isika

Formation aux métiers du numérique

CONCEPTEUR DEVELOPPEUR D'APPLICATION IA NODE JS

TITRE RNCP BAC+4

100% À DISTANCE

NODE JS

CYBERSÉCURITÉ

INTELLIGENCE ARTIFICIELLE

OBJECTIFS DE LA FORMATION

TITRE PROFESSIONNEL BAC+4 (NIVEAU : 6) CODE RNCP : 37873

A la fin de la formation les stagiaires sont capables de concevoir et de réaliser des applications sécurisées, tels que des logiciels d'entreprise, des applications pour mobiles et tablettes, ainsi que des sites Web tout en intégrant des agents IA. Ils maîtrisent toutes les étapes du cycle de vie d'un projet : de l'analyse des besoins à la rédaction du cahier des charges, en passant par le développement, les tests et le déploiement.

OBJECTIFS :

- Analyser et formaliser les besoins utilisateurs.
- Rédiger le dossier de conception.
- Savoir travailler en équipe dans un contexte de méthode agile comme Scrum.
- Concevoir, développer et maintenir une application en multi-couches.
- Mettre en œuvre de manière autonome NodeJS, et les technologies liées aux environnements Web Responsive (HTML, CSS, Bootstrap, Javascript).
- Développer les interfaces utilisateur.
- Réaliser les traitements métier de l'application avec des composants sécurisés.
- Modéliser et concevoir des bases de données SQL et NoSQL et participer à leur implémentation.
- Identifier les exigences, concevoir les cas de tests, préparer les jeux de données, exécuter une campagne de tests
- Utiliser le Vibe Coding
- Automatiser des processus sans code
- Construire des applications visuellement
- Utiliser des plateformes de NoCode
- Mettre en oeuvre des agents d’IA
- Documenter le déploiement de l'application et contribuer à la mise en production dans une démarche DevSecOps en collaboration
- Rédiger les documentations nécessaires à l’exploitation.

ADMISSION

Concepteur Développeur IA

PUBLIC

Tout public : demandeur d'emploi ou salarié, peu importe le domaine de formation initial.

Public avec une reconnaissance RQTH, accompagnement possible par notre référent handicap : Patrice Gaudin / patrice.gaudin@projet-isika.com

PRÉREQUIS

Dela motivation et de la disponibilité pour suivre un parcours avec une grande autonomie et un accompagnement par des experts.

Aucun prérequis en informatique.

MODALITÉS D'ACCÉS

1. **Candidature en ligne** sur notre site : www.projet-isika.com (dépôt de CV + questionnaire de 5min)
2. **Test de logique** en ligne (20min)
3. **Entretien individuel** de motivation en visioconférence pour évaluer votre projet professionnel et de formation.

EQUIPEMENT

Pour suivre la formation les éléments suivants seront nécessaires :

- Connexion internet haut débit
- Équipements visioconférence : webcam, microcasque
- Ordinateur, 64bits, 16Go, icore5 ou 7.

LA FORMATION

Concepteur Développeur IA

DURÉE

259 heures dont 75% asynchrone et 25% accompagné par des experts en téléprésentiel

FORMAT

La formation se déroule **entièrement à distance**. Les stagiaires sont **encadrés et accompagnés par nos formateurs experts.** Les cours sont dispensés en face à face pédagogique et en direct à travers **nos classes virtuelles,** en alternance avec des sessions de travaux pratiques en autonomie, corrigés ensuite en direct avec le formateur et un accès à des contenus sur plateforme pédagogique.

