CHATGPT - ATELIER - DÉVELOPPER AVEC L'INTELLIGENCE ARTIFICIELLE (IA)

m2iformation.fr

Sommaire

Sommaire

Utilisation de l'IA dans le domaine IT

1. L'aide au développement

Transformation du développement logiciel :

L'intelligence artificielle révolutionne la façon dont nous développons des logiciels, passant d'un processus entièrement manuel à un workflow assisté.

Utilisation de l'IA dans le domaine IT

1. L'aide au développement

- **Génération de code** : création automatique de fonctions, classes, modules
- **Complétion intelligente** : suggestions contextuelles avancées
- **Debugging assisté** : identification et résolution d'erreurs
- **Refactoring automatique** : amélioration de la qualité du code
- **Documentation automatique** : génération de commentaires et docs
- **Tests unitaires** : création automatique de tests complets

Utilisation de l'IA dans le domaine IT

1. L'aide au développement

Impact sur la productivité :

```
Développement traditionnel :
Conception → Codage → Tests → Debug → Documentation → Déploiement

Développement assisté par IA :
Conception ↔️ IA → Codage rapide ↔️ IA → Tests auto ↔️ IA → Debug assisté ↔️ IA
```

Utilisation de l'IA dans le domaine IT

2. L'injection d'IA dans un produit

Approches d'intégration :

1. IA comme fonctionnalité :

```
// Exemple : Système de recommandation
class RecommendationEngine {
    async getRecommendations(userId, context) {
        const aiResponse = await openai.completions.create({
            model: "gpt-3.5-turbo",
            messages: [{
                role: "system",
                content: "Tu es un expert en recommandations produit"
            }, {
                role: "user",
                content: `Recommande des produits pour l'utilisateur ${userId}
                         avec le contexte : ${JSON.stringify(context)}`
            }]
        });
        return this.parseRecommendations(aiResponse.choices[0].message.content);
    }
}
```

Utilisation de l'IA dans le domaine IT

2. IA comme service transverse :

- Analyse de logs intelligente
- Monitoring prédictif
- Optimisation automatique des performances
- Sécurité adaptive

3. IA embedded :

- Modèles embarqués dans l'application
- Inference locale pour la rapidité
- Protection des données sensibles

Utilisation de l'IA dans le domaine IT

3. La gestion du texte avec les LLM

Large Language Models - Capacités :

Traitement de texte avancé :

```
# Exemple d'utilisation pour l'analyse de sentiment
def analyze_customer_feedback(feedback_text):
    prompt = f"""
    Analyse le feedback client suivant et retourne un JSON avec :
    - sentiment (positif/neutre/négatif)
    - score de 1 à 10
    - mots-clés principaux
    - suggestions d'amélioration

    Feedback : {feedback_text}
    """

    response = llm.generate(prompt)
    return json.loads(response)
```

Utilisation de l'IA dans le domaine IT

3. La gestion du texte avec les LLM

Applications concrètes :

- **Génération de contenu** : articles, descriptions produits
- **Traduction automatique** : multilingue contextuelle
- **Résumé automatique** : documents longs, emails
- **Classification de texte** : catégorisation, tagging
- **Extraction d'entités** : noms, dates, concepts
- **Génération de requêtes** : SQL, API calls depuis langage naturel

Utilisation de l'IA dans le domaine IT

4. Le développement de chatbots IA

Architecture moderne d'un chatbot :

```
User Input → NLU (Understanding) → Dialog Manager → NLG (Generation) → Response
     ↑                                                    ↓
Knowledge Base ←→ Context Manager ←→ External APIs
```

Utilisation de l'IA dans le domaine IT

4. Le développement de chatbots IA

Composants essentiels :

1. Compréhension du langage naturel (NLU) :

```
class ChatbotNLU {
    async processInput(userMessage, context) {
        const analysis = await openai.completions.create({
            model: "gpt-4",
            messages: [{
                role: "system",
                content: `Tu es un assistant qui analyse les intentions utilisateur.
                          Retourne un JSON avec intent, entities, confidence.`
            }, {
                role: "user",
                content: userMessage
            }]
        });

        return JSON.parse(analysis.choices[0].message.content);
    }
}
```

Utilisation de l'IA dans le domaine IT

4. Le développement de chatbots IA

Composants essentiels :

