

架构的搭建

- config.py

```
1  # 数据库的配置
2  HOSTNAME = '10.1.88.88'
3  PORT     = '3306'
4  DATABASE = 'egkjqa'
5  USERNAME = 'root'
6  PASSWORD = '123456'
7  DB_URI = 'mysql+pymysql://{username}:{password}@{hostname}:{port}/{database}?
            charset=utf8'.format(username=USERNAME, password=PASSWORD, hostname=HOSTNAME, port=PORT, database=DATABASE)
8
9  SQLALCHEMY_DATABASE_URI = DB_URI
10 SQLALCHEMY_TRACK_MODIFICATIONS = True
11
12 SECRET_KEY = "SDFASDFASDFASDFASDFASDF"
```

- exts.py

```
1  from flask_sqlalchemy import SQLAlchemy
2  db = SQLAlchemy()
```

- app.py

```
1  from flask import Flask
2  import config
3  from exts import db
4  app = Flask(__name__)
5  app.config.from_object(config)
6  db.init_app(app)
```

- blueprints

- __init__.py
- qa.py
- user.py

- __init__.py

```
1  from .qa import bp as qa_bp
2  from .user import bp as user_bp
```

- qa.py

```
1  from flask import Blueprint
2  bp = Blueprint("qa", __name__, url_prefix="/")
3
4  @bp.route("/")
5  def index():
6      return "首页"
```

- user.py

```
1 from flask import Blueprint
2 bp = Blueprint("user", __name__, url_prefix="/user")
3
4 @bp.route("/login")
5 def login():
6     return "登陆"
```

- app.py

```
1 from flask import Flask
2 import config
3 from exts import db
4 from blueprints import qa_bp, user_bp
5
6 app = Flask(__name__)
7 app.config.from_object(config)
8 db.init_app(app)
9
10 app.register_blueprint(qa_bp)
11 app.register_blueprint(user_bp)
12
13 if __name__ == '__main__':
14     app.run()
15
```



首页



登陆

base.html页面实现

- bootstrap
 - bootstrap.4.6.min.css(官网给的参考用法, 可以选择下载下来)
 - <https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/css/bootstrap.min.css>

```
1
2 <link rel="stylesheet"
  href="https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/css/bootstrap.min
  .css" integrity="sha384-
  zCbKRCUGaJDkqS1kPbPd7TveP5iyJE0EjAuZQTgFLD2yLzuqKfdkllfg/esrtxukn"
  crossorigin="anonymous">
3
```

- templates
 - base.html
 - index.html

base.html

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <title>{% block title %}
6   {% endblock %}</title>
7
8   <link rel="stylesheet" href="{% url_for('static', filename =
  "bootstrap/bootstrap.4.6.min.css") %}">
9
10   {% block head %}
11   {% endblock %}
12 </head>
13 <body>
14 <nav class="navbar navbar-expand-lg navbar-light bg-light">
15   <div class="container">
16     <a class="navbar-brand" href="#">英格问答</a>
17     <button class="navbar-toggler" type="button" data-toggle="collapse" data-
  target="#navbarSupportedContent" aria-controls="navbarSupportedContent"
  aria-expanded="false" aria-label="Toggle navigation">
18       <span class="navbar-toggler-icon"></span>
19     </button>
20
21     <div class="collapse navbar-collapse" id="navbarSupportedContent">
22       <ul class="navbar-nav mr-auto">
23         <li class="nav-item active">
24           <a class="nav-link" href="#">首页 <span class="sr-only">(current)
  </span></a>
25         </li>
26         <li class="nav-item">
27           <a class="nav-link" href="#">发布问答</a>
28         </li>
29         <li class="nav-item ml-5">
```

```

30         <form class="form-inline my-2 my-lg-0">
31             <input class="form-control mr-sm-2" type="search"
placeholder="关键字" aria-label="Search">
32             <button class="btn btn-outline-success my-2 my-sm-0"
type="submit">搜索</button>
33         </form>
34     </li>
35
36 </ul>
37 <ul class="navbar-nav ">
38     <li class="nav-item ">
39         <a class="nav-link" href="#">登陆 <span class="sr-only">(current)
</span></a>
40     </li>
41     <li class="nav-item">
42         <a class="nav-link" href="#">注册</a>
43     </li>
44
45 </ul>
46 </div>
47 </div>
48 </nav>
49 <div class="container">
50     {% block body %}
51     {% endblock %}
52 </div>
53
54 </body>
55 </html>

```

index.html

```

1  {% extends "base.html" %}
2  {% block title %}
3      英格问答首页
4  {% endblock %}
5
6  {% block head %}
7
8  {% endblock %}
9
10 {% block body %}
11 <h1>
12  首页
13 </h1>
14 {% endblock %}

```

- qa.py

```

1  from flask import Blueprint, render_template
2  bp = Blueprint("qa", __name__, url_prefix="/")
3
4  @bp.route("/")
5  def index():
6      return render_template("index.html")

```

首页

注册登陆页面

- user.py

```
1 from flask import Blueprint, render_template
2 bp = Blueprint("user", __name__, url_prefix="/user")
3
4 @bp.route("/login")
5 def login():
6     return render_template("login.html")
7
8 @bp.route("/register")
9 def register():
10    return render_template("register.html")
```

- static
 - css
 - init.css

```
1 body{
2     background-color: #f3f3f3;
3 }
```

- base.html(增加一个css样式, 增加两个url_for反转函数)

