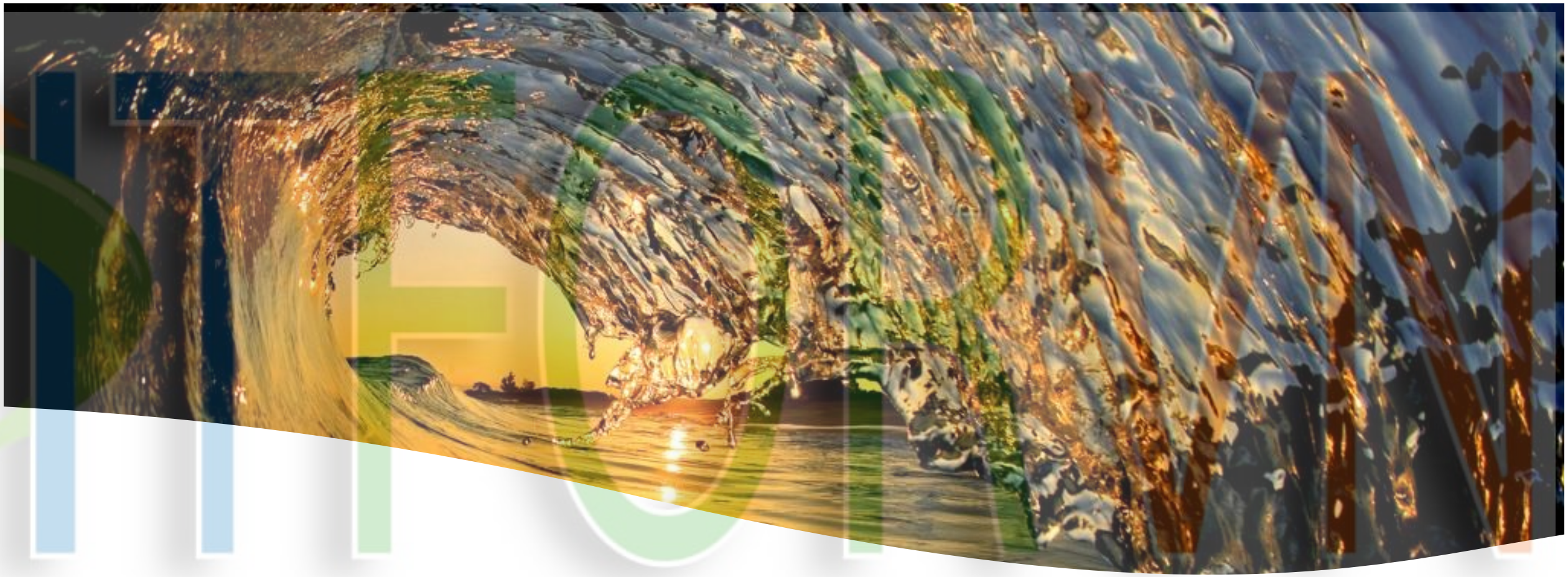


# Python: RESTful API application

ITFROVN - Python 2

# Python Flask

03

# 1. Flask environment prepare

1. Install virtual environment và libpq-dev:

   apt install python3-venv libpq-dev

2. Download Flask boilerplate:

   git clone -b db https://github.com/abpabab/flask-boilerplate

3. Tạo mới virtual environment:

   cd flask-boilerplate

