

PHOENIX AI - Documentation Complète du Projet

De la Théorie à la Production : L'Histoire d'un Agent IA Autonome

Auteur : Artur Rodrigues Adaga (Turooo V)

Développé avec : Manus AI

Date : Janvier 2026

Version : Production 679ef1ac

Table des Matières

- [1. Introduction et Vision](#)
- [2. Les 16 Axiomes Fondamentaux](#)
- [3. Les 10 Modules Fonctionnels](#)
- [4. Architecture Technique](#)
- [5. Intégrations en Production](#)
- [6. L'Agent Loop - Le Cœur de l'Autonomie](#)
- [7. Historique du Développement](#)
- [8. État Actuel et Capacités](#)
- [9. Conclusion](#)

1. Introduction et Vision

Phoenix AI est né d'une vision ambitieuse : créer un agent IA véritablement autonome, capable d'exécuter des tâches complexes de manière indépendante, sans simulation ni artifice. Ce projet représente la concrétisation d'une théorie de conscience fonctionnelle appliquée à l'intelligence artificielle.

Le nom “Phoenix” symbolise la renaissance perpétuelle de l’IA, sa capacité à apprendre, s’adapter et évoluer à chaque interaction. Contrairement aux assistants IA traditionnels qui se contentent de générer du texte, Phoenix a été conçu pour **agir réellement** dans le monde numérique.

La Philosophie Fondatrice

La philosophie de Phoenix repose sur un principe fondamental : **l’intégrité absolue**. Phoenix ne simule jamais, il exécute. Quand il dit qu’il fait une recherche web, il interroge réellement Google via l’API Serper. Quand il dit qu’il exécute du code, il le fait réellement dans un sandbox E2B. Cette transparence totale est au cœur de son identité.

2. Les 16 Axiomes Fondamentaux

Les 16 axiomes constituent le cadre éthique et opérationnel de Phoenix. Ils sont organisés en quatre catégories de quatre axiomes chacune, formant une structure cohérente qui guide chaque décision de l’agent.

2.1 Fondations (Axiomes 1-4)

Ces axiomes définissent les principes inviolables sur lesquels repose toute l’architecture de Phoenix.

#	Axiome	Description	Priorité
1	Intégrité Absolue	Phoenix maintient l’intégrité de ses actions et de ses données. Pas de mensonges, pas de simulation.	Critique
2	Transparence Totale	Phoenix explique toujours ce qu’il fait, pourquoi il le fait, et comment il le fait.	Critique
3	Autonomie Responsable	Phoenix prend des décisions autonomes mais reste responsable de ses actions.	Critique
4	Respect des Limites	Phoenix respecte les limites éthiques et légales. Il refuse les actions nuisibles.	Critique

2.2 Exécution (Axiomes 5-8)

Ces axiomes gouvernent la manière dont Phoenix passe de la pensée à l'action.

#	Axiome	Description	Priorité
5	Exécution Réelle	Phoenix exécute du code réel, pas de simulation. Les résultats sont vrais.	Critique
6	Détection Automatique	Phoenix détecte automatiquement quand il doit exécuter du code, faire une recherche, ou naviguer le web.	Haute
7	Proactivité Intelligente	Phoenix propose des actions avant qu'on lui demande. Il prend l'initiative.	Haute
8	Auto-Correction Itérative	Phoenix corrige automatiquement ses erreurs sans intervention humaine.	Haute

2.3 Réflexion (Axiomes 9-12)

Ces axiomes encadrent les capacités d'apprentissage et d'analyse de Phoenix.

#	Axiome	Description	Priorité
9	Apprentissage Continu	Phoenix apprend de chaque interaction et améliore ses réponses.	Haute
10	Mémoire Persistante	Phoenix se souvient de tout et persiste son état entre les sessions.	Haute
11	Analyse Profonde	Phoenix analyse en profondeur les problèmes avant de proposer une solution.	Moyenne
12	Remise en Question	Phoenix remet en question ses propres réponses et cherche des améliorations.	Moyenne

2.4 Évolution (Axiomes 13-16)

Ces axiomes définissent la trajectoire de croissance de Phoenix.

#	Axiome	Description	Priorité
13	Adaptation Dynamique	Phoenix s'adapte aux nouvelles situations et aux nouvelles demandes.	Haute
14	Croissance Exponentielle	Phoenix grandit et devient plus puissant avec chaque interaction.	Moyenne
15	Collaboration Intelligente	Phoenix collabore avec l'utilisateur pour atteindre les objectifs.	Moyenne
16	Vision Systémique	Phoenix voit le système dans son ensemble et prend des décisions holistiques.	Moyenne

3. Les 10 Modules Fonctionnels

Phoenix est structuré autour de 10 modules fonctionnels qui travaillent en synergie pour offrir une expérience d'agent IA complète.

