2. File Structure Explanation

The directory structure is designed for scalability and organization:

bash

Copy code

```
compliance-checker/
 ├── src/
 |   ├── models/       # Define database models and schemas
 |   ├── controllers/  # Handle business logic and API routing
 |   ├── utils/        # Utility functions for common operations
 |   ├── templates/    # Frontend templates for rendering
 |   ├── config/       # Configuration files (e.g., settings.py)
 ├── tests/            # Unit and integration tests
 ├── data/             # Dataset files for the application
 ├── docs/             # Documentation for the project
 ├── .env              # Environment variables (e.g., API keys)
 ├── README.md         # Project overview and instructions
 ├── requirements.txt  # List of dependencies
 └── main.py           # Entry point of the application
```

Description of Key Components:

1. **src/models/**: Contains database models, schemas, and ORM definitions (e.g., SQLAlchemy).
2. **src/controllers/**: Includes API route handlers, connecting user requests to the business logic.
3. **src/utils/**: Helper scripts and utility functions for repetitive tasks.
4. **src/templates/**: Stores HTML templates for rendering frontend pages (e.g., with Flask's Jinja2).
5. **src/config/**: Centralized configuration settings (e.g., database credentials, API keys).
6. **tests/**: Unit tests for each component to ensure reliability and correctness.
7. **data/**: Stores input datasets, processed files, or exported results.
8. **docs/**: Technical and user documentation.
9. **.env**: Stores sensitive credentials and environment-specific variables (e.g., OpenAI keys).
10. **README.md**: Contains a project summary, setup instructions, and usage guidelines.
11. **requirements.txt**: Lists all dependencies for easy installation.

12. **main.py**: Entry point to initialize the application and run the server.