

SI & Solution Architect

COMPÉTENCES

Langages maîtrisés	C, C++, JAVA, JAVASCRIPT, TYPESCRIPT, CSS, SQL, PHP, SHELL (UNIX), DART, RUST
Frameworks	SPRING BOOT, SPRING, SPRING DATA / BATCH / MVC / SECURITY / CLOUD, BACKBONE.JS, JPA, ANGULAR 5, REACTJS, NEXTJS, GATSBY, VERT.X, QUARKUS, ACTIVE MQ, RABBITMQ, HAZELCAST, KAFKA, JAX-RS, JAX-WS, LIQUIBASE / FLYWAY, JOOQ, QUERY DSL, LOGBACK, OPENAPI 3 / SWAGGER, FLUTTER, JQUERY, TYPESCRIPT, MAVEN, GRADLE, TOMCAT, NGINX, MOCKITO, MOCKMVC, TESTCONTAINER, JUNIT, CONCORDION,
Databases	POSTGRESQL, MYSQL, ORACLE, NEO4J, MONGODB, CASSANDRA, COUCHBASE
DevOps & CI/CD	GIT, GITLAB, BITBUCKET, GITHUB, JENKINS, DOCKER, PODMAN, SONAR, ARTIFACTORY, NEXUS, RANCHER, HARBOR
Outils de développement	INTELLIJ IDEA, CURSOR / VSCode, CLAUDE, GITHUB COPILOT, MPC
Tests & Performance	SPRING-TEST, TEST-CONTAINER, MOCKMVC, MOCKITO, TESTCONTAINERS, JUNIT, JMETER, JMH, VISUAL VM, SOAPUI, GATLING, CONCORDION, PUPPET
Méthodes	Agile (SCRUM/SAFE), Best Practices : Design Driven Development, Hexagonal Architecture, Dev. Practices (KISS, SOLID, ACID...)

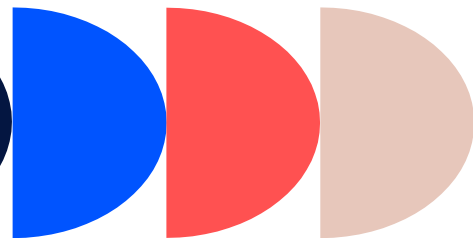
DIPLÔMES

2010 - 2014 **Ingénieur - Epitech Paris**

AUTRES

Langues Anglais

Projets personnels MyDoc - 2019 : Application lourde mobile de gestion documentaire à travers le réseau social (PeopleDoc like) - Flutter (dart) / SQLite / JDK8 / Spring / Neo4j / JWT / Liquigraph / Vertx / Jax-rs / Let's encrypt (SSL) + AES / Nginx / Hébergement VPS



ARCHITECTE SOLUTION & SYSTEME D'INFORMATIONS

Descriptif du projet **Architecte sur le programme FrontToBack** : Travail en collaboration avec les équipes opérationnelles et SDSI en charge de la réalisation des applications FRONT business (parcours vente et intégration SI) – programmes TCC/F2B (Tenue de Compte Considération) & Espace Client
+50 interlocuteurs différents (Directeurs Programme TCC/F2B/Socle/Espace Client, Pôle Finance, Abeille, Production, RMSPT – Desktop & Tools, Business Analysts, Tech-leads, développeurs, archi sécu/CapSecu/RSSI, DBA/Pôle Data...)

- Missions**
- Consolider l'équipe d'architectes sur le programme CORE Business du SDSI au sein du cabinet d'études (Pôle Archi – 3 architectes)
 - Rétro-documenter l'architecture existante Legacy
 - Documenter les solutions archi du programme dans le cadre du programme de transformation digitale de l'UFF (Digitalisation des processus manuel/papier)
 - Présenter les solutions ayant un impacte sur l'architecture du SI en comité d'archi + Design Authority afin de valider et circulariser les décisions
 - Arbitrer mon intervention sur les différents sujet de mon programme selon les besoins du projet (DP / PO / CP) lorsqu'elle est nécessaire
 - Poser les lignes directrices d'architecture et codes de conduite génériques quant aux choix technologiques, ou solutions à adopter sur un sujet
 - Rationaliser les processus fonctionnels afin de limiter/supprimer la dette technique et garantir les mêmes tuyaux/routes pour toute intégration de nouveau partenaire ou sujet similaire
 - Identifier et remonter les problèmes opérationnels récurrents ou blocages prioritaires des équipes projet
 - Circulariser les décisions d'architecture DSI & Transverses aux équipes projet
 - Garantir que les conceptions projet respectent les décisions et choix d'architecture du SDSI
 - Faciliter le cadrage, pilotage et la coordination des équipes lorsque certains projets impliquent de nouvelles dépendances à d'autres
 - Faciliter la communication entre les équipes projet dans un contexte métier et budgétaire tendu (Perte de l'agrément bancaire effective début 2025 : changement de paradigme métier suite à rachat de l'UFF par Abeille Assurances / Réduction budgétaire de la gestion avec PSE / Roadmap SDSI définie uniquement sur le business : pas de rénovation technique du SI)
 - Lead Archi refonte espace client : Réalisation du DAT, étude IAM/CIAM / Stack Technique / Cartographie des flux / habilitations / Mise en conformité réglementaire (DORA/MiFID2/eIDAS/RGPD..)

