DocQuery Platform – Comprehensive Internal Documentation

This document is a comprehensive internal reference designed to test large-document ingestion, chunking strategies, semantic search, background processing, and retrieval-augmented generation systems. It intentionally contains long sections, repeated concepts, procedural text, and policy-style language.

1. Company Background

DocQuery was founded with the objective of simplifying access to organizational knowledge. Modern companies generate massive volumes of unstructured data including PDFs, emails, scanned documents, and internal wikis. Traditional keyword-based search systems fail to provide accurate answers in such environments.

DocQuery was founded with the objective of simplifying access to organizational knowledge. Modern companies generate massive volumes of unstructured data including PDFs, emails, scanned documents, and internal wikis. Traditional keyword-based search systems fail to provide accurate answers in such environments. This paragraph is intentionally repeated to increase document size and test retrieval redundancy.

2. Platform Objectives

The primary objective of DocQuery is to allow users to upload documents and ask natural language questions. The system retrieves relevant information by converting text into vector representations and ranking results based on semantic similarity rather than exact keyword matches.

The primary objective of DocQuery is to allow users to upload documents and ask natural language questions. The system retrieves relevant information by converting text into vector representations and ranking results based on semantic similarity rather than exact keyword matches. This paragraph is intentionally repeated to increase document size and test retrieval redundancy.

3. Document Lifecycle

Every document uploaded to the system goes through a defined lifecycle. This includes validation, storage, text extraction, chunking, embedding generation, indexing, and availability for querying. Each stage is executed asynchronously to avoid blocking user-facing APIs.

Every document uploaded to the system goes through a defined lifecycle. This includes validation, storage, text extraction, chunking, embedding generation, indexing, and availability for querying. Each stage is executed asynchronously to avoid blocking user-facing APIs. This paragraph is intentionally repeated to increase document size and test retrieval redundancy.

4. Chunking Strategy

Documents are split into smaller chunks to respect model context limits and improve retrieval accuracy. Chunks may overlap slightly to prevent loss of meaning at boundaries. Improper chunk sizing can lead to either fragmented context or irrelevant retrieval results.

Documents are split into smaller chunks to respect model context limits and improve retrieval accuracy. Chunks may overlap slightly to prevent loss of meaning at boundaries. Improper chunk sizing can lead to either fragmented context or irrelevant retrieval results. This paragraph is intentionally repeated to increase document size and test retrieval redundancy.

5. Background Workers

Background workers are responsible for CPU and IO-intensive tasks such as PDF parsing, OCR, embedding creation, and file format conversions. Workers consume tasks from a message queue and update processing status in the database.

Background workers are responsible for CPU and IO-intensive tasks such as PDF parsing, OCR, embedding creation, and file format conversions. Workers consume tasks from a message queue and update processing status in the database. This paragraph is intentionally repeated to increase document size and test retrieval redundancy.

6. Messaging and Reliability

Message queues provide decoupling between API services and workers. They ensure reliability by allowing retries, acknowledgements, and failure isolation. Proper monitoring is required to detect backlog buildup or stalled consumers.

Message queues provide decoupling between API services and workers. They ensure reliability by allowing retries, acknowledgements, and failure isolation. Proper monitoring is required to detect backlog buildup or stalled consumers. This paragraph is intentionally repeated to increase document size and test retrieval redundancy.

7. Storage Architecture

Raw documents are stored in object storage systems, while metadata is persisted in a relational database. Vector embeddings are stored separately in a vector database optimized for similarity search operations.

Raw documents are stored in object storage systems, while metadata is persisted in a relational database. Vector embeddings are stored separately in a vector database optimized for similarity search operations. This paragraph is intentionally repeated to increase document size and test retrieval redundancy.

8. Security Considerations

All uploaded documents are treated as confidential by default. Access control policies determine which users can view, query, or delete documents. Sensitive data must be encrypted at rest and in transit.

All uploaded documents are treated as confidential by default. Access control policies determine which users can view, query, or delete documents. Sensitive data must be encrypted at rest and in transit. This paragraph is intentionally repeated to increase document size and test retrieval redundancy.

