Smartrentals - Technical Architecture Report

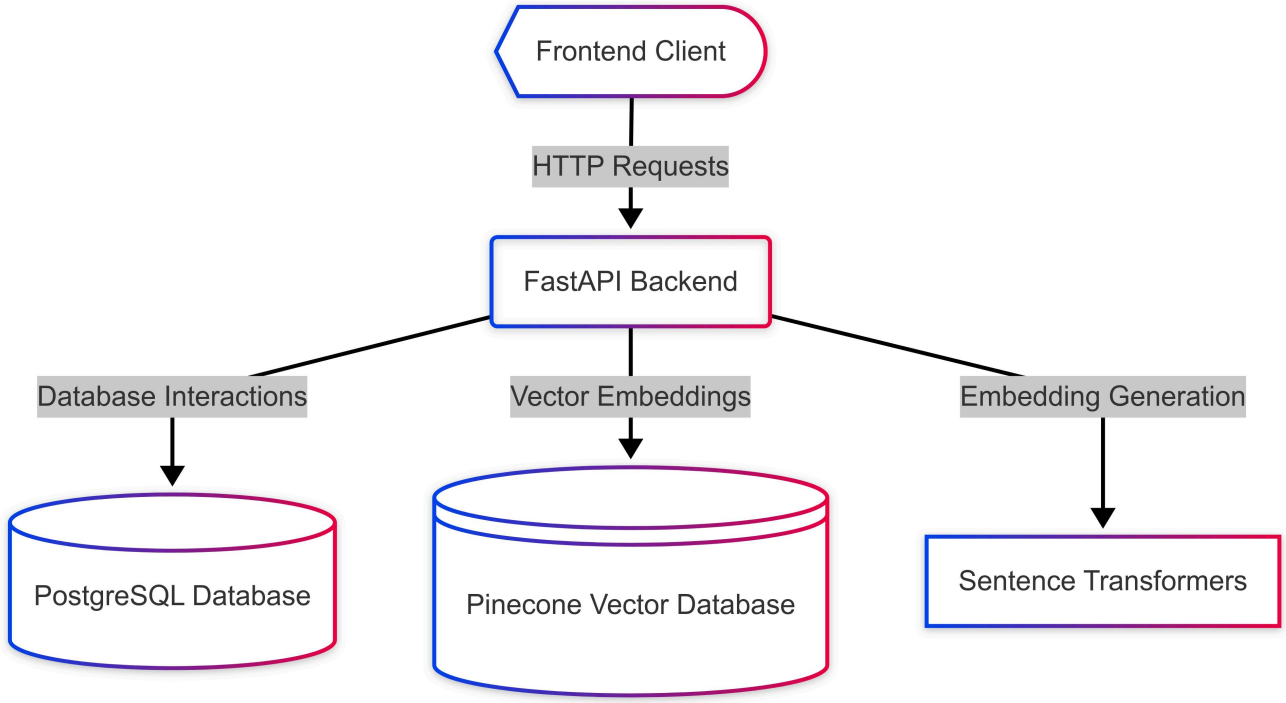
Author : Shriniwas Kulkarni

Passionate Python and Java Developer, PCCOE 2026 CSE(AIML)

Overview

This is a sophisticated property recommendation system built using FastAPI, leveraging advanced technologies for secure, scalable, and intelligent property matching.

System Architecture



Key Technical Components

1. Backend Framework

- **FastAPI**: High-performance, modern Python web framework
- Enables rapid API development with automatic OpenAPI (Swagger) documentation
- Implements robust request validation using Pydantic models

2. Authentication Mechanism

- **Token-Based Authentication**
  - Custom token generation using Fernet encryption

    - 10-minute session validity
    - Secure token validation process
- Prevents multiple simultaneous logins
- Cryptographically secure token generation

3. Database Integration

- **PostgreSQL**: Relational database for user management
- Stores user credentials and transaction logs
- Tables:
    - `credentials` : User registration details
    - `transactions` : User activity tracking

4. Vector-Based Recommendation System

- **Pinecone Vector Database**: Enables semantic property search
- **Sentence Transformers**: Converts property addresses to high-dimensional embeddings
- Advanced recommendation algorithm using cosine similarity
- Supports intelligent, context-aware property recommendations

5. Security Features

- **CORS Middleware**: Configurable cross-origin resource sharing
- Environment variable management with `python-dotenv`
- Parameterized database queries to prevent SQL injection
- Cryptographic token generation

Technical Highlights

Advanced Search Capability

```
def get_recommendations(pinecone_index, search_term, top_k=10):
    embed = get_embeddings([search_term])
    res = pinecone_index.query(vector=embed, top_k=top_k, include_metadata=True)
    return res
```

- Converts search terms into vector embeddings
- Retrieves semantically similar properties
- Supports flexible, intelligent search

Robust Data Validation

```
class Property(BaseModel):
    PropertyTypes: Literal['1 Bedroom', '2 Bedroom', ...]
    Security: Literal['Not Applicable', 'Gated Community', ...]
    # ... other strictly typed fields
```

- Uses Pydantic for type enforcement
- Ensures data integrity
- Supports predefined value sets for specific fields

Technology Stack

- **Web Framework**: FastAPI
- **Database**: PostgreSQL
- **Vector DB**: Pinecone
- **ML Model**: Sentence Transformers
- **Encryption**: Cryptography (Fernet)
- **ORM/Database Driver**: Psycopg2

Scalability and Performance Considerations

- Serverless Pinecone vector index
- Efficient embedding generation
- Stateless API design
- Minimal external dependencies

Potential Improvements

- Implement more advanced authentication (e.g., JWT)
- Add rate limiting
- Enhance error handling
- Implement more sophisticated recommendation algorithms

Conclusion

A modern, scalable property recommendation system demonstrating expertise in:

- API design
- Machine learning integration
- Database management
- Security implementation