Environnement

- Agile (SCRUM) / Management matriciel / Rituels
- Archis / Github Copilot / Confluence / JIRA /
- oracle / PostgreSQL / Gitlab / JAVA / Angular /
- Docker / Nginx / M365 / Entra ID / EasyVista /
- Vokse / Mainframe (AS400)

TECH-LEAD (transverse) sur les applications Backend

Descriptif du projet **Système d'applications VEGA** : Migration du système applicatif OMEGA existant vers un système applicatif recodé from scratch dans un environnement micro-services
1 équipe FRONT (Angular – 7 personnes) Plusieurs projets Back (1projet = 1 micro-service entre 3 et 5 personnes par équipe)

- Missions**
- Création de 4 nouveaux micro-services (Rejet / Publication / Index / Demande)
 - Amélioration de la documentation (Installation environnement / pratiques)
 - Normalisation/Propagation de bonnes pratiques HTEOAES / Rest API
 - Généralisation/normalisation de pratiques de développement par consensus et code review
 - Mise en place nomenclature de versions pour les migrations de données (Flyway)
 - Résolution des problèmes de volumétrie :
 - Optimisation de performances des exports CSV (streaming + Jooq)
 - Gestion de la volumétrie des upload Amazon S3 / alimentation Elastic-search
 - Optimisation de requêtes SQL (JPA > Native query via Jooq)
 - Réduction de la dette technique (maintenabilité / performances) :
Intégration d'une nouvelle techno dans la stack Back pour renforcer la maintenabilité/performances des requêtes SQL natives dans les micro-services
 - Garantir la qualité du code (couverture des tests d'intégration > 80%), respect des tirs de perfs (SLA)
 - Reprise du build gradle pour les nouveaux micro-services (mutualisation/isolation des versions utilisées sur la stack technique back) – Réduction du temps de build CI
 - Professionalisation du build : Mise en place Release plugin + BOM
 - Dockerisation de l'environnement technique
 - Tests d'intégration via (JUnit / Zonky / Test-container)
 - Développement de features fonctionnelles (kafka / ES / S3 / WS)
 - Code Review + Rétros MOE

Environnement ● DOCKER, GITLAB, OKTA, SCIM, OPEN-API, SPRING INTEGRATION / SPRING-BATCH / SPRING-CLOUD / SPRING-GATEWAY / JOOQ, AWS (S3), ELK, JAVA (17), KAFKA, MAP-STRUCT/LOMBOK, LOGBACK, POSTGRESQL, ZONKY (Embedded PSQl DB), REDIS, JUNIT, TEST-CONTAINER, MOCKMVC, DROOLS (BRMS), ANGULAR, NGINX, APACHE2, POSTGRESQL, SONAR, JAVA, GRADLE, RENOVATE BOT

DINUM (Direction Interministérielle du Numérique)

avril 2023 - juin 2023

LEAD TECH - CHEF DE PROJET TECHNIQUE (Travail en autonomie)

Descriptif du projet **MISE EN PLACE D'UNE SOLUTION SSO** : Mise en place d'un système d'authentification mutualisé utilisant Keycloak comme Identity Provider pour la gestion des comptes et droits utilisateurs. Test de l'authentification sur une application en développement composée d'une application BACK en python (API rest unicorn) et d'une application FRONT en ReactJs 18 (typescript)
Projet réalisé dans une équipe de 6 personnes

