知乎论坛业务模型系统详细设计说明书

1. 系统功能概述

1. 访问知乎网站

用户可以通过浏览器访问知乎网站，主页展示了当前热门问题、推荐内容、最新动态，以及根据用户兴趣和历史浏览记录定制的个性化内容。除了这些，用户还可以看到各种专题、活动以及社交功能，如关注、消息和通知。

2. 用户登录或注册账号

对于首次访问知乎的用户，需要进行账号注册。用户点击注册按钮后，可以选择使用邮箱或手机号码注册账号，并设置一个安全的密码。已有账号的用户可以直接在登录界面输入他们的账号和密码进行登录。如果用户选择以游客身份浏览知乎，他们可以有限制地访问内容，但无法进行互动操作如点赞、评论或发布内容。

3. 发布问题

用户如果有问题需要解答，可以通过点击主页上的“提问”按钮进入提问页面。在提问页面，用户需要填写问题的标题和详细内容，完成这些步骤后，用户点击“发布”按钮提交问题。提交后的问题会展示在相关话题页面以及首页推荐区域，吸引其他用户的关注和回答。

4. 回答问题

用户在浏览知乎时，可以在问题详情页查看其他用户发布的问题。如果对某个问题有见解或解决方案，用户可以在页面下方的回答框中输入自己的回答内容。用户在回答前，可以查看其他用户的回答，此外，知乎还可能推荐一些高质量的回答和相关问题，帮助用户更好地理解问题背景并提供更有价值的回答。

5. 查看文章

知乎提供了专栏文章功能，主要供会员用户使用。会员用户可以查看和发布长篇文章，文章内容覆盖广泛，包括专业知识分享、个人经历、新闻评论、深度分析等。非会员用户如果想查看这些文章，需要通过充值或购买会员资格来获得访问权限。这种机制不仅提升了内容的质量，也鼓励用户成为会员，享受更多的特权和优质内容。

2.系统功能模块结构

前端功能

1.用户注册与登录

注册：新用户可以通过输入邮箱或手机号码及设置密码来注册账号。

登录：已有账号的用户可以通过输入账号和密码进行登录。

游客模式：允许用户在不注册或登录的情况下浏览部分内容，但无法进行互动操作如点赞、评论或发布内容。

2.问题的提出与回答

提出问题：用户可以点击“提问”按钮，填写问题的标题和详细内容.

回答问题：在问题详情页，用户可以输入回答内容。用户还可以浏览其他回答。

3.文章的发布与浏览

发布文章：会员用户可以在专栏中发布长篇文章。

浏览文章：会员用户可以浏览并阅读其他会员发布的文章，非会员用户则需充值或购买会员资格才能访问。

4.用户信息的查看与修改

查看信息：用户可以在个人主页查看自己的基本信息。

修改信息：用户可以更新个人资料，包括昵称、头像、简介、联系方式等。

5.问题的搜索

用户可以通过搜索框输入关键词，快速查找相关问题和文章。搜索功能支持模糊查询和分类筛选，帮助用户更高效地找到所需内容。

WEB 服务端

1.处理用户请求

接收来自前端的各种请求，包括注册、登录、提问、回答、发布文章、修改信息、搜索等操作。对请求进行身份验证和权限检查，确保用户合法且具备相应权限。

2.进行业务逻辑处理

根据用户请求，执行相应的业务逻辑，如用户注册时的验证、提问时的相关问题推荐、回答时的内容审核、文章发布时的分类和标签管理等。对于涉及敏感内容或违规行为的操作进行拦截和处理，维护平台的安全和秩序。