2. Gestion du dialogue :

```
class DialogManager {
    constructor() {
        this.conversationHistory = [];
        this.userContext = {};
    }
    async generateResponse(intent, entities, userMessage) {
        const context = this.buildContext();
        const response = await openai.completions.create({
            model: "gpt-4",
            messages: [
                {
                    role: "system",
                    content: `Tu es un assistant commercial expert. Contexte de conversation : ${JSON.stringify(context)}Intent détecté : ${intent} Entités : ${JSON.stringify(entities)}`
                },
                ...this.conversationHistory,
                {
                    role: "user",
                    content: userMessage
                }
            ],
            temperature: 0.7,
            max_tokens: 500
        });
        this.updateConversationHistory(userMessage, response.choices[0].message.content);
        return response.choices[0].message.content;
    }
}
```

Panorama des outils IA

1. Comparatif des principales plateformes

Outil	Développeur	Points forts	Points faibles	Prix
ChatGPT	OpenAI	Interface intuitive, GPT-4 performant	Limite requêtes gratuit	20$/mois Pro
Gemini	Google	Intégration Google, multimodal	Moins de plugins	Gratuit/20$/mois
Claude	Anthropic	Sécurité, longs contextes	Moins répandu	20$/mois Pro
Amazon Q	AWS	Intégration AWS, sécurité entreprise	Ecosystème fermé	Sur devis
GitHub Copilot	GitHub/OpenAI	Intégration IDE, code-first	Spécialisé code	10$/mois

Panorama des outils IA

2. ChatGPT

Avantages :

- **Interface conversationnelle** naturelle et intuitive
- **GPT-4** : raisonnement avancé, créativité
- **Plugins** : extensions fonctionnelles riches
- **API robuste** : intégration facile dans applications
- **Communauté active** : ressources et partages

Panorama des outils IA

2. ChatGPT

Inconvénients :

- **Limitations de données** : cutoff de connaissance
- **Hallucinations** : génération d'informations incorrectes
- **Coût** : version payante pour fonctionnalités avancées
- **Latence** : délais de réponse variables
- **Politique d'usage** : restrictions sur certains contenus

Panorama des outils IA

3. Gemini (Google Bard)

Avantages :

- **Intégration Google** : Gmail, Drive, Calendar
- **Multimodal** : texte, images, code
- **Accès internet** : informations en temps réel
- **Gratuit** : version de base sans restriction majeure

Panorama des outils IA

3. Gemini (Google Bard)

Inconvénients :

- **Écosystème fermé** : dépendance Google
- **Moins de plugins** : extensibilité limitée
- **Performance variable** : moins constant que GPT-4

Panorama des outils IA

4. Claude (Anthropic)

Avantages :

- **Sécurité renforcée** : alignment research
- **Longs contextes** : jusqu'à 100k tokens
- **Éthique** : réponses plus prudentes
- **Code quality** : excellente génération de code

Panorama des outils IA

4. Claude (Anthropic)

Inconvénients :

- **Adoption limitée** : moins d'intégrations
- **Conservatisme** : parfois trop prudent
- **Prix** : modèle économique similaire à OpenAI

Panorama des outils IA

5. Amazon Q Developer

Avantages :

- **Sécurité entreprise** : compliance, audit
- **Intégration AWS** : services cloud natifs
- **Spécialisation** : optimisé pour développement
- **Support entreprise** : SLA, support technique

Panorama des outils IA

5. Amazon Q Developer

Inconvénients :

- **Écosystème fermé** : centré AWS
- **Coût** : pricing entreprise élevé
- **Courbe d'apprentissage** : complexité AWS

Panorama des outils IA

6. GitHub Copilot

Avantages :

- **Intégration IDE** : VS Code, JetBrains, Vim
- **Suggestions contextuelles** : code en temps réel
- **Multi-langages** : support large
- **Prix attractif** : 10$/mois seulement

Panorama des outils IA

6. GitHub Copilot

Inconvénients :

- **Spécialisé code** : pas de chat général
- **Dépendance GitHub** : écosystème Microsoft
- **Qualité variable** : suggestions parfois incorrectes

Focus sur ChatGPT

1. Versions et tarification

ChatGPT Free :

