données (No SQL / SQL) |
| 2.7 | En tant qu’utilisateur je veux choisir la base de données de production (MySQL, Cassandra, mongo dB...) afin de ….. |
| 2.8 | En tant qu’utilisateur je veux consulter la liste des applications générées |
| 2.9 | En tant qu’utilisateur je veux télécharger les applications |
| Génération application Front-End | 3.1 | En tant qu’utilisateur je veux avoir au moins home et login component dans mon application généré |
| 3.2 | En tant qu’utilisateur je veux avoir pour chaque table demandée un component correspond. |
| 3.3 | En tant qu’utilisateur je veux avoir les routing configuré |
| Génération application Back-End | 4.1 | En tant qu’utilisateur je veux avoir les couches persistance, service et model générés dans mon application |
| 4.2 | En tant qu’utilisateur je veux avoir les dépendances nécessaires ajoutées à mon application |
| Génération d’application Full Stack | 5.1 | En tant qu’administrateur je veux avoir une application avec un front moderne, élégant et répondre à la charte Orange ainsi qu’un back solide |
| 5.2 | En tant qu’utilisateur je veux avoir les configurations nécessaires dans l’application (router, dépendances…) |
| 5.3 | En tant qu’utilisateur je veux avoir pour chaque table de base de données des components ainsi que la couche service, controller, entity, repository |
| 5.4 | En tant qu’utilisateur je veux avoir dans l’application générée pour chaque table dans la base de données de crud (Create, Read, Update, Delete) |
| Containerisation de l’application | 6 | En tant qu’utilisateur je veux que mon application soit conteneuriser afin d’isoler l’exécution des applications. |
| Orchestration d’application | 7 | En tant qu’utilisateur je veux que mon application garantie l’orchestration des conteneurs docker |
| Intégration continue | 8 | En tant qu’utilisateur je veuxmon application assure l’intégration continue. |
| Déploiement continue | 9 | En tant qu’utilisateur je veux que mon application assure le déploiement continu. |

**La planification des Sprints**

Nous présentons maintenant l’une des phases les plus importantes dans un projet Scrum. Dans cette partie on va mettre en contexte la planification du projet ainsi que la durée de réalisation de chaque sprint.

Le Tableau 2.3 illustre la planification des Sprints de notre projet.

**Tableau 2.3:** Tableau de planification des Sprints

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Release | Sprint | Description de sprint | Date début | Date Fin |
| Release 1 : génération et téléchargement d’application Simple | Sprint 1 | Authentification et génération des structures basiques des applications Front-End, Back-End, Full Stack | 03/02/2020 | 28/02/2020 |
| Sprint 2 | Télécharger les applications générer | 02/03/2020 | 21/03/2020 |
| Release 2 : génération d’application avec base de données | Sprint 3 | Génération des applications avec les bases de données | 23/03/2020 | 23/05/2020 |
| Release 3 : Devops | Sprint 4 | Containerisation, Orchestration, déploiement et intégration continue d’application | 25/05/2020 | 30/06/2020 |

**Conclusion**

Dans ce chapitre on a précisé les besoins de notre générateur, les différentes fonctionnalités mentionnées dans le backlog du produit ainsi qu’on a identifié l’acteur principal de notre application. Outre, on a modélisé le processus métier du notre générateur pour donner une idée claire sur le workflow du notre générateur.

Le chapitre suivant est

**Chapitre 3 : Initialisation du projet**

**Introduction**

Après la planification des releases et sprints dans le chapitre précédent, nous allons maintenant introduire notre projet en mettant en évidence l’architecture technique et logicielle de notre application, dans un deuxième temps nous avons illustré une conception détaillée, et pour finir nous allons spécifier l’environnement de travail que nous avons adopté.

**Architecture du générateur**

Dans cette section, nous allons présenter l’architecture du notre générateur. L’architecture d’une application informatique se définit par sa décomposition en plusieurs-couches dont le but de rependre aux besoins déjà définit précédemment, diminué les dépendances entre les services.

**Architecture technique**

Nous avons suivre une architecture 3 tiers dans notre générateur pour garantir la meilleur performance et cout.

