PricePulse 📈

Smart Price Monitoring & Alert System for Amazon Products

PricePulse is a full-stack web application that automatically tracks Amazon product prices and sends alerts when prices drop below your target threshold. Built with React.js frontend and Node.js backend, it uses web scraping technology to monitor prices in real-time.

🌟 Features

- **Real-time Price Tracking**: Automated price monitoring using Playwright web scraping
- **Smart Alerts**: Get notified when product prices drop below your target price
- **User-friendly Dashboard**: Clean, responsive interface to manage tracked products
- **Historical Price Data**: Track price trends over time
- **Bot Protection Handling**: Advanced anti-detection mechanisms for reliable scraping
- **Scheduled Monitoring**: Background service for continuous price checking
- **RESTful API**: Well-structured API endpoints for all operations

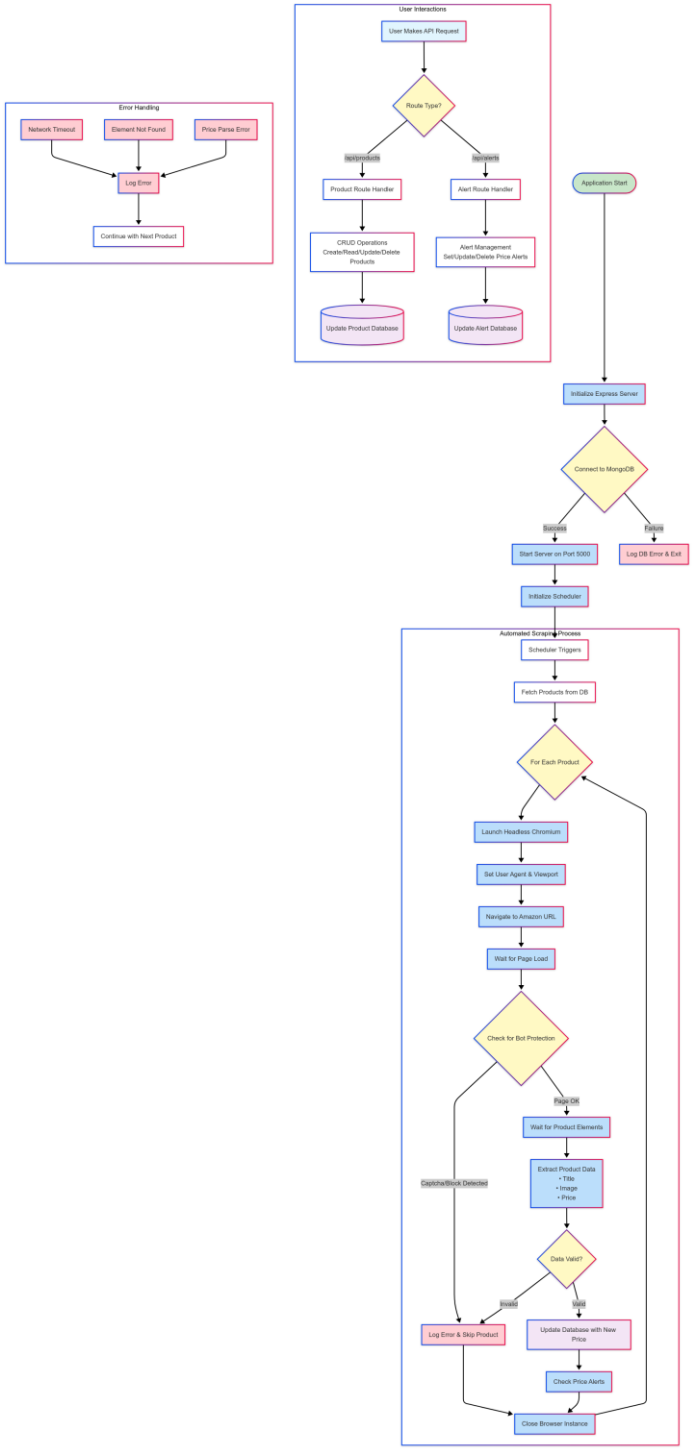
