🔍 UptimeWarden - Application de Surveillance de Disponibilité

Application web complète pour surveiller la disponibilité et les performances de sites web, serveurs et API en temps réel.

📋 Fonctionnalités

Surveillance Multi-Protocoles

- **HTTP/HTTPS** - Surveillance des sites web et API
- **Ping** - Vérification de la connectivité réseau
- **Port TCP** - Surveillance des ports spécifiques
- **Mot-clé** - Vérification de la présence de contenu

Fonctionnalités Principales

✅ Contrôles en Temps Réel

- Intervalles configurables (1-60 minutes)
- Vérifications automatiques continues
- Historique détaillé des checks

🔔 Système d'Alertes Multi-Canaux

- Email (SMTP)
- SMS (Twilio)
- Slack
- Discord

📊 Tableau de Bord Interactif

- Statistiques de disponibilité en temps réel
- Graphiques de temps de réponse
- Historique des incidents
- Métriques de performance

🌐 Pages de Statut

- Pages publiques ou privées
- Personnalisables (logo, couleurs)
- Partage avec clients/équipe

🔐 Authentification & Gestion

- Système de connexion sécurisé (JWT)
- Gestion des rôles utilisateurs
- Préférences de notification

🔒 Vérificateur SSL

- Validation des certificats SSL
- Alertes d'expiration (30 jours)
- Surveillance continue

📈 Statistiques & Probabilités

- Calcul d'uptime automatique
- Prédictions basées sur l'historique
- Rapports détaillés

🏗️ Architecture Technique

Frontend

- **React.js** - Framework UI
- **Tailwind CSS** - Styling
- **Recharts** - Visualisations
- **Lucide React** - Icônes
- **WebSocket** - Temps réel

Backend

- **Node.js** - Runtime
- **Express.js** - Framework API
- **MongoDB** - Base de données
- **Mongoose** - ODM
- **JWT** - Authentication
- **node-cron** - Scheduler
- **WebSocket (ws)** - Communication temps réel

Services Externes

- **Nodemailer** - Envoi d'emails
- **Twilio** - SMS (optionnel)
- **Slack/Discord** - Webhooks (optionnel)

🚀 Installation

Prérequis

- Node.js >= 16.0.0
- MongoDB >= 5.0
- npm >= 8.0.0

Étapes d'installation

1. **Cloner le repository**

```bash
git clone https://github.com/votre-username/uptimewarden.git
cd uptimewarden
```

2. **Installer les dépendances**

```bash
```

```
# Installer les dépendances backend
npm install

# Installer les dépendances frontend
cd client
npm install
cd ..
```

3. **Configuration de l'environnement**

```
bash

# Copier le fichier d'exemple
cp .env.example .env

# Éditer le fichier .env avec vos configurations
nano .env
```

4. **Démarrer MongoDB**

```
bash

# Sur Linux/Mac
sudo systemctl start mongod

# Sur Windows avec MongoDB Community
net start MongoDB
```

5. **Lancer l'application**

Mode développement:

```
bash

# Terminal 1 - Backend
npm run dev

# Terminal 2 - Frontend
npm run client
```

Mode production:

```
bash

# Build du frontend
npm run build

# Démarrer le serveur
npm start
```

L'application sera accessible sur:

- Frontend: http://localhost:3000
- Backend API: http://localhost:5000

📝 Configuration

Variables d'environnement

Créez un fichier `.env` à la racine du projet:

```
env

# Serveur
PORT=5000
NODE_ENV=production

# MongoDB
MONGODB_URI=mongodb://localhost:27017/uptimewarden

# JWT
JWT_SECRET=votre-cle-secrete-tres-securisee

# Email (Gmail)
EMAIL_USER=votre-email@gmail.com
EMAIL_PASSWORD=votre-mot-de-passe-app

# SMS (Twilio - optionnel)
TWILIO_ACCOUNT_SID=
TWILIO_AUTH_TOKEN=
TWILIO_PHONE_NUMBER=

# Webhooks (optionnel)
SLACK_WEBHOOK_URL=
DISCORD_WEBHOOK_URL=
```