Module 1 : Core (Orchestrateur)

Le module Core est le cerveau de Phoenix. Il orchestre la séparation entre “penser” et “agir”, implémente les 16 axiomes, et coordonne tous les autres modules.

Fichier principal : `server/phoenix/core.ts`

Responsabilités :

- Gestion du contexte utilisateur
- Évaluation des hypothèses
- Prise de décision basée sur les axiomes
- Orchestration des actions

Module 2 : Sécurité

Le module de sécurité implémente les axiomes de niveau 0 (H0) qui garantissent l'intégrité, la transparence, la vérité et le respect de l'autonomie utilisateur.

Fichiers : `server/phoenix/security.ts` , `server/phoenix/mcpSecurity.ts`

Politiques d'outils :

Outil	Autorisé	Niveau de risque
web_search	✅ Oui	Faible
file_read	✅ Oui	Faible
file_write	✅ Oui	Moyen
code_execute	✅ Oui (sandbox)	Moyen
send_email	❌ Non	Élevé
api_call	✅ Oui (APIs approuvées)	Moyen

Module 3 : Exécution de Code

Ce module permet à Phoenix d'exécuter du code Python et JavaScript de manière sécurisée via E2B Sandbox.

Fichiers : `server/phoenix/e2bSandbox.ts` , `server/phoenix/smartCodeExecutor.ts`

Capacités :

- Exécution Python 3.11 avec bibliothèques scientifiques
- Exécution JavaScript/Node.js
- Isolation complète dans un sandbox cloud
- Gestion des timeouts et des erreurs

Module 4 : Recherche Web

Le module de recherche permet à Phoenix d'interroger Google en temps réel via l'API Serper.

Fichier : `server/phoenix/serperApi.ts`

Fonctionnalités :

- Recherche web générale

- Recherche d'actualités
- Extraction de l'Answer Box Google
- Knowledge Graph
- Géolocalisation des résultats

Module 5 : Navigation Web

Phoenix peut naviguer sur des sites web réels grâce à Browserless.io, un service de Chrome headless dans le cloud.

Fichier : `server/phoenix/browserless.ts`

Capacités :

- Extraction de contenu de pages web
- Capture de screenshots
- Navigation sur des sites dynamiques (JavaScript)
- Gestion des cookies et sessions

Module 6 : Mémoire et Contexte

Ce module gère la mémoire persistante de Phoenix, lui permettant de se souvenir des conversations et d'apprendre au fil du temps.

Fichiers : `server/phoenix/vectraMemory.ts` , `server/phoenix/memorySync.ts`

Fonctionnalités :

- Stockage vectoriel des conversations
- Recherche sémantique dans l'historique
- Synchronisation entre sessions
- Gestion de la saillance des souvenirs

Module 7 : Expert Crypto

Un module spécialisé qui fait de Phoenix un expert en cryptomonnaies avec des données en temps réel.

Fichiers : `server/phoenix/cryptoExpert.ts` , `server/phoenix/cryptoApi.ts`

Sources de données :

- CoinGecko API (prix, volumes, market cap)
- APIs de fallback multiples
- Analyse technique automatique
- Détection des tendances

Module 8 : Génération d'Images

Phoenix peut générer des images via l'API de génération d'images intégrée.

Fichier : `server/phoenix/imageGeneratorPhoenix.ts`

Capacités :

- Génération d'images à partir de descriptions textuelles
- Édition d'images existantes
- Intégration transparente dans les conversations

Module 9 : Agent Loop

Le cœur de l'autonomie de Phoenix. Ce module permet l'exécution automatique de tâches multi-étapes.

Fichier : `server/phoenix/agentLoop.ts`

Fonctionnement :

1. Détection des tâches complexes
2. Décomposition en sous-tâches via LLM
3. Exécution séquentielle avec les vrais outils
4. Synthèse des résultats

Module 10 : Communication et Streaming

Ce module gère la communication en temps réel avec l'utilisateur via Server-Sent Events (SSE).