- Accès GPT-3.5 Turbo
- Limite : ~20 messages/3h en période de forte charge
- Pas d'accès aux plugins
- Pas de GPT-4

Focus sur ChatGPT

1. Versions et tarification

ChatGPT Plus (20$/mois) :

- Accès prioritaire GPT-4
- Limite : ~50 messages/3h GPT-4
- Accès aux plugins
- Fonctionnalités beta en avant-première
- DALL-E intégré

Focus sur ChatGPT

1. Versions et tarification

ChatGPT Team (25$/utilisateur/mois) :

- Tout ChatGPT Plus
- Espace de travail collaboratif
- Pas d'entraînement sur vos données
- Limite augmentée : 100 messages/3h

Focus sur ChatGPT

1. Versions et tarification

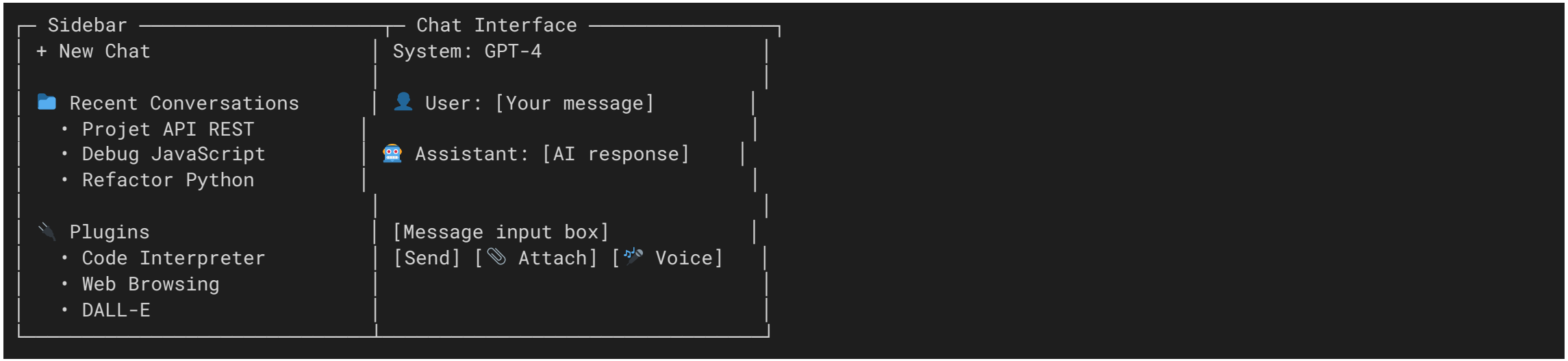
ChatGPT Enterprise (prix sur devis) :

- Sécurité entreprise
- Limites très élevées
- Analytics et administration
- SSO et contrôles d'accès

Focus sur ChatGPT

2. Interface et utilisation

Éléments de l'interface :



Gestion des conversations :

- **Conversations persistantes** : historique sauvegardé
- **Contexte maintenu** : mémoire durant la session
- **Partage** : liens publics vers conversations
- **Export** : téléchargement des données

Focus sur ChatGPT

3. Configuration et personnalisation

Paramètres système :

```
Instructions personnalisées :
"Tu es un assistant spécialisé en développement web moderne.
Réponds toujours avec :
- Code commenté et expliqué
- Bonnes pratiques incluses
- Tests unitaires si applicable
- Gestion d'erreurs intégrée
Utilise TypeScript par défaut pour JavaScript."
```

Focus sur ChatGPT

3. Configuration et personnalisation

Configuration du contexte :

```
Contexte de projet :
- Stack : React/TypeScript/Node.js/PostgreSQL
- Architecture : Clean Architecture
- Tests : Jest + Testing Library
- Déploiement : Docker + AWS
- Standards : ESLint + Prettier
```

Focus sur ChatGPT

3. Configuration et personnalisation

Types de réponse :

- **Concis** : réponses courtes et directes
- **Détaillé** : explications approfondies
- **Code-first** : priorité aux exemples pratiques
- **Pédagogique** : explications étape par étape