Configuration Gmail

Pour utiliser Gmail pour les alertes email:

1. Activer l'authentification à deux facteurs
2. Générer un mot de passe d'application
3. Utiliser ce mot de passe dans `EMAIL_PASSWORD`

🔧 Utilisation

Créer un moniteur

1. Connectez-vous à l'application
2. Cliquez sur "Ajouter un Moniteur"
3. Remplissez le formulaire:
   - Nom du moniteur
   - Type (HTTP/HTTPS/Ping/Port)
   - URL ou adresse
   - Intervalle de vérification
   - Méthodes d'alerte

Gérer les alertes

1. Accédez aux paramètres du moniteur
2. Activez/désactivez les canaux d'alerte
3. Configurez les seuils d'alerte

Créer une page de statut

1. Allez dans "Pages de Statut"
2. Créez une nouvelle page
3. Sélectionnez les moniteurs à afficher
4. Personnalisez l'apparence
5. Partagez l'URL publique

📊 API Endpoints

Authentication

```
POST /api/auth/register - Créer un compte
POST /api/auth/login    - Se connecter
```

Monitors

```
GET    /api/monitors          - Liste des moniteurs
POST   /api/monitors          - Créer un moniteur
PUT    /api/monitors/:id      - Modifier un moniteur
DELETE /api/monitors/:id      - Supprimer un moniteur
GET    /api/monitors/:id/history - Historique des checks
```

Incidents

```
GET /api/incidents - Liste des incidents
```

Status Pages

```
POST /api/status-pages      - Créer une page de statut
GET  /api/status-pages/:slug - Accéder à une page de statut
```

Statistics

```
GET /api/stats - Statistiques globales
```

🔐 Sécurité

- Authentification JWT
- Mots de passe hashés (bcrypt)
- Protection CORS
- Validation des entrées
- Rate limiting (recommandé pour production)
- HTTPS recommandé en production

🧪 Tests

```bash
```

```
# Tests unitaires
npm test

# Tests d'intégration
npm run test:integration

# Coverage
npm run test:coverage
```

📦 Déploiement

Docker

dockerfile

```
# Créer une image Docker
docker build -t uptimewarden .

# Lancer le container
docker run -p 5000:5000 -e MONGODB_URI=mongodb://mongo:27017/uptimewarden uptimewarden
```

Heroku

bash

```
heroku create uptimewarden
heroku addons:create mongolab
git push heroku main
```

VPS/Cloud

1. Configurer MongoDB
2. Installer Node.js
3. Cloner le repository
4. Configurer les variables d'environnement
5. Utiliser PM2 pour la gestion des processus

bash

```
npm install -g pm2
pm2 start server.js --name uptimewarden
pm2 startup
pm2 save
```

🤝 Contribution

Les contributions sont les bienvenues !

1. Fork le projet
2. Créer une branche (`git checkout -b feature/AmazingFeature`)
3. Commit les changements (`git commit -m 'Add AmazingFeature'`)
4. Push vers la branche (`git push origin feature/AmazingFeature`)
5. Ouvrir une Pull Request

📄 Licence

Ce projet est sous licence MIT. Voir le fichier `LICENSE` pour plus de détails.

👥 Auteurs

- **UptimeWarden Team** - GitHub

🙏 Remerciements

- Inspiré par UptimeRobot
- Communauté React et Node.js
- Tous les contributeurs

📞 Support

- Documentation: docs.uptimewarden.com
- Issues: GitHub Issues
- Email: support@uptimewarden.com

🗺️ Roadmap

- [ ] Application mobile (React Native)
- [ ] Intégrations supplémentaires (PagerDuty, Telegram)
- [ ] Tests de performance (Load testing)
- [ ] Surveillance de certificats multiples
- [ ] API webhooks sortants
- [ ] Rapports PDF automatiques
- [ ] Dashboard multi-utilisateurs
- [ ] Authentification SSO
- [ ] Mode dark natif

Note: Cette application est conçue à des fins éducatives et professionnelles. Assurez-vous de respecter les politiques des services que vous surveillez.