3.与数据库交互

将用户操作结果持久化到数据库中，如保存注册信息、存储提问和回答内容、更新用户资料等。从数据库中读取所需数据并返回给前端，如用户信息、问题和回答列表、文章内容等。

数据库端

1.存储用户数据

保存用户的基本信息、账号凭证、个人资料、历史活动记录等，确保数据的安全性和隐私保护。

2.存储问题数据

记录用户提出的所有问题，包括问题的标题、内容、分类、关联话题、发布时间等，便于后续的检索和展示。

3.存储回答数据

存储用户对各类问题的回答内容，包括回答的文字、图片、视频、链接等，以及点赞、评论和分享等互动数据。

4.存储会员数据

记录会员用户的信息和会员资格的有效期、权限等，支持会员功能的管理和使用。

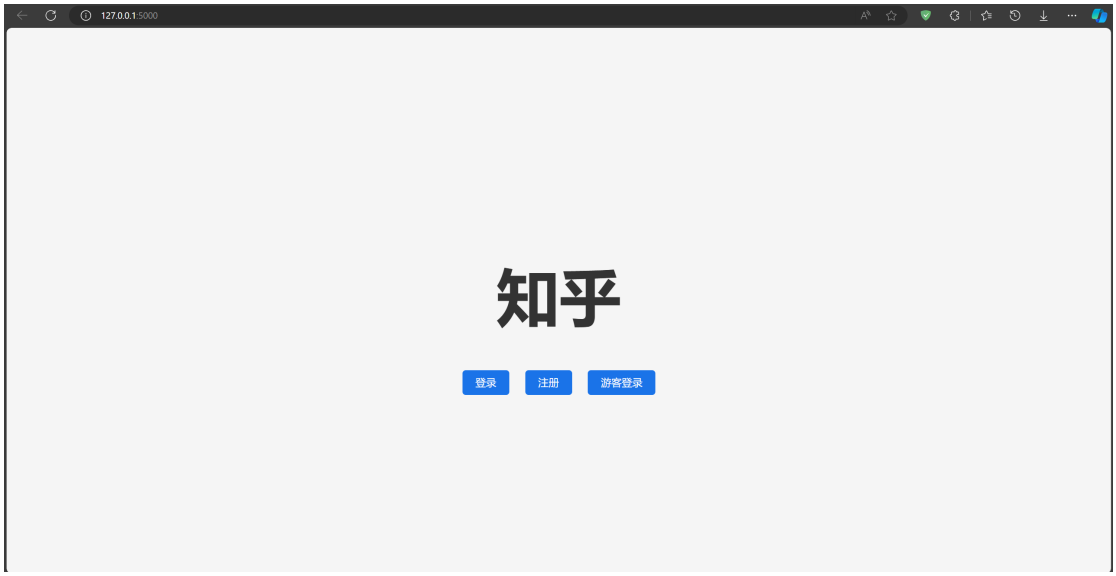
5.存储文章数据

保存会员用户发布的文章内容，包括文章的标题、正文、封面图片、多媒体内容、分类、标签等，确保文章的完整性和可访问性。

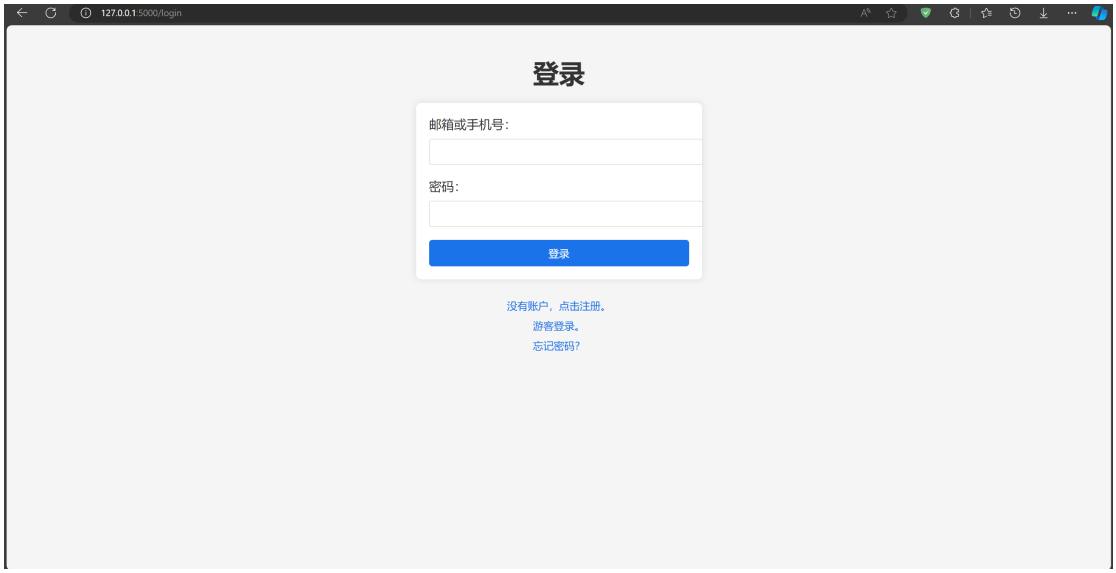
3.系统界面设计

用户注册与登录界面

index.html:



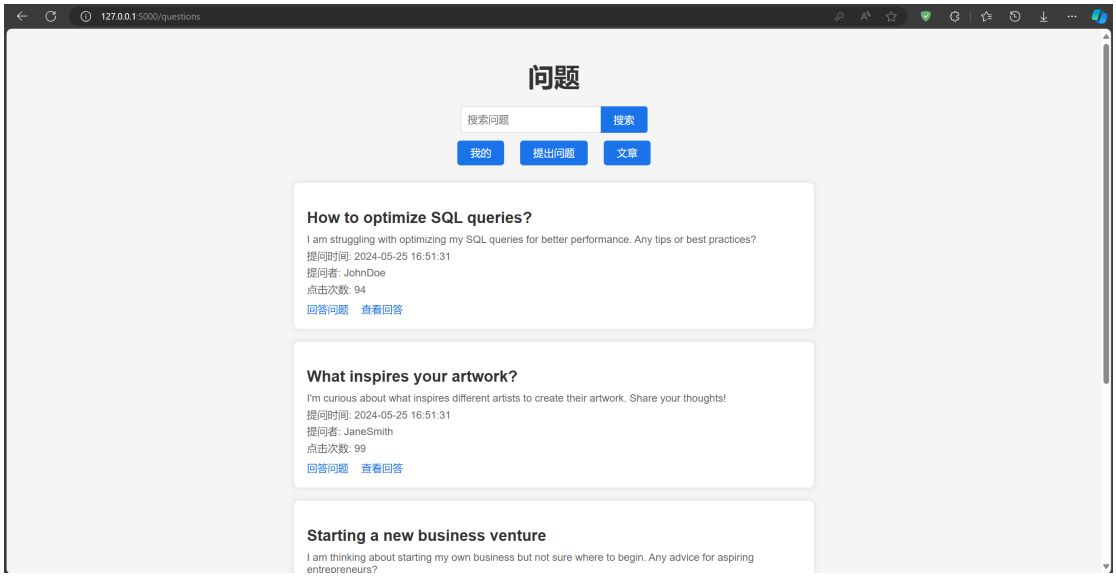
login.html:



register.html:

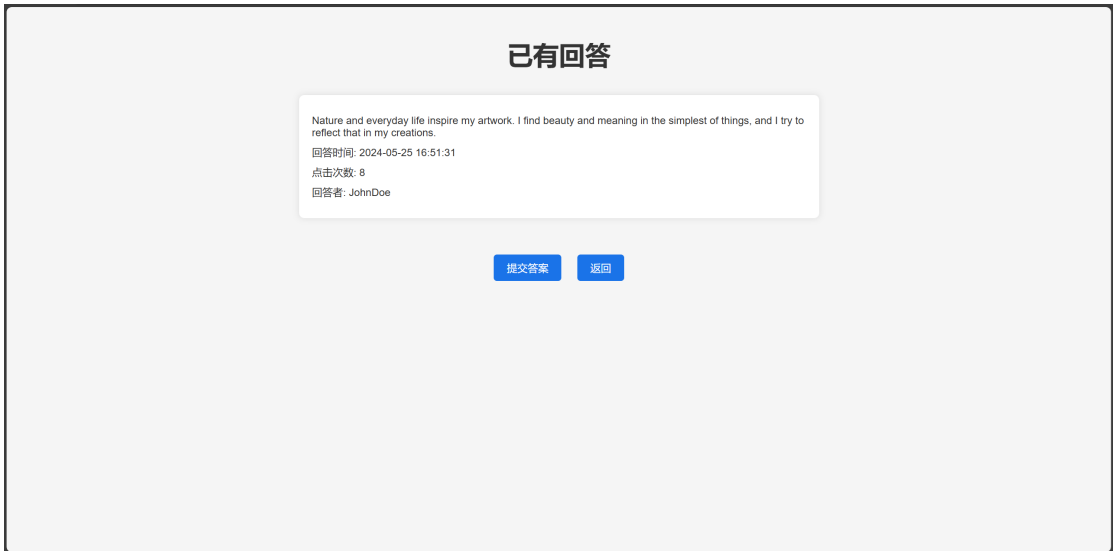
问题浏览界面

questions.html:



问题详情与回答界面

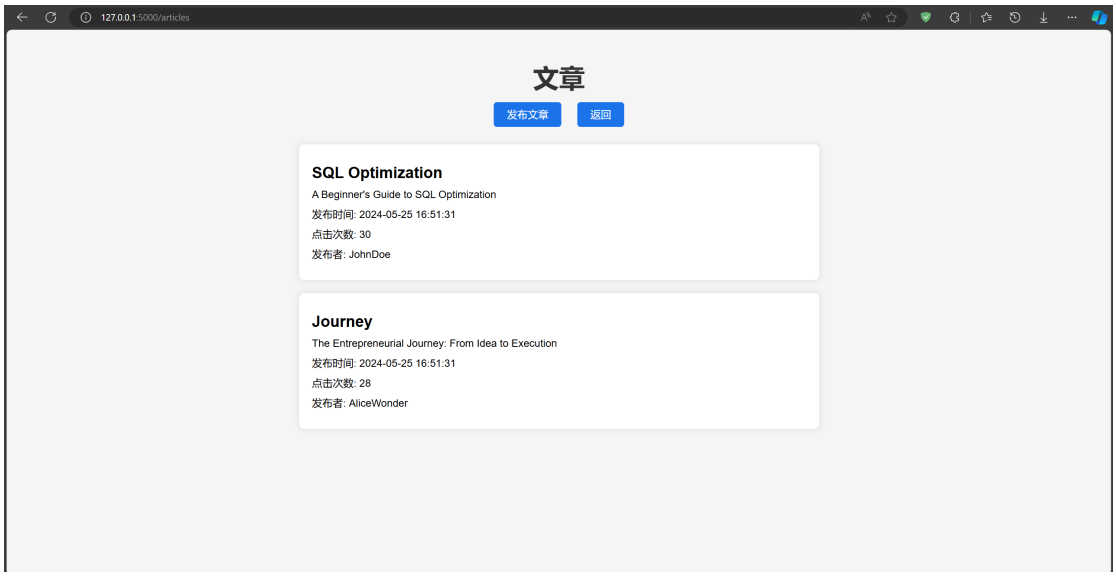
view_answers.html:



answer.html:

文章浏览与发布界面

articles.html:



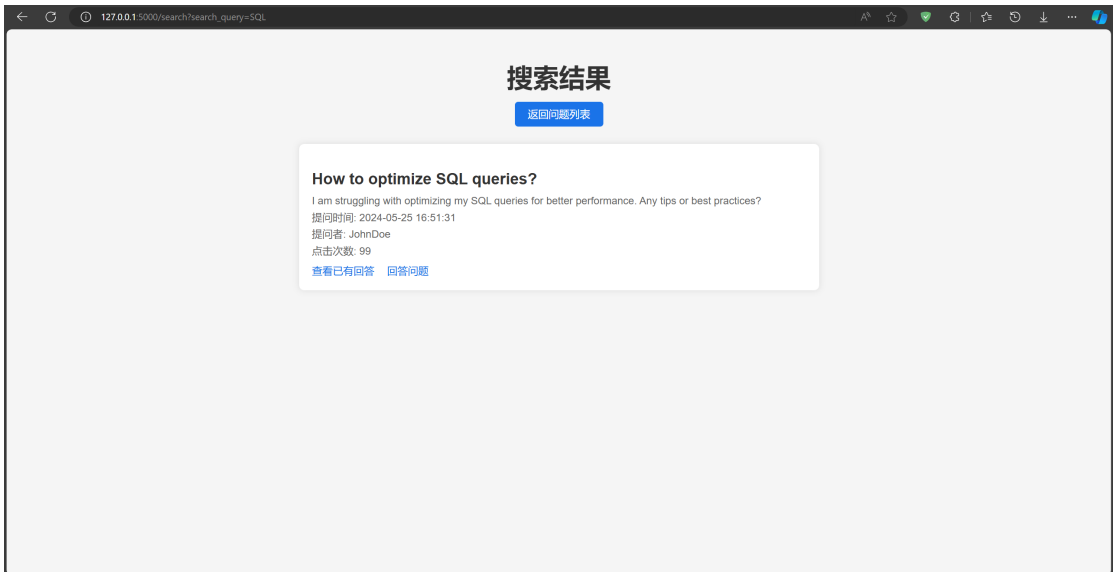
publish_article.html:

用户信息查看与修改界面

profile.html:

搜索功能界面

search_results.html:



4. 系统物理模型

表：用户表、问题表、回答表、会员表、文章表

表名		用户表（User）			
数据库用户		root			
主键		用户账号 ID (UserID)			
外键		无			
排序字段		用户账号 ID (UserID)			
索引字段		无			
字段名称	数据类型	允许为空	唯一	默认值	约束条件
用户账号 ID	Int	N	Y	无	主键
用户个人简介	VARCHAR(255)	Y	N	无	无
用户昵称	VARCHAR(50)	N	N	无	无
用户性别	ENUM ('male', 'female', 'other')	N	N	无	取值范围 ('male', 'female', 'other')
用户年龄	Int	Y	N	无	CHECK (age >= 0 AND age <= 100)
用户密码	VARCHAR(50)	N	N	无	无
用户教育经历	VARCHAR(255)	Y	N	无	无

用户职业经历	VARCHAR(255)	Y	N	无	无

表名		问题表（Question）			
数据库用户		root			
主键		问题编号 (QuestionID)			
外键		提问者用户 ID (AskerUserID)			
排序字段		问题编号 (QuestionID)			
索引字段		无			
字段名称	数据类型	允许为空	唯一	默认值	约束条件
问题编号	Int	N	Y	无	主键
提问者用户 ID	Int	N	N	无	外键
发布时间	Datetime	N	N	无	无
问题内容	Text	N	N	无	无
点击次数	Int	Y	N	0	无
问题标题	Varchar	N	N	无	无

表名		回答表（Answer）			
数据库用户		root			
主键		答案编号 (AnswerID)			
外键		回答者用户 ID (ResponderUserID)			
排序字段		答案编号 (AnswerID)			
索引字段		无			
字段名称	数据类型	允许为空	唯一	默认值	约束条件
答案编号	Int	N	Y	无	主键
回答时间	Datetime	N	N	无	无
回答内容	Text	N	N	无	无
回答者用户 ID	Int	N	N	无	外键
被查看次数	Int	Y	N	0	无

表名		会员表（Member）			
数据库用户		root			
主键		(member_id)			
外键		(user_id)			
排序字段		(member_id)			
索引字段		无			
字段名称	数据类型	允许为空	唯一	默认值	约束条件
member_id	Int	N	Y	无	主键
user_id	Int	N	Y	无	外键
remaining_time	Int	N	N	无	无