**La Figure 3.1** présente l’architecture technique du générateur.

**Architecture logicielle**

L’architecture logicielle concerne la répartition et définition des éléments d’application ainsi que la division logique de système et le regroupement des composants selon le traitement.

**La Figure 2.3** illustre l’architecture logicielle de notre système

**Environnement de travail**

La partie suivante est d’idée pour présenter l’environnement matériel et logiciel d’application qui nous permettons de réaliser notre générateur.

**Environnement matériel**

Pour réaliser notre système, nous avons utilisé comme environnement matériel un poste de travail ayant les caractéristiques suivantes :

Le Tableau 3.1 représente la description de l’environnement matériel utilisé.

**Tableau 3.1:** Table descriptive de l’environnement matériel

|  |  |
| --- | --- |
| **Caractéristique** | **Type** |
| **Processor** | Processeur Intel Core i5-4210U |
| **Mémoire installée (RAM)** | 16 Go |
| **Disque dur** | 500Go |
| **Système d’exploitation** | Windows 10 Entreprise |

**Environnement logiciel**

Pour mettre en évidence notre générateur, nous avons eu recours aux ensembles des logiciels qui sont :

* **Visual Studio code (VSC) :** Visual Studio Code est un éditeur de code source léger mais puissant qui s'exécute sur votre bureau et est disponible pour Windows, MacOs et Linux. Il est livré avec un support intégré pour JavaScript, Type Script et Node.js et possède un riche écosystème d'extensions pour d'autres langages (tels que C ++, C #, Java, Python, PHP, Go) et des runtimes (tels que .NET et Unity).

Source : <https://code.visualstudio.com/docs>

* **IntelliJ IDEA :** IntelliJ IDEA est un environnement de programmations spéciales ou un environnement de développement intégré (IDE) largement destiné à Java. Cet environnement est utilisé notamment pour le développement de programmes.

Source : <https://www.techopedia.com/definition/7755/intellij-idea>

* **Mongo DB :** est un serveur de base de données No SQL où les données sont stockées dans des documents JSON. Il assure une haute disponibilité, une distribution géographique et une mise à l’échelle puisque la base de données est distribuée.

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

* **PostgreSQL :** est un puissant système de base de données relationnelle objet open source qui utilise et étend le langage SQL combiné à de nombreuses fonctionnalités qui stockent et adaptent en toute sécurité les charges de travail de données les plus complexes.

**Source :** <https://www.postgresql.org/about/>

* **MySQL :** MySQL, le système de gestion de base de données SQL Open Source le plus populaire, est développé, distribué et pris en charge par Oracle Corporation.

<https://dev.mysql.com/doc/refman/8.0/en/what-is-mysql.html>

* **Apache Tomcat :** Apache Tomcat « 9 » est un outil de serveur Web open source développé par Apache Software Fondation (ASF). C'est l'un des nombreux produits open source liés à Apache utilisés par les professionnels de l'informatique pour diverses tâches et objectifs. Il permet d’exécuter des servlets et des pages serveur java (JSP).

Source : <https://www.techopedia.com/definition/15735/apache-tomcat>

* **Angular CLI :** Cela signifie Angular Command Line Interface. Comme son nom l’indique, c’est un outil de ligne de commande pour créer des applications angulaires.

<https://www.quora.com/What-is-Angular-CLI>

**Framework et Outils**

* **Angular (version 8) :** Angular est une plate-forme et un Framework développé par Google pour créer des applications clientes à page unique en utilisant HTML et TypeScript. Angular est écrit en TypeScript. Il implémente les fonctionnalités de base et facultatives sous la forme d'un ensemble de bibliothèques TypeScript que vous importez dans vos applications.

Source : <https://angular.io/guide/architecture>

* **Spring Boot :** est un Framework libre utilisé pour faciliter le développement des applications basées sur Spring.

Spring Boot simplifie :

-la création des applications Spring en offrant des outils permettant le développement des applications autonomes packagées en jar, war ... ;

-La configuration en fournissant des dépendances de démarreur ;

-Le développement en offrant des fonctionnalités prêtes à la production ( la configuration externalisée, les mesures ...).

