FastAPI Minimal Backend

一个最小可运行的 FastAPI 后端项目，只保留基础结构，便于快速启动，后续可按需扩展。

✨ 特性

- 🚀 **FastAPI** - 高性能的现代 Web 框架
- 🗄️ **SQLAlchemy 2.0** - 异步 ORM，支持 PostgreSQL
- 🔐 **JWT 认证** - 完整的用户认证和授权系统
- 📝 **Pydantic** - 数据验证和序列化
- 🎯 **分层架构** - Repository、Service、API 三层架构
- 🐳 **Docker** - 完整的容器化支持
- 📊 **日志系统** - 结构化日志（loguru）
- 📚 **API 文档** - 自动生成的 Swagger/ReDoc 文档
- 🛡️ **异常处理** - 统一的异常处理机制

📁 项目结构（精简版）

```
.
├── app/
│   ├── __init__.py
│   ├── main.py                 # 应用入口（仅健康检查）
├── .gitignore
├── pyproject.toml
└── README.md
```

🚀 快速开始

本地开发（Poetry）

1. 克隆项目

```
git clone <repository-url>
cd fastapi-enterprise-backend
```

2. 使用 Poetry 安装依赖

```
pip install poetry
poetry install --no-root
```

3. 启动应用

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

应用将在 http://localhost:8000 启动

- API 文档: http://localhost:8000/api/docs
- ReDoc 文档: http://localhost:8000/api/redoc

常见问题

- 如果你使用 Poetry 2.x，`poetry shell` 默认不可用。可用：
  - `poetry env activate`（按提示激活），或
  - 安装插件：`poetry self add poetry-plugin-shell` 后使用 `poetry shell`

使用 Makefile (Linux/Mac)

```
# 查看所有可用命令
make help

# 安装依赖
make install

# 运行应用
make run

# 运行测试
make test

# 代码格式化
make format

# 代码检查
make lint

# Docker 操作
make docker-up
make docker-down
make docker-logs
```

API 使用

- **健康检查**

```
curl -s http://localhost:8000/health
```

测试

- **运行所有测试**

```
pytest
```

运行特定测试

```
pytest tests/test_api/test_auth.py
```

生成覆盖率报告

```
pytest --cov=app --cov-report=html
```

安全性

- JWT 令牌认证
- 密码使用 bcrypt 加密
- CORS 配置
- 请求速率限制（可选）
- SQL 注入防护（SQLAlchemy ORM）
- XSS 防护

⚠ **生产环境注意事项：**

1. 修改 `SECRET_KEY` 为强随机字符串
2. 使用 HTTPS
3. 配置正确的 CORS 源
4. 使用环境变量管理敏感信息
5. 启用速率限制
6. 定期更新依赖

代码格式化

```
black app tests
isort app tests
```

代码检查

```
flake8 app tests
mypy app
```

技术栈

核心框架

- **FastAPI** - Web 框架
- **Uvicorn** - ASGI 服务器
- **SQLAlchemy 2.0** - ORM
- **Pydantic** - 数据验证

> 后续可按需添加数据库、认证、日志、中间件等模块。当前仓库不包含 Docker 相关文件，若需容器化可再补充。