📐 Architecture

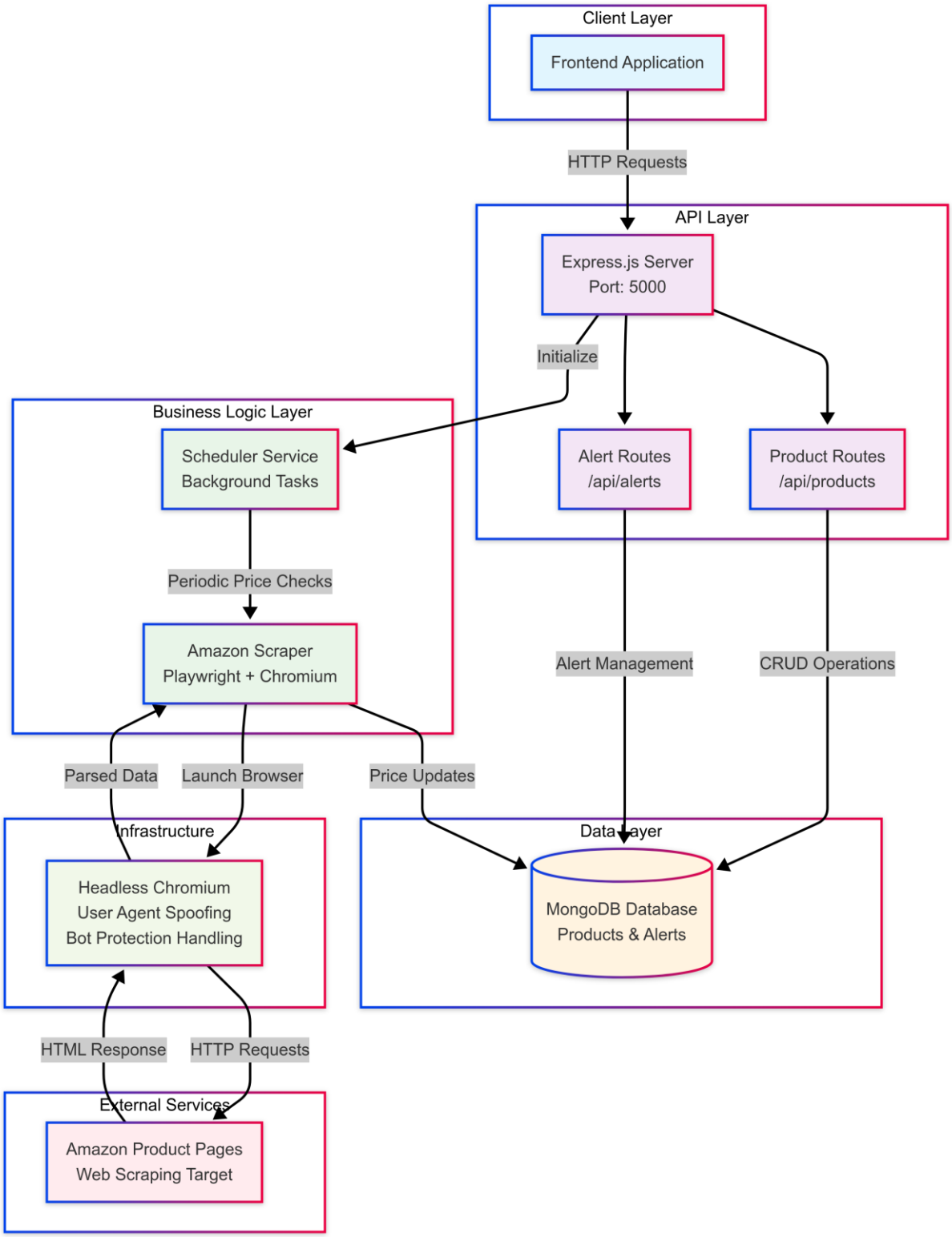
System Architecture

```
graph TB
    subgraph "Client Layer"
        UI[Frontend Application<br/>React.js]
    end

    subgraph "API Layer"
        SERVER[Express.js Server<br/>Port: 5000]
        ROUTES1[Product Routes<br/>/api/products]
        ROUTES2[Alert Routes<br/>/api/alerts]
    end

    subgraph "Business Logic Layer"
        SCHEDULER[Scheduler Service<br/>Background Tasks]
        SCRAPER[Amazon Scraper<br/>Playwright + Chromium]
    end
```

```
subgraph "Data Layer"
    MONGO[(MongoDB Database<br/>Products & Alerts)]
end

subgraph "External Services"
    AMAZON[Amazon Product Pages<br/>Web Scraping Target]
end

UI -->|HTTP Requests| SERVER
SERVER --> ROUTES1
SERVER --> ROUTES2
ROUTES1 -->|CRUD Operations| MONGO
ROUTES2 -->|Alert Management| MONGO
SCHEDULER -->|Periodic Checks| SCRAPER
SCRAPER -->|Web Scraping| AMAZON
SCRAPER -->|Price Updates| MONGO
```