Source : <https://spring.io/projects/spring-boot>

* **Boosted Orange :** Orange Boosted est une bibliothèque HTML, CSS et JS d'Orange de Framework accessible, ergonomique et de marque Orange basée sur Bootstrap 4.5.0,la boîte à outils open source frontale la plus populaire au monde.

Source : <https://boosted.orange.com/>

**Langages de programmation**

* **Java :** La technologie Java définit à la fois un langage de programmation orienté objet et une plateforme informatique. Crée par l'entreprise Sun Microsystems, la technologie Java est indissociable du domaine de l'informatique et du Web.

Source :<https://www.journaldunet.fr/web-tech/dictionnaire-du-webmastering/1203555-java-definition/>

* **TypeScript :** Le TypeScript est un langage de programmation développé par Microsoft en 2012. Son ambition principale est d’améliorer la productivité de développement d’applications complexes.

C’est un langage open source, développé comme un sur-ensemble de JavaScript. Ce qu’il faut comprendre par-là, c’est que tout code valide en JavaScript l’est également en TypeScript.

Source : <https://blog.cellenza.com/developpement-specifique/web-developpement-specifique/introduction-a-typescript/>

* **HTML5 :** C’est un langage de balisage conçu pour la création des pages Web et des applications Web utilisé dans la vue d’un composant Angular.

Source : <https://docs.microsoft.com/fr-fr/dotnet/architecture/modern-web-apps-azure/common-client-side-web-technologies#:~:text=HTML%20est%20le%20langage%20de,Web%20et%20des%20applications%20Web.&text=Ce%20document%20HTML%20peut%20r%C3%A9f%C3%A9rencer,la%20forme%20de%20code%20JavaScript.>

**Conclusion**

Dans ce chapitre, nous avons décrit l’architecture du notre application. En outre, nous avons mis en évidence notre environnement de travail matériel et logiciel suivant les besoins de notre projet. Dans le chapitre suivant nous allons entamer la partie réalisation, nous présentons en premier lieu le release 1.

**Release 1 : Génération et téléchargement d’application** **Simple**

**Introduction**

Au cours de ce chapitre, nous allons présenter la première phase de réalisation du générateur. Ce dernier regroupe les deux premiers Sprints que nous avons réalisés.

Nous allons commencer par le sprint 1 qui a pour l’objectif d’établi le module d’authentification ainsi que la génération des applications simples. Ensuite, nous attaquons le sprint 2 dont lequel on va présenter la partie de la consultation et téléchargement des applications générées.

**Sprint 1 : Authentification et génération des applications simples**

Ce Sprint consiste la phase initiale du notre générateur. Ce dernier présente la partie d’authentification qui est une étape primordiale et fondamentale avec laquelle on peut accéder au générateur.

D’autres parts, la génération des applications simples front, back ou full stack avec les components de base nécessaires.

**Backlog du Sprint 1**

Le Tableau 4.1 montre le backlog Sprint de l’authentification et génération d’applications simples.