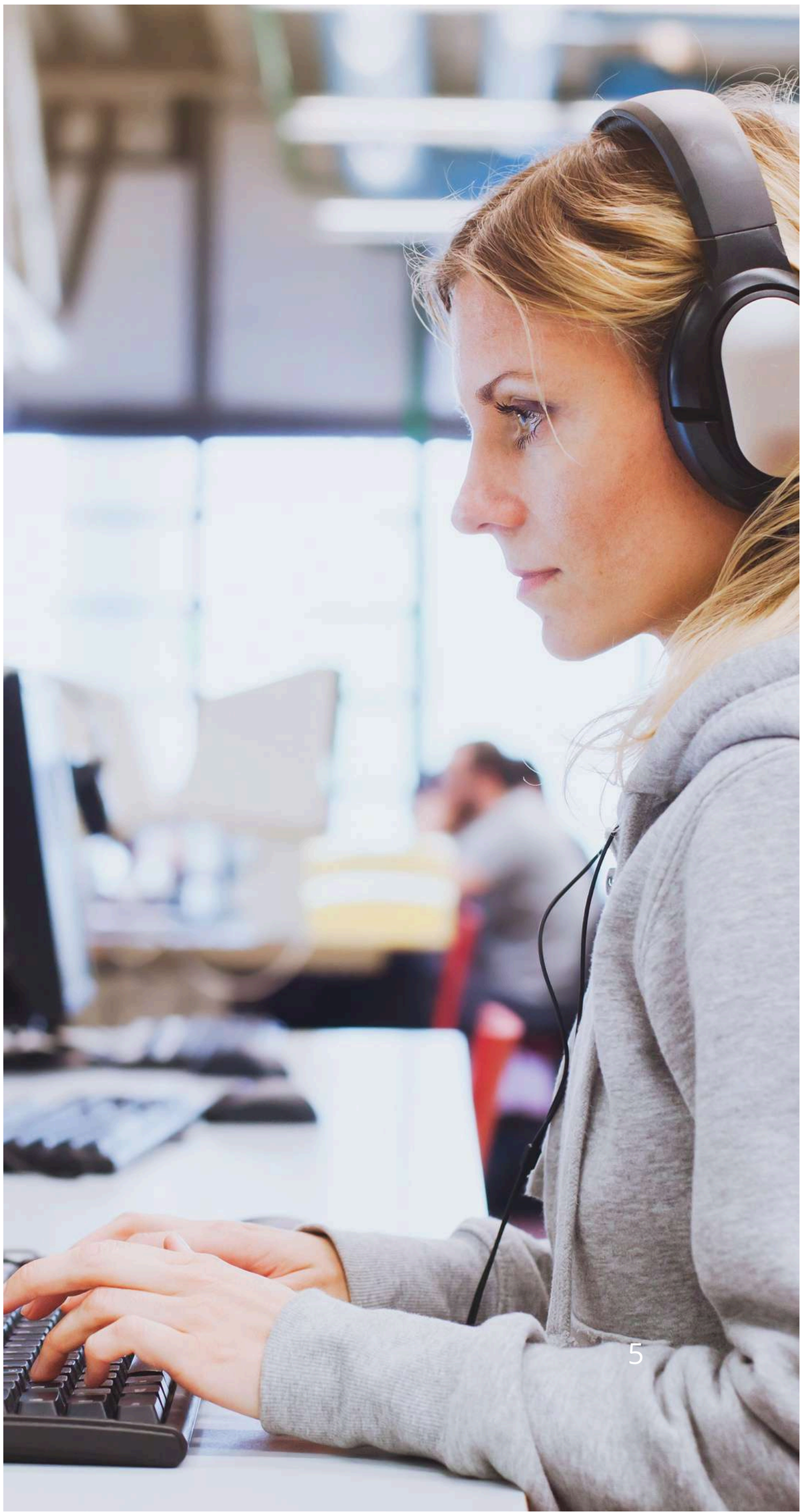
ACCOMPAGNEMENT VERS L'EMPLOI

Ce parcours forme des spécialistes immédiatement opérationnels, capables d'assumer dès la fin de la formation des fonctions de Concepteur Développeur ou d'Ingénieur de Développement dans le domaine des agents IA et des application web.

Les technologies mises en œuvre sont en adéquation avec le marché.

RYTHME

Grâce à l'autonomie et la liberté que propose le mode asynchrone, la formation peut être effectuée en 2 mois en mode intensif à temps plein ou s'étaler sur plusieurs mois pour ménager vie professionnelle et formation.



259 HEURES POUR DEVENIR CONCEPTEUR DÉVELOPPEUR IA

TITRE PROFESSIONNEL BAC+4 (NIVEAU : 6) CODE RNCP : 37873

UNE FORMATION TOURNÉE VERS L'EMPLOI

Ce parcours de 259 **heures flexibles** entièrement **à distance** forme des techniciens **immédiatement opérationnels,** capables d'assumer dès la fin de la formation des fonctions de **Concepteur Développeur Web et IA ou d'Ingénieur de Développement .**

Les apprenants peuvent aussi suite à ce parcours assurer des fonctions de Consultant Fonctionnel ou d'Assistant à Maîtrise d'Ouvrage (AMO).

UN ACCOMPAGNEMENT AU QUOTIDIEN DEPUIS CHEZ VOUS

L'enseignement est dispensé en direct par nos formateurs à travers nos classes virtuelles. **La pédagogie hybride** permet d'alterner les sessions de cours avec des travaux pratiques et des périodes d'autonomie et de projets en situation réelle comme en entreprise.
Une grande quantité de contenu est également disponible sur une plateforme dédiée.

UN PROGRAMME ACTUALISÉ

Ce programme de haut niveau, enrichi pour maîtriser les nouveaux outils de **Cybersécurité** et d'**Intelligence Artificielle** est constamment actualisé pour répondre aux besoins du marché et des entreprises de la tech.