```
1 <link rel="stylesheet" href="{{ url_for("static", filename =
2 "css/init.css") }}">
3
4
5 <li class="nav-item ">
6     <a class="nav-link" href="{{ url_for("user.login") }}">登陆 <span
7 class="sr-only">(current)</span></a>
8 </li>
9 <li class="nav-item">
10    <a class="nav-link" href="{{ url_for("user.register") }}">注册</a>
11 </li>
```

- login.html

```
1 {% extends "base.html" %}
2 {% block title %}
3     英格问答首页
4 {% endblock %}
5
```

```

6  {% block head %}
7
8  {% endblock %}
9
10 {% block body %}
11     <div class="row mt-4">
12         <div class="col"></div>
13         <div class="col">
14             <form>
15                 <div class="form-group">
16                     <label for="exampleInputEmail1">邮箱地址</label>
17                     <input type="email" class="form-control" id="exampleInputEmail1"
18 aria-describedby="emailHelp">
19                     <small id="emailHelp" class="form-text text-muted">我们不会把邮箱用于
20 其他用途</small>
21                 </div>
22                 <div class="form-group">
23                     <label for="exampleInputPassword1">密码</label>
24                     <input type="password" class="form-control"
25 id="exampleInputPassword1">
26                 </div>
27                 <button type="submit" class="btn btn-primary btn-block">立即登陆
28 </button>
29             </form>
30         </div>
31     </div>
32 {% endblock %}

```

- register.html

```

1  {% extends "base.html" %}
2  {% block title %}
3      英格问答首页
4  {% endblock %}
5
6  {% block head %}
7
8  {% endblock %}
9
10 {% block body %}
11     <div class="row mt-4">
12         <div class="col"></div>
13         <div class="col">
14             <form>
15                 <div class="form-group">
16                     <label for="exampleInputEmail1">邮箱地址</label>
17                     <input type="email" class="form-control" id="exampleInputEmail1"
18 aria-describedby="emailHelp">
19                     <small id="emailHelp" class="form-text text-muted">我们不会把邮箱用于
20 其他用途</small>
21                 </div>
22                 <div class="form-group">
23                     <label for="exampleInputEmail1">获取验证码</label>
24                     <div class="input-group">

```

```

24         <input type="text" class="form-control" >
25         <div class="input-group-append">
26             <button class="btn btn-outline-secondary" type="button">获取验证
码</button>
27         </div>
28     </div>
29 </div>
30
31     <div class="form-group">
32         <label for="exampleInputPassword1">密码</label>
33         <input type="password" class="form-control"
id="exampleInputPassword1">
34     </div>
35
36     <div class="form-group">
37         <label for="exampleInputPassword1">确认密码</label>
38         <input type="password" class="form-control" >
39     </div>
40
41     <button type="submit" class="btn btn-primary btn-block">立即注册
</button>
42 </form>
43 </div>
44 <div class="col"></div>
45 </div>
46
47 {% endblock %}

```

[<](#)
[>](#)
[刷新](#)
[地址簿](#)
[☆](#)
[http://127.0.0.1:5000/user/login](#)
[🔍](#)
[🔖](#)
[☆](#)
[合肥23岁女子失联](#)

[\[bbj1030\]](#)
[百度](#)
[工作台 · 图](#)
[哔哩哔哩](#)
[Google](#)
[python](#)
[GitHub](#)
[博客收藏](#)
[英语](#)
[小众软件](#)
[下载Zabbix](#)
[课程导读 | 1](#)
[云原生技术](#)
[OpenStack](#)
[go](#)

[英语问答](#)
[首页](#)
[发布问答](#)

[登陆](#)
[注册](#)

邮箱地址

我们不会把邮箱用于其他用途

密码

[<](#)
[>](#)
[刷新](#)
[地址簿](#)
[☆](#)
[http://127.0.0.1:5000/user/register](#)
[🔍](#)
[🔖](#)
[☆](#)
[合肥23岁女子失联](#)

[\[bbj1030\]](#)
[百度](#)
[工作台 · 图](#)
[哔哩哔哩](#)
[Google](#)
[python](#)
[GitHub](#)
[博客收藏](#)
[英语](#)
[小众软件](#)
[下载Zabbix](#)
[课程导读 | 1](#)
[云原生技术](#)
[OpenStack](#)
[go](#)

[英语问答](#)
[首页](#)
[发布问答](#)

[登陆](#)
[注册](#)

邮箱地址

我们不会把邮箱用于其他用途

获取验证码

密码

确认密码

注册登陆功能

邮箱配置

- pip install Flask-Mail

- config.py

```
1 # 邮箱配置
2 # 项目中用的是QQ邮箱
3 MAIL_SERVER = "smtp.qq.com"
4 MAIL_PORT = 465
5 MAIL_USE_TLS = False
6 MAIL_USE_SSL = True
7 MAIL_DEBUG = True
8 MAIL_USERNAME = "2177780569@qq.com"
9 MAIL_PASSWORD = "znndrwwhkr...."
10 MAIL_DEFAULT_SENDER = "2177780569@qq.com"
```

- exts.py

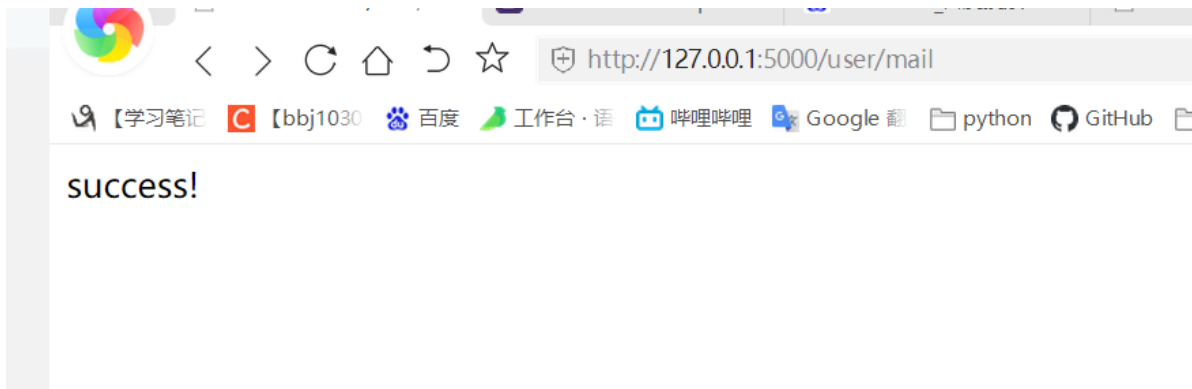
```
1 from flask_mail import Mail
2 mail = Mail()
```

- app.py