表名	文章表（articles）
数据库用户	root

主键		(article_id)			
外键		(publish_member_id)			
排序字段		(article_id)			
索引字段		无			
字段名称	数据类型	允许为空	唯一	默认值	约束条件
article_id	Int	N	Y	无	主键
publish_member_id	Int	N	Y	无	外键
article_content	Int	N	N	无	无
submit_time	DATETIME	N	N	无	无
click_count	Int	N	N	0	无

```sql
-- 创建用户表
CREATE TABLE users (
    user_id INT AUTO_INCREMENT PRIMARY KEY,
    nickname VARCHAR(50) NOT NULL UNIQUE,
    email_or_phone VARCHAR(50) NOT NULL UNIQUE,
    password VARCHAR(50) NOT NULL,
    gender ENUM('male', 'female', 'other') NOT NULL,
    age INT CHECK (age >= 0 AND age <= 100) NOT NULL,
    profile VARCHAR(255),
    education VARCHAR(255),
    occupation VARCHAR(255)
);

-- 创建问题表
CREATE TABLE questions (
    question_id INT AUTO_INCREMENT PRIMARY KEY,
    asker_user_id INT,
    publish_time DATETIME,
    question_title VARCHAR(255) NOT NULL,
    question_content TEXT,
    click_count INT DEFAULT 0,
    FOREIGN KEY (asker_user_id) REFERENCES users(user_id) ON DELETE
CASCADE
);

-- 创建回答表
CREATE TABLE answers (
    answer_id INT AUTO_INCREMENT PRIMARY KEY,
    question_id INT,
    responder_user_id INT,
    answer_time DATETIME,
    answer_content TEXT,
    click_count INT DEFAULT 0,
```

```
    FOREIGN KEY (responder_user_id) REFERENCES users(user_id) ON DELETE
CASCADE,
    FOREIGN KEY (question_id) REFERENCES questions(question_id) ON
DELETE CASCADE
);

-- 创建会员表
CREATE TABLE members (
    member_id INT AUTO_INCREMENT PRIMARY KEY,
    user_id INT UNIQUE,
    remaining_time INT,
    FOREIGN KEY (user_id) REFERENCES users(user_id) ON DELETE CASCADE
);

-- 创建文章表
CREATE TABLE articles (
    article_id INT AUTO_INCREMENT PRIMARY KEY,
    publish_member_id INT,
    article_title TEXT,
    article_content TEXT,
    submit_time DATETIME,
    click_count INT DEFAULT 0,
    FOREIGN KEY (publish_member_id) REFERENCES members(member_id) ON
DELETE CASCADE
);

-- 插入用户数据
INSERT INTO users (nickname, email_or_phone, password, gender, age,
profile, education, occupation) VALUES
('JohnDoe', 'john@example.com', 'password123', 'male', 28, 'A software
engineer with a passion for technology.', 'Bachelor of Computer
Science', 'Software Engineer'),
('JaneSmith', 'jane@example.com', 'pass456', 'female', 25, 'An aspiring
artist exploring the world through creativity.', 'Bachelor of Fine
Arts', 'Artist'),
('AliceWonder', 'alice@example.com', 'abc123', 'female', 35, 'An
adventurous soul seeking new experiences and challenges.', 'Master of
Business Administration', 'Entrepreneur');

-- 插入问题数据
INSERT INTO questions (asker_user_id, publish_time, question_title,
question_content, click_count) VALUES
```

```
(1, NOW(), 'How to optimize SQL queries?', 'I am struggling with
optimizing my SQL queries for better performance. Any tips or best
practices?', 10),
(2, NOW(), 'What inspires your artwork?', 'I''m curious about what
inspires different artists to create their artwork. Share your
thoughts!', 15),
(3, NOW(), 'Starting a new business venture', 'I am thinking about
starting my own business but not sure where to begin. Any advice for
aspiring entrepreneurs?', 8);

-- 插入回答数据
INSERT INTO answers (question_id, responder_user_id, answer_time,
answer_content, click_count) VALUES
(1, 3, NOW(), 'One way to optimize SQL queries is to ensure proper
indexing on frequently queried columns. Additionally, avoiding
unnecessary joins and using efficient WHERE clauses can help improve
performance.', 5),
(2, 1, NOW(), 'Nature and everyday life inspire my artwork. I find
beauty and meaning in the simplest of things, and I try to reflect that
in my creations.', 7),
(3, 2, NOW(), 'Starting a new business can be daunting but also
exciting. My advice would be to start with a solid business plan,
conduct market research, and seek mentorship from experienced
entrepreneurs.', 12);

-- 插入会员数据
INSERT INTO members (user_id, remaining_time) VALUES
(1, 30),
(3, 20);

-- 插入文章数据
INSERT INTO articles (publish_member_id, article_title,
article_content, submit_time, click_count) VALUES
(1, 'SQL Optimization', 'A Beginner''s Guide to SQL Optimization',
NOW(), 20),
(2, 'Journey', 'The Entrepreneurial Journey: From Idea to Execution',
NOW(), 18);
```

5. 系统安全体系设计

角色划分：

1. 普通用户

权限：浏览内容、提问、回答问题、点赞、评论、收藏、查看公开文章。

限制：无法发布长篇文章、无法查看会员专属内容、无法进行高级数据分析和管理。

2.会员用户

权限：拥有普通用户的所有权限，外加发布长篇文章、查看和评论会员专属内容、获得系统推荐优先权。

限制：无法进行系统管理和维护操作。

定期备份数据库：

策略：进行数据备份，确保在发生数据丢失或损坏时能够及时恢复。

恢复机制：提供自动化的恢复脚本，可以根据需要选择恢复某一天的备份数据，并在恢复前进行数据一致性检查。

6.系统运行环境设计与部署结构

运行环境：Python、Flask、MySQL、HTML/CSS/JavaScript

环境的主要配置信息为

```
Flask==2.3.2
PyMySQL==1.0.2
```

部署结构：使用 Flask 作为后端框架，以及 pymysql 模块来连接 MySQL 数据库。同时通过在 HTML 文件中附加部分 python 代码的方式减少了 python 程序的代码量。

7．源代码列表及说明

zhihu.py：主应用程序文件，定义了路由和处理函数

templates 目录：存放 HTML 模板文件

requirements.txt：存放 python 环境的主要版本信息

create_table.sql：存放建表语句和插入数据语句的脚本文件