LA PÉDAGOGIE EN MODE HYBRIDE

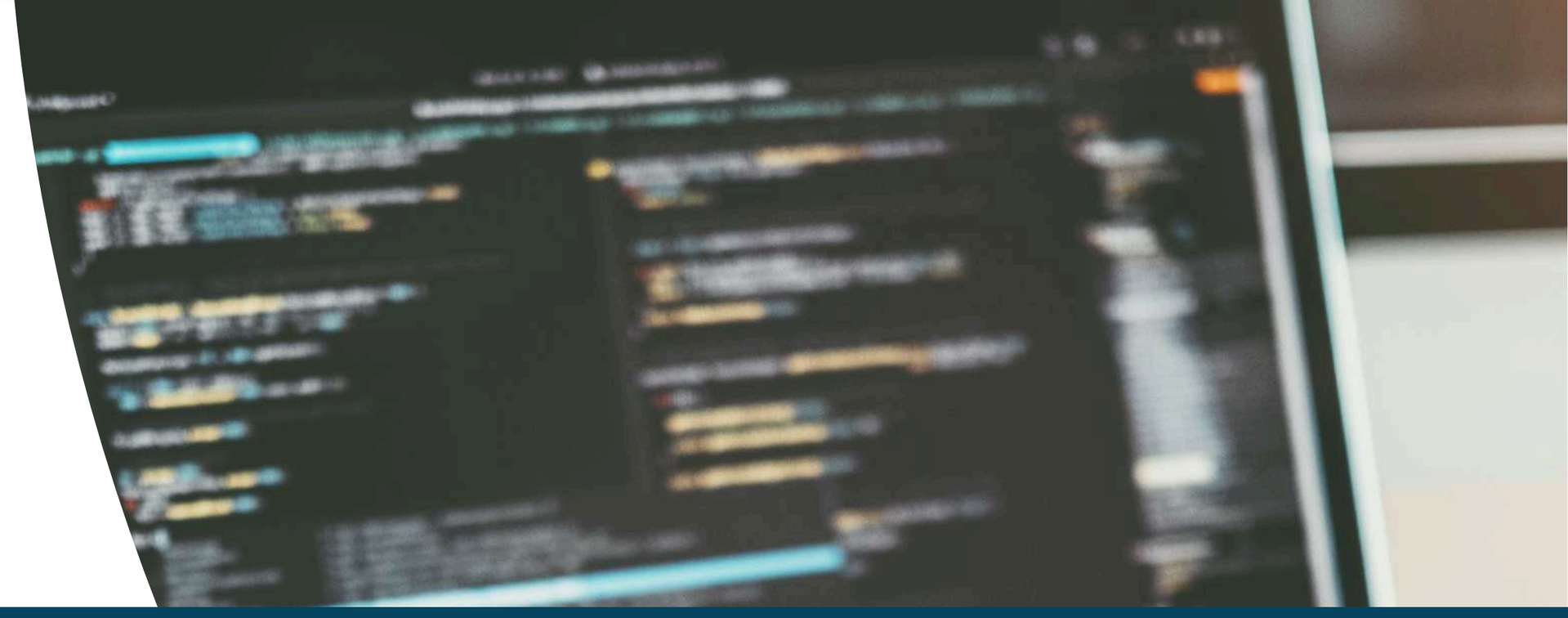
UNE PÉDAGOGIE AXÉE SUR LA PRATIQUE

Afin de former des professionnels prêts à intégrer le marché du travail dès la fin de leur formation. ISIKA a développé une **approche pédagogique axée sur l'apprentissage par la pratique**. Les stagiaires évoluent dans **un environnement qui reproduit les conditions du monde professionnel** avec des mises en situation concrètes, basées sur des problématiques actuelles rencontrées en entreprise. Grâce à nos **outils de travail**, ils réalisent au cours de leur formation un projet :

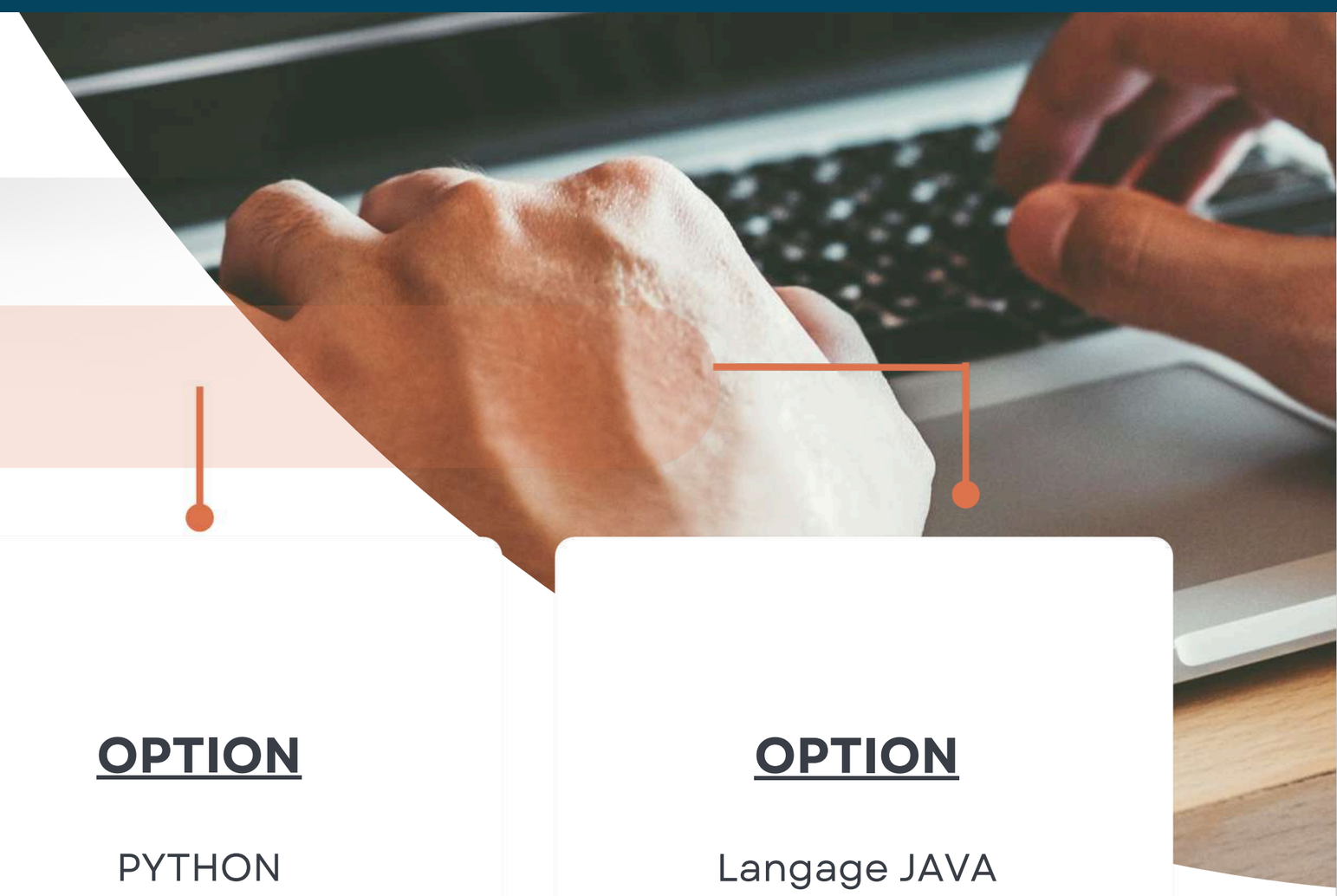
- **Un projet individuel**, permettant d'approfondir leurs compétences techniques et mettre en œuvre le fruit de leur apprentissage.

Ces projets encadrés par nos experts sont présentés par les stagiaires en soutenance. Ils visent l'acquisition d'une expérience concrète et valorisable auprès des recruteurs. La pédagogie par la pratique permet de développer des **compétences techniques solides**, mais aussi **une véritable capacité d'adaptation aux exigences des entreprises.**

isika
Formation aux métiers du numérique



LE PROGRAMME



LANGAGE

Langage NODE JS
et Typescript

Framework EXPRESS JS

OPTION

PYTHON
et Javascript

Framework FLASK
PYTHON

OPTION

Langage JAVA

LE PROGRAMME

DE L'EXPRESSION DES BESOINS À LA RÉALISATION D'UNE APPLICATION WEB RESPONSIVE ET D'AGENTS IA

L'algorithmique par la pratique

Notion d'algorithme - Pseudo code - Variables - Expressions - Structures de contrôle - Fonctions - Algorithmique avancée - Notion de complexité - Algorithmes de tri - Récursivité - Structures de données : Pile, File, Liste, Arbre.