```
1 from exts import db, mail
2 mail.init_app(app)
```

- user.py

```
1 from exts import mail
2 from flask_mail import Message
3 @bp.route("/mail")
4 def my_mail():
5     message = Message(
6         subject="邮箱测试",
7         recipients=["441732861@qq.com"],
8         body = "这是一篇测试邮件",
9     )
10    mail.send(message)
11    return "success!"
```

数据库配置

- models.py

```
1 from exts import db
2 from datetime import datetime
3 class EmailCpatchaModel(db.Model):
4     __tablename__ = "email_cattcha"
5     id = db.Column(db.Integer, primary_key=True, autoincrement=True)
6     email = db.Column(db.String(100), nullable=False, unique=True)
7     captcha = db.Column(db.String(10), nullable=False)
8     create_time = db.Column(db.DateTime, default=datetime.now)
```

- user.py

```
1 from models import EmailCpatchaModel
```

- app.py

```
1 from flask_migrate import Migrate
2
3 migrate = Migrate(app, db)
```

- 终端

```
1 flask db init
2 flask db migrate
3 flask db upgrade
4
```

验证码存储

- user.py

```
1 from flask import Blueprint, render_template, request
2 from exts import mail, db
3 from flask_mail import Message
4 from models import EmailCpatchaModel
5 import string
6 import random
7 import datetime
8
9
```

```

10 @bp.route("/captcha")
11 def get_captcha():
12     email = request.args.get("email")
13     print(email)
14     my_string = string.digits + string.ascii_letters
15     captcha = "".join(random.sample(my_string, 4))
16     print(captcha)
17     if email:
18         message = Message(
19             subject="验证码",
20             recipients=[email],
21             body=f"英格科技问答，您的注册验证码是：{captcha}，请不要告诉任何人哦",
22         )
23         mail.send(message)
24
25     # 定义验证码数据
26     captcha_model =
EmailCaptchaModel.query.filter_by(email=email).first()
27     if captcha_model:
28         captcha_model.captcha = captcha
29         captcha_model.create_time = datetime.now()
30         db.session.commit()
31     else:
32         captcha_model = EmailCaptchaModel(email=email, captcha=captcha)
33         db.session.add(captcha_model)
34         db.session.commit()
35     return "success!"
36 else:
37     return "没有传递邮箱"

```

注册功能

- UserModel.py

```

1 class UserModel(db.Model):
2     __tablename__ = "user"
3     id = db.Column(db.Integer, primary_key=True, autoincrement=True)
4     username = db.Column(db.String(150), nullable=False, unique=True)
5     password = db.Column(db.String(200), nullable=False)
6     email = db.Column(db.String(100), nullable=False, unique=True)
7     join_time = db.Column(db.DateTime, default=datetime.now)

```

- 终端

```

1 flask db init
2 flask db migrate
3 flask db upgrade

```

- blueprints
 - forms.py

```

1 import wtforms
2 from wtforms.validators import length, email, EqualTo
3 from models import EmailCaptchaModel, UserModel
4

```

```

5 class RegisterForm(wtforms.Form):
6     username = wtforms.StringField(validators=[length(min=3,max=20)])
7     captcha = wtforms.StringField(validators=[length(min=4,max=4)])
8     password = wtforms.StringField(validators=[length(min=6,max=20)])
9     password_confirm = wtforms.StringField(validators=[EqualTo("password")])
10    email = wtforms.StringField(validators=[email()])
11
12    def validate_captcha(self, field):
13        captcha = field.data
14        email = self.email.data
15        captcha_model = EmailCaptchaModel.query.filter_by(email =
email).first()
16        if not captcha_model or captcha_model.captcha.lower() !=
captcha.lower():
17            print("邮箱验证码错误")
18            raise wtforms.ValidationError("邮箱验证码错误")
19
20    def validate_email(self, field):
21        email = field.data
22        user_model = UserModel.query.filter_by(email=email).first()
23        if user_model:
24            print("邮箱已经被注册过")
25            raise wtforms.ValidationError("邮箱已经被注册过")

```

- user.py

```

1 from werkzeug.security import generate_password_hash
2
3 @bp.route("/register", methods=["GET", "POST"])
4 def register():
5     if request.method == "GET":
6         return render_template("register.html")
7     else:
8         form = RegisterForm(request.form)
9         if form.validate():
10            email = form.email.data
11            username = form.username.data
12            password = form.password.data
13            hash_password = generate_password_hash(password)
14            user = UserModel(email=email, username=username,
password=hash_password)
15            db.session.add(user)
16            db.session.commit()
17            return redirect(url_for("user.login"))
18        else:
19            print("validate 失败")
20            return redirect(url_for("user.register"))

```

- register.html