**Tableau 4.1:** Backlog du Sprint 1

|  |  |  |  |
| --- | --- | --- | --- |
| **Id Story** | **Story** | **Taches** | **Jours** |
| 1.1 | En tant qu’utilisateur je dois être capable de m’authentifier pour accéder à la plateforme. | Implémentation de la classe génératrice de Tokens JWT | 3 |
| Implémentation des méthodes de configuration (configure, CORS, etc.) |
| Implémentation du web service REST d’authentification. |
| 1.2 | En tant qu’utilisateur je veux me déconnecter de l’application | Création et l’implémentation du bouton logout. | 3 |
| Synchronisation de la déconnexion de Spring Security avec le web service logout. |
| 1.3 | En tant qu’utilisateur je veux générer des applications Front-End, Back-End et Full-stack | Développement de la partie front permettant l’initialisation du nom et le type d’applications. | 5 |
| Développement du web service pour récupérer nom et type d’application dans le backend. |
| 1.4 | En tant qu’utilisateur je veux avoir une application Front End générée qui assure au moins home et login component | Création du page front end assure le choix du Framework et la version a utilisée dans la génération d’application. | 7 |
| Développement du service coté backend qui récupère les données nécessaires pour la génération d’application Front. |
| Développement du service backend pour générer des components de base login et home page, ainsi que la page html 404-not found. |
| Développement du service backend pour configurer des routes nécessaires dans « app.routing.module.ts » et « app.module.ts » pour garantir le bon fonctionnement d’application. |
| 1.5 | En tant qu’utilisateur je veux avoir une application Back End générée avec les couches entité, service, persistance et Controller | Développement de la partie Front qui aide l’utilisateur à faire les choix nécessaires pour générer une application Back End. | 7 |
| Développement d’un service qui récupère les choix et les données d’utilisateurs fondamentaux pour générer une application Back end. |
| Développement d’un service coté backend qui génère la couche entité d’application |
| Développement d’un service coté backend qui génère la couche entité d’application |
| Développement d’un service coté backend qui génère la couche service d’application |
| Développement d’un service coté backend qui génère la couche persistante d’application |
| Développement d’un service coté backend qui génère la couche controlleur d’application |
| 1.6 | En tant qu’utilisateur je veux que le type d’authentification d’application générée soit JWT | Développement d’un service backend qui génère l’implémentation de la classe génératrice de Tokens JWT, des méthodes de configuration et du web service REST d’authentification coté backend d’application générer. | 5 |

**Analyse**

Après avoir bien comprendre les techniques utilisées dans le premier Sprint. Nous allons maintenant expliquer les termes techniques et fondamentaux ayant mentionnée dans ce Sprint.

**JWT :** JSON Web Token est une norme ouverte qui définit une manière compacte et autonome de transmettre en toute sécurité des informations entre les parties en tant qu'objet JSON

Source : <https://jwt.io/introduction/>

**Token :** Les tokens d’authentification (ou jetons d’authentification) représentent un concept puissant d’authentification. Il permet d’autoriser des utilisateurs à entrer leur nom d’utilisateur et leur mot de passe afin d’obtenir un jeton qui leur permet d’accéder à une ressource spécifique sans réutiliser un nom d’utilisateur et un mot de passe. Une fois son jeton obtenu, l’utilisateur peut profiter du jeton qui offre un accès à une ressource spécifique pour une période de temps précise sur un site distant.

Source : <https://www.supinfo.com/articles/single/2759-tokens-authentification>

**Conception**

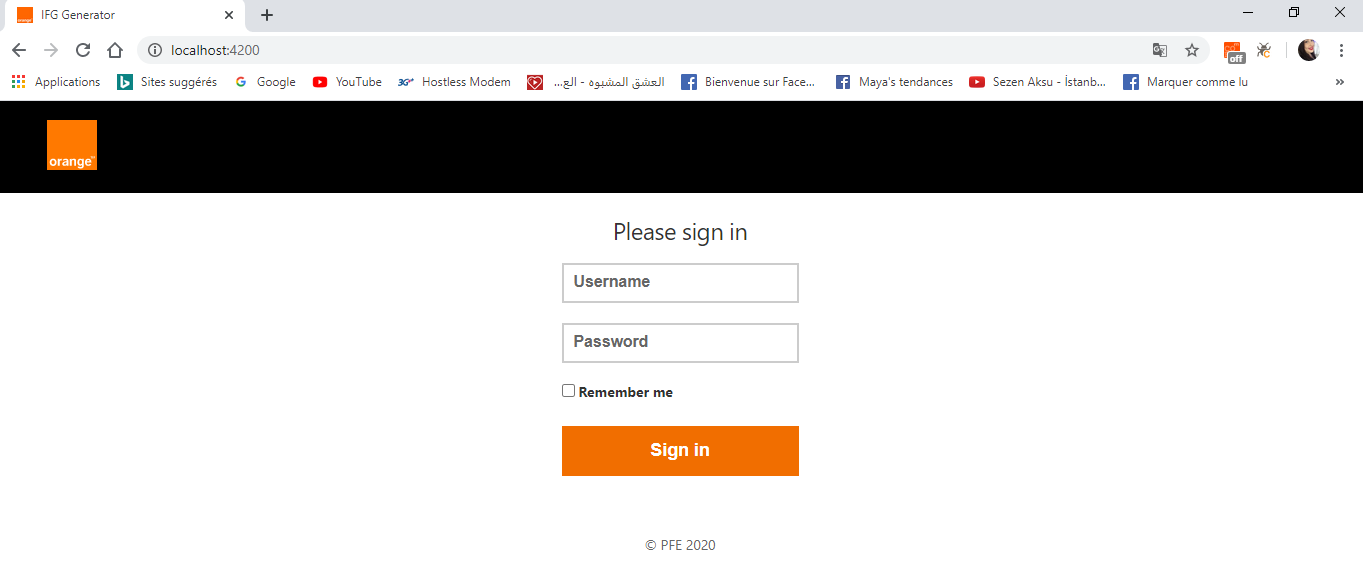
**Diagramme de séquence objet du cas « S’authentifier »**

