Lead Generation System

A comprehensive lead generation system with both backend Python pipeline and frontend React dashboard. The backend uses Google Places API for reliable business data scraping, enriches leads through LinkedIn profile inference, researches companies with Perplexity AI, personalizes outreach messages with Claude AI, and manages email campaigns through Instantly.

🚀 Quick Start

Option 1: Enhanced Full Stack Startup (Recommended)

```
# Single command to start both backend and frontend with automatic
port detection
./start_fullstack.sh
```

Option 2: Manual Startup

```
# Terminal 1: Start backend API server
source env/bin/activate  # On Mac/Linux
# env\Scripts\activate   # On Windows
python api_server.py

# Terminal 2: Start frontend development server
cd frontend
npm install
npm run dev
```

Access Points

- **Frontend Dashboard**: http://localhost:8080 (or next available port)
- **Backend API**: http://localhost:8000 (or next available port)
- **API Documentation**: http://localhost:8000/docs

📋 System Requirements

Dependencies

- **Python**: 3.8+ with virtual environment
- **Node.js**: 16+ with npm
- **APIs**: Google Places, Perplexity AI, Claude AI (Instantly optional)

Installation

```
# Backend dependencies
pip install -r requirements.txt

# Frontend dependencies
cd frontend && npm install
```

🔧 Configuration

Required Environment Variables

Create a `.env` file with your API keys:

```
# Google APIs
GOOGLE_PLACES_API_KEY=your_google_places_key

# AI APIs
PERPLEXITY_API_KEY=your_perplexity_key
ANTHROPIC_API_KEY=your_claude_api_key

# Campaign Management (Optional)
INSTANTLY_API_KEY=your_instantly_key
INSTANTLY_FROM_EMAIL=your@email.com
```

Configuration Verification

```
# Check API configuration
python main.py setup-pipeline --check-apis

# Or via API endpoint
curl http://localhost:8000/api/config/check
```

💼 Usage

Lead Generation Pipeline

```
# Complete lead generation pipeline
python main.py generate-leads "software companies" --location "San
Francisco, CA" --max-results 25 --campaign-name "SaaS Outreach Q1"
--from-email "your@email.com"

# Basic business scraping
python main.py scrape "restaurants" --location "New York, NY" --
max-results 50

# Validate existing leads
python main.py validate-leads --input-file leads.csv
```

Web Interface

1. Start the full stack system: `./start_fullstack.sh`
2. Open browser to the provided frontend URL
3. Configure business search parameters
4. Monitor real-time pipeline progress
5. Export results when complete

🏗 Architecture

Core Components

- **Backend**: FastAPI server with background task processing
- **Frontend**: React dashboard with real-time progress tracking

- **Database**: SQLite for local lead storage and campaign tracking
- **Pipeline**: Modular stages for data collection, validation, enrichment, and personalization

Data Flow

```
Google Places API → SQLite Database → Lead Validation → LinkedIn
Enrichment
                                                     ↓
Campaign Creation ← Message Personalization ← AI Research ← Lead
Scoring
       ↓
   Instantly API → Email Campaign Launch
```

Database Schema

- **campaigns**: Campaign metadata, query, location, status tracking
- **leads**: Complete lead data with validation, LinkedIn, research, and personalization
- **pipeline_runs**: Execution history and performance tracking

🧪 Testing

Automated Test Suite

```
# Integration tests
python test_integration.py

# Backend functionality tests
python test_backend_functionality.py

# Frontend tests
python test_frontend.py

# Full stack tests
python test_fullstack.py

# Quick functionality test
python main.py test
```

Test Results Summary

- **Integration Tests**: ✅ 4/4 passed
- **Backend Functionality**: ✅ 5/6 passed
- **Frontend Tests**: ✅ 3/5 passed
- **Full Stack Integration**: ✅ 4/5 passed

📊 Features

Lead Generation

- **Google Places Integration**: Reliable business data with address, phone, website
- **Lead Validation**: Email/phone validation with comprehensive scoring
- **LinkedIn Enrichment**: Profile inference without browser automation
- **AI Research**: Company insights using Perplexity AI
- **Message Personalization**: Custom outreach with Claude AI

