Here's a **starter README template** for your FastAPI project. I've made it clean, extensible, and included placeholders for guides, changelog, and future expansion 👇

FastAPI Project 🚀

A modular and scalable **FastAPI** application following a clean architecture with clear separation of concerns:

- **API Layer** (`app/api`)
- **Core Utilities** (`app/core`)
- **CRUD Layer** (`app/crud`)
- **Database Models & Sessions** (`app/db`)
- **Schemas** (`app/schemas`)
- **Services / Business Logic** (`app/services`)
- **Tests** (`tests`)

📂 Project Structure

```
fastapi_project/
│
├── app/
│   ├── api/                # API routes (organized by version/endpoints)
│   ├── core/               # Configs, security, middleware, roles
│   ├── crud/               # Database CRUD operations
│   ├── db/                 # Database models and sessions
│   ├── schemas/            # Pydantic models for request/response
│   ├── services/           # Business logic
│   ├── main.py             # FastAPI app entrypoint
│   └── logger.py           # Centralized logging
│
├── tests/                  # Unit and integration tests
├── requirements.txt        # Project dependencies
├── README.md               # Project documentation
└── .env                    # Environment variables (excluded from VCS)
```

⚡ Features

- JWT Authentication (`/api/login`, `/api/logout`)
- User Management (`/api/users`) with Role Assignment
- Product Management with Category Assignment
- Category & Role Management APIs
- Centralized Logging and Middleware Support
- Scalable and Modular Design

🚀 Getting Started

1️⃣ Prerequisites

- Python 3.10+
- FastAPI
- Uvicorn
- SQLAlchemy

2️⃣ Installation

Clone the repo and install dependencies:

```
git clone https://github.com/your-username/fastapi_project.git
cd fastapi_project
python -m venv venv
source venv/bin/activate  # On Windows: venv\Scripts\activate
pip install -r requirements.txt
```

3️⃣ Environment Variables

Create a `.env` file in the project root:

```
DATABASE_URL=sqlite:///./test.db
SECRET_KEY=your-secret-key
ALGORITHM=HS256
ACCESS_TOKEN_EXPIRE_MINUTES=30
```

4️⃣ Run the Application

```
uvicorn app.main:app --reload
```

Visit http://localhost:8000/docs for Swagger UI.

📖 API Endpoints (v1)

🔑 Auth

- `POST /api/login` – Login with credentials
- `POST /api/logout` – Logout and revoke token

👤 Users

- `GET /api/users` – List all users (pagination supported)

- `POST /api/users` – Create a new user
- `GET /api/users/{id}` – Get a user by ID
- `PUT /api/users/{id}/role` – Assign a role to a user

📦 Products

- `GET /api/products` – List products (pagination/filtering)
- `POST /api/products` – Create product
- `GET /api/products/{id}` – Get product by ID
- `PUT /api/products/{id}` – Update product
- `DELETE /api/products/{id}` – Delete product
- `POST /api/products/{id}/category` – Assign product to category
- `DELETE /api/products/{id}/category` – Remove product from category

🏷️ Categories

- `GET /api/categories` – List categories
- `POST /api/categories` – Create category
- `PUT /api/categories/{id}` – Update category
- `DELETE /api/categories/{id}` – Delete category

🛡️ Roles

- `GET /api/roles` – List roles
- `POST /api/roles` – Create role
- `PUT /api/roles/{id}` – Update role
- `DELETE /api/roles/{id}` – Delete role

🧪 Running Tests

```
pytest
```

📌 Development Guide

- All routes live inside `app/api/v1/endpoints/`
- CRUD logic inside `app/crud/`
- Business rules inside `app/services/`
- Pydantic schemas inside `app/schemas/`
- Central configs & security inside `app/core/`

✅ Follow this structure when adding new features.

✅ Use versioned APIs (`v1`, `v2`, …) for backward compatibility.