# FinSolve Technologies

CODE BASICS

*AI-Powered Internal Chatbot with Role-Based Access Control*

# Agenda

- Problem Overview
- Our AI Solution
- Technical Architecture
- Key Features & Innovation
- Business Impact
- Live Demonstration

# The Challenge We Faced

### Communication Delays

Teams struggling with slow information flow

### Data Silos

Departments are isolated from relevant information

### Decision Bottlenecks

Strategic planning hindered by access barriers

### Inefficient Processes

Manual search through multiple systems

# Our AI-Powered Solution

## RAG-Based Chatbot with Role-Based Access Control

An intelligent assistant that provides instant, secure, role-specific access to company information while maintaining data security and compliance.

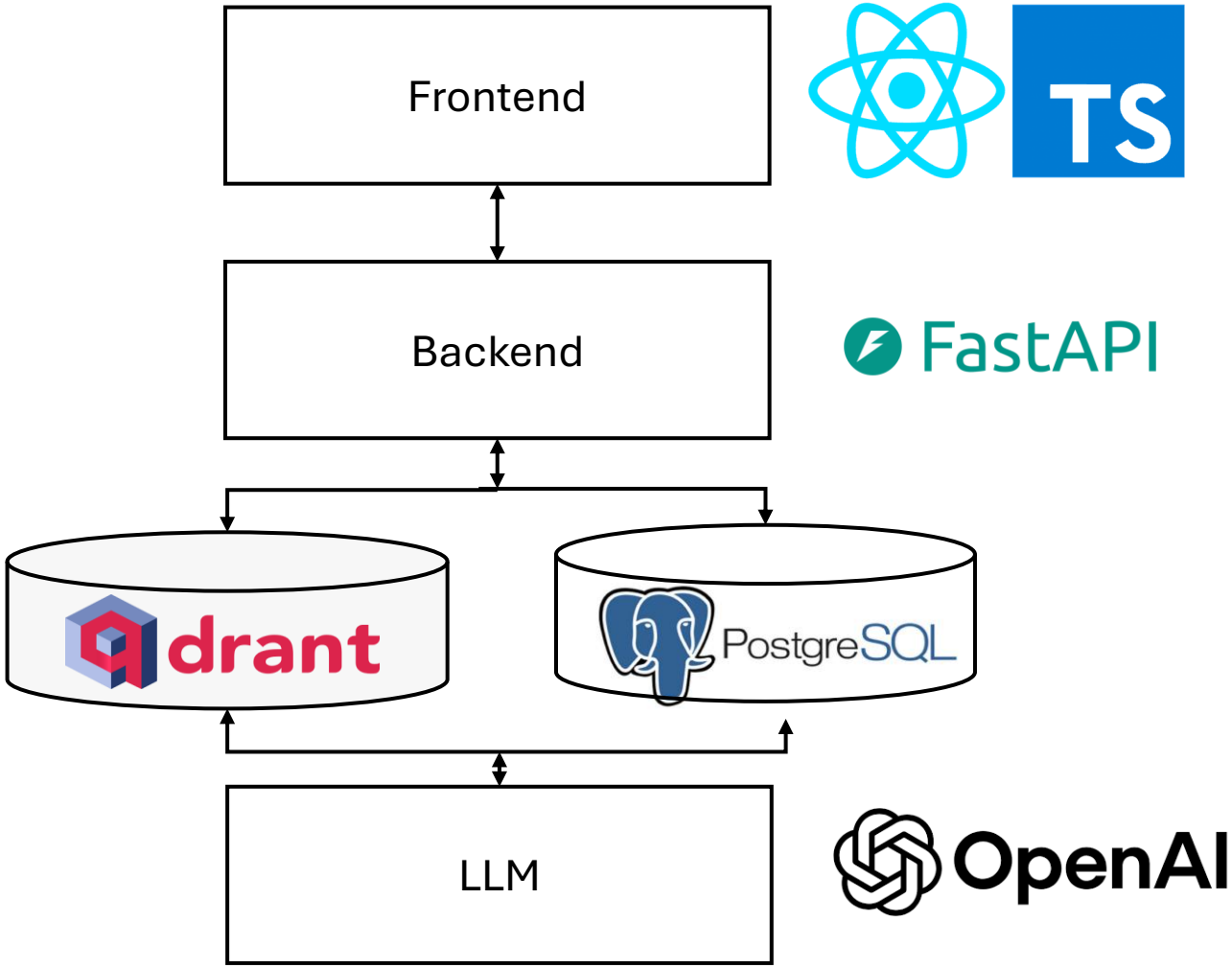
**Key Benefits:**

- Instant access to relevant information
- Secure role-based data protection
- Natural language query processing
- Contextual, accurate responses

**Business Impact:**

- Reduced communication delays
- Faster decision-making
- Improved productivity
- Enhanced data security