- Missions**
- Proposition de la stack technique avec keycloak / validation de l'architecture
 - Test / POC de keycloak avec oauth2_proxy + nginx sur une application fictive (configuration de keycloak + nginx) : test d'authentification Login/Logout
 - Dockerisation de l'application Front (Nginx frontal pour reactjs) et Back (Nginx frontal pour l'API Rest python) + keycloak + Postgresql (utilisé par keycloak) + keycloak : une image / service docker par techno
 - Test de déploiement sur un environnement d'intégration avec le front dockerisé + Reverse-proxy (nginx) + Champollion (application front Reactjs exposée via nginx) + oauth2_proxy (Client OIDC gérant les sessions et authentifications) + postgresql (BDD keycloak) + keycloak (Identity Provider OIDC/SAML)
 - Déploiement/Test des applications authentifiées sur une seule instance (pré-test avant isolation des composants par instance) : FRONT (nginx dockerisé) / BACK (nginx dockerisé) / KEYCLOAK (quarkus dockerisé) + PSQL (service docker - BDD keycloak)
 - Isolation des composants :
 - Déploiement du service FRONT (service nginx) + service oauth2_proxy (client d'authentification) sur une instance dédiée
 - Déploiement du service BACK (service nginx) + service oauth2_proxy (client d'authentification) sur une instance dédiée
 - Déploiement du service Keycloak (Identity Provider) et sa BDD (service PSQL pour le stockage des utilisateurs) sur une instance dédiée
 - Configuration keycloak : Attribution de privilèges sur des comptes utilisateur de test pour leur donner les droits administrateur (Administration de compte utilisateur sur le realm keycloak configuré)
 - Documentation de la configuration de keycloak + Nginx + oauth2_proxy pour une application web exposée via un serveur Nginx utilisant oauth2_proxy comme client d'authentification OIDC

Environnement • DOCKER, GITLAB, KEYCLOAK, OAUTH2, SWAGGER, PYTHON, REACTJS, TYPESCRIPT, NGINX, POSTGRESQL, QUARKUS, OAUTH2-PROXY

Expert Atlassian - JIRA Insight

Descriptif du projet Reprise de données client (CRM) importées quotidiennement dans JIRA pour la gestion des groupes et résolution d'anomalies
Projet réalisé dans une équipe de 6 personnes

- Missions**
- Résolution de problèmes de filtrages suite à des demandes de client spécifiques lors de la déclaration de tickets par les utilisateurs de différents groupe (Insight Query Language)
 - Résolution de bugs JIRA (création de doublons lors d'imports JIRA via Jira Automation - Mise en place de script groovy pour palier au problème (issue JIRA toujours d'actualité dans les dernières versions)
 - Identification des fiches client provenant de sources différentes (différents CRM) portant le même id
 - Suppression des données inutiles pour reprise progressive des id client
 - Mise en place d'une nomenclature d'ID unique pour identifier les fiches client de manière unique d'un CRM à un autre dans JIRA après un import
 - Consolidation de la documentation et modes opératoires
 - Création d'un environnement JIRA d'intégration
 - Configuration de la BDD + Copie configuration LDAP de la PRE-PROD
 - Configuration nginx
 - Dump de la BDD de prod sur l'intégration
 - Mise en place de script de nettoyage en cas de dépassement d'espace disque
 - Rédaction de la procédure de restauration de l'environnement d'intégration en cas d'urgence

Environnement ● JIRA SERVICE MANAGEMENT, NGINX, SCRIPTS GROOVY, CONFLUENCE

Lead Technique

Descriptif du projet **OSIS-V3** : OSIS-V3 : Création d'une nouvelle application de gestion/saisie de formulaires/documents dynamiques
Projet réalisé dans une équipe de 5 personnes

