# Cahier des charges fonctionnel et technique

## 1. Contexte & objectifs

Projet : Plateforme de veille et de décision d'investissement en bourse  
Objectif principal :  
- Offrir une interface de visualisation en temps réel du sentiment (positif / négatif / neutre) et du volume d’articles financiers par marché (CAC40, marchés américains, etc.)  
- Aider à la prise de décision pour l’investissement en bourse  
  
Motivations :  
- Apprentissage et montée en compétence Angular + SCSS  
- Produit commercialisable pour investissement personnel et tiers  
  
Critère de réussite :  
- Application développée de bout en bout et fonctionnelle selon le périmètre décrit

## 2. Périmètre fonctionnel

|  |  |
| --- | --- |
| Fonctionnalité | Description |
| Scraping | Collecte d’articles financiers toutes les 5 minutes depuis plusieurs sites (flux HTML/API) |
| Analyse de sentiment | Traitement BERT spécialisé pour classifier chaque article (positif / négatif / neutre) |
| Dashboard graphiques | Courbes d’évolution et diagrammes camembert du sentiment par marché ; Nombre d’articles en temps réel |
| Filtres & agrégations | Filtrer par date, source, marché, mot-clé |

## 3. Authentification & gestion des utilisateurs

- Inscription libre : email / mot de passe  
- Profils : Gratuit (1–2 marchés), Standard (sous-ensemble de marchés), Premium (tous les marchés)  
- Permissions : lecture selon forfait  
- Sécurité : OAuth2 + JWT

## 4. Monétisation & facturation

- Stripe : abonnement mensuel récurrent  
- Plans : Gratuit, Standard, Premium  
- Facturation : factures automatiques, relances, gestion TVA via Stripe

## 5. UX/UI & Design System

- Framework UI : Angular Material + theming SCSS  
- Responsive : mobile/tablette ; futur APK/IOS  
- Charte graphique : à définir (logo, palette, typo)  
- Wireframes : à esquisser avant dev

## 6. Architecture technique

- Front-end : Angular + SCSS  
- Bibliothèques : ng2-charts (Chart.js), Angular Material  
- Back-end : à définir (Node.js/Express ou autre)  
- Base de données : SQL/NoSQL selon volumétrie  
- Hébergement : VPS Linux (IONOS)  
- CI/CD : Docker + GitHub Actions

## 7. Performance & scalabilité

- Fréquence : scraping et mise à jour toutes les 5 minutes  
- Volumétrie : à chiffrer  
- Monitoring : Sentry, Datadog, etc.

## 8. Sécurité & conformité

- RGPD : consentement, anonymisation  
- MFA : à envisager  
- API : rate limiting, sécurisation JWT/OAuth2

## 9. Tests & maintenance

- Tests unitaires : Jest  
- Tests e2e : Cypress  
- Support & SLA : à définir (J+1, J+4)

## 10. Planning & budget

|  |  |  |
| --- | --- | --- |
| Étape | Livrable | Durée estimée |
| Spécifications détaillées | Cahier des charges validé | 1 semaine |
| Design UI/UX | Wireframes + maquettes | 2 semaines |
| Mise en place infra | CI/CD, Docker, serveur | 1 semaine |
| Développement front | Composants clés du dashboard | 4 semaines |
| Développement back | API scraping + analyse BERT | 4 semaines |
| Intégration & tests | Tests unitaires et e2e | 2 semaines |
| Préparation lancement | Documentation + déploiement | 1 semaine |

Budget estimé : 2 000 €  
Équipe : 1 lead dev, soutien possible d’un développeur ami

