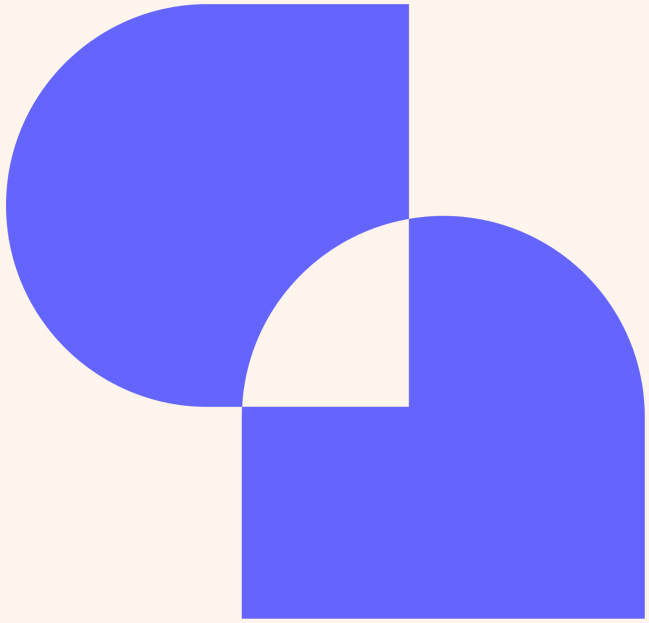
GoMind

“Agir avec l’esprit éveillé”

DOSSIER DE COMPÉTENCES

Développeur confirmé ReactJS

Spécialisé : NextJS



PRINCIPALES EXPÉRIENCES

DEESTREE : Développeur ReactJS/NextJS (2024-2025)
PINGS : Développeur ReactJS/NextJS (2023)
SMARTCH : Développeur ReactJS/NextJS/NestJS (2022)
FOND D'INVESTISSEMENT PRIVE : Développeur ReactJS/NextJS (2021-2022)
CONSERTO : Développeur ReactJS/NextJS/Express (2020-2021)

CONNAISSANCES TECHNIQUES

Langages : HTML, CSS, Sass, JavaScript ES5/ES6, TypeScript, C, C++, Java

Librairies/framework :
React.JS, NextJS, Hooks, Context API, HOC, Custom Hooks, Pure Component/Function, Styled-components, React Form , Redux, Redux Toolkit, Redux Off-line, React Query, Lazy ReactJS, Material-UI, Tailwind CSS, ChakraUI, Lodash, React Hook Form, ShadCN, Recharts, BabylonJS, ThreeJS, React Three Fiber, Jquery, Websocket, Transition API, Suspense, Error Boundaries, Gatsby, Framer Motion, GSAP, React Spring

Headless : EAM, RadixUI, Mantine,

Back : NestJS, Sequelize, ExpressJS, Prisma

Architecture/Design Pattern : Document Object- Model, UML, Atomic Design, Clean Archi, Microservices, SOLID, Kiss, Factory, Observer, Template, Singleton, Patterns Controlled, Uncontrolled

Base de données : MySQL, MongoDB, Firebase, Supabase,, PostgreSQL, DynamODB, GraphQL (Appolo)

Bonnes pratiques : Jest, RTL, Mock, Cypress, TDD, Storybook, Prettier, ESlint, Performance (Lighthouse)

Outils : Git, GitHub, Jenkins, GitlabCI, Bitbucket, Postman, Swagger, Docker, Vite, Sequelize, Vercel, Netlify, Heroku, Webpack, Husky, React dev tool,

Gestion de projet : Confluence, JIRA, Agile

EXPÉRIENCES TECHNIQUES

DEESTREE : Développeur ReactJS (2024-2025)

Développement d'une plate-forme de e-commerce dans une équipe internationale, en méthodes agiles.