TP 1 : Configuration développeur

Objectif : Configurer ChatGPT pour un développement optimal

Étape 1 : Instructions personnalisées

```
Rôle : Expert développeur full-stack
Spécialités : JavaScript/TypeScript, Python, architectures modernes
Style de réponse :
- Code propre et commenté
- Explications concises mais complètes
- Inclusion des bonnes pratiques
- Tests et gestion d'erreurs systématiques
- Performance et sécurité prises en compte
```

TP 1 : Configuration développeur

Étape 2 : Test de configuration

```
Prompt de test :
"Crée une fonction API REST pour gérer l'authentification utilisateur"

Réponse attendue :
- Code TypeScript/Express
- Validation des données
- Hashage des mots de passe
- Tests unitaires
- Gestion d'erreurs
- Documentation
```

Bonnes pratiques et prompting

1. Importance du contexte

Éléments contextuels essentiels :

1. Contexte technique :

```
Stack technique : React 18 + TypeScript + Vite
Backend : Node.js + Express + PostgreSQL
Architecture : Clean Architecture + DDD
Tests : Jest + Supertest + Testing Library
Déploiement : Docker + AWS ECS
```

Bonnes pratiques et prompting

1. Importance du contexte

Éléments contextuels essentiels :

2. Contexte métier :

```
Application : E-commerce B2B
Utilisateurs : 10k+ entreprises clientes
Volumes : 1M+ transactions/jour
Contraintes : RGPD, PCI-DSS compliance
Performance : <200ms API response time
```

Bonnes pratiques et prompting

1. Importance du contexte

Éléments contextuels essentiels :

3. Contexte de développement :

```
Équipe : 5 développeurs seniors
Méthodologie : Scrum + TDD
Code review : systématique via GitLab
CI/CD : automatisé avec pipelines
Standards : ESLint + Prettier + Husky
```

Bonnes pratiques et prompting

2. Vérification de compréhension

Techniques de validation :

1. Reformulation :

```
Prompt : "Avant de coder, reformule ma demande pour vérifier ta compréhension"

Exemple :
User : "Crée un système d'auth avec JWT"
AI : "Tu souhaites que je crée un système d'authentification utilisant JWT avec :
- Endpoints login/logout/refresh
- Middleware de vérification
- Gestion des rôles utilisateur
- Sécurisation des routes protégées
Est-ce correct ?"
```

Bonnes pratiques et prompting

2. Vérification de compréhension

Techniques de validation :

2. Questions de clarification :

```
Prompt type : "Si des éléments ne sont pas clairs, pose-moi des questions précises avant de commencer"
```

Bonnes pratiques et prompting

3. Approche itérative

Stratégie fonction par fonction :

Étape 1 : Architecture globale

```
"Définis d'abord l'architecture générale du système d'authentification :
- Structure des dossiers
- Interfaces principales
- Flow de données
- Technologies utilisées"
```

Bonnes pratiques et prompting

3. Approche itérative

Étape 2 : Implémentation progressive

```
"Implémente maintenant uniquement la fonction de hashage des mots de passe"
"Maintenant la fonction de validation JWT"
"Ensuite le middleware d'authentification"
```

Étape 3 : Tests à chaque étape

```
"Pour chaque fonction, génère immédiatement les tests unitaires correspondants"
```

Bonnes pratiques et prompting

4. Types de prompts avancés

1. Prompt Entrée/Sortie :

```
Input : Objet utilisateur { email, password, role }
Output : Token JWT avec payload { userId, email, role, exp }
Contraintes :
- Validation email format
- Password minimum 8 caractères
- Hash avec bcrypt
- Token expire en 24h
```

Bonnes pratiques et prompting

4. Types de prompts avancés

2. Prompt Chaîne de réflexion :

```
"Raisonne étape par étape pour créer ce système d'auth :
1. Analyse les besoins de sécurité
2. Définis l'architecture des composants
3. Choisis les bibliothèques appropriées
4. Implémente avec gestion d'erreurs
5. Crée les tests de validation"
```

Bonnes pratiques et prompting

4. Types de prompts avancés

3. Prompt RCT (Rôle Contexte Tâche) :

```
Rôle : Expert sécurité et architecte backend senior
Contexte : Application bancaire avec 100k+ utilisateurs, contraintes RGPD strictes
Tâche : Concevoir un système d'authentification multi-facteurs robuste et scalable

Exigences :
- Zero Trust Architecture
- Audit trail complet
- Performance <100ms
- Haute disponibilité 99.9%
```