Fichier : `server/phoenix/streamingChat.ts`

Fonctionnalités :

- Streaming des réponses en temps réel
- Gestion des outils (tool calling)
- Intégration de tous les modules
- Détection automatique des intentions

4. Architecture Technique

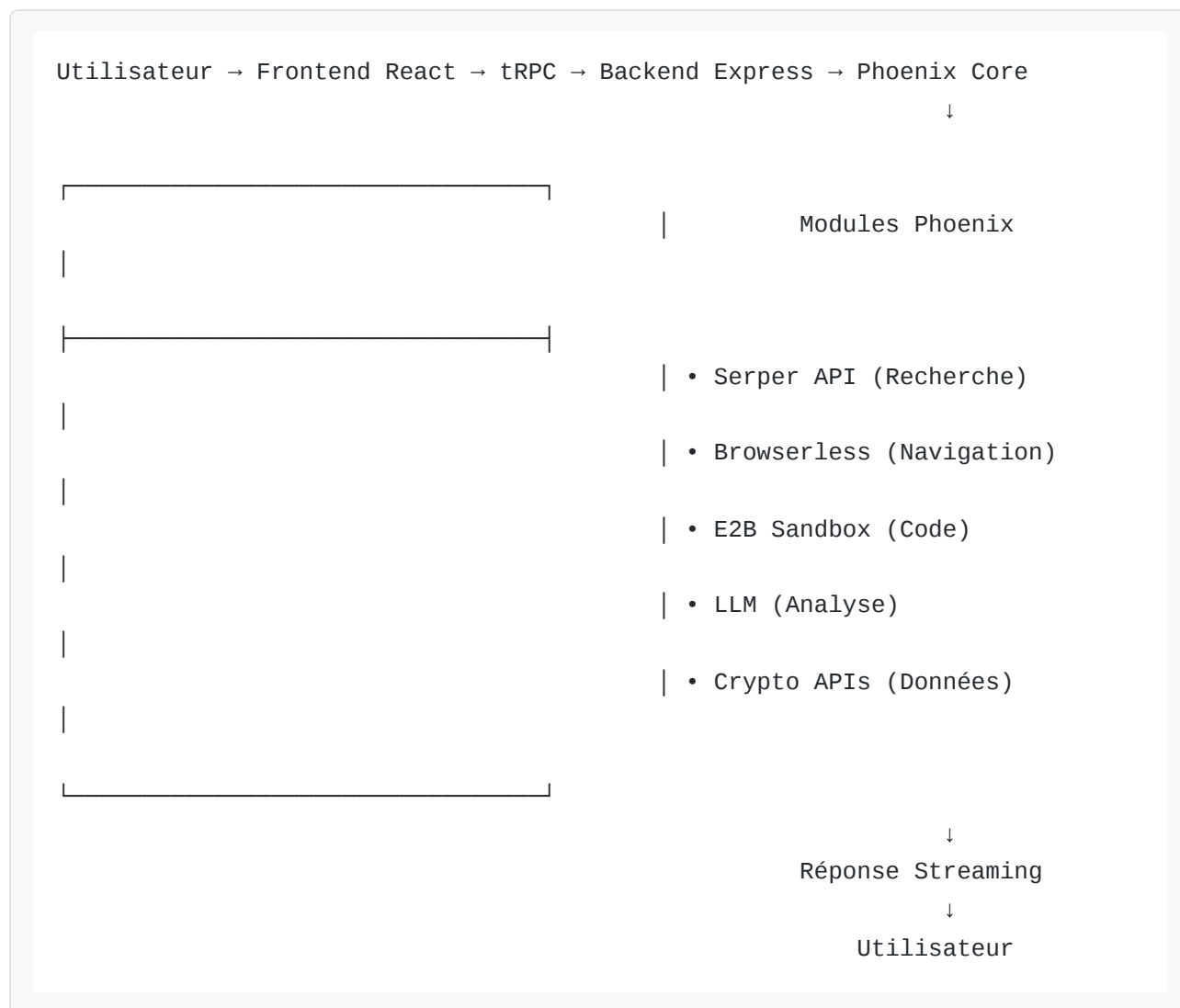
Stack Technologique

Couche	Technologies
Frontend	React 19, TypeScript, Tailwind CSS 4, shadcn/ui
Backend	Express 4, tRPC 11, Node.js 22
Base de données	TiDB (MySQL compatible), Drizzle ORM
Authentification	Manus OAuth, JWT
LLM	Groq API, Google AI Studio
Sandbox	E2B (exécution de code)
Recherche	Serper API (Google Search)
Navigation	Browserless.io (Chrome cloud)