Le diagramme de séquence objet a pour le but d’illustrer les interactions entre les différents objets de notre système. La figure 4 représente le diagramme de séquence de l’authentification.

**Réalisation**

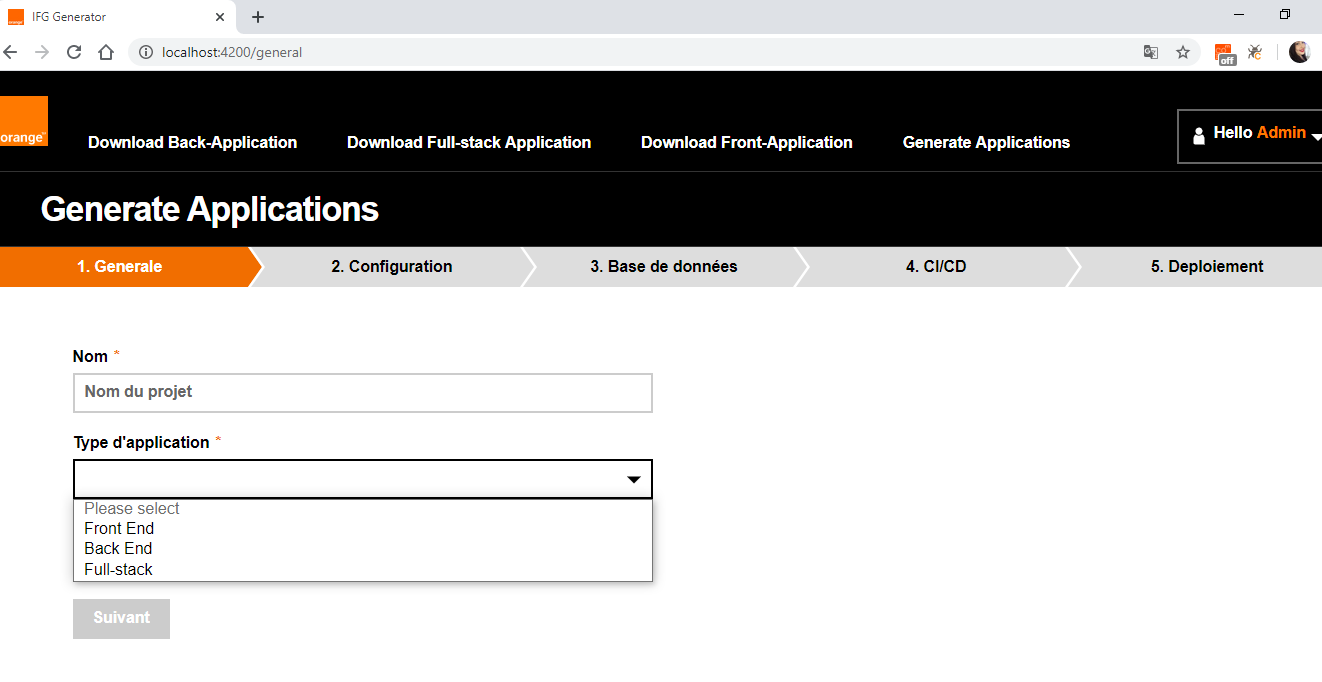
Nous illustrons dans ce qui suit les interfaces qui répondent aux besoins déjà spécifiés dans le backlog Sprint. En premier lieu, nous présentons dans la Figure 4.1 l’interface d’authentification avec laquelle l’utilisateur peut accéder au notre générateur.