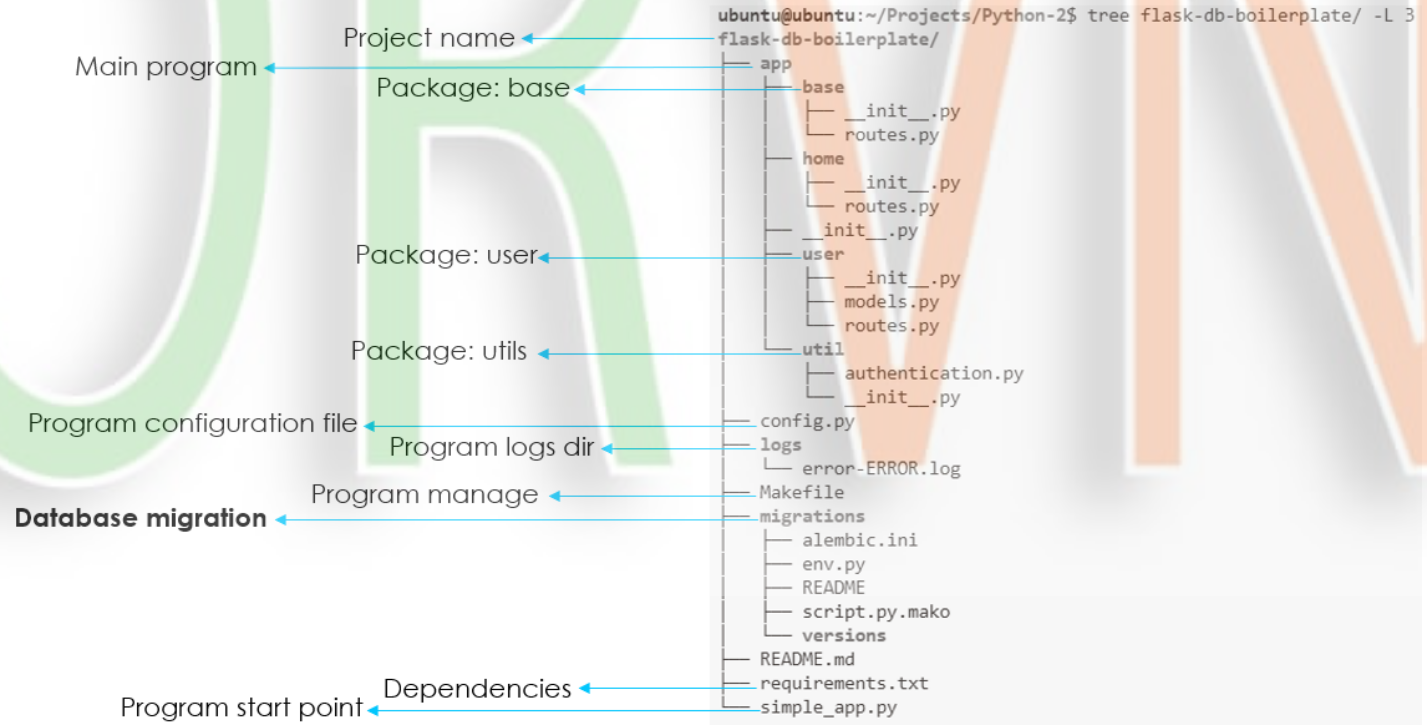
   python3 -m venv .venv

4. Kích hoạt virtual environment:

   source .venv/bin/activate

5. Cài đặt project dependency modules:

   pip3 install -r requirements.txt



# 2. Database prepare - MySQL

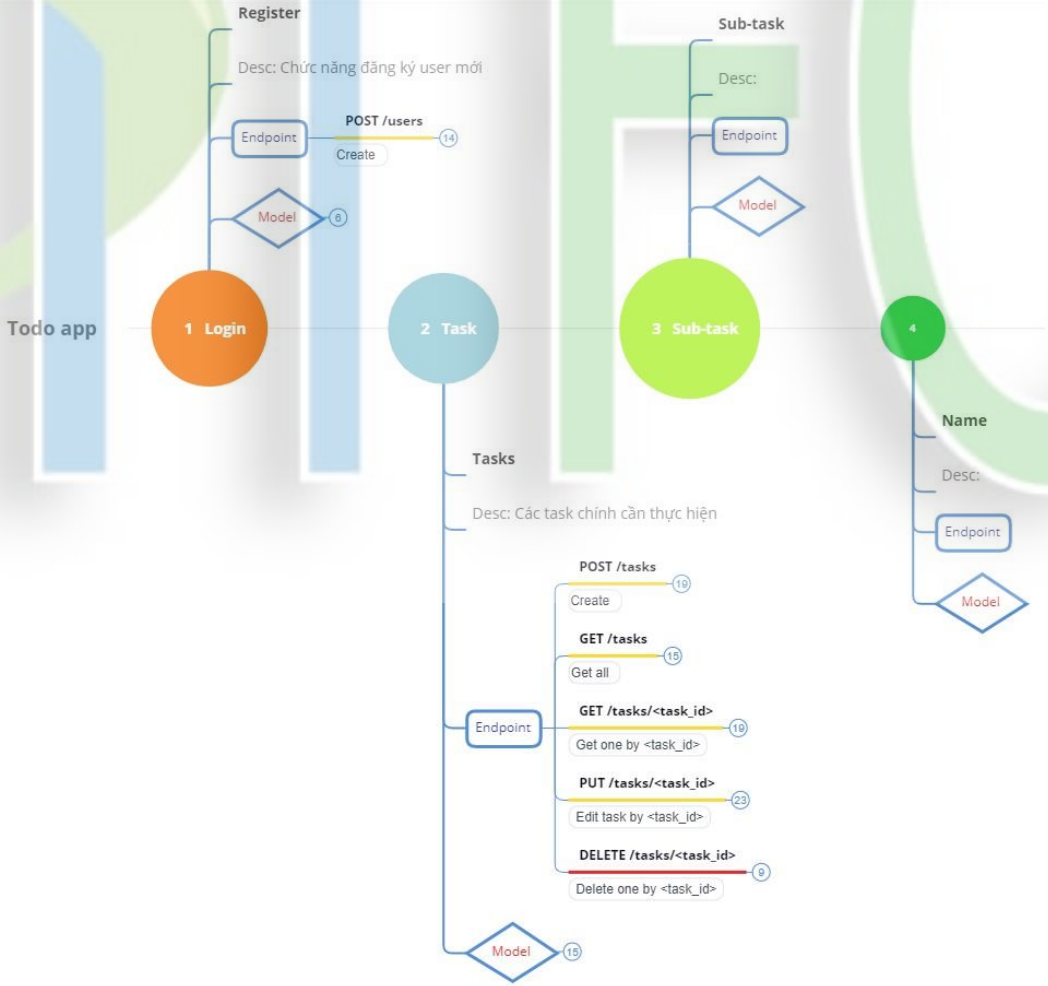
- MySQL 8.0 on Ubuntu 20.04:
  - Install MySQL server: apt install mysql-server
  - Create Database and User:

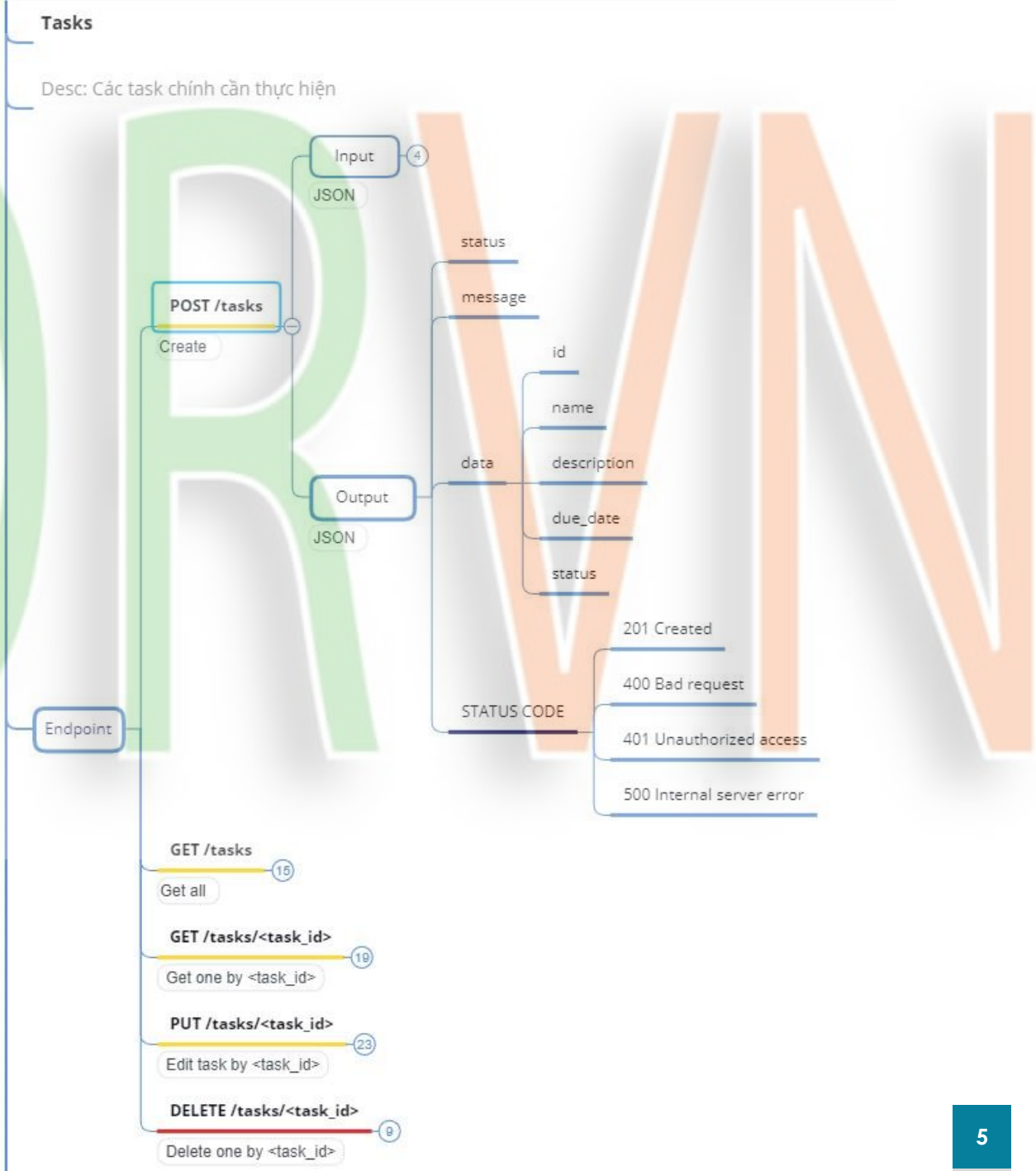
```
(root) # mysql

mysql> CREATE DATABASE tasks_db;
mysql> CREATE USER 'tasks_user'@'localhost' \
    -> IDENTIFIED WITH mysql_native_password \
    -> BY 'MyStrongPassword';
mysql> GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, INDEX, DROP, ALTER, CREATE TEMPORARY TABLES, LOCK TABLES \
    -> ON tasks_db.* \
    -> TO 'tasks_user'@'localhost';
```