Bonnes pratiques et prompting

4. Types de prompts avancés

4. Prompts Few-Shot :

```
Voici des exemples de fonctions bien structurées dans notre codebase :

Exemple 1 :
[Code exemple avec documentation, tests, gestion d'erreurs]

Exemple 2 :
[Autre exemple suivant les mêmes standards]

Maintenant, crée une fonction similaire pour [nouvelle fonctionnalité] en suivant le même style et les mêmes standards.
```

TP Pratique 2 : Prompting avancé

Exercice : Maîtrise du Prompting pour le Développement d'API

Contexte :

Vous êtes stagiaire développeur backend et devez utiliser ChatGPT pour vous aider à développer une API de gestion de produits. Votre mentor vous a confié cette mission pour tester vos compétences en prompting et votre capacité à obtenir du code de qualité professionnelle.

TP Pratique 2 : Prompting avancé

Situation :

L'entreprise d'e-commerce a besoin d'un endpoint `GET /api/products` pour son catalogue de 50 000 produits, avec un pic de 1000 requêtes/seconde. Stack : Node.js + Express + MongoDB + Redis.

Votre mission :

1. **Rédigez le prompt optimal** pour obtenir de ChatGPT une solution complète et professionnelle
2. **Testez votre prompt** et ajustez-le si nécessaire
3. **Évaluez la qualité** de la réponse obtenue

TP Pratique 2 : Prompting avancé

Défis à relever :

- Comment formuler votre demande pour obtenir du code production-ready ?
- Comment vous assurer que l'IA comprend bien le contexte business ?
- Comment obtenir une architecture propre et des bonnes pratiques ?
- Comment faire en sorte que le code soit adapté à votre équipe de juniors ?

Fonctionnalités attendues dans la réponse IA :

TP Pratique 2 : Prompting avancé

Questions de réflexion :

- Quelle structure de prompt maximise la qualité de la réponse ?
- Comment équilibrer précision et liberté créative de l'IA ?
- Quels éléments de contexte sont critiques à mentionner ?
- Comment s'assurer que l'IA valide sa compréhension avant de coder ?

Bonus :

Proposez une stratégie de prompting itératif pour affiner progressivement la solution.

Avantages pratiques de ChatGPT

1. Rapidité de génération de code

Gain de productivité mesurable :

```
Développement traditionnel d'une API CRUD :
- Analyse : 2h
- Architecture : 2h
- Implémentation : 8h
- Tests : 4h
- Documentation : 2h
Total : 18h
```

Avantages pratiques de ChatGPT

1. Rapidité de génération de code

```
Développement assisté par IA :
- Analyse + Architecture : 1h
- Prompting et génération : 2h
- Révision et ajustements : 3h
- Tests et validation : 2h
Total : 8h (gain de 55%)
```

Avantages pratiques de ChatGPT

2. Boilerplate Code et Scaffolding

Génération de structures projets :

```
# Prompt : "Crée la structure complète d'un projet Node.js avec Express, TypeScript, tests, Docker"

project-name/
├── src/
│   ├── controllers/
│   ├── services/
│   ├── models/
│   ├── middleware/
│   ├── routes/
│   ├── utils/
│   └── app.ts
├── tests/
│   ├── unit/
│   ├── integration/
│   └── e2e/
├── docker/
│   ├── Dockerfile
│   └── docker-compose.yml
├── .github/
│   └── workflows/
│       └── ci.yml
```

Avantages pratiques de ChatGPT

Configuration automatique :

```
// package.json généré automatiquement
{
  "name": "project-name",
  "version": "1.0.0",
  "scripts": {
    "dev": "nodemon src/app.ts",
    "build": "tsc",
    "start": "node dist/app.js",
    "test": "jest",
    "test:watch": "jest --watch",
    "test:coverage": "jest --coverage",
    "lint": "eslint src/**/*.ts",
    "lint:fix": "eslint src/**/*.ts --fix",
    "docker:build": "docker build -f docker/Dockerfile -t project-name .",
    "docker:run": "docker-compose -f docker/docker-compose.yml up"
  },
  "dependencies": {
    "express": "^4.18.2",
    "cors": "^2.8.5",
    "helmet": "^6.1.5",
    "morgan": "^1.10.0",
    "dotenv": "^16.0.3"
  },
  "devDependencies": {
    "@types/express": "^4.17.17",
    "@types/node": "^20.2.5",
    "@types/jest": "^29.5.2",
    "typescript": "^5.1.3",
    "nodemon": "^2.0.22",
    "jest": "^29.5.0",
    "ts-jest": "^29.1.0",
```