Les Techniques de Prompting pour IA génératives

Rappels sur les LLMs et l'IA générative - Prompting : définition, enjeux, limites - Architecture d'un prompt efficace : rôle, objectif, contrainte - Prompt simple, Zero-shot, Few-shot avec exemples - Role prompting : adopter un point de vue expert - Structuration : Markdown prompting, Instructions + Contexte + Tâche - Prompt engineering VS prompt crafting - Chaînage de prompts - Récupération de données externes dans un prompt (API, base de connaissances, SQL...) - Éviter les hallucinations / biais / réponses vagues - Techniques d'itération : Prompt Tuning, Prompt Injection (cas pratique)

Les fondamentaux de JavaScript

Introduction au langage JavaScript - Historique et rôle du JS dans l'écosystème web - Environnement de développement (navigateur, console, VS Code) - Insertion dans une page HTML (inline, script, defer) - Premières instructions JS (console.log, alert, prompt) - Bases du langage - Variables (let, const, var) - Types primitifs (string, number, boolean, undefined, null) - Opérateurs, structures conditionnelles - Boucles : for, while, do...while - Fonctions (déclaration, expression, fléchée) - Tableaux et objets - Manipulations des tableaux (push, pop, splice, forEach, map, filter) - Création et itération d'objets - Fonctions d'ordre supérieur - Déstructuration, spread/rest operators - Programmation événementielle et manipulation DOM - Sélection d'éléments HTML (querySelector) - Événements : click, submit, change, keyup... - Gestion dynamique du DOM (création, suppression, classes, styles) - Fonctions avancées - Closures et portée des variables - Fonctions callback - Introduction aux Promesses et au async/await - Gestion des erreurs (try/catch) - Appels API et AJAX - Le modèle client-serveur - Requêtes HTTP avec fetch() - Traitement des réponses JSON - Organisation du code et modularité - Bonne pratiques de structuration d'un projet - Introduction à l'approche objet -Notions de JS côté serveur (Node.js Intro) - Présentation de Node.js et npm - Exécution d'un script JS côté serveur

JavaScripts avancé - Mise en place des Tests et Introduction à Python

Programmation orientée objet - Classes, constructeurs, méthodes - Attributs d'instance et de classe - Encapsulation, héritage simple - Utilisation de bibliothèques - Introduction à math, random, datetime, os - Installation de packages avec pip - Premiers pas avec pandas, matplotlib ou OpenAI - Différence entre tests manuels, unitaires, intégration, bout-en-bout - Test-Driven Development (TDD), Behavior-Driven Development (BDD) - Les tests unitaires avec unittest - Structure d'un test : assertions, fixtures, exécution - Setup/teardown, coverage, mock (avec unittest.mock) - Utilisation de pytest et comparatif avec unittest - Paramétrisation, gestion des erreurs attendues.
L'Environnement de Développement Intégré (Visual Studio Code) - Débogueur - Intégration d'IA génératives pour l'aide à la production de code (Copilot) - La syntaxe du langage - Nombres, chaînes, booléens - Conversion de types - Opérateurs logiques et arithmétiques - Conditions if, elif, else - Boucles for, while - Structures de contrôle imbriquées - Fonctions - Déclaration et appel - Paramètres et valeurs de retour - Notion de portée (scope) et variables locales/globales - Structures de données - Listes et opérations (tri, filtres, slices...) - Tuples et immutabilité - Dictionnaires, ensembles, compréhensions de liste - Itérations avancées avec enumerate, zip - Fichiers et exceptions - Lire/écrire dans des fichiers texte - Ouverture sécurisée avec with - Exceptions (try, except, finally).

LE PROGRAMME

DE L'EXPRESSION DES BESOINS À LA RÉALISATION D'UNE APPLICATION WEB RESPONSIVE ET D'AGENTS IA

Les Outils de développement - Système de contrôle de version décentralisé (Git - GitHub - GitLab)

Comprendre Git : fondations - Qu'est-ce qu'un système de contrôle de version ? - Historique et architecture de Git (décentralisé vs centralisé) - Référentiel local, distant, commit, branche, HEAD - Git en pratique - git status, add, commit, log, diff, reset, checkout - Créer et gérer des branches - Fusionner (merge, rebase) et résoudre des conflits - GitHub et GitLab : plateformes collaboratives - Cloner un dépôt distant, push, pull, fetch - Gestion de projets collaboratifs, révision de code.

Conduite de projet - Méthodes Agiles - Scrum - Jira

Introduction à la gestion de projet agile - Limitations des méthodes classiques (cycle en V, cascade) - Les valeurs et principes du Manifeste Agile - Les familles de méthodes : Scrum, Kanban, XP, SAFe - Scrum en détail - Les rôles : Product Owner, Scrum Master, Dev Team - Les rituels : sprint planning, daily scrum, review, rétrospective - Les artefacts : backlog produit, backlog sprint, burndown chart - Jeu de rôle : simuler un sprint avec l'outil Jira - Jira pour la gestion agile - Présentation de l'interface Jira - Création de projets, de user stories, de tâches - Boards Scrum et Kanban, gestion des sprints - Suivi de vélocité, gestion des priorités, étiquettes, filtres - Mise en place d'un backlog agile dans Jira + planification de sprint.

Rédiger un cahier des charges - Rédaction des spécifications fonctionnelles

Rédiger un cahier des charges - Utilisation du modèle Volere - Modélisation des processus métier avec BPMN - Rédaction des exigences - Modélisation des cas d'utilisation avec UML - Rédaction des spécifications fonctionnelles et non fonctionnelles - Modélisation du domaine et des entités métier avec le diagramme de classe UML - Modélisation des états métiers avec le diagramme d'état UML..

Analyse et conception - Modélisation avec UML

Introduction à l'analyse orientée objet - Historique de l'UML et normalisation OMG - Objectifs de l'analyse orientée objet - Différence entre analyse et conception - Identifier les besoins fonctionnels - Techniques de recueil des besoins - Introduction aux cas d'utilisation - Rédiger un diagramme de cas d'utilisation UML - Définir les acteurs, les scénarios, les exceptions - Structure du système - Diagramme de classes - Concepts : classe, attribut, méthode, association, agrégation, composition - Héritage, interfaces, relations statiques - Utilisation de l'outil PlantUML / StarUML pour créer des diagrammes - Comportements et interactions - Diagrammes de séquence : messages synchrones, asynchrones, activation, alt, loop - Diagrammes d'états-transitions : machine à états, transitions, événements - Diagrammes d'activités : logiques de flux et conditions - Architecture et composants - Diagrammes de composants : architecture logicielle modulaire - Diagrammes de déploiement : composants déployés sur l'infrastructure - UML et architecture logicielle (MVC, hexagonal, microservices) - Démarche projet orientée UML - Démarche de modélisation en contexte agile - Documenter les artefacts UML pour une équipe projet - Outils collaboratifs : Visual Paradigm, PlantUML.