```
{% block body %}
<div class="row mt-4">
  <div class="col"></div>
  <div class="col">
    <form method="POST", action="{{ url_for("user.register") }}">
      <div class="form-group">...
```

发送验证码

- static
 - js
 - register.js
 - jquery
 - jquery.3.6.min.js
 - <https://cdn.bootcdn.net/ajax/libs/jquery/3.6.0/jquery.min.js>
- jquery.3.6.min.js

```
1 // 来自https://cdn.bootcdn.net/ajax/libs/jquery/3.6.0/jquery.min.js
```

- register.html

```
1 {% block head %}
2   <script src="{{ url_for("static", filename = "jquery/jquery.3.6.min.js") }}"></script>
3   <script src="{{ url_for("static", filename = "js/register.js") }}"></script>
4 {% endblock %}
5 -----
6   <button class="btn btn-outline-secondary" type="button"
7   id="captcha-btn">获取验证码</button>
```

- user.py

```

1  @bp.route("/captcha", methods=["POST"])
2  def get_captcha():
3      email = request.form.get("email")
4
5      -----
6
7      # code: 200, 成功的、正常的请求
8      return jsonify({"code": 200})
9  else:
10     # code: 400, 客户端错误
11     return jsonify({"code": 400})

```

- register.js

```

1  function bindCaptchaBtnClick(){
2      $("#captcha-btn").on("click", function (event){
3          var email = $("input[name='email']").val();
4          if(!email){
5              alert("请先输入邮箱");
6              return ;
7          }
8          //通过js发送网络请求: ajax。Async JavaScript And XML (JSON)
9          $.ajax({
10             url: "/user/captcha",
11             method: "POST",
12             data: {
13                 "email": email
14             },
15             success: function (res){
16                 var code = res["code"];
17                 if (code == 200){
18                     alert("验证码发送成功");
19                 }else {
20                     alert(res["message"]);
21                 }
22             }
23         })
24     });
25 }
26
27 //等网页文档所有元素加载完成之后再执行
28 $(function (){
29     bindCaptchaBtnClick();
30 });

```

- register.js 增加了验证码倒计时功能

```

1  function bindCaptchaBtnClick(){
2      $("#captcha-btn").on("click", function (event){
3          var $this = $(this);
4          var email = $("input[name='email']").val();
5          if(!email){
6              alert("请先输入邮箱");
7              return ;
8          }
9          //通过js发送网络请求: ajax。Async JavaScript And XML (JSON)

```

```

10     $.ajax({
11         url: "/user/captcha",
12         method: "POST",
13         data: {
14             "email": email
15         },
16         success: function (res){
17             var code = res["code"];
18             if (code == 200){
19                 //取消点击事件
20                 $this.off("click");
21                 //开始倒计时
22                 var countDown = 60;
23                 var timer = setInterval(function (){
24                     countDown -= 1;
25                     if (countDown > 0){
26                         $this.text(countDown+"秒后重新发送");
27                     }else {
28                         $this.text("获取验证码");
29                         //重新执行一下这个函数，重新绑定点击事件
30                         bindCaptchaBtnClick()
31                         //如果不需要倒计时了，那么就要记得清楚倒计时了，否则会一直
32                         //执行下去
33                         clearInterval(timer)
34                     }
35                 },1000)
36                 alert("验证码发送成功");
37             }else {
38                 alert(res["message"]);
39             }
40         })
41     });
42 }
43
44 //等网页文档所有元素加载完成之后再执行
45 $(function (){
46     bindCaptchaBtnClick();
47 });

```

实现效果

登陆注册

邮箱地址

441732861@qq.com

我们不会把邮箱用于其他用途

获取验证码

58秒后重新发送

用户名

密码

登陆功能

登陆基本功能

- login.html

```
1  {% block body %}
2      <div class="row mt-4">
3          <div class="col"></div>
4          <div class="col">
5              <form action="{{ url_for('user.login')}}" method="post">
6                  <div class="form-group">
7                      <label for="exampleInputEmail">邮箱地址</label>
8                      <input type="email" class="form-control" id="exampleInputEmail"
9                          aria-describedby="emailHelp" name="email">
10                 </div>
11                 <div class="form-group">
12                     <label for="exampleInputPassword1">密码</label>
13                     <input type="password" class="form-control"
14                         id="exampleInputPassword1" name="password">
15                 </div>
16                 <button type="submit" class="btn btn-primary btn-block">立即登陆
17             </button>
18         </form>
19     </div>
20 </div>
21 </div>
22 {% endblock %}
```

- forms.py

```
1  class LoginForm(wtforms.Form):
2      email = wtforms.StringField(validators=[email()])
3      password = wtforms.StringField(validators=[length(min=6,max=20)])
```

- login.py

```
1
2  @bp.route("/login", methods=["GET", "POST"])
```

```

3  def login():
4      if request.method == "GET":
5          return render_template("login.html")
6      else:
7          form = LoginForm(request.form)
8          if form.validate():
9              email = form.email.data
10             password = form.password.data
11             user = UserModel.query.filter_by(email=email).first()
12             if user and check_password_hash(user.password, password):
13                 session["user_id"] = user.id
14                 return redirect("/")
15             else:
16                 return redirect(url_for("user.login"))
17         else:
18             return redirect(url_for("user.login"))

```

登陆错误提示

- login.py 增加了登陆错误提示

```

1  @bp.route("/login", methods=["GET", "POST"])
2  def login():
3      if request.method == "GET":
4          return render_template("login.html")
5      else:
6          form = LoginForm(request.form)
7          if form.validate():
8              email = form.email.data
9              password = form.password.data
10             user = UserModel.query.filter_by(email=email).first()
11             if user and check_password_hash(user.password, password):
12                 session["user_id"] = user.id
13                 return redirect("/")
14             else:
15                 flash("邮箱密码不匹配!! ")
16                 return redirect(url_for("user.login"))
17         else:
18             flash("邮箱或密码格式错误!! ")
19             return redirect(url_for("user.login"))

```

- login.html

