Monitoring and Logging Recommended Plan

Logging Strategy

While ACT does not currently include a formal logging infrastructure, future implementation of application-level logging is recommended for **troubleshooting**, **security auditing**, and **usage analysis**. Below is a proposed strategy for incorporating logging capabilities into the system:

Logging Levels

Use a standardized logging library (e.g., Python's logging module, Node's winston, or pino) to support:

- **INFO** – Normal operations (e.g., user logged in, translation requested)
- **WARN** – Suspicious behavior (e.g., repeated failed login attempts)
- **ERROR** – Failures in translation, database connections, etc.
- **DEBUG** – Used during development for detailed tracebacks

Log Storage Options

- **Development/Local**: Log to rotating .log files (e.g., logs/api.log)
- **Production**: Recommend forwarding logs to a centralized system such as:
    - **ELK Stack** (Elasticsearch + Logstash + Kibana)
    - **Datadog** or **Grafana Loki**
    - **Sentry** (for error and frontend logging)
- Container logs (e.g., via docker logs) can also be aggregated with **Fluentd** or **Filebeat**.

Security Consideration

Logs must **never include sensitive data** (e.g., tokens, full Delphi code snippets, or personal user info). Use **redaction and masking** policies before logging input.

Alerts & Monitoring

Although no automated monitoring is currently in place, the following tools and approaches are suggested for future expansion:

🧠 *What to Monitor*

Component	Metric or Behavior
Translation API	Request latency, error rate (e.g., 500s, timeouts)
Authentik (OIDC)	Login attempts, failure rates
CouchDB / PostgreSQL	Availability, storage usage, write failures
Frontend UI (optional)	JavaScript errors, navigation drop-offs

🔭 *Tools for Implementation*

- **Prometheus + Grafana** for metrics and dashboards
- **UptimeRobot or StatusCake** for external uptime monitoring
- **Sentry** for frontend and backend exception tracking
- **Docker Healthchecks** for container-level monitoring

⚠️ *Alerting Strategy*

Integrate alerting via:

- **Slack or Microsoft Teams** for dev notifications
- **Email alerts** for critical failures (e.g., backend crashes, Authentik downtime)
- **PagerDuty / Opsgenie** for scalable enterprise alerting (if hosted on internal servers)

🚨 *Suggested Thresholds*

Metric	Threshold	Action
Translation API latency	> 2 seconds average over 1 minute	Notify dev team
Authentik failures	> 3 failed logins per user/minute	Trigger security alert
CouchDB availability	No response for 5 consecutive pings	Auto-restart or alert

Future Steps

1. Add backend logging using winston (Node.js) and logging (Python) with log rotation.
2. Configure Docker log forwarding to external tool (e.g., Logstash).
3. Create a Grafana dashboard linked to Prometheus metrics (e.g., request rate, error rate).
4. Set up uptime monitoring for ports 3000 and 9000.