LE PROGRAMME

DE L'EXPRESSION DES BESOINS À LA RÉALISATION D'UNE APPLICATION WEB RESPONSIVE ET D'AGENTS IA

Créer et exploiter des bases de données relationnelles SQL

Les concepts du relationnel - Définition d'une base de données - Définition d'un SGBD - Le modèle de données relationnel - L'algèbre relationnelle - Installation de XAMPP (MySQL/MariaDB) - Data Definition Language (DDL : create, alter, drop) - Data Manipulation Language (DML ; insert, select, update, delete) - Les opérateurs - Les jointures - Les sous-requêtes - Les fonctions d'agrégation et instructions de regroupement - Gestion des utilisateurs, rôles et mots de passe. Notion de MCD - MLD - MPD.

Les bases de données non relationnelles NoSQL

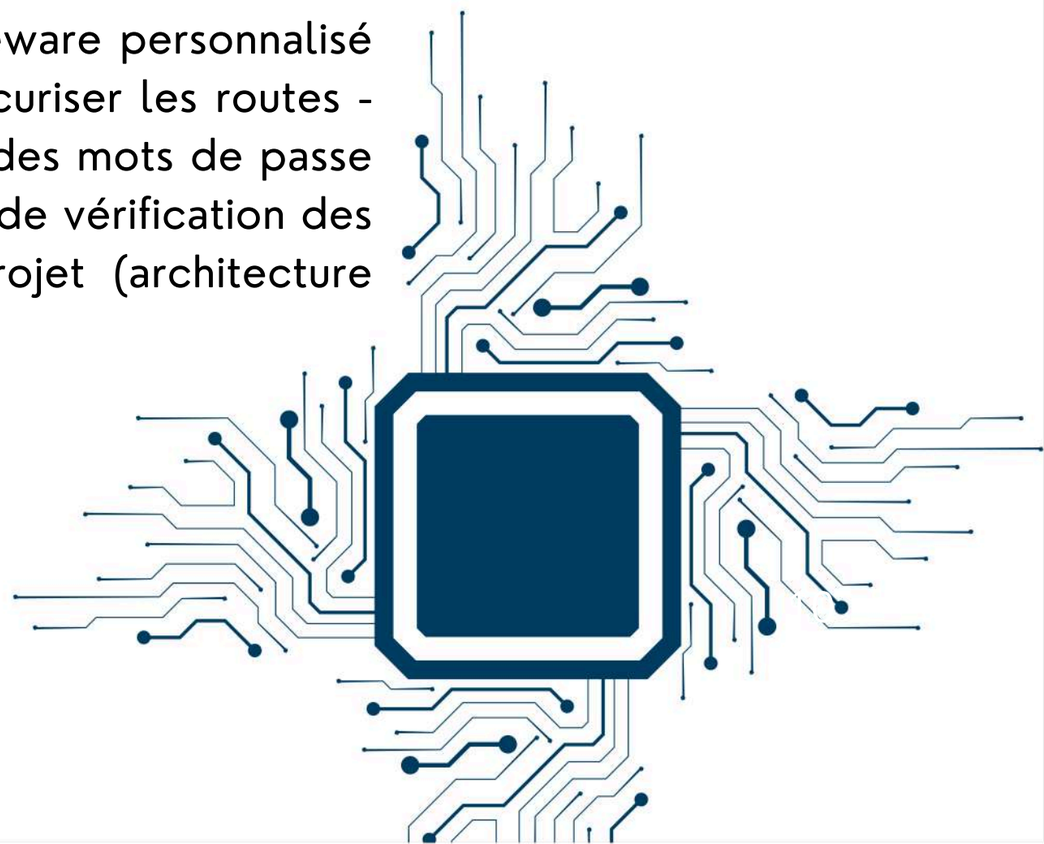
Mise en oeuvre de MongoDB, Serveur et Shell - Format de stockage - Database, Collection, Document - Interface graphique Compass - Gestion des bases de données - Gestion des collections - Insérer, rechercher, mettre à jour et supprimer des documents - Les jointures - Gestion des utilisateurs, rôles et mots de passe.

Développer des pages Web / Responsive Design avec HTML, CSS et Bootstrap

Structure de base d'une page HTML, balises HTML essentielles, liens, images - CSS de Base: Sélecteurs CSS, propriétés et valeurs, mise en forme de base (couleurs, polices, marges, etc.) - Bootstrap de Base: Installation de Bootstrap, structure de base, grille Bootstrap, composants de base - Le Responsive Design - Concepts de Base: Qu'est-ce que le Responsive Design ? Importance de l'adaptabilité, méthodologies (mobile-first, desktop-first) - Media Queries: Utilisation des media queries pour des styles adaptatifs, syntaxe et exemples - Grille Bootstrap Avancée: Grilles flexibles, systèmes de colonnes, alignement et disposition - Composants et UI Bootstrap - Composants de Base: Boutons, badges, barres de navigation, cartes, formulaires - Composants Avancés: Modals, carrousels, tooltips, accordéons - Personnalisation: Thématisation, variables Sass, extensions et plugins.