```

1      <div class="form-group">
2          {% for message in get_flashed_messages() %}
3              <div>
4                  <div class="text-danger">{{ message }}</div>
5              </div>
6          {% endfor %}
7      </div>
8      <div class="form-group">
9          <button type="submit" class="btn btn-primary btn-block">立即登陆
10     </button>
11 </div>

```


实现效果

邮箱地址

密码

邮箱密码不匹配！！

立即登陆

邮箱地址

密码

邮箱或密码格式错误！！

立即登陆

访问控制

登陆成功进行渲染

- app.py

```
1  # 在所有请求之前会执行的函数
2  @app.before_request
3  def before_request():
4      user_id = session.get("user_id")
5      if user_id:
6          try:
7              user = UserModel.query.get(user_id)
8              # 给g绑定一个user属性，这个属性就是user这个变量
9              # setattr(g, "user", user)
10             g.user = user
11         except:
12             g.user = None
13 # 请求来了 -> before_request -> 视图函数 -> 视图函数中返回模板 ->
context_processor
14 # 渲染的所有的模板都会执行这个函数
15 @app.context_processor
16 def context_processor():
17     if hasattr(g, "user"):
18         return {"user": g.user}
19     else:
20         return {}
21
22 if __name__ == '__main__':
23     app.run()
```

- base.html

```
1      {% if user %}
2          <li class="nav-item ">
3              <span class="nav-link">{{ user.username }}</span>
4              </li>
5          <li class="nav-item">
6              <a class="nav-link" href="#">退出登陆</a>
7              </li>
8      {% else %}
9          <li class="nav-item ">
10             <a class="nav-link" href="{{ url_for("user.login") }}">登陆
11 <span class="sr-only">(current)</span></a>
12             </li>
13             <li class="nav-item">
14                 <a class="nav-link" href="{{ url_for("user.register") }}">注册
15 </a>
16             </li>
17     {% endif %}
```

退出登陆

- user.py

```

1 | @bp.route("/logout")
2 | def logout():
3 |     # 清楚所有的数据
4 |     session.clear()
5 |     return redirect(url_for("user.login"))

```

- base.html

```

1 | <a class="nav-link" href="{{ url_for("user.logout") }}">退出登陆
   | </a>

```

装饰器功能

在需要用到的功能之前增加装饰器，用来判断是否登陆

- decorators.py

```

1 | from flask import g, redirect, url_for
2 | from functools import wraps
3 |
4 | def login_required(func):
5 |     @wraps(func)
6 |     def wrapper(*args, **kwargs):
7 |         if hasattr(g, "user"):
8 |             return func(*args, **kwargs)
9 |         else:
10 |             return redirect(url_for("user.login"))
11 |     return wrapper

```

- qa.py

```

1 | from decorators import login_required
2 |
3 | @bp.route("/question/public")
4 | @login_required
5 | def public_question():
6 |     return render_template("public_question.html")

```

- public_question.html

```

1 | {% extends "base.html" %}
2 | {% block title %}发布问答{% endblock %}
3 | {% block body %}
4 | <h1>发布问答</h1>
5 | {% endblock %}

```

- base.html

```

1         <a class="nav-link" href="/">首页 <span class="sr-only">
(current)</span></a>
2
3
4         <a class="nav-link" href="{% url_for('qa.public_question')
}}">发布问答</a>

```

发布问答

发布问题

- public_question.html

```

1  {% extends "base.html" %}
2  {% block title %}发布问答{% endblock %}
3  {% block body %}
4  <div class="row" style="margin-top: 20px">
5  <div class="col"></div>
6  <div class="col-6">
7      <h1 style="text-align: center">发布问答</h1>
8      <form action="{% url_for('qa.public_question') %}" method="post">
9          <div class="form-group">
10             <input type="text" name="title" class="form-control" placeholder="请输入标题">
11             </div>
12             <div class="form-group">
13                 <textarea name="content" class="form-control" rows="10"
placeholder="请输入内容"></textarea>
14             </div>
15             <div class="form-group" style="text-align: right">
16                 <button class="btn btn-primary">发布</button>
17             </div>
18         </form>
19     </div>
20 <div class="col"></div>
21 </div>
22 {% endblock %}

```

[英格问答](#)
[首页](#)
[发布问答](#)

bbj1030
退出登陆

发布问答

请输入标题

请输入内容

发布

- froms.py

```

1 class QuestionForm(wtforms.Form):
2     title = wtforms.StringField(validators=[length(min=3, max=150)])
3     content = wtforms.StringField(validators=[length(min=4)])

```

- models.py

```

1 class QuestionModel(db.Model):
2     __tablename__ = "question"
3     id = db.Column(db.Integer, primary_key=True, autoincrement=True)
4     title = db.Column(db.String(150), nullable=False)
5     content = db.Column(db.Text, nullable=False)
6     author_id = db.Column(db.Integer, db.ForeignKey("user.id"))
7     create_time = db.Column(db.DateTime, default=datetime.now)
8
9     author = db.relationship("UserModel", backref="questions")
10
11     -----
12
13 flask db migrate
14 flask db upgrade


```


- qa.py