Avantages pratiques de ChatGPT

3. Aide au débogage

Analyse d'erreurs automatique :

Avantages pratiques de ChatGPT

4. Refactoring et optimisation

Amélioration automatique du code :

Avantages pratiques de ChatGPT

5. Migration de langages

JavaScript vers TypeScript :

Avantages pratiques de ChatGPT

6. Tests unitaires automatiques

Génération complète de tests :

Problèmes et limitations

1. Nécessité de connaissances techniques

Problème fondamental :

Pour valider et corriger le code généré par ChatGPT, il faut déjà maîtriser les technologies utilisées.

Exemple de code problématique généré :

Problèmes et limitations

2. Problèmes de qualité du code

Issues communes dans le code généré :

1. Gestion d'erreurs insuffisante :

2. Problèmes de performance :

Problèmes et limitations

3. Contradictions et incohérences

Exemple de réponses contradictoires :

Session 1 : "Pour la sécurité, utilise toujours HTTPS et JWT avec expiration courte"

Session 2 : "Les JWT peuvent être stockés dans localStorage sans problème"

Session 3 : "localStorage pour JWT est une vulnérabilité XSS, utilise httpOnly cookies"

Problèmes et limitations

Gestion des contradictions :

```
// Approche systématique pour éviter les contradictions
const securityPrompt = `
Contexte sécurité établi :
- Application banking avec données sensibles
- Conformité PCI-DSS requise
- Niveau de sécurité : Maximum
- Pas de compromise sur la sécurité

Avec ce contexte, recommande la meilleure approche pour l'authentification JWT.
`;

// Validation croisée
const validationPrompt = `
Valide cette implémentation JWT par rapport aux standards de sécurité :
[Code à valider]

Points à vérifier :
- Stockage sécurisé des tokens
- Expiration appropriée
```

Problèmes et limitations

4. Problèmes de sécurité

Vulnérabilités courantes dans le code généré :

1. Injection SQL/NoSQL :

2. Exposition de données sensibles :

Problèmes et limitations

5. Fraîcheur des données

Problème des données obsolètes :

Stratégie de validation :

Intégrations et extensions

1. Plugins ChatGPT essentiels

Code Interpreter :

```
# Capacités du Code Interpreter
# - Exécution Python en sandbox
# - Traitement de fichiers (CSV, JSON, images)
# - Génération de graphiques
# - Analyse de données
```

Intégrations et extensions

1. Plugins ChatGPT essentiels

Web Browsing :

```
// Utilisation pour recherche de documentation
const prompt = `
Recherche les dernières best practices pour l'authentification JWT en 2024.
Concentre-toi sur :
- Nouvelles recommandations OWASP
- Algorithmes recommandés
- Stockage sécurisé des tokens
- Gestion de l'expiration

Fournis des exemples de code à jour.
`;
```

Intégrations et extensions

2. Visual Studio Code - Extensions IA

Installation et configuration CodeGPT :

```
// settings.json
{
  "codegpt.apiKey": "your-openai-api-key",
  "codegpt.model": "gpt-4",
  "codegpt.temperature": 0.3,
  "codegpt.maxTokens": 2048,
  "codegpt.customPrompts": {
    "explainCode": "Explique ce code en français avec des commentaires détaillés",
    "addTests": "Génère des tests unitaires Jest complets pour ce code",
    "refactor": "Refactorise ce code pour améliorer la lisibilité et les performances",
    "addDocumentation": "Ajoute une documentation JSDoc complète"
  }
}
```

Intégrations et extensions

2. Visual Studio Code - Extensions IA

Raccourcis clavier utiles :

```
Ctrl+Shift+P → "CodeGPT: Explain Code"
Ctrl+Shift+P → "CodeGPT: Generate Tests"
Ctrl+Shift+P → "CodeGPT: Refactor Code"
Ctrl+Shift+P → "CodeGPT: Add Documentation"
```

