Product Requirements & Specification Document

Project Name

CivicSync - Public Service Request Optimizer

Description

CivicSync is a government web platform leveraging agentic-AI to analyze, prioritize, and route citizen service requests. The system predicts resource needs, optimizes response times, and generates actionable reports for city officials. Features include secure authentication, real-time dashboards, and collaborative tools for public sector teams.

---

1. Goals & Objectives

Goal	Objective
Efficient Request Management	Automate analysis, prioritization, and routing of service requests
Resource Optimization	Predict and allocate resources for optimal response
Actionable Insights	Provide real-time dashboards and reports for decision-making
Secure Collaboration	Enable authenticated, role-based access and team collaboration

---

2. Core Features

Feature	Description
AI-Powered Request Analysis	Agentic-AI analyzes and categorizes incoming requests
Prioritization & Routing	AI prioritizes and routes requests to appropriate departments/teams
Resource Prediction	System predicts required resources and estimated response times
Real-Time Dashboards	Live dashboards display request status, workload, and KPIs
Actionable Reporting	Generate and export reports for city officials
Secure Authentication	OAuth2-based authentication with role-based access control
Team Collaboration Tools	Internal chat, notes, and assignment features for public sector teams

---

3. User Roles & Permissions

Role	Permissions
Admin	Full access: manage users, settings, reports, and all requests
City Official	View dashboards, generate reports, oversee request status

Team Lead	Assign/monitor requests, manage team collaboration
Team Member	View/resolve assigned requests, participate in collaboration
Citizen (External)	Submit/view own requests, receive status notifications

---

4. Functional Requirements

4.1 Request Lifecycle

```
Citizen submits request -> AI analyzes & categorizes -> System prioritizes & rou
```

4.2 AI Analysis & Routing

- Use agentic-AI to:
  - Classify request type/severity
  - Predict required resources
  - Suggest optimal routing based on workload and location

4.3 Dashboards & Reporting

- Real-time visualization of:
  - Open/closed requests
  - Response times
  - Resource allocation
- Exportable reports (PDF/CSV)

4.4 Collaboration Tools

- Internal chat per request
- Notes and attachments
- Assignment and status updates

4.5 Authentication & Security

- OAuth2 login (city SSO integration)
- Role-based access control
- Audit logs for all actions

---

5. Non-Functional Requirements

Requirement	Specification
Performance	<2s response time for dashboard and request ops
Scalability	Support 10,000+ concurrent users
Security	Encrypted data at rest and in transit
Availability	99.9% uptime

Compliance	GDPR, local data protection laws

6. Technical Specifications

Component	Technology	Notes
Frontend	Vue.js	SPA, responsive, real-time updates
Backend	Python (FastAPI)	REST API, AI integration
AI Engine	Agentic-AI	Request analysis, prioritization
Database	PostgreSQL	Relational data storage
Containerization	Docker	All services containerized
Authentication	OAuth2	SSO integration
Deployment	Docker Compose/K8s	Scalable, cloud-ready

7. API Overview

```
POST   /api/requests           # Submit new request
GET    /api/requests           # List/filter requests
PATCH  /api/requests/{id}      # Update request status/assignment
GET    /api/dashboard          # Dashboard data
GET    /api/reports            # Generate/export reports
POST   /api/auth/login         # User authentication
```

8. Milestones & Timeline

Milestone	Target Date
Requirements Finalized	Week 1
MVP Prototype	Week 4
AI Integration Complete	Week 6
Dashboard & Reporting	Week 8
Security Audit	Week 9
Beta Release	Week 10

9. Risks & Mitigations

Risk	Mitigation
AI misclassification	Human-in-the-loop review, continuous training

Data privacy breaches	Strong encryption, regular audits
Scalability bottlenecks	Load testing, container orchestration

---

10. Success Metrics

- 90%+ automated request routing accuracy
- 30% reduction in average response time
- 99.9% system uptime
- Positive user feedback from city officials and teams

End of Document