```





1
2 @bp.route("/question/public", methods=["GET", "POST"])
3 @login_required
4 def public_question():
5     if request.method == "GET":
6         return render_template("public_question.html")
7     else:
8         form = QuestionForm(request.form)
9         if form.validate():
10             title = form.title.data
11             content = form.content.data
12             question = QuestionModel(title=title, content=content,
13             author=g.user)
14             db.session.add(question)
15             db.session.commit()
16             return redirect("/")
17         else:
18             flash("标题或者内容格式错误!!")
19             return redirect(url_for("qa.public_question"))

```

 Eagle







 egkjqa

 ▼ 表

-  alembic_version
-  email_cattcha
-  **question**
-  user

对象

question @egkjqa (Eagle) - 表

 开始事务
  文本
  筛选
  排序
  导入
  导出

id	title	content	author_id
1	如何看待王力宏事件	大家各抒己见	3

- public_question.html

```

1 {% extends "base.html" %}

```

```

2  {% block title %}发布问答{% endblock %}
3  {% block body %}
4      <div class="row" style="margin-top: 20px">
5          <div class="col"></div>
6          <div class="col-6">
7              <h1 style="text-align: center">发布问答</h1>
8              <form action="{{ url_for('qa.public_question') }}" method="post">
9                  <div class="form-group">
10                     <input type="text" name="title" class="form-control"
placeholder="请输入标题">
11                 </div>
12                 <div class="form-group">
13                     <textarea name="content" class="form-control" rows="10"
placeholder="请输入内容"></textarea>
14                 </div>
15                 {% for message in get_flashed_messages() %}
16                     <div>
17                         <div class="text-danger">{{ message }}</div>
18                     </div>
19                 {% endfor %}
20                 <div class="form-group" style="text-align: right">
21                     <button class="btn btn-primary">发布</button>
22                 </div>
23             </form>
24         </div>
25     </div class="col"></div>
26 </div>
27 {% endblock %}

```

发布问答

标题或者内容格式错误！！

发布

展示问答

- static
 - CSS
 - index.css

```
1  .question-ul li {
2      padding: 10px;
3      overflow: hidden;
4      display: flex;
5      background-color: #fff;
6      border-bottom: 1px solid #eee;
7  }
8
9  .side-question {
10     flex-basis: 38px ;
11     height: 100%;
12 }
13
14 .side-question-avatar {
15     width: 38px;
16     height: 38px;
17     border-radius: 3px;
18 }
19
20 .question-main {
21     flex: 1;
22     width: 660px;
23     margin-left: 10px;
24     overflow: hidden;
25 }
26
27 .question-title a {
28     color: #259;
29     font-size: 14px;
30     font-weight: 900;
31 }
32
33 .question-author {
34     font-size: 12px;
35     margin-top: 5px;
36 }
37
38 .question-content {
39     margin-top: 5px;
40     font-size: 12px;
41 }
42
43 .question-detail {
44     text-align: right;
45     margin-top: 10px;
46 }
47
48 .question-detail .question-author {
49     margin-right: 10px;
50 }
```

- templates

o index.html

```
1 {% extends "base.html" %}
2 {% block title %}
3     英格问答首页
4 {% endblock %}
5
6 {% block head %}
7     <link rel="stylesheet" href="{% url_for( 'static', filename =
8 'css/index.css') %}" class="rel">
9 {% endblock %}
10
11 {% block body %}
12     <div class="row" style="margin-top: 20px">
13         <div class="col"></div>
14         <div class="col-8">
15             <ul class="question-ul">
16                 <li>
17                     <div class="side-question">
18                         
20                         </div>
21                         <div class="question-main">
22                             <div class="question-title"><a href="#">钢铁是怎样练成的</a></div>
23                             <div class="question-content">XXX</div>
24                             <div class="question-detail">
25                                 <span class="question-author">bbj1030</span>
26                                 <span class="question-time">2020-12-24</span>
27                             </div>
28                         </div>
29                     </li>
30                 </ul>
31             </div>
32         <div class="col"></div>
33     </div>
34 {% endblock %}
```

英格问答 首页 发布问答 关键字 搜索 bbj1030 退出



钢铁是怎样练成的

XXX

bbj1030 2020-12-24

- qa.py


```

1 @bp.route("/")
2 def index():
3     questions = QuestionModel.query.order_by(db.text("-create_time")).all()
4     return render_template("index.html", questions=questions)

```

- index.html 使用数据库的数据对首页进行渲染

```

1 {% extends "base.html" %}
2 {% block title %}
3     英格问答首页
4 {% endblock %}
5
6 {% block head %}
7     <link rel="stylesheet" href="{{ url_for( 'static', filename =
8         'css/index.css') }}" class="rel">
9 {% endblock %}
10
11 {% block body %}
12     <div class="row" style="margin-top: 20px">
13         <div class="col"></div>
14         <div class="col-8">
15             <ul class="question-ul">
16                 {% for question in questions %}
17                     <li>
18                         <div class="side-question">
19                             
21                             </div>
22                             <div class="question-main">
23                                 <div class="question-title"><a href="#">{{ question.title }}</a>
24                                 </div>
25                                 <div class="question-content">{{ question.content }}</div>
26                                 <div class="question-detail">
27                                     <span class="question-author">b{{ question.author.username }}
28                                     </span>
29                                     <span class="question-time">{{ question.create_time }}</span>
30                                 </div>
31                             </li>
32                         {% endfor %}
33                     </ul>
34                 </div>
35             <div class="col"></div>
36         </div>
37     {% endblock %}

```

问答详情页面

- index.html

```

1     <div class="question-title"><a href="{{
2         url_for("qa.question_detail", question_id=question.id) }}">{{ question.title
3         }}</a></div>

```

- qa.py