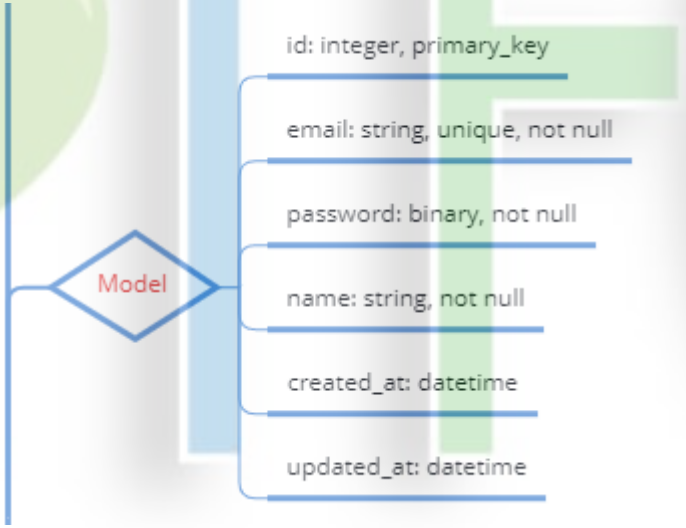
Database nên phân quyền **đúng** và **đủ** cho user, tránh dùng “GRANT ALL”

# 3. Functional design
# 4. Endpoint design



# 5. Model design




```python
from bcrypt import gensalt, hashpw
from flask_login import UserMixin
from datetime import datetime
from sqlalchemy import (
    Integer,
    BINARY,
    DateTime,
    String
    )
from app import db, login_manager


class User(db.Model, UserMixin):
    """
    User model
    """

    __tablename__ = 'User'

    id                  = db.Column(Integer, primary_key=True)
    email               = db.Column(String(100), unique=True, nullable=False)
    password            = db.Column(BINARY, nullable=False)
    name                = db.Column(String(50), nullable=False)
    created_time        = db.Column(DateTime, default=datetime.utcnow)
    updated_time        = db.Column(DateTime, default=datetime.utcnow, onupdate=datetime.utcnow)

    def __init__(self, user_data):
        for key, value in user_data.items():
            if key == 'password':
                value = hashpw(value.encode('utf8'), gensalt())
            setattr(self, key, value)

    def __repr__(self):
        return str(self.email)
```

# 6. Coding …

1. Edit config.py
2. Create all app/<package> structure:
    a. __init__.py
    b. routes.py
    c. models.py
3. Create models
4. Database migration: make db_upgrade

*Phải thực hiện import model vào routes.py trước khi thực hiện migration*

5. Coding routing logic

# 6.1 Get posted JSON

- Import:

  from bcrypt import checkpw, hashpw, gensalt

- Register - generate password:

  hashpw(value.encode('utf8'), gensalt())

- Login - check password:

  checkpw(submit_pwd.encode('utf8'), stored_pwd):

- Get posted JSON:

```
"""
{
    "email": "admin@gmail.com",
    "name": "Johny Deep"
}
"""
from flask import request

try:
    data = request.get_json()
except:
    abort(400)
```

- Lưu ý:
  - request header Content-Type: application/json
  - request.get_json(force = True): sẽ bỏ qua bước check header

# 6. Database: sqlalchemy

- SQLAlchemy is the Python SQL toolkit and Object Relational Mapper that gives application developers the full power and flexibility of SQL.
- Insert:

```
from app import db
from app.user.models import User

user = User({'email':'admin@gmail.com', 'name': 'Johny Deep'})
db.session.add(user)
db.session.commit()

user.id # returned inserted id
```

- Delete:

```
from app import db
from app.user.models import User

User.query.filter_by(id=123).delete()
## --- or --- ##
User.query.filter(User.id == 123).delete()

db.session.commit()
```