**La Figure 4.1** représente l’interface d’authentification



**Figure 4.1:** Interface d’authentification

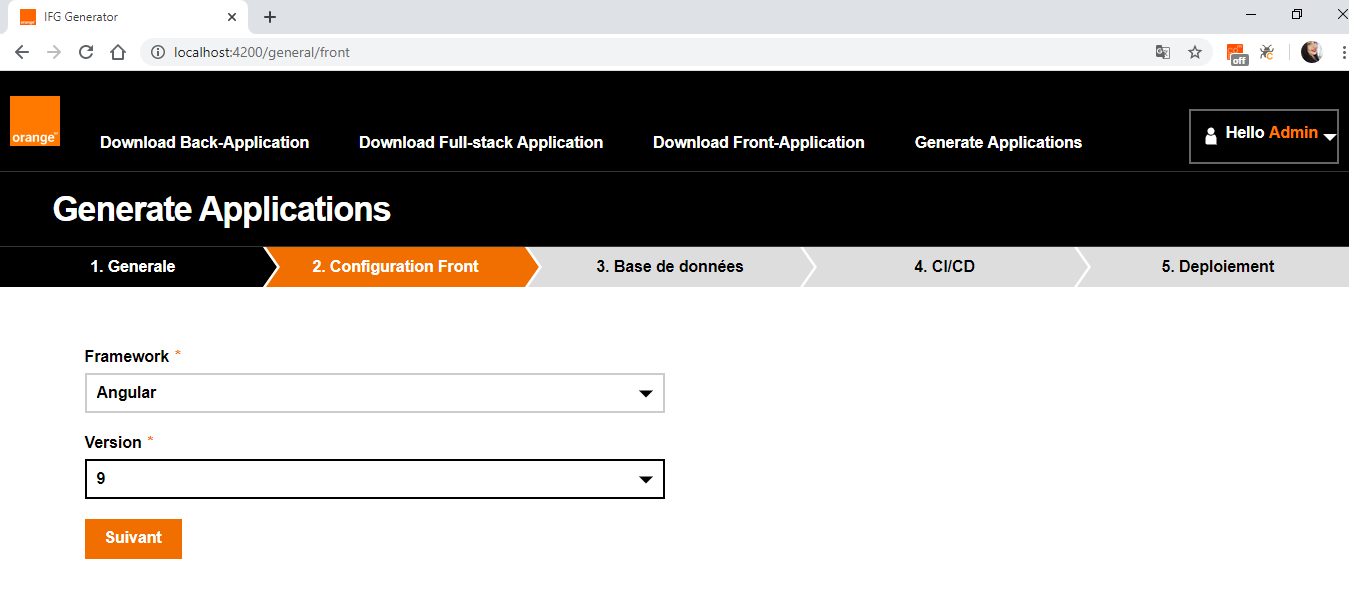
En outre, l’interface suivante 4.2 représente la première étape de la configuration d’application dont laquelle l’utilisateur choisir le nom ainsi que le type d’application a généré.