```
1 @bp.route("/question/<int:question_id>")
2 def question_detail(question_id):
3     question = QuestionModel.query.get(question_id)
4     return render_template("detail.html", question=question)
```

- static
 - css
 - detail.css

```
1 .page-title{
2     text-align: center;
3 }
4
5 .question-info {
6     text-align: center;
7 }
8
9 .comment-group-title {
10     margin-top: 20px;
11 }
12
13 .form-container {
14     width: 500px;
15     margin-top: 10px;
16     text-align: right;
17 }
18
19 .comment-group li {
20     border-bottom: 1px solid #eee;
21     padding: 10px 0;
22 }
23
24 .comment-group li .user-info {
25     height: 40px;
26     line-height: 40px;
27     color: #9b9b9b;
28     font-size: 16px;
29 }
30
31 .user-info .avatar {
32     height: 40px;
33     width: 40px;
34     float: left;
35     border-radius: 50%;
36 }
37
38 .user-info .username {
39     margin-left: 10px;
40 }
41
42 .user-info .create-time {
43     float: right;
44 }
45
46 .comment-content {
47     margin-left: 50px;
```

- detail.html

```
1  {% extends "base.html" %}
2
3  {% block title %}问答详情{% endblock %}
4
5  {% block head %}
6      <link rel="stylesheet" href="{% url_for("static",
7      filename="css/detail.css") %}">
8  {% endblock %}
9
10 {% block body %}
11     <div class="row" style="margin-top: 20px">
12         <div class="col"></div>
13         <div class="col-8" style="background-color: #fff;">
14             <h3 class="page-title">钢铁是怎样练成的</h3>
15             <p class="question-info">
16                 <span>作者：尼古拉</span>
17                 <span>时间：2020-12-24</span>
18             </p>
19             <hr>
20             <p class="question-content">xxx</p>
21             <hr>
22         </div>
23     </div>
24 {% endblock %}
```

首页 发布问答

关键字

搜索

bbj1030 退出登陆

钢铁是怎样练成的

作者：尼古拉 时间：2020-12-24

xxx

评论功能

评论页面

- models.py

```
1
2  class AnswerModel(db.Model):
3      __tablename__ = "answer"
4      id = db.Column(db.Integer, primary_key=True, autoincrement=True)
5      content = db.Column(db.Text, nullable=False)
6      create_time = db.Column(db.DateTime, default=datetime.now)
7      question_id = db.Column(db.Integer, db.ForeignKey("question.id"))
```

```

8     author_id = db.Column(db.Integer, db.ForeignKey("user.id"))
9
10    question = db.relationship("QuestionModel", backref = "answers")
11    author = db.relationship("UserModel", backref="answers")
12    -----
13
13 flask db migrate
14 flask db upgrade

```

- detail.css

```

1 .comment-group{
2     list-style: none;
3     padding-left: 0;
4 }

```

- detail.html

```

1 {% extends "base.html" %}
2
3 {% block title %}{{ question.title }}{% endblock %}
4
5 {% block head %}
6     <link rel="stylesheet" href="{{ url_for("static",
7 filename="css/detail.css") }}">
8 {% endblock %}
9
10 {% block body %}
11     <div class="row" style="margin-top: 20px">
12         <div class="col"></div>
13         <div class="col-8" style="background-color: #fff;">
14             <h3 class="page-title">{{ question.title }}</h3>
15             <p class="question-info">
16                 <span>作者: {{ question.author.username }}</span>
17                 <span>时间: {{ question.create_time }}</span>
18             </p>
19             <hr>
20             <p class="question-content">{{ question.content }}</p>
21             <hr>
22             <h4 class="comment-group-title">评论 (10) : </h4>
23             <form action="#" method="post">
24                 <input type="hidden" name="question_id" value="{{ question.id }}">
25                 <div class="form-group">
26                     <input type="text" placeholder="请填写评论" class="form-control">
27                 </div>
28                 <div class="form-group" style="text-align: right">
29                     <button class="btn btn-primary">评论</button>
30                 </div>
31             </form>
32             <ul class="comment-group">
33                 <li>
34                     <div class="user-info">
35                         
37                         <span class="username">bbj1030</span>
38                         <span class="create-time">2020-12-24</span>

```

```

37         </div>
38         <p class="comment-content">我觉得挺好的啊</p>
39     </li>
40 </ul>
41
42 </div>
43 <div class="col"></div>
44 </div>
45 {% endblock %}

```

rm -rf /* 什么意思？

作者：bbj1030 时间：2021-12-24 16:49:16

上班同事说这个很好用，是真的吗？

评论 (10) :

请填写评论

评论



bbj1030

2020-12-24

我觉得挺好的啊

评论

- forms.py

```

1 class AnswerForm(wtforms.Form):
2     content = wtforms.StringField(validators=[length(min=1)])

```

- qa.py

```

1
2 @bp.route("/answer/<int:question_id>", methods=["GET", "POST"])
3 def answer(question_id):
4     form = AnswerForm(request.form)
5     if form.validate():
6         content = form.content.data
7         answer_model = AnswerModel(content=content, author =
g.user, question_id = question_id)
8         db.session.add(answer_model)
9         db.session.commit()
10        return redirect(url_for("qa.question_detail", question_id =
question_id))
11    else:
12        print(11111)

```

```

13     flash("表单验证失败!!")
14     return redirect(url_for("qa.question_detail",
15                             question_id=question_id))

```

- detail.html