Application Flow




Client Layer
Frontend Application
HTTP Requests
API Layer
Express.js Server
Port: 5000
Initialize
Business Logic Layer
Scheduler Service
Background Tasks
Alert Routes
/api/alerts
Product Routes
/api/products
Periodic Price Checks
Amazon Scraper
Playwright + Chromium
Alert Management
CRUD Operations
Parsed Data
Launch Browser
Price Updates
Infrastructure
Data Layer
Headless Chromium
User Agent Spoofing
Bot Protection Handling
MongoDB Database
Products & Alerts
HTML Response
HTTP Requests
External Services
Amazon Product Pages
Web Scraping Target

🛠️ Technology Stack

Frontend

- **React.js** - User interface framework
- **TailwindCSS** - Styling and responsive design
- **Axios** - HTTP client for API calls

Backend

- **Node.js** - Runtime environment
- **Express.js** - Web application framework
- **Playwright** - Web scraping and browser automation
- **Chromium** - Headless browser for scraping

Database

- **MongoDB** - NoSQL database for storing products and alerts
- **Mongoose** - MongoDB object modeling

DevOps & Tools

- **dotenv** - Environment variables management
- **CORS** - Cross-origin resource sharing
- **node-cron** - Task scheduling

📁 Project Structure

```
PricePulse/
├── backend/
|   ├── models/
|   |   ├── Product.js        # Product data model
|   |   └── Alert.js          # Alert data model
|   ├── routes/
|   |   ├── productRoutes.js    # Product API endpoints
|   |   └── alertRoutes.js      # Alert API endpoints
|   ├── scraper.js            # Amazon web scraper
|   ├── scheduler.js          # Background job scheduler
```

```
|   ├── server.js             # Main server file
|   └── package.json          # Backend dependencies
├── frontend/
|   ├── src/
|   |   ├── components/       # React components
|   |   ├── pages/            # Application pages
|   |   ├── services/         # API service functions
|   |   └── App.js            # Main React component
|   ├── public/               # Static assets
|   └── package.json          # Frontend dependencies
└── README.md
```

Web Scraping Process

The scraper handles Amazon's anti-bot measures:

```
// User agent spoofing
userAgent: "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36"


// Captcha detection
if (bodyText.includes("Enter the characters you see below")) {
  throw new Error("Blocked by Amazon captcha");
}
// Price extraction with multiple selectors
const priceElement =
  document.querySelector(".a-price .a-offscreen") ||
  document.querySelector("#priceblock_ourprice") ||
  document.querySelector("#priceblock_saleprice");
```

🔄 **Automated Monitoring**

The scheduler runs background jobs to:

1. **Fetch all tracked products** from database
2. **Scrape current prices** using Playwright
3. **Compare with target prices** set by users
4. **Update price history** in database

5. **Trigger alerts** when prices drop below targets
6. **Handle errors gracefully** and continue monitoring

Built with ❤️ by Parth Maheta

PricePulse - Never miss a great deal again!