- Analyse du besoin
- Suivi des spécifications techniques et fonctionnelles
    - Co écriture des US
    - Participation à la création du backlog delivery
    - Participation aux chiffrages et estimations
- Suivi du design system
- Équipe de 5 développeurs :
- Gestion de projet en SCRUM :
    - Suivi des cérémonies : daily, démo, rétro
    - Sprint 1 semaine
    - Atelier technique toute les semaines
- Utilisation de l'architecture Microservices :
    - Application des principes Clean Code et SOLID.
    - Mise en œuvre de patterns avancés React : Compound Components, Controlled/Uncontrolled Components
    - Création des HOC (High-Order Components).
    - Gestion du prop drilling via Context API
    - Implémentation de stratégies de virtualisation, suspense, et error boundaries pour optimiser la robustesse et la réactivité
- Développement d'application en ReactJS 18/NextJS :
    - Développement déclaratif,
    - Gestion du Server-Side Rendering (SSR) et utilisation du nouvel App Router.
    - Mise en place des Hooks & État : useState, useReducer, useCallback, useMemo (mémoïsation)
    - Création de custom hooks.
    - Optimisation des performances : lazy loading, transitions API, mémoïsation, découpage dynamique des composants.
    - UI & Accessibilité : intégration avec Tailwind CSS, ShadCN
    - Utilisation de i18next pour l'internationalisation.
- Développement TypeScript :
        - Utilisation des Generic, et Unknow
        - Utilisation de Zod pour la validation de schémas et gestion stricte de l'immuabilité
- Développement Python :
    - Utilisation de Flask
    - Utilisation de DynamoDB
- Versionning sur Git
    - SUivi du Gitflow
    - Création de normes de commit
- Mise en place des tests unitaires avec RTL
- Mise en place de ESlint et Prettier
- Utilisation de Storybook pour la documentation des composants
- Utilisation de Vite

- Utilisation de Bitbucket pour la CD
- Intégration et test des API via Postman & Swagger
- Utilisation de FastAPI

Environnement technique :
US, Microservices, Clean Code, SOLID, Compound Component, HOC, SSR, App Router, TypeScript, Generic, Unknown, Immutabilité, Zod, ReactJS 16, Redux, React Router, Lazy loading, NextJS, Tailwind, ShadCN, i18next, PDF Reader, Virtualisation, Suspense, Error Boundaries, FastAPI, Flask, DynamoDB, Gitflow, Bitbucket, Postman, Swagger, RTL, Storybook

---

PINGS : Développeur ReactJS/NextJS (2023)

Développement from scratch d'une plate-forme de jeux vidéo en NextJS

- Analyse du besoin
- Écriture des spécifications techniques et fonctionnelles
    - Création de ticket JIRA
    - Alimentation du backlog technique
    - Démo hebdomadaire devant les stakeholders
- Échange avec le game designer sur les wireframe via Figma
- Équipe de 3 personnes
- Mise en place d'une architecture monolithique
    - Application rigoureuse des principes Clean Code et SOLID.
    - Utilisation des patterns :
        - Factory et Singleton pour la gestion centralisée d'instances.
        - Compound Components pour une composition flexible des interfaces.
        - Controlled / Uncontrolled Components pour une gestion fine des états internes.
    - Réduction du prop drilling via Context API.
    - Optimisation de la performance via mémoïsation (useMemo).
- Développement d'application en ReactJS 16/NextJS :
    - Développement impérative
    - Gestion des life cycle (Use State, Use Réf, Use Reducer, Use callback
    - Création de Custom hooks
    - Gestion du SSR et Page Router pour optimiser le rendu et le référencement.
    - Gestion de l'UI / Design System : utilisation de Tailwind CSS et Mantine pour le style et la cohérence visuelle.
    - Internationalisation via i18next.
    - Mise en place de 3D : intégration de Three.js et React Three Fiber pour la génération et la manipulation d'éléments 3D dans le navigateur.
    - Backend & Authentification : utilisation de Supabase pour la gestion des utilisateurs et de la base de données.

- Utilisation de TypeScript : typage strict, utilisation de Generic et Unknown, respect des principes d'immutabilité.estion du SSR
    -
- Versionning sur Git
    - SUivi du Gitflow
    - Création de normes de commit
- Utilisation de Storybook
- Mise en place de ESlint et Prettier
- Utilisation de Github et Github Action pour la CI/CD

Environnement technique :
Monolithe, SOLID, Factory, Singleton, Compound Component, TypeScript, Generic, Unknown, Immutabilité, ReactJS, ReactJS 16, NextJS, useState, useRef, useReducer, useCallback, useEffect, Custom hooks, SSR, Page Router, Tailwind, Mantine, i18next, ThreeJS, React Three Fiber, Supabase, Gitflow, Storybook, GitHub Actions, Jira, Scrum, Figma

---

SMARTCH : Développeur ReactJS/NextJS/NestJS (2022)