```

1  {% extends "base.html" %}
2
3  {% block title %}{{ question.title }}{% endblock %}
4
5  {% block head %}
6      <link rel="stylesheet" href="{{ url_for("static",
7          filename="css/detail.css") }}">
8  {% endblock %}
9
10 {% block body %}
11     <div class="row" style="margin-top: 20px">
12         <div class="col"></div>
13         <div class="col-8" style="background-color: #fff;">
14             <h3 class="page-title">{{ question.title }}</h3>
15             <p class="question-info">
16                 <span>作者: {{ question.author.username }}</span>
17                 <span>时间: {{ question.create_time }}</span>
18             </p>
19             <hr>
20             <p class="question-content">{{ question.content }}</p>
21             <hr>
22             <h4 class="comment-group-title">评论 (10) : </h4>
23             <form action="{{ url_for("qa.answer", question_id=question.id) }}"
24                 method="post">
25                 <div class="form-group">
26                     <input type="text" placeholder="请填写评论" name="content"
27                         class="form-control">
28                 </div>
29                 <div class="form-group" style="text-align: right">
30                     <button class="btn btn-primary">评论</button>
31                 </div>
32             </form>
33             <div class="form-group">
34                 {% for message in get_flashed_messages() %}
35                     <div>
36                         <div class="text-danger">{{ message }}</div>
37                     </div>
38                 {% endfor %}
39             </div>
40             <ul class="comment-group">
41                 <li>
42                     <div class="user-info">
43                         
45                         <span class="username">bbj1030</span>
46                         <span class="create-time">2020-12-24</span>
47                     </div>
48                     <p class="comment-content">我觉得挺好的啊</p>
49                 </li>
50             </ul>

```

```

47
48     </div>
49     <div class="col"></div>
50 </div>
51 {% endblock %}

```

The screenshot shows a web application interface for a database. On the left is a sidebar with a tree view of database schemas: 'egkjqa' (expanded) contains 'alembic_version', 'answer' (selected), 'email_cattcha', 'question', and 'user'. Below this are other schemas: 'information_schema', 'mysql', 'performance_schema', and 'test'. The main area displays a table view for the 'answer' table. The table has columns: 'id', 'content', 'create_time', 'question_id', and 'author_id'. It contains 4 rows of data, with the 4th row highlighted in blue. The table view includes a toolbar with options like '开始事务', '文本', '筛选', '排序', '导入', and '导出'.

渲染评论页面

- detail.html

```

1  {% extends "base.html" %}
2
3  {% block title %}{{ question.title }}{% endblock %}
4
5  {% block head %}
6      <link rel="stylesheet" href="{{ url_for("static",
7          filename="css/detail.css") }}">
8  {% endblock %}
9
10 {% block body %}
11     <div class="row" style="margin-top: 20px">
12         <div class="col"></div>
13         <div class="col-8" style="background-color: #fff;">
14             <h3 class="page-title">{{ question.title }}</h3>
15             <p class="question-info">
16                 <span>作者: {{ question.author.username }}</span>
17                 <span>时间: {{ question.create_time }}</span>
18             </p>
19             <hr>
20             <p class="question-content">{{ question.content }}</p>
21             <hr>
22             <h4 class="comment-group-title">评论 ({{ question.answers|length }}) :
23             </h4>
24             <form action="{{ url_for("qa.answer", question_id=question.id) }}"
25                 method="post">
26                 <div class="form-group">
27                     <input type="text" placeholder="请填写评论" name="content"
28                         class="form-control">
29                     </div>
30                     <div class="form-group" style="text-align: right">

```

```

27         <button class="btn btn-primary">评论</button>
28     </div>
29 </form>
30
31 <div class="form-group">
32     {% for message in get_flashed_messages() %}
33         <div>
34             <div class="text-danger">{{ message }}</div>
35         </div>
36     {% endfor %}
37 </div>
38
39 <ul class="comment-group">
40     {% for answer in question.answers %}
41         <li>
42             <div class="user-info">
43                 
44                 <span class="username">{{ answer.author.username }}</span>
45                 <span class="create-time">{{ answer.create_time }}</span>
46             </div>
47             <p class="comment-content">{{ answer.content }}</p>
48         </li>
49     {% endfor %}
50 </ul>
51
52 </div>
53 <div class="col"></div>
54 </div>
55 {% endblock %}

```

- models.py

```

1 class AnswerModel(db.Model):
2     __tablename__ = "answer"
3     id = db.Column(db.Integer, primary_key=True, autoincrement=True)
4     content = db.Column(db.Text, nullable=False)
5     create_time = db.Column(db.DateTime, default=datetime.now)
6     question_id = db.Column(db.Integer, db.ForeignKey("question.id"))
7     author_id = db.Column(db.Integer, db.ForeignKey("user.id"))
8
9     question = db.relationship("QuestionModel", backref =
db.backref("answers", order_by=create_time.desc()))
10    author = db.relationship("UserModel", backref="answers")

```

搜索功能

- qa.py


```
35         </form>
36     </li>
37
38 </ul>
39 <ul class="navbar-nav ">
40     {% if user %}
41         <li class="nav-item ">
42             <span class="nav-link">{{ user.username }}</span>
43         </li>
44         <li class="nav-item">
45             <a class="nav-link" href="{{ url_for("user.logout") }}">退出登陆
46         </a>
47         </li>
48     {% else %}
49         <li class="nav-item ">
50             <a class="nav-link" href="{{ url_for("user.login") }}">登陆
51             <span class="sr-only">(current)</span></a>
52         </li>
53         <li class="nav-item">
54             <a class="nav-link" href="{{ url_for("user.register") }}">注册
55         </a>
56         </li>
57     {% endif %}
58 </ul>
59 </div>
60 </div>
61 </nav>
62 <div class="container">
63     {% block body %}
64     {% endblock %}
65 </div>
66 </body>
67 </html>
```