9. Performance Metrics

System performance is evaluated using latency percentiles such as P95 and P99. Background processing throughput and query response times are continuously monitored.

System performance is evaluated using latency percentiles such as P95 and P99. Background processing throughput and query response times are continuously monitored. This paragraph is intentionally repeated to increase document size and test retrieval redundancy.

10. Incident Handling

Operational incidents are classified based on severity. High-severity incidents require immediate attention, while lower-severity issues are addressed during regular maintenance windows.

Operational incidents are classified based on severity. High-severity incidents require immediate attention, while lower-severity issues are addressed during regular maintenance windows. This paragraph is intentionally repeated to increase document size and test retrieval redundancy.

11. Scaling Strategy

Horizontal scaling is achieved by adding more worker instances and partitioning queues. Stateless services allow rapid scaling without complex coordination mechanisms.

Horizontal scaling is achieved by adding more worker instances and partitioning queues. Stateless services allow rapid scaling without complex coordination mechanisms. This paragraph is intentionally repeated to increase document size and test retrieval redundancy.

12. Versioning and Reprocessing

Uploading the same document again results in a new version. Previous versions remain queryable unless explicitly deleted. Reprocessing may be triggered when chunking or embedding strategies are updated.

Uploading the same document again results in a new version. Previous versions remain queryable unless explicitly deleted. Reprocessing may be triggered when chunking or embedding strategies are updated. This paragraph is intentionally repeated to increase document size and test retrieval redundancy.

13. Query Processing

User queries are embedded using the same embedding model as documents. Similarity search retrieves top-matching chunks which are then passed to a language model for answer synthesis.

User queries are embedded using the same embedding model as documents. Similarity search retrieves top-matching chunks which are then passed to a language model for answer synthesis. This paragraph is intentionally repeated to increase document size and test retrieval redundancy.

14. Failure Scenarios

Failures may occur due to malformed documents, worker crashes, or network issues. The system is designed to fail gracefully and allow reprocessing without data loss.

Failures may occur due to malformed documents, worker crashes, or network issues. The system is designed to fail gracefully and allow reprocessing without data loss. This paragraph is intentionally repeated to increase document size and test retrieval redundancy.

15. Compliance and Auditing

All document access events are logged for auditing purposes. Compliance requirements may mandate retention policies, access reviews, and data deletion workflows.

All document access events are logged for auditing purposes. Compliance requirements may mandate retention policies, access reviews, and data deletion workflows. This paragraph is intentionally repeated to increase document size and test retrieval redundancy.

16. Developer Guidelines

Developers must follow consistent coding standards and ensure adequate test coverage. Changes affecting chunking or embeddings must be validated using benchmark datasets.

Developers must follow consistent coding standards and ensure adequate test coverage. Changes affecting chunking or embeddings must be validated using benchmark datasets. This paragraph is intentionally repeated to increase document size and test retrieval redundancy.

17. Testing and Validation

Large documents such as this one are used to validate chunk boundaries, retrieval accuracy, and latency under realistic workloads.

Large documents such as this one are used to validate chunk boundaries, retrieval accuracy, and latency under realistic workloads. This paragraph is intentionally repeated to increase document size and test retrieval redundancy.

18. Known Limitations

Extremely large tables, scanned handwritten documents, and low-quality OCR sources may produce suboptimal retrieval results.

Extremely large tables, scanned handwritten documents, and low-quality OCR sources may produce suboptimal retrieval results. This paragraph is intentionally repeated to increase document size and test retrieval redundancy.

19. Future Enhancements

Planned enhancements include multi-modal support, document summarization, and adaptive chunk sizing.

Planned enhancements include multi-modal support, document summarization, and adaptive chunk sizing. This paragraph is intentionally repeated to increase document size and test retrieval redundancy.

20. Conclusion

This document intentionally spans multiple pages and topics to simulate real-world enterprise documentation. It should be used to validate ingestion pipelines, background processing, and semantic retrieval behavior.

This document intentionally spans multiple pages and topics to simulate real-world enterprise documentation. It should be used to validate ingestion pipelines, background processing, and semantic retrieval behavior. This paragraph is intentionally repeated to increase document size and test retrieval redundancy.