**Figure 4.2:** Interface

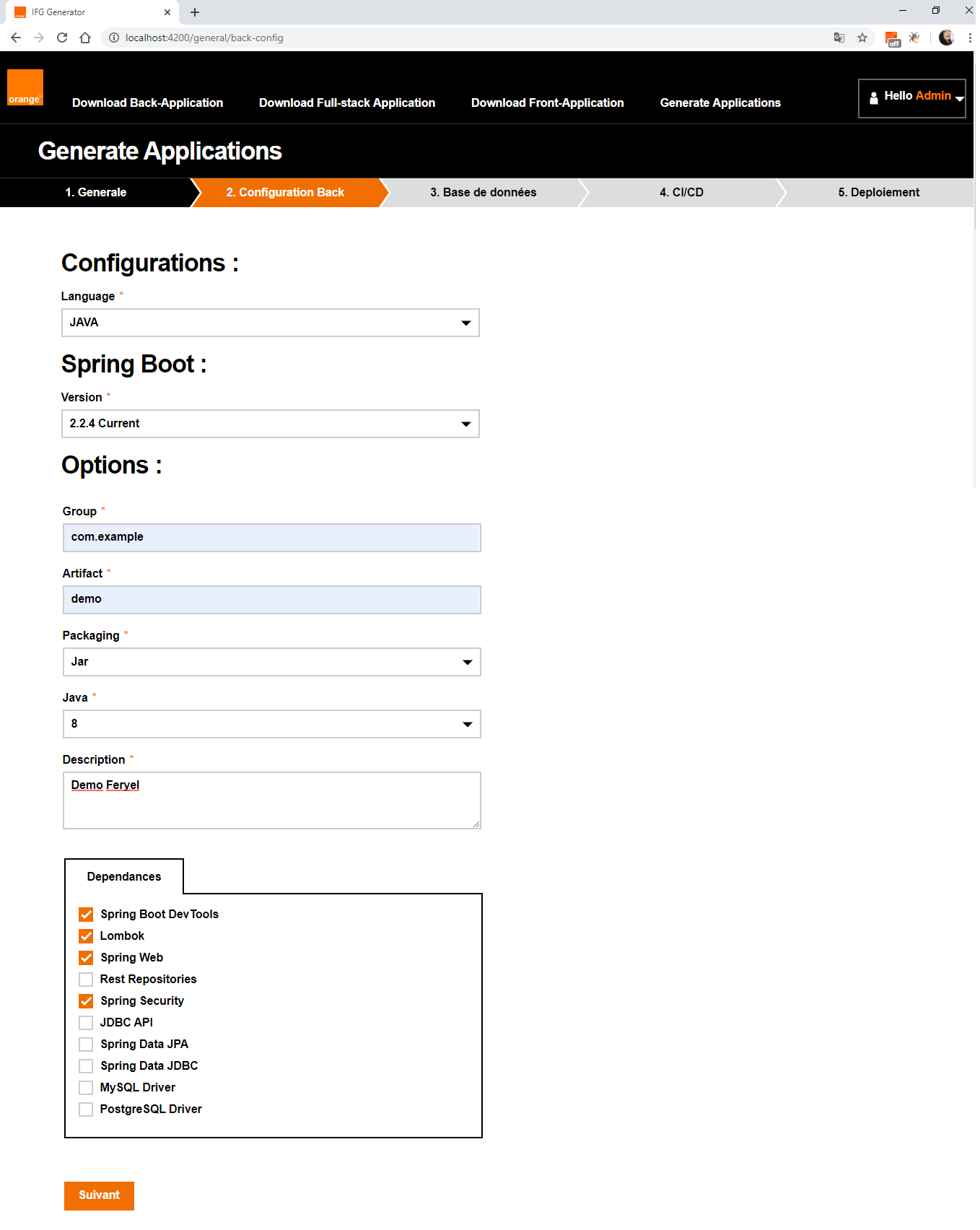
Après avoir choisir le type et le nom d’application à générer, le développeur est chargé maintenant de suivre la configuration d’application. Une liste de configuration nécessaire s’affiche à l’utilisateur suite à son choix concernant le type d’application.

Si l’utilisateur a choisi du générer une application Front End, donc il est chargé de choisir le type de Framework ainsi que la version comme il est illustré dans la figure 4.3.