## 11. Découpage en tickets

| ID | Titre | Description | Domaine |
| --- | --- | --- | --- |
| INFRA-001 | Init dépôt Git & branche principale | Création du repo, protection de la branche principale, règles de merge (pull requests obligatoires). | Infra |
| INFRA-002 | Configuration CI/CD | Mise en place GitHub Actions + Docker build/push + déploiement sur VPS IONOS. | Infra |
| INFRA-003 | Définir et provisionner l’environnement | Script d’infra (ansible, terraform ou scripts bash) pour config serveur Linux (nginx, docker). | Infra |
| DESIGN-001 | Créer moodboard & charte graphique | Proposition de palette, typographie, composants UI (butterflies sur Angular Material). | Design/UX |
| DESIGN-002 | Wireframes dashboard principal | Maquettes low-fi des écrans : login, liste marchés, vue graphique, filtres. | Design/UX |
| DESIGN-003 | Prototype interactif sur Figma | Liens cliquables entre écrans, transitions clés, validations UX. | Design/UX |
| FE-001 | Init projet Angular + SCSS | Génération Angular CLI, configuration SCSS global, theming Angular Material. | Front-end |
| FE-002 | Module Core & Shared | Création des modules pour services réutilisables, interceptors HTTP, composants partagés. | Front-end |
| FE-003 | Routing & layout | Configuration des routes (login, dashboard, settings) et du layout principal avec menu latéral. | Front-end |
| FE-004 | Service API HTTP | Création du service Angular pour consommer les endpoints back-end (fetch marchés, sentiments, articles). | Front-end |
| FE-005 | Composant Login | Formulaire de connexion, validation, appel AuthService, gestion des tokens JWT. | Front-end |
| FE-006 | Composant Tableau de bord | Structure de la page dashboard, intégration des placeholders pour charts et filtres. | Front-end |
| FE-007 | Intégration ng2-charts | Installation et configuration des modules Chart.js, composants pour courbes et camemberts. | Front-end |
| FE-008 | Filtres & sélection marché | UI pour filtrer par date, source, mot-clé et sélectionner un marché. | Front-end |
| FE-009 | Théming & responsive design | Ajustements SCSS pour mobile/tablette, tests breakpoints, amélioration UX sur petits écrans. | Front-end |
| BE-001 | Init projet back-end (Node.js/Express) | Mise en place du projet, structure dossiers (controllers, services, models), TypeScript config. | Back-end |
| BE-002 | Schéma base de données | Modélisation des entités (Users, Subscriptions, Markets, Articles, Sentiments). | Back-end |
| BE-003 | Service de scraping | Tâche cron toutes les 5 min pour récupérer HTML/API et stocker articles bruts. | Back-end |
| BE-004 | Intégration BERT | Pipeline pour lancer le modèle BERT sur les articles et stocker sentiment (+ score). | Back-end |
| BE-005 | Endpoints markets & sentiments | Routes GET pour lister marchés, récupérer tendances de sentiment et volumes d’articles. | Back-end |
| BE-006 | Authentification OAuth2 + JWT | Endpoints login/signup, génération/validation tokens, refresh tokens. | Back-end |
| BE-007 | Intégration Stripe Subscriptions | Webhooks Stripe, création plans, gestion des statuts d’abonnement, facturation automatique. | Back-end |
| BE-008 | Sécurisation API | Rate-limiting, helmet, CORS, validation des payloads, tests de vulnérabilité basique. | Back-end |
| QA-001 | Tests unitaires front (Jest) | Couverture des services et composants critiques (AuthService, ChartComponent). | QA |
| QA-002 | Tests unitaires back (Jest/Mocha) | Coverage pour la logique de scraping, BERT, AuthService, StripeService. | QA |
| QA-003 | Tests end-to-end (Cypress) | Scénarios : connexion, navigation dashboard, filtres, paiement test Stripe. | QA |
| QA-004 | Automation déploiement & rollback tests | Vérifier qu’après un build CI, le déploiement ne casse pas l’application (smoke tests). | QA |
| DOC-001 | Rédaction README & guide de contribution | Instructions pour lancer localement, normes de code, pull request template. | Documentation |
| DOC-002 | Guide utilisateur & manuel d’installation | PDF ou page web décrivant le parcours utilisateur et la mise en prod sur IONOS. | Documentation |