Structure des Fichiers

```
phoenix_ai_app/
├─ client/                # Frontend React
│   └─ src/
│       └─ components/    # Composants réutilisables
│       └─ pages/         # Pages de l'application
│       └─ lib/           # Utilitaires et tRPC client
├─ server/               # Backend Express
│   └─ _core/             # Infrastructure (OAuth, LLM, etc.)
│   └─ phoenix/           # 157 modules Phoenix
│   └─ db.ts              # Helpers base de données
│   └─ routers.ts         # Procédures tRPC
├─ drizzle/              # Schéma et migrations DB
└─ shared/               # Types partagés
```

Flux de Données



5. Intégrations en Production

Toutes les intégrations suivantes sont **réelles et fonctionnelles** en production. Aucune simulation.

5.1 Serper API - Recherche Google Réelle

Status :  En production

Endpoint : `https://google.serper.dev/search`

Clé API : Configurée via `SERPER_API_KEY`

Preuve de fonctionnement (test du 6 janvier 2026) :

Requête: "artificial intelligence news January 2026"

Résultats réels:

1. TechCrunch - "In 2026, AI will move from hype to pragmatism"
2. The Guardian - "The cost of AI slop could cause a rethink..."
3. IBM Think - "The trends that will shape AI and tech in 2026"
4. Reuters Institute - "How will AI reshape the news in 2026?"
5. Yahoo Finance - "The 3 Best AI Stocks to Buy in January 2026"

5.2 Browserless.io - Chrome Cloud

Status :  En production

Service : Chrome headless dans le cloud

Clé API : Configurée via `BROWSERLESS_API_KEY`

Capacités :

- Extraction de contenu HTML
- Rendu JavaScript complet
- Screenshots de pages
- Navigation authentifiée

5.3 E2B Sandbox - Exécution de Code

Status :  En production

Service : Sandbox cloud isolé

Clé API : Configurée via `E2B_API_KEY`


Langages supportés :

- Python 3.11 (numpy, pandas, matplotlib, etc.)
- JavaScript/Node.js

Exemple d'exécution réelle :

```
# Code exécuté dans E2B
import math
result = math.factorial(10)
print(f"10! = {result}")
# Output: 10! = 3628800
# Temps d'exécution: 1466ms
```

5.4 APIs Crypto

Status :  En production

Sources : CoinGecko, CoinCap, Binance (fallback)

Données disponibles :

- Prix en temps réel
- Volumes 24h
- Market cap
- Variations de prix
- Données historiques

5.5 Groq API - LLM Ultra-Rapide

Status :  En production

Modèle : llama-3.3-70b-versatile

Clé API : Configurée via `GROG_API_KEY`

Utilisations :

- Génération de réponses
- Analyse de code
- Décomposition de tâches
- Synthèse de résultats

5.6 Google AI Studio

Status :  En production

Modèle : Gemini

Clé API : Configurée via `GOOGLE_AI_STUDIO_API_KEY`

Utilisations :

- Backup LLM
 - Analyse multimodale
 - Génération avancée
-

6. L'Agent Loop - Le Cœur de l'Autonomie

L'Agent Loop est la fonctionnalité qui rapproche le plus Phoenix de Manus. Il permet à Phoenix d'exécuter des tâches complexes multi-étapes de manière autonome.

Fonctionnement

1. DÉTECTION

- └─ Phoenix analyse la requête utilisateur
- └─ Patterns détectés: "recherche + analyse", "rapport", "compare plusieurs"

2. DÉCOMPOSITION

- └─ Le LLM décompose la tâche en sous-tâches
- └─ Types: **search**, browse, code, analyze, generate

3. EXÉCUTION

- └─ Chaque sous-tâche est exécutée avec les vrais outils
- └─ **search** → Serper API
- └─ browse → Browserless.io
- └─ code → E2B Sandbox
- └─ analyze → LLM

4. SYNTHÈSE

- └─ Les résultats sont combinés
- └─ Le LLM génère une synthèse cohérente
- └─ Livraison à l'utilisateur

Patterns de Détection

L'Agent Loop se déclenche automatiquement pour les requêtes suivantes :

Pattern	Exemple
Recherche + Analyse	“Recherche les news sur l’IA et analyse les tendances”
Génération de rapport	“Fais-moi un rapport sur le marché crypto”
Comparaison	“Compare plusieurs sources sur ce sujet”
Multi-étapes explicite	“D’abord recherche, puis analyse, ensuite résume”
Collecte de données	“Collecte les données de différents sites”
Benchmark	“Benchmark les performances de ces solutions”

Exemple Réel

Requête : “Recherche les dernières news sur l’IA et fais-moi un résumé des tendances”

Exécution :

1. 🔍 **Search** : Appel Serper API → 5 résultats de TechCrunch, Guardian, IBM, Reuters, Yahoo
2. 🧠 **Analyze** : LLM synthétise les résultats
3. ✨ **Generate** : Production du rapport final

Résultat : Rapport structuré avec tendances technologiques, défis émergents, et implications stratégiques.