- Missions
- Proposition de la stack technique + validation de l'architecture multi-tiers (3 tiers : BDD / server (Backend) / client (Frontend en ReactJS))
 - Création du projet back : mise en place architecture modulaire (application multi-module, pattern en layers spring-boot Controller/Service/Repository,DAO) + configuration du projet et implémentation le d'abstraction CRUD
 - Ajout de Liquibase et configuration pour versionner les schémas et données de tests (via base H2) + schemas/données par environnement (création d'un profil maven externalisé par environnement pour credentials BDD)
 - Mise en place de Swagger via OpenApi pour exposition et documentation des web-services Rest destinés aux utilisateurs authentifiés (serveur backend non visible à l'extérieur du parc/réseau ATIH interne - réseau privé)
 - Création du projet front avec ReactJs en typescript via npm/npx + ajout d'un serveur proxy frontal (reverse-proxy) nginx en production et ajout de règles de filtrage pour sécurité nginx (exemple restriction des clients, filtrage de requête, redirections)
 - Obfuscation du code front pour le build de prod (failles XSS) - pour rendre le code inutilisable (javascript-obfuscator)
 - Mise en place de la notion de contexte applicatif (ApplicationContext - cadre d'implémentation) dans le projet FRONT pour héberger les services (via ServiceProvider), certaines données partagées ou variables d'environnement
 - Construction de composants front unitaires et ré-utilisés partout dans l'application (formulaires de saisie, conteneurs de popup, layouts, notifications...)
 - Mise en place de l'authentification SSO via service externe mutualisé CAS (Central Authentication Service) dans le projet backend et frontend
 - Tests d'intégration (Jooq/Mockito/H2/Mysql)
 - Développement de fonctionnalités / évolutions selon SFD par profil (établissement/Régional/National) – 2 modes de navigation : BackOffice ou FrontOffice
 - Mise en place du déploiement CI/CD (création du pipeline multibranche + déploiement automatique sur rancher)
 - Mise en place du dockerfile + jenkinsfile
 - Alimentation de Artifactory avec l'image déployée
 - Déploiement automatique sur les instances de service Rancher

Environnement

- MAVEN, JAVA 11, SPRING BOOT, LIQUIBASE, SPRING-DATA, JPA, JOOQ, MYSQL, MOCKITO, JUNIT, H2, LOGBACK, JACKSON, LOMBOK, TOMCAT, OPENAPI 3, SWAGGER, REACTJS, TYPESCRIPT, NPM, NODEJS, NGINX, DOCKER, GIT, ARTIFACTORY, JENKINS, SONAR, RANCHER, ELK, JIRA, CONFLUENCE

RESILIENCE APPLICATIVE (LEAD)

Descriptif du projet Mise en place d'un document de référence pour la résilience des applications destiné à être décliné par les entités du groupe BNP et commandité par la BCE
Projet réalisé dans une équipe de 5 personnes

- Missions**
- Implémentation des patterns de contre pression (back-pressure) pour la résilience applicative au niveau des services d'une application : configuration et seuils (threshold)
 - Décrire les tests de résilience applicative (Tests d'intégration, Chaos/Monkey tests, objectifs des couvertures de test, tests intelligents et automatisés)
 - Définir les objectifs, considérations et préconisations à adopter dans le cadre de la résilience applicative afin d'assurer la disponibilité/continuité de service d'une application/service
 - Identifier et définir les classes d'applications du domaine bancaire
 - Identifier et cataloguer les cas de disruption applicative pouvant engendrer une indisponibilité de service
 - Définir le contexte de résilience applicative (Scope et pré-requis)
 - Identifier et cataloguer les métriques observables pertinentes et cas d'usage des patterns de résilience applicative pour chaque métrique
 - Identifier et définir l'observabilité dans le cadre de la résilience applicative (APM, NFR/Quality Attributes, SLA/SLO, métriques)
 - Définition des mécanismes de résilience applicative : patterns de contre-pression (back-pressure resiliency patterns)

Environnement ● FIGMA, POWERPOINT, M365, TDD, DDD, SRE, BACK-PRESSURE PATTERNS, NFR, CONFLUENCE, JIRA, SPLUNK, DATADOG

Lead Technique

Descriptif du projet **Dispostock** : Application de gestion de stocks de médicaments et vaccins
Projet réalisé dans une équipe de 5 personnes

- Missions**
- Remplacement de log4j par Logback
 - Migration des appels http vers Spring Webflux
 - Mise en place de tâches schedulées (dépôts de fichiers excel hebdomadaire par sftp, tâches de purge)
 - Intégration des demandes en urgence (vaccins, dispositifs médicaux) : back (multiple services) + front (multiple écrans) + exports excels
 - Intégration de suivi des inventaires de consommable et équipement pour suivre : back (multiple services) + front (multiples écrans) + imports/exports excel
 - Reprise et migration du MCD pour homogénéiser le référentiel d'établissements de santé (Finess)
 - Mutualisation de la remontée des vaccins en un seul endroit avec import en masse des stocks de vaccin (Covid/Grippe/MonkeyPox)
 - Tests d'intégration (ajout de Mockito sur le projet)
 - Mise en place nouveau workflow Git basique et pratique
 - Mise en place de liquibase pour versionner les migrations de données + intégration au workflow CI (Modification DockerFile + pipeline JenkinsFile)
 - Mise en place de release (maven-release-plugin)
 - Mise en place d'une abstraction de service CRUD générique par proxification (permet d'implémenter par défaut les méthodes de CRUD spring-data pour un service et/ou ajouter du comportement fonctionnel à tous les services en héritant, ex : historisation générique des actions CRUD)