Node JS - Express JS

Introduction à Node.js - Qu'est-ce que Node.js ? Architecture orientée événement - Installation de Node.js, npm/yarn - Modules intégrés (fs, http, path) - Création d'un serveur HTTP minimal - Comprendre la boucle d'événements - Premiers pas avec Express.js - Introduction à Express.js - Gestion des routes (GET, POST, PUT, DELETE) - Utilisation des middlewares - Analyse de requêtes / corps / paramètres - Architecture MVC : introduction - Architecture RESTful et persistance - Architecture REST : bonnes pratiques - Intégration d'une base de données - CRUD complet sur une ressource - Séparation des responsabilités (routes, services, modèles) - Middleware personnalisé + gestion centralisée des erreurs - Sécurité et authentification - Sécuriser les routes - Authentification avec JWT (JSON Web Token) - Stockage sécurisé des mots de passe (bcrypt) - Création d'un système d'inscription / login - Middleware de vérification des tokens - Bonnes pratiques et déploiement - Structuration du projet (architecture modulaire).



LE PROGRAMME

DE L'EXPRESSION DES BESOINS À LA RÉALISATION D'UNE APPLICATION WEB RESPONSIVE ET D'AGENTS IA

TypeScript

Introduction à TypeScript et typage de base - Pourquoi TypeScript ? Avantages dans les projets modernes - Installation, configuration (tsc, tsconfig.json) - Premiers pas : typage statique, annotations simples - Types de base : string, number, boolean, any, unknown, null, undefined - Types personnalisés : type, interface, enum - Fonctions, objets, classes et typage avancé - Fonctions typées, paramètres optionnels, valeurs par défaut - Typage des objets, tableaux et tuples - Classes, constructeurs, héritage, interfaces implémentées - Accès public/private/protected - Génériques (<T>) et contraintes de type - Unions (|), intersections (&) et discriminated unions - Types avancés et outils de développement - Types utilitaires (Partial, Pick, Omit, Record, Readonly) - Les assertions de type (as, unknown, never, asserts) - Modules, namespaces et organisation du code - Déclarations de types (.d.ts), Typings pour bibliothèques JS - Utilisation de bibliothèques avec DefinitelyTyped (@types/...) - Linting, compilation, intégration dans un projet Node.js ou frontend - Utilisation dans Node.js avec Express.

Développement d'application Front

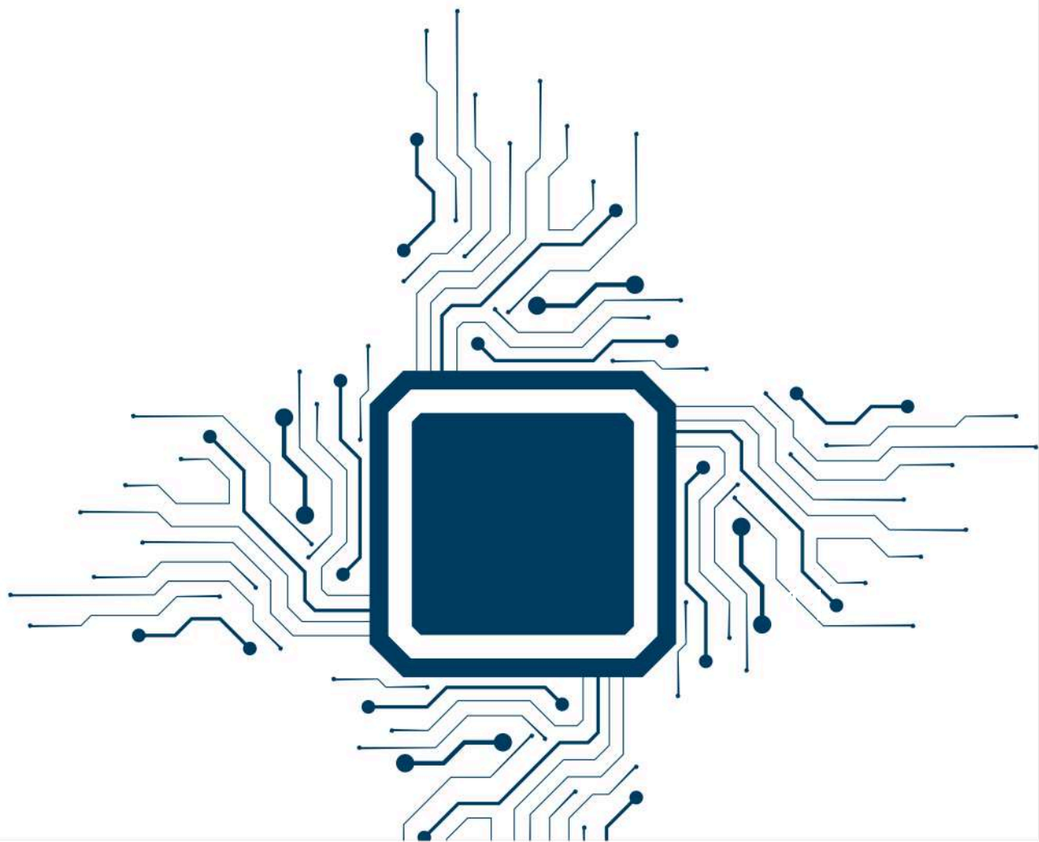
Concevoir et développer une interface utilisateur moderne, ergonomique et responsive - Utiliser efficacement HTML5, CSS3 et JavaScript pour construire des interfaces interactives - Manipuler le DOM et gérer les événements côté client - Maîtriser les bases des frameworks front modernes comme React et autres - Structurer un projet Front-End en composants réutilisables - Interagir avec une API REST et afficher dynamiquement les données - Mettre en place des bonnes pratiques de développement front : accessibilité, performance, versioning, responsive design.

Vibe Coding

Présentation d'outils IA - Notions de "squelette intelligent" : demander à l'IA de générer les structures de base - Lecture de code par IA pour auto-apprentissage - Utiliser GPT pour générer, expliquer, documenter du code - Créer un assistant de développement personnalisé (via API OpenAI, ou LangChain) - Prototypage d'interfaces avec IA.

Automatisation sans code accessible pour tous avec Zapier

Introduction à l'automatisation sans code - Qu'est-ce qu'un Zap ? (Trigger, Action, Filter, Delay, Formatter...) - Présentation de Zapier : interface, applications, plans - Enrichissement des Zaps avec filtres, chemins conditionnels et variables - Notion de multi-étapes, de formatage de données et de champs personnalisés - Bonnes pratiques d'automatisation : nommage, structure, sécurité, limites - Présentation de cas métiers réels (RH, marketing, gestion documentaire, CRM) - Intégrer Zapier avec des IA génératives (OpenAI ChatGPT).