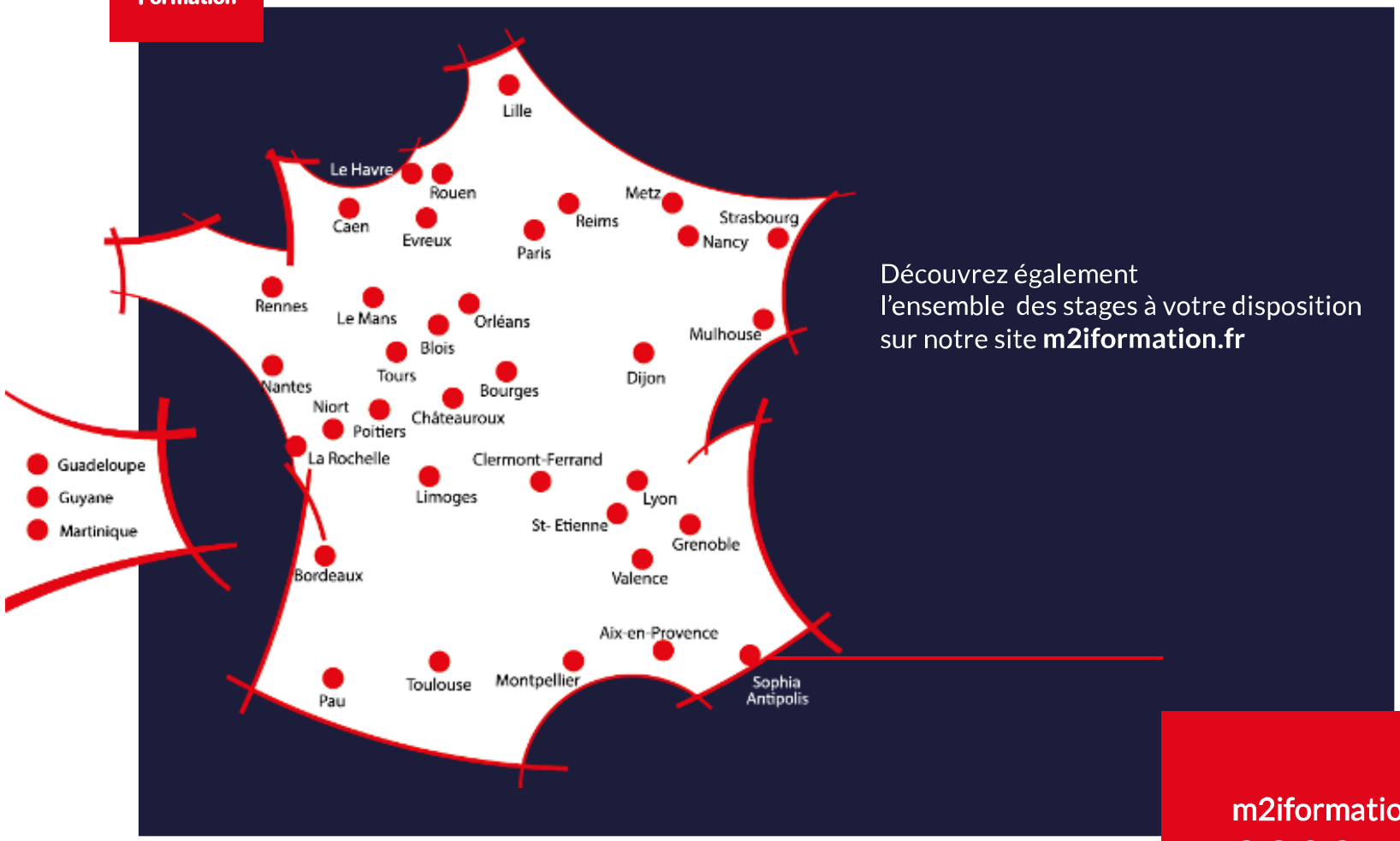
TP 3 : Configuration IDE complète

Objectif : Configurer un environnement de développement optimal avec IA

Étape 1 : Extensions VS Code

Étape 2 : Configuration workspace

Étape 3 : Workflow avec IA



Découvrez également
l'ensemble des stages à votre disposition
sur notre site **m2iformation.fr**

m2iformation.fr