Appendices

1. API Reference

This section provides an overview of the key API endpoints and their usage in the RAG system.

Common Endpoints:

- `POST /api/documents/upload` — Upload a new document
- `GET /api/documents` — List all documents (with optional filters)
- `GET /api/documents/{id}` — Retrieve a specific document
- `POST /api/rag/query` — Submit a query to the RAG system and receive a response
- `GET /api/projects` — List all projects
- `POST /api/projects` — Create a new project

Authentication:

- Most endpoints require user authentication (token-based or session-based)

Refer to your project's route files (`routes/api.php`, `routes/web.php`) and controller classes for detailed request/response formats.

---

2. Configuration Guide

Environment Variables:

- `APP_ENV`, `APP_DEBUG`, `APP_KEY` — Laravel environment settings
- `DB_CONNECTION`, `DB_HOST`, `DB_DATABASE`, `DB_USERNAME`, `DB_PASSWORD` — Database configuration
- `OPENAI_API_KEY` — OpenAI API key for embeddings and completions
- `ELASTICSEARCH_HOST` — Elasticsearch connection URL
- `CACHE_DRIVER`, `SESSION_DRIVER` — Caching and session management

Config Files:

- `config/database.php` — Database connections
- `config/filesystems.php` — File storage
- `config/cache.php` — Caching
- `config/services.php` — Third-party services (OpenAI, Elasticsearch)

---

3. Troubleshooting Guide

Common Issues:

- **Elasticsearch Not Running:** Ensure Docker is running and the Elasticsearch container is healthy.
- **OpenAI API Errors:** Check your API key and network connectivity.
- **File Upload Fails:** Verify file size limits and storage permissions.
- **No Search Results:** Check that embeddings are generated and indexed correctly.

- **Performance Issues:** Monitor server resources and Elasticsearch health.

Logs:

- Application logs: `storage/logs/laravel.log`
- Elasticsearch logs: Docker container logs or `/var/log/elasticsearch/`

Debugging Tips:

- Use Laravel's built-in debugging tools (`php artisan tinker`, `php artisan log:clear`)
- Check browser console for frontend errors

---

4. Best Practices

- Use environment variables for all sensitive configuration.
- Regularly back up your database and Elasticsearch indices.
- Keep dependencies up to date and monitor for security advisories.
- Use HTTPS in production.
- Limit API and document access based on user roles and projects.
- Monitor logs and set up alerts for critical errors.
- Test your deployment and rollback procedures.

---

5. Additional Resources

- Laravel Documentation
- Elasticsearch Documentation
- OpenAI API Documentation
- Docker Documentation
- PHP Official Site
- Tailwind CSS Documentation
- PhpWord Documentation