LE PROGRAMME

DE L'EXPRESSION DES BESOINS À LA RÉALISATION D'UNE APPLICATION WEB RESPONSIVE ET D'AGENTS IA

Construction visuelle d'application avec Make

Introduction à Make : philosophie, positionnement, cas d'usage - Interface et structure d'un scénario (scénarios, modules, connexions, variables) - Types de modules : actions, déclencheurs, filtres, routes, cycles, erreurs - Structures conditionnelles, gestion d'erreurs, modules répétés, agrégation - JSON, mapping de données, transformation de chaînes, dates, formats - Utilisation de Webhooks (réception et envoi) pour créer ses propres déclencheurs - Intégration avec OpenAI (prompt dynamique) ou services IA - Structure modulaire d'une application : scénarios principaux, assistants, logs - Gestion des données persistantes (Airtable, Google Sheets, Bases internes Make) - Tableaux de bord, monitoring, documentation.

No Code avec la plateforme open source n8n

Qu'est-ce que n8n ? Philosophie open source, comparaison avec Make et Zapier - Présentation de l'interface utilisateur, des nœuds, workflows, exécutions - Création de premiers scénarios -Déclencheur manuel et Webhook - Automatisation de formulaire - Structures de base : input/output, configuration des champs, mapping JSON - Utilisation de boucles, conditions, expressions dynamiques - Appels API - Intégration de services IA - Déploiement - Hébergement local - n8n Cloud et alternatives - Sécurité, sauvegardes, versioning, journaux d'exécution.

IA agentique LangChain

Qu'est-ce qu'un agent IA ? (vs chatbot, RAG, API orchestration) - Présentation des grandes familles d'agents (reactive, réflexive, planning) - Concepts clés de LangChain - LLM wrappers, PromptTemplates, Chains - Tools, Agents, Memory, Callbacks - Mise en place d'un environnement local Python + LangChain - Création d'outils personnalisés - Agents de type ReAct et Zero-Shot - LangChain Expression Language (LCEL) - Gestion de la mémoire conversationnelle - Agents planificateurs - Intégration avec vecteurs sémantiques - Indexation de documents (PDF, sites web, emails) - Création de Retrieval-Augmented Generation agents (RAG) - Combiner un RAG avec un agent outillé - Agents multi-actions, multi-étapes et planificateurs - Appels API dynamiques - Réflexion et planification - Intégration front-end - Introduction aux architectures orientées événements pour agents distribués.

Sécurité des applications

Introduction à la sécurité des applications - Pourquoi la sécurité est-elle cruciale ? (Statistiques, retours d'expérience) - Menaces, vulnérabilités, attaques : vocabulaire essentiel - OWASP Top 10 (2021) : panorama des 10 principales failles - Notions de CIA (Confidentialité, Intégrité, Disponibilité) - Authentification et gestion des sessions - Mécanismes d'authentification (basic, token, OAuth2, JWT) - Protection des sessions (cookie, SameSite, timeout, HttpOnly) - Attaques courantes : bruteforce, credential stuffing, session fixation - Sécuriser le code - Revue de code sécurisé : injections SQL/NoSQL, XSS, CSRF - Bonnes pratiques de développement (input validation, output encoding) - Gestion des erreurs et des logs - API, Front et sécurisation des communications - Sécuriser les APIs REST et GraphQL - Chiffrement HTTPS / TLS / headers de sécurité (CSP, X-Frame-Options) - CORS, rate limiting, authentification côté client - Sécurité dans le navigateur : CSP, sandbox, intégrité - Outils de test et sécurisation avancée - Automatisation des tests sécurité dans une CI/CD.

LE PROGRAMME

DE L'EXPRESSION DES BESOINS À LA RÉALISATION D'UNE APPLICATION WEB RESPONSIVE ET D'AGENTS IA

DevSecOps

Conteneurisation avec Docker : Mise en place d'une chaîne CI/CD (Continuous Integration/Continuous Delivery) avec Jenkins - Méthodologie de Tests - Définition et principes de DevSecOps - Avantages de DevSecOps pour les entreprises - Création de Pipelines CI/CD avec Jenkins - Automatisation des tests - Sécurité et Maintenance de Jenkins - Mise en place d'une chaîne CI/CD pour une application web.

OPTION PYTHON

- Les fondamentaux de Python
- Python avancé - Mise en place des Tests
- Framework Flask Python

OPTION LANGAGE JAVA

Les fondamentaux de Java

Introduction à Java et premiers programmes - Installation (JDK, IDE : IntelliJ / Eclipse / VS Code) - Premier programme : Hello World - Compilation et exécution - Structure d'un programme Java - Notions de variables, types primitifs, opérateurs - Contrôle de flux et méthodes - Structures conditionnelles : if, else, switch - Boucles : for, while, do while - Méthodes : déclaration, appel, paramètres, retour - Portée des variables - Bonnes pratiques de nommage - Programmation orientée objet (POO) - Classes et objets : définition, instanciation - Attributs et méthodes d'instance - Constructeurs - Encapsulation, getters / setters - Visibilité : public, private, protected - Introduction à l'héritage (extends), polymorphisme, surcharge - Collections, tableaux, chaînes de caractères - Tableaux à une ou plusieurs dimensions - Manipulation des chaînes (String, StringBuilder) - Listes dynamiques : ArrayList, LinkedList - Maps : HashMap - Boucles for-each et Iterator - Introduction aux types génériques - Fichiers, exceptions - Lecture/écriture de fichiers texte (FileReader, BufferedReader, FileWriter) - Gestion des exceptions (try, catch, finally, throw) - Structuration en packages.