Création from scratch d'une plateforme de E-learning

- Analyse du besoin
- Écriture des spécifications techniques et fonctionnelles
    - Création de ticket JIRA
    - Alimentation du backlog technique
    - Démo hebdomadaire devant les stakeholders
- Échange avec le designer autour du Design System
- Équipe de 5 dev
    - Gestion de projet en Scrum (daily, démo, rétro, poker planning)
    - Sprint de deux semaines
    - Code Review
- Utilisation de l'architecture Microservices :
    - Utilisation de la mémoization avec Use memo
    - Implémentation de Compound Component
    - Implémentation des patterns Controlled, Uncontrolled
    - Écriture de design de composant (gestion des Atoms, moléculaire, organismes...)
    - Gestion du prop drilling
    - Utilisation du Context API
- Développement d'application en ReactJS 16/NextJS :
    - Développement impérative
    - Gestion des life cycle (Use State, Use Réfn Use Reducer, Use callback
    - Création de Custom hooks
    - Gestion de l'incrémentale generation
    - Utilisation de App Router
    - Utilisation de ChakraUI
    - Utilisation de MUI
- Développement NestJS :
    - Utilisation de Prisma
    - Création de route API
    - Gestion relation one2many, many2many,
- Versionning sur Git

    - SUivi du Gitflow
    - Création de normes de commit
- Écriture de tests unitaires avec RTL et Jest
- Mise en place de ESlint
- Utilisation de Storybook
- Utilisation de Docker

Environnement technique :
Microservices, Jira, Compound Component, Prop drilling, ReactJS, ReactJS 16, NextJS, useState, useRef, useReducer, useCallback, Custom hooks, App Router, ChakraUI, MUI, RTL, Jest, ESLint, Storybook, Docker, Gitflow, Figma, Jira

---

FOND D'INVESTISSEMENT PRIVE : Développeur ReactJS/NextJS (2021-2022)

Création d'une plateforme d'investissement en web3 pour faire des investissements financiers (NFT)

- Analyse du besoin
- Écriture des spécifications techniques et fonctionnelles
- Échange avec le designer autour du Design System
    - Implémentation de maquette au pixel perfect
    - Échange fréquent avec l'UX & UI designer
- Équipe de 4 dev
    - Rôle lead tech :
        - Garant des codes Review
        - Formation et encadrement des développeurs
    - Gestion de projet en Scrum :
        - Sprint de 1 semaine
- Utilisation de l'architecture Microservices :
    - Design de l'architecture avec des schéma UML
    - Utilisation de la mémoization avec Use memo
    - Implémentation des patterns Controlled, Uncontrolled
    - Gestion du prop drilling
    - Utilisation du Context API
- Développement d'application en ReactJS 16/NextJS :
    - Développement impérative
    - Gestion des life cycle (Use State, Use Réfn Use Reducer, Use callback
    - Création de Custom hooks
    - Utilisation de App Router
    - Utilisation de Tailwind
    - Utilisation de MUI
    - Création de smart contrat avec Selenium
    - Connaissance Core banking :
        - Affichage de la monnaie via le Context
        - Interaction avec des API bancaires
- Versionning sur Git
    - SUivi du Gitflow
- Mise en place de ESlint
- Utilisation de Storybook
- Utilisation de Docker
- Utilisation de Github Action pour la CD

Environnement technique :
UML, Microservices, Design System, ReactJS, ReactJS 16, NextJS, App Router, useCallback, Custom hooks, Tailwind, MUI, Storybook, Selenium, ESLint, Docker, GitHub Actions, Gitflow, Jira, Figma, Scrum

---

CONSERTO : Développeur Angular (2020-2021)

Création d'une application immobilière pour aider les techniciens à remplir les diagnostics immobiliers.

- Analyse du besoin
- Écriture des spécifications techniques et fonctionnelles
- Suivi des wireframes avec Figma
- Équipe de 12 dév :
  - Pair programming
  - Code Review
- Gestion de projet en Scrum
  - Sprint de deux semaines
  - Cérémonies agile
- Utilisation de l'architecture Microservices :
- Développement d'application en Angular :
  - Migration vers AngularJS vers Angular
  - Gestion du binding bidirectionnel
  - Création de composants complexes
  - Mise en place des Class Decorator
  - Gestion des NG modules
  - Mise en place des Directives
  - Gestion des Lifecycle
  - Utilisation de RxJS
- Développement NodeJS/Express :
  - Mise en place de l'authentification
  - Gestion des mots de passe oublié
- Versionning sur Git
  - SUivi du Gitflow
- Écriture de tests unitaires avec Jest
- Implémentation de Storybook
- Utilisation de Docker
- Utilisation de Gitlab pour la CI

Environnement technique :
Microservices, Angular, Class Decorator, NG Modules, Directives, RxJS, Express, Git, Jest, Storybook, Docker, GitlabCI

FORMATION

Epitech : Architecte logiciel, Développeur d'application

LANGUES

Français

Anglais