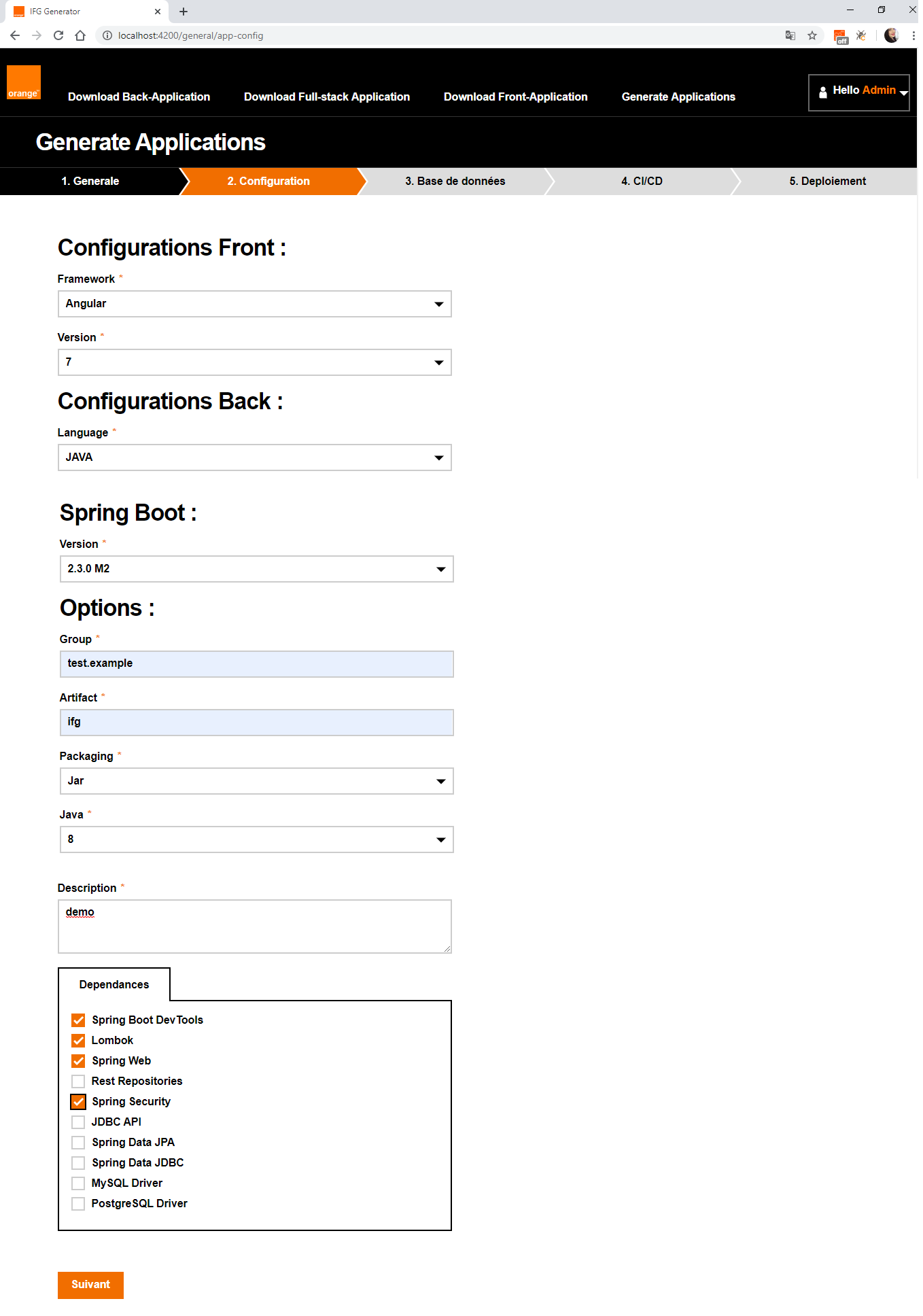
**Figure 4.3:** Interface de la configuration d’application Front

Sinon si pour que l’utilisateur génère une application Back End simple, il faut qu’il remplisse la liste des choix nécessaires présenté dans la figure 4.4.



**Figure 4.4:** Interface de la configuration d’application Back

Et pour finir, si l’utilisateur a choisi de générer une application Full-Stack, il doit remplir le formulaire suivant pour garantir la génération d’application selon ses choix.



**Figure 4.5:** Interface de la configuration d’application Full-Stack

**Sprint 2 : Télécharger les applications générer**

L’objet principale de ce Sprint c’est de garantir le téléchargement d’application généré selon les choix et les besoins d’utilisateur sous forme Zip.

|  |  |  |  |
| --- | --- | --- | --- |
| **Id Story** | **Story** | **Taches** | **Jours** |
| 1.8 | En tant qu’utilisateur je veux consulter les détails des applications générées | Création de la partie front end permettant l’affichage des applications dans des datables selon le type d’application |  |
|  |
|  |
| 1.9 | En tant qu’utilisateur je veux télécharger les applications générées à tout moment |  |  |