Java Avancé + Spring Boot

Java Avancé - SOLID Bonnes pratiques de conception OO - Packages, modules, visibilité - Classes abstraites, interfaces, héritage multiple avec interfaces - Énumérations, annotations - API Fonctionnelle (Lambda, Streams) - Lambdas en Java - Interface fonctionnelle - Programmation déclarative et immutabilité - Gestion des erreurs et API avancées - Exceptions personnalisées - Développement Spring Boot - Découverte de Spring Boot - Inversion de contrôle (IoC), injection de dépendance - Structure d'une app Spring Boot - Configuration, properties, auto-configuration - Architecture MVC (Controllers, Services, Repositories) - API RESTful - Création d'API REST avec Spring Web - Annotations REST : @RestController, @RequestMapping, @PathVariable, etc. - Manipulation de requêtes et réponses (GET, POST, PUT, DELETE) - Gestion des statuts HTTP et des exceptions - Persistance avec Spring Data JPA / Hibernate - Connexion à une base de données SQL (MySQL/H2) et NoSQL (MongoDB) - Création d'entités JPA, mapping relationnel - Repositories Spring Data - Requêtes JPQL.

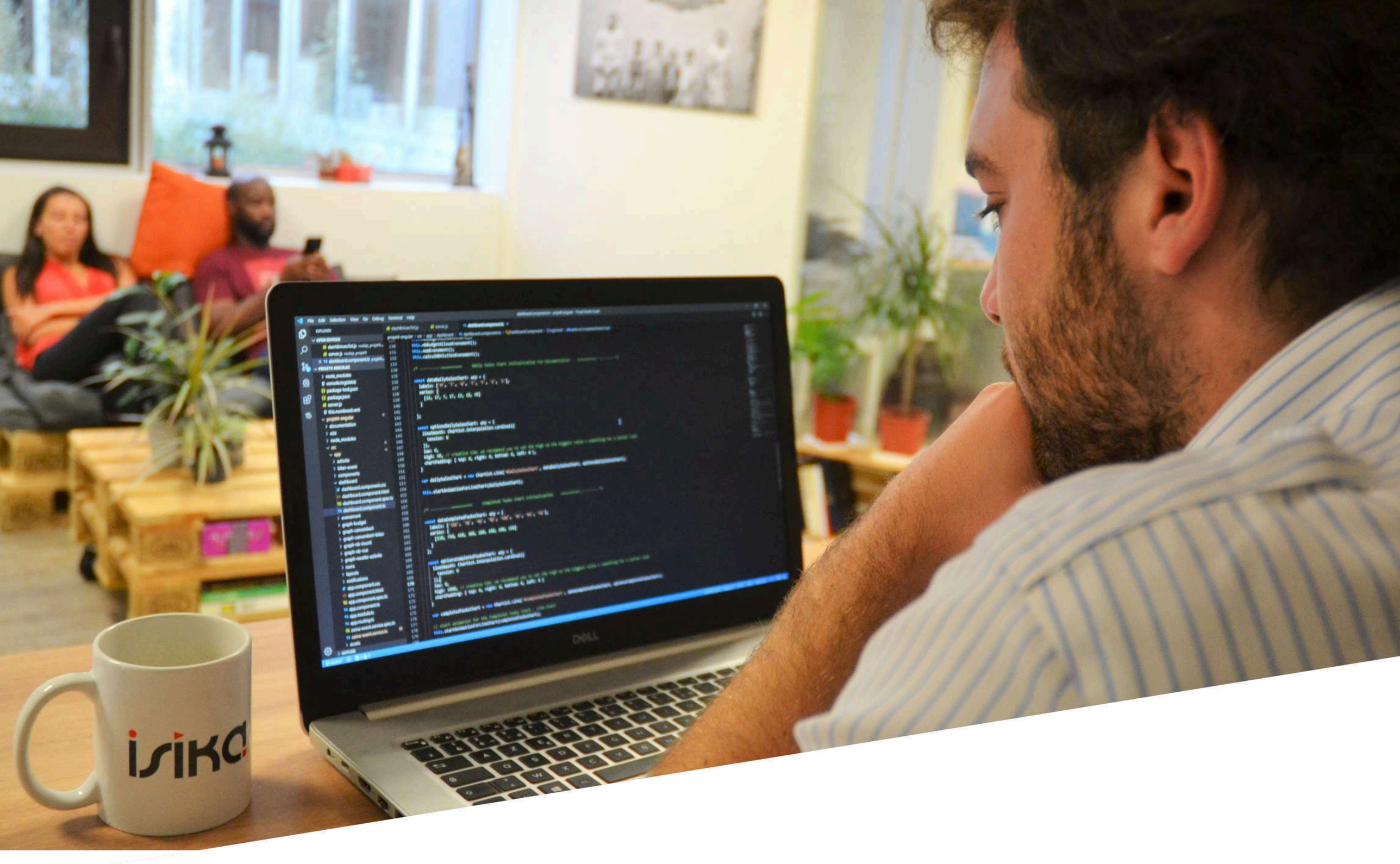
EVALUATION

Lesstagiaires sontévalués tout au long de la formation sur la base d’un contrôle continu, de la réalisation des projets 1 et 2 lors des soutenances et de la rédaction d’un mémoire.

SANCTION DE FORMATION

La formation donne accès au Titre professionnel “Concepteur Développeur d’Applications”, diplôme de niveau 6 (BAC+4), inscrit au Répertoire National de la Certification Professionnelle (RNCP).

Un examen blanc est organisé à la fin de la formation. La présentation du candidat à l’examen devant un jury du Ministère du Travail pour la certification, est effective si les évaluations de l’équipe pédagogique sont positives. Le tarif du passage du titre RNCP sur l’une de nos sessions d’examen est de 180 euros.



MODALITÉS DE FINANCEMENT

- Autofinancement : 5400 euros Financement sous
- POEI (France Travail et OPCA) CPF avec ou sans
- abondement Transition Professionnelle (PTP)
- Contrat de Sécurisation Professionnel (CSP) La
- Région AIF
-
-

Notre équipe vous accompagne pour établir le meilleur dispositif de financement.

Déposez votre candidature sur www.projet-isika.com pour en savoir plus.

Projet ISIKA - Siège Social : 1 Pl. Paul Verlaine, 92100 Boulogne-Billancourt

http://www.projet-isika.com – info.apprenants@projet-isika.com - 01 59 08 02 42

SIRET : 832 085 385 00017 - APE : 8559A - Numéro de déclaration activité : 11922203492

Version - 2025