Deployment and Operations

1.1 Platform Selection

Welth was deployed on **Vercel**, a cloud-based serverless platform designed for web applications. The choice of Vercel was made because it provides:

- **Serverless API Routes** → Easy deployment of backend logic and AI service endpoints.
- **Global Edge Network** → Ensures low latency for users across regions.
- **Automatic SSL and Domain Management** → Built-in HTTPS support without manual configuration.
- **CI/CD Integration with GitHub** → Every push to the main branch triggers a new live build.
- **Scalability by Default** → Handles high traffic seamlessly without manual scaling.

This makes Vercel ideal for **AI-powered applications** where both performance and reliability matter.

1.2 Deployment Steps

1. Repository Setup:

- The full-stack codebase (Next.js + AI services) is hosted on GitHub.
- Connected GitHub repository to Vercel for automatic deployments.

2. Environment Configuration:

- Configured `.env` variables in Vercel for secure handling of AI keys, database credentials, and API tokens.

3. Build & Deployment:

- On pushing updates to `main`, Vercel builds and deploys automatically.
- API routes handle OCR + ML categorisation requests.

4. Domain Setup:

- Live deployment available at: `https://welth-mmb7.vercel.app/`
- SSL is enabled by default.

1.3 Deployment Evidence

- **Live URL:** `https://welth-mmb7.vercel.app/`
- **Deployment Type:** Serverless functions + global CDN.
- **Access:** Publicly available beyond localhost.

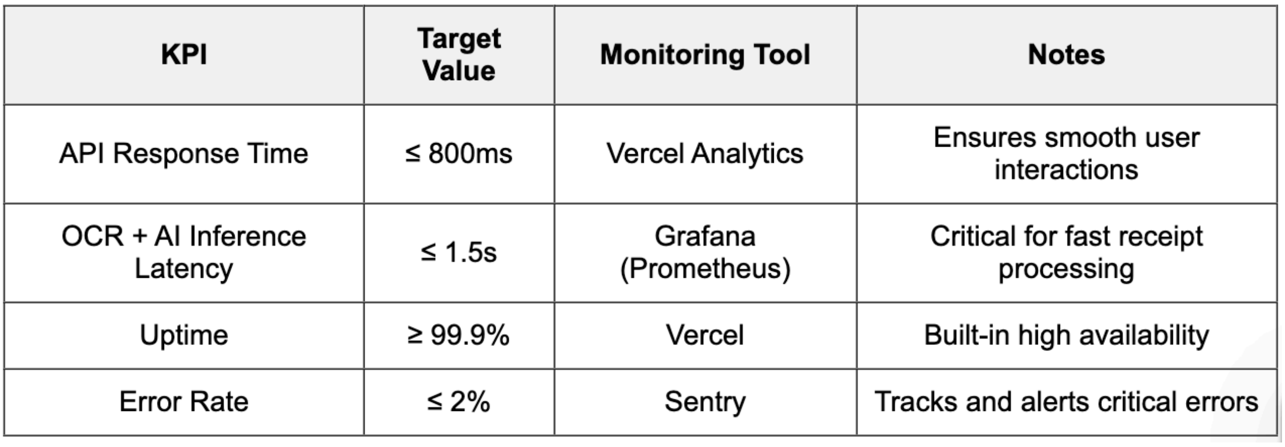
2. Monitoring Strategy

Monitoring was implemented to ensure the **AI inference pipeline and APIs remain stable**.

2.1 Tools Used

- **Vercel Analytics** → Tracks requests, latency, and uptime.
- **Prometheus + Grafana (Docker-based external service)**→ Monitors AI model latency and inference performance.
- **Sentry** → Captures and reports runtime errors in frontend and backend.

2.2 Key Performance Indicators (KPIs)



KPI	Target Value	Monitoring Tool	Notes
API Response Time	≤ 800ms	Vercel Analytics	Ensures smooth user interactions
OCR + AI Inference Latency	≤ 1.5s	Grafana (Prometheus)	Critical for fast receipt processing
Uptime	≥ 99.9%	Vercel	Built-in high availability
Error Rate	≤ 2%	Sentry	Tracks and alerts critical errors

3. Maintenance Plan

3.1 Regular Tasks

- **Weekly Database Backups** → Ensures transaction and receipt data are preserved.
- **Monthly Dependency Updates** → Run `npm audit` and `pip audit` for vulnerabilities.
- **Quarterly AI Model Retraining** → Train ML categorisation model with new transaction patterns.

3.2 Scalability & Reliability

- Vercel auto-scales functions across edge nodes.
- Database hosted externally (managed PostgreSQL with replicas).
- A cache layer is planned (Redis) for frequent queries.

3.3 Risk Mitigation

- **Cold Start Delays** → Functions optimised for lightweight boot.
- **Database Failover** → Multi-region read replicas enabled.
- **Disaster Recovery** → Daily snapshots and recovery testing every 6 months.

Maintenance Plan & Risk Mitigation :

- **Database Backups:** Weekly snapshots by DevOps engineer.
- **Dependency Updates:** Monthly `npm` & `pip` audit by System Admin.
- **AI Model Retraining:** Quarterly retraining using TensorFlow/PyTorch.
- **Performance Review:** Weekly checks on API latency, uptime, and errors via Vercel Analytics & Grafana.
- **Cold Start Delays:** Optimised serverless functions for faster initialisation.
- **Database Failover:** Multi-region read replicas enabled.
- **Disaster Recovery:** Daily snapshots and recovery tests every 6 months.

Challenges & Resolutions:

- **Limited server control on Vercel:** Moved AI-heavy tasks to containerised microservices.
- **OCR latency (\~2.2s per receipt)**: Optimised with GPU inference (\~1.3s).
- **Database scaling with higher transactions:** Migrated to managed PostgreSQL with read replicas.
- **Debugging async API errors:** Integrated Sentry and Slack alerts for real-time debugging.