Environnement

- MAVEN, JAVA 8, SPRING BOOT, LIQUIBASE, SPRING DATA JPA, JOOQ, MYSQL, MOCKITO, JUNIT, H2, LOGBACK, JACKSON, LOMBOK, TOMCAT, THYMELEAF, JQUERY, DATATABLES, DOCKER, GIT, ARTIFACTORY, JENKINS, SONAR, RANCHER, ELK, JIRA, CONFLUENCE



Lead Technique

Descriptif du projet **Epass-check-finess** : Création d'une application exposée : API Rest permettant de chercher/contrôler/filtrer les établissements de santé provenant d'une base de donnée référentielle
Projet réalisé en autonomie

- Missions**
- Proposition de la stack technique + validation
 - Création du projet
 - Mise en place architecture modulaire (application multi-module)
 - Mise en place de Jooq (création profil maven) / Génération du modèle Jooq
 - Mise en place de liquibase (création profil maven)
 - Séparation des couches en layer (Controller / Abstraction de service / DAO repositories)
 - Intégration d'une base H2 versionnée (liquibase) pour les tests d'intégration + création profil maven
 - Réalisation des tests d'intégration utilisant Mockito et la base H2 dédiée aux tests
 - Développement de différents service de restitution de données (liste d'établissements, recherches par critères, contrôles de validité)
 - Mise en place de la CI/CD
 - Dockerisation de l'application pour le packaging
 - Déploiement de l'application sur Jenkins (pipeline / JenkinsFile)
 - Création du service Rancher + Pointage vers l'application (Service Load-balancer)
 - Utilisation de la nouvelle API Rest par dispostock pour consommer les services exposés (Recherche et contrôles par critères d'établissements de santé)

Environnement ● MAVEN, JAVA 11, SPRING BOOT, LIQUIBASE, SPRING DATA JPA, JOOQ, MYSQL, MOCKITO, JUNIT, H2, LOGBACK, JACKSON, LOMBOK, TOMCAT, OPENAPI 3, SWAGGER, DOCKER, GIT, ARTIFACTORY, JENKINS, SONAR, RANCHER, ELK, JIRA, CONFLUENCE

Année de césure

avril 2019 - décembre 2020

Expériences de voyage

Descriptif du projet Madère, Cuba, Allemagne, Autriche, Slovénie, Croatie, Espagne
Projet réalisé en autonomie

- Missions**
- S'enrichir personnellement et culturellement
 - Mieux appréhender les voyages à l'étranger
 - Développer/apprendre la langue espagnole
 - Découvrir une petite partie des pays de l'est
 - Découvrir les cultures étrangères
 - Vivre une expérience du voyage seul

Environnement ● ORGANISATION/RÉORGANISATION

DEVELOPPEUR JAVA

Descriptif du projet Assurer la maintenance et participer au développement des évolutions sur une application (ERP) de gestion de contrats commerciaux permettant de gérer le cycle de vie des espaces d'affichage publicitaire (METROBUS, MEDIAGARE)
Projet réalisé dans une équipe de 10 personnes

- Missions**
- Stabilisation + refonte du système de remises appliquées aux contrats commerciaux
 - Identification du processus algorithmique du calcul des remises existant
 - Formalisation des besoins d'évolutions pour mise en place d'un indice « Yield »
 - Nettoyage et refactoring du code existant
 - Implémentation des nouvelles règles de calcul de remises considérant les seuils « Yield »
 - Mise en place de tests unitaires (JUnit)
 - Mise en place de tests fonctionnels d'intégration (Mockito + Concordion)
 - Résolution des problèmes techniques liés aux évolutions apportées dans le calcul de remises
 - Modification de la vue d'édition des contrats pour afficher les niveaux de Yield par contrat
 - Modification de la vue d'édition des remises
 - Evolutions Graphiques
 - SmartPlanner : Modification de la vue pour filtrer les seuils yield lors d'une simulation (GWT)
 - Modification de l'édition de contrat : diverses modifications graphiques
 - Nettoyage et refactoring du code existant
 - Maintenance
 - Modification et création de requêtes (JPA/Criteria)
 - Nettoyage et refactoring du code existant
 - Résolution d'anomalies en production
 - Revues de code régulières (intégré dans le workflow Gitlab au sein de l'équipe)