# System Architecture




TS

FastAPI

OpenAI

# Technology Stack

- React + TypeScript
- FastAPI
- OpenAI GPT-4
- Qdrant Vector DB
- Sentence Transformers
- PostgreSQL
- Docker
- Python

# Role-Based Access Control

**Finance Team**

Financial reports, expenses, reimbursements

**Marketing Team**

Campaign data, customer feedback, sales metrics

**HR Team**

Employee data, attendance, payroll, reviews

**Engineering**

Technical docs, architecture, processes

**C-Level**

Full access to all company data

**General Employees**

Policies, events, FAQs

# Advanced Features

**User Experience**

- Intuitive chat interface
- Real-time responses
- Source document references
- Mobile-responsive design

**Security**

- HTTP Basic Authentication
- Role-based data filtering
- Secure document handling

**Admin Dashboard**

- User management system
- Document indexing control
- System statistics
- Vector store management

**Scalability**

- Modular architecture
- Docker containerization
- Extensible role system

# RAG Pipeline Process

**Document Ingestion**

Markdown and CSV files are processed, chunked, and embedded using Sentence Transformers

**Query Processing**

User queries are embedded and matched against vector database with role-based filtering

**Context Retrieval**

Relevant document chunks are retrieved based on semantic similarity and user permissions

**Response Generation**

GPT-4 generates contextual responses with source document references

# Document Loading Architecture

- Directory Structure

```
resources/data/
├── engineering/
│   └── *.md files
├── finance/
│   └── *.md, *.csv files
├── hr/
│   └── *.md, *.csv files
├── marketing/
│   └── *.md, *.csv files
└── general/
    └── *.md files
```

Processing Pipeline

1. Directory Traversal
2. File Type Detection (.md or .csv)
3. Content Processing
4. Vector Generation

# Intelligent Document Chunking

## Markdown Processing

- Header-Based Sectioning
  - Extract hierarchical structure
  - Preserve section context
  - Maintain heading levels
- Overlapping Chunks
  - Word-based chunking
  - Semantic continuity
  - Configurable chunk size

## CSV Processing

- Row-Based Chunking
  - Each row as structured data
  - String representation for search
  - Preserve original structure
- Metadata Enrichment
  - Column headers preserved
  - Row context maintained
  - Searchable field mapping

# Embedding Generation & Vector Storage

Sentence Transformers Model: all-MiniLM-L6-v2

Rich Metadata Payload

- **role:** Department/access level
- **source:** Original filename
- **section_title:** Document section
- **chunk_index:** Position in document
- **word_count:** Chunk size metrics
- **row_data:** Structured CSV data

Qdrant Configuration

- **Vector Size:** 384 dimensions
- **Distance Metric:** Cosine similarity
- **Storage:** On-disk for scalability
- **Batch Size:** 100 documents/batch
- **Indexing:** Role-based organization

# Prompt Construction

**Parameters:**

- **query:** The user's question
- **docs:** Document chunks from vector store
- **user_role:** User's role (engineering, finance, etc.)
- **use_enhanced_prompt:** Role-aware prompt toggle
- **temperature, max_tokens, max_chunks:** API controls

**Enhanced Prompt:**

- Role-specific instructions
- Context guidelines
- Metadata formatting
- Section titles & sources

**Simple Prompt**

- Basic context format
- Question-answer structure
- Minimal metadata
- Faster processing

# OpenAI API Call

**Key Configuration Details:**

- Model: GPT-4
  - Latest generation model for high-quality responses
- Temperature: 0.2 (Default)
  - Low randomness for consistent, deterministic answers
- Max Tokens: Optional Limit
  - Controls response length when specified

# Innovation Highlights

### Smart Role Filtering

Dynamic document access based on user roles at query time

### Modular Design

Separation of concerns: API, vector store, UI, and access control

### Source Transparency

Every response includes document sources for verification

### Admin Analytics

Comprehensive system stats and management tools

# Expected Business Impact

**Efficiency Gains**

- 80% reduction in information search time
- Instant access to relevant documents
- Streamlined decision-making process
- Reduced cross-department delays

**Competitive Advantages**

- Enhanced data security compliance
- Improved employee productivity
- Faster strategic planning cycles
- Scalable AI infrastructure

# Scalability & Future Enhancements

### Scalability Features

- Modular architecture
- Docker deployment
- Extensible role system
- Batch document processing

### Future Enhancements

- OAuth/JWT authentication
- Advanced analytics dashboard
- Multi-language support
- Integration with existing systems

### Expansion Possibilities

- Additional document formats
- Real-time document updates
- Voice interface integration
- Custom role hierarchies

# Live Demonstration

Let's explore the chatbot in action!

# Thank you

GitHub: [prateeksharma1809/ds-rpc-01]