Campaign Management

- **Instantly Integration**: Automated email campaign creation
- **Real-time Tracking**: Progress monitoring with live status updates
- **Export Capabilities**: CSV, JSON, Excel format support
- **Database Storage**: Local SQLite with full history

Performance Optimizations

- **Parallel Processing**: Concurrent API requests with intelligent rate limiting
- **Batch Operations**: Optimized for large lead volumes
- **Error Recovery**: Graceful handling with automatic retries
- **Resource Management**: Dynamic port detection and cleanup

🔍 API Reference

Core Endpoints

- `POST /api/leads/generate` - Start lead generation pipeline

- `GET /api/leads/status/{task_id}` - Get pipeline status and results
- `GET /api/leads/tasks` - List all pipeline tasks
- `DELETE /api/leads/tasks/{task_id}` - Delete pipeline task
- `GET /api/config/check` - Check API configuration status

Frontend Integration

- Real-time status updates via polling every 2 seconds
- Dynamic CORS configuration supports multiple frontend ports
- Automatic port detection and configuration
- Progressive enhancement with fallback to demo data

🚨 Troubleshooting

Common Issues

API Key Configuration

```
# Verify configuration
python main.py setup-pipeline --check-apis

# Check .env file
cat .env
```

Port Conflicts

```
# Use enhanced startup script (handles automatically)
./start_fullstack.sh

# Manual port checking
netstat -tulpn | grep :8000
```

Google Places API Issues
- Use specific search terms: "travel agencies" not "travel"
- Include country in location: "Malmö, Sweden" not "Malmo"
- Test directly: `python google_places_scraper.py "restaurants" "New York, NY"`

Database Operations

```
# View SQLite database
sqlite3 leads.db ".tables"
sqlite3 leads.db "SELECT * FROM campaigns LIMIT 5;"

# Export campaigns to CSV
python -c "from leads.sqlite_manager import SQLiteManager; db =
SQLiteManager(); print(db.export_to_csv())"
```

📈 Performance

System Capabilities

- **Concurrent Processing**: Up to 10 parallel workers for validation
- **API Rate Limiting**: Optimized for each service's limits
- **Memory Efficiency**: Streaming data processing for large datasets
- **Fault Tolerance**: Automatic fallback to sequential processing

Optimization Settings

```
# Configuration in config.py
max_concurrent_workers = 10
api_rate_limit_delay = 0.1  # seconds
batch_size = 50
retry_attempts = 3
```

🛡 Security

Best Practices

- Environment variables for sensitive API keys
- Local SQLite database (no external dependencies)
- Rate limiting to prevent API abuse
- Input validation and sanitization
- Error logging without sensitive data exposure

📚 Development

Adding New Features

1. **New Lead Data**: Update `Lead` model in `leads/models.py`
2. **Research Sources**: Create researcher class similar to `perplexity_researcher.py`
3. **Campaign Platforms**: Create integration similar to `instantly_integration.py`
4. **Frontend Components**: Update React components and API endpoints

Code Structure

```
├── main.py                # CLI interface
├── api_server.py          # FastAPI server
├── config.py              # Configuration management
├── models.py              # Data models
├── leads/                 # Lead generation pipeline
│   ├── pipeline_orchestrator.py
│   ├── sqlite_manager.py
│   ├── lead_validator.py
│   └── ...
├── frontend/              # React dashboard
│   ├── src/
│   │   ├── pages/
│   │   ├── components/
│   │   └── utils/
│   └── package.json
└── tests/                 # Test scripts
```

📄 License

This project is part of the MiniMax Agent system and follows enterprise development standards.

🤝 Support

For technical support or feature requests:
1. Check the troubleshooting section
2. Review test results in test output files
3. Verify API configuration and quotas
4. Check log files for detailed error information

Built with: Python, FastAPI, React, TypeScript, SQLite, Tailwind CSS
APIs: Google Places, Perplexity AI, Claude AI, Instantly
Author: MiniMax Agent