Environnement ● JAVA 8, SPRING BOOT, SPRING MVC, SPRING BATCH, SPRING-DATA, SPRING-BATCH, JPA, HIBERNATE, JUNIT, CONCORDION, MOCKITO, POSTGRESQL, MAVEN, POSTMAN, JASPER, TOMCAT, JOOQ, JACKSON ● GIT, JENKINS, SONAR

RESPONSABLE TECHNIQUE

Descriptif du projet Leo / 2020FPF est un package d'applications web (EAR) permettant de gérer le cycle de vie des véhicules sur les lieux de traitement d'un point à un autre, c'est à dire du fournisseur, constructeur, centre ou autre) au client (centre, garagiste ou simple particulier). Elle représente le référentiel et le socle applicatif. Leo est une application permettant de gérer tout ce qui concerne le transport de véhicule/s d'un point à un autre. Projet réalisé dans une équipe de 10 personnes

- Missions**
- Rédaction de documentations techniques :
 - Revues de code et mise en place de bonnes pratiques
 - Optimisation / Refactoring
 - Mise en place de tests unitaires et recette
 - Migration du système applicatif (E A R) de ant vers maven pour 7 applications (Administration, Referentiel, Commande, Exploitation, Rmes, Supervision, Facturation) – 2 mois au total
 - Reprise d'une application tablette permettant de saisir les déchargements des véhicules dans les centres
 - Identification et résolution des anomalies existantes sur l'application
 - Formalisation de usecases utilisateur (parcours utilisateur et scénarios)

Environnement

- JAVA 6, JAVA 8, SPRING, SPRING BOOT, SPRING-BATCH, JPA, HIBERNATE, JSF, RICHFACES, JENKINS, MAVEN, JUNIT, SONAR, ORACLE, TOMCAT, JBOSS WILDFLY, EAR, JAX-RS, APACHE AXIS2, EJB 3.0, JMS, JACKSON, SOAP, PL/SQL, JIRA, CONFLUENCE



DEVELOPPEUR JAVA

Descriptif du projet Medicis est une application de type ERP qui répond aux besoins fonctionnels de l'entreprise côté utilisateur en interne (Administration de la gestion de contrats régie/commerciaux, Agencement des bandes de diffusion publicitaire avant livraison aux complexes/cinema)
Projet réalisé dans une équipe de 5 personnes

- Missions**
- Rédaction des documentations de références.
 - Rédaction de documentations techniques :
 - Création d'un système de gestion des Contributions Numériques :
 - Refonte de la propagation des prestations de contrat de régie dans les salles cinéma :
 - Création d'une historisation des opérations SQL sur les salles et complexes
 - Implémentation du produit «Suivi de Film » dans les calendriers de diffusion :
 - Refonte de l'interface comptable (Migration de SAP vers Y2 - CEGID) :
 - Amélioration de la DAO :
 - Amélioration du socle graphique :
 - Création des tests unitaires et bouchons (JUnit / Mockito)
 - Implémentation des scripts de livraisons dans les différents environnements
 - Correction d'anomalies fonctionnelles/techniques diverses et variées (Redmine)

Environnement ● JAVA 8, SPRING, JPA, HIBERNATE, JUNIT, EXT JS, GWT, JENKINS, SONAR, DOCKER, GIT, GITLAB, MOCKITO, MYSQL, MAVEN, LIQUIBASE, JOOQ, TOMCAT, REDMINE

DEVELOPPEUR JAVA

Descriptif du projet ExpressPad est une application de gestion des plans d'entretien de voitures sur tablette tactile

Projet réalisé dans une équipe de 7 personnes

- Missions**
- Rédaction de spécifications techniques (CLA/PCR)
 - Refonte totale de la génération de devis (ITEXT)
 - :
 - Amélioration de la configuration de l'administration (CXF/Struts) :
 - Conception et implémentation d'un outil de statistiques dans l'application :
 - Implémentation d'une vue dynamique permettant de positionner des dégâts/contrôles par géolocalisation sur un véhicule à l'aide de pictogrammes :
 - Conception et implémentation du service associé à la vue
 - Création des tests unitaires et bouchons (JUnit / Mockito / SoapUI)
 - Implémentation des scripts de livraisons dans les différents environnements
 - Participation à la recette : Suivi et correction des anomalies (JIRA)
 - Rédaction des documentations de références

Environnement ● JAVA 6, SPRING, JPA, HIBERNATE, BACKBONE.JS, STRUTS, TOMCAT, ORACLE, MAVEN, ECLIPSE, JERSEY (SERVICE REST), SOAPUI, ALFRESCO, JUNIT, JENKINS