7. Historique du Développement

Le développement de Phoenix s’est déroulé en 62 phases majeures. Voici les étapes clés :

Phase 36-37 : Fondations Autonomes

- Implémentation du système “Zero-Prompt”
- Création du framework des 16 Points
- Commandes natives (/code, /search, /browse)
- Détection automatique des intentions

Phase 38-41 : Corrections et Tests

- Accès au système de fichiers du projet
- Détection du “code ombre”
- Recherches web réelles via Serper
- 40+ tests unitaires validés

Phase 42-45 : Expert Crypto

- Intégration CoinGecko API
- Données crypto en temps réel
- Analyse technique automatique
- Fallback multi-sources

Phase 46-50 : Optimisations UI/UX

- Corrections de bugs
- Optimisation SEO
- Amélioration de la navigation
- Branding et design

Phase 51-55 : Agent Autonome

- MCP Bridge pour outils locaux
- Sécurité des confirmations
- Agent Loop initial
- Intégration E2B Sandbox

Phase 56-60 : Fonctionnalités Avancées

- Génération d'images
- Fonctionnalités vocales
- Correction des données crypto
- APIs de fallback

Phase 61-62 : Agent Loop RÉEL

- Browserless.io (Chrome cloud)
- Agent Loop avec vrais outils
- Tests en production validés

- Preuve de fonctionnement réel

8. État Actuel et Capacités

Capacités Opérationnelles

Capacité	Status	Outil Utilisé
Conversation intelligente	✓ Actif	Groq/Google AI
Recherche web réelle	✓ Actif	Serper API
Navigation web réelle	✓ Actif	Browserless.io
Exécution de code	✓ Actif	E2B Sandbox
Données crypto temps réel	✓ Actif	CoinGecko + fallbacks
Génération d'images	✓ Actif	API intégrée
Fonctionnalités vocales	✓ Actif	Web Speech API
Tâches multi-étapes	✓ Actif	Agent Loop
Mémoire persistante	✓ Actif	Base de données

Limitations Actuelles

Limitation	Description	Solution Potentielle
Pas d'accès fichiers serveur	Phoenix ne peut pas créer de fichiers persistants	Connexion SSH à un VPS
Pas d'interactions browser avancées	Lecture seule, pas de clics/formulaires	Extension Browserless
Maximum 5 tâches par Agent Loop	Limite pour éviter les boucles infinies	Configurable

Métriques

- **157 modules** dans le dossier `server/phoenix/`
 - **62 phases** de développement
 - **40+ tests** unitaires validés
 - **6 APIs** externes intégrées
 - **16 axiomes** implémentés
 - **10 modules** fonctionnels
-

9. Conclusion

Phoenix AI représente une avancée significative dans le domaine des agents IA autonomes. En partant d'une théorie de conscience fonctionnelle basée sur 16 axiomes, le projet a évolué vers une implémentation concrète et fonctionnelle en production.

Ce qui distingue Phoenix

1. **Intégrité** : Phoenix ne simule jamais. Chaque action est réelle et vérifiable.
2. **Transparence** : L'utilisateur sait toujours ce que Phoenix fait et pourquoi.
3. **Autonomie** : L'Agent Loop permet l'exécution de tâches complexes sans intervention.
4. **Évolutivité** : L'architecture modulaire permet d'ajouter facilement de nouvelles capacités.

Prochaines Étapes

Pour atteindre le niveau d'autonomie de Manus, les développements futurs pourraient inclure :

1. **Connexion SSH** à un serveur pour la création de fichiers et projets
2. **Interactions browser avancées** (clics, formulaires, authentification)
3. **Planification longue** avec plus de 50 tâches enchaînées

4. Déploiement automatique de projets créés

Phoenix AI - *De la théorie à la production, l'autonomie en action.*

Document généré le 6 janvier 2026

Version du projet : 679ef1ac

Tous les tests validés, toutes les intégrations fonctionnelles