课设题目

数据库原理课程设计报告

学　　号　**2152809**

姓　　名　曾崇然

专　　业　计算机科学与技术

授课老师　关佶红老师

目录

摘 要 随着日本决定将福岛核电站的处理水排放入海，公众对全球海洋环境和食品安全的担忧显著增加。为此，我开发了一个海洋核污染和食品安全查询平台，旨在提供一个用户友好的界面，通过集成污染信息查询、公告发布、食品信息查询以及智能辅助功能，来增强公众对海洋核污染事件的了解和食品安全的监控。该平台利用网络技术和数据库管理系统，能够实时更新和展示受污染区域的数据，包括污染程度、影响范围及相关食品安全通报。此外，智能辅助功能能够通过用户的自然描述来完成信息的查询和组件的动态渲染显示任务，帮助用户更有效地获取所需信息。我们的研究和开发工作不仅展示了技术在环境监测与公共健康领域的应用潜力，也为政府和民众提供了一个强大的决策支持工具，以应对可能的环境与健康危机。本次实验报告为最终报告设计，包含需求分析以及可行性分析、概念设计、逻辑设计、具体实现、成果展示和项目总结……

关键词：数据库；核污染；食品安全

一.概述

1.1 课题背景

在全球化的今天，环境问题已成为跨国界、影响深远的挑战。特别是海洋环境污染，由于其复杂性和广泛性，对生态系统和人类生活产生了严重影响。2021 年，日本政府宣布计划从福岛第一核电站释放经过处理的含放射性物质的废水入海，此举引发了全球范围内对海洋环境安全和食品安全的关切。放射性物质的扩散不仅威胁海洋生物的健康，还可能通过食物链影响到人类食品的安全，尤其是海鲜产品。

随着环境污染事件的增多，公众对于环境信息的透明度和实时性要求日益增高。然而，现有的信息发布通道常常存在信息更新不及时、不够透明或难以获取的问题。这不仅影响了人们对环境风险的正确评估，也加剧了公众的不安情绪。

在这样的背景下，开发一个集污染信息查询、公告发布、食品安全信息查询和智能辅助服务于一体的平台显得尤为重要。该平台将采用网络技术和数据库技术，不仅能提供环境污染数据和食品安全警报，还为用户提供通过自然描述查询数据的功能，极大地提升信息的可获取性和透明度。通过这一平台，我们希望能够为政府、研究机构、企业及公众提供一个可靠的环境监测和食品安全评估工具，以更好地应对环境危机和保护公共健康。

1.2 编写目的

本文档旨在全面指导和记录“海洋核污染与食品安全情况查询平台”项目的开发过程。作为项目唯一的开发者，本文档的编写目的包括但不限于以下几点：

- 明确项目目标和需求：通过需求分析章节，详细阐述平台的功能需求和用户期望，确保开发过程中的目标明确和需求具体。
- 评估项目可行性：通过可行性分析，评估技术实现的可能性、经济效益以及项目的社会影响，为开发决策提供科学依据。
- 规划设计和实现路径：通过概念设计和逻辑设计章节，详细规划平台的架构和组件设计，以确保开发过程有序进行。
- 优化资源和时间管理：项目管理章节将规划时间表和资源分配，帮助合理安排开发周期，确保按时完成项目。
- 提供项目总结和反思：通过项目实现和总结部分，记录开发过程中的关键成就和遇到的挑战，提供未来改进的方向。

本文档不仅是项目开发的指导书，也是个人学习和成长的记录。通过系统地编写和维护本文档，可以加深对项目管理和软件开发流程的理解，提高个人在项目规划、实施和问题解决方面的能力。

二.需求分析

2.1 功能需求

污染信息查询功能：

- 目的：提供实时的海洋核污染数据查询，包括污染源的地理位置、浓度值、浓度等级等信息。
- 用户交互：用户可以通过地图界面选择特定区域或特定日期进行查询。
- 预期结果：系统将展示所查询区域的详细污染信息，包括但不限于污染物浓度和趋势图等。

公告功能：

- 目的：发布关于海洋核污染及食品安全的最新公告和警报。
- 用户交互：用户可以在平台首页浏览最新的公告和警报。
- 预期结果：用户能及时获取官方及权威机构的最新信息，包括预防措施和食品安全指南。

食品信息查询功能：

- 目的：提供受核污染影响区域内的食品安全信息查询，特别是海产品。
- 用户交互：用户可通过搜索具体的食品名称或扫描条形码查询食品安全信息。
- 预期结果：系统将提供该食品的安全级别、来源地、检测报告和消费建议。

智能辅助功能：

- 目的：为用户提供使用自然语言进行信息查询的方式。。
- 用户交互：用户输入自然语言的查询请求，页面上显示动态渲染的组件。
- 预期结果：用户将可以使用更加便捷自由的方式获取到想要的信息。

2.2 数据字典

数据流字典：

名称	来源	去向	说明
原始数据	管理员输入	数据处理模块	管理员输入的原始的污染和产品信息
处理后的数据	数据处理模块	存储数据的数据库	将原始信息进行处理之后的处理数据信息
更新信息	管理员	更新信息处理模块	管理员向系统发出的申请更新数据库数据的信息
处理后的更新信息	更新信息处理模块	数据库	经过校验和处理的管理员的更新信息
查询请求	用户	请求处理模块	用户通过与前端交互产生的信息查询请求

查询语句	请求处理模块	数据库	请求处理模块将用户的原始的请求根据逻辑转化成插叙语句与数据库进行交互
渲染后的数据	数据处理和渲染模块	用户	从数据库拿到的数据金国前端的处理和渲染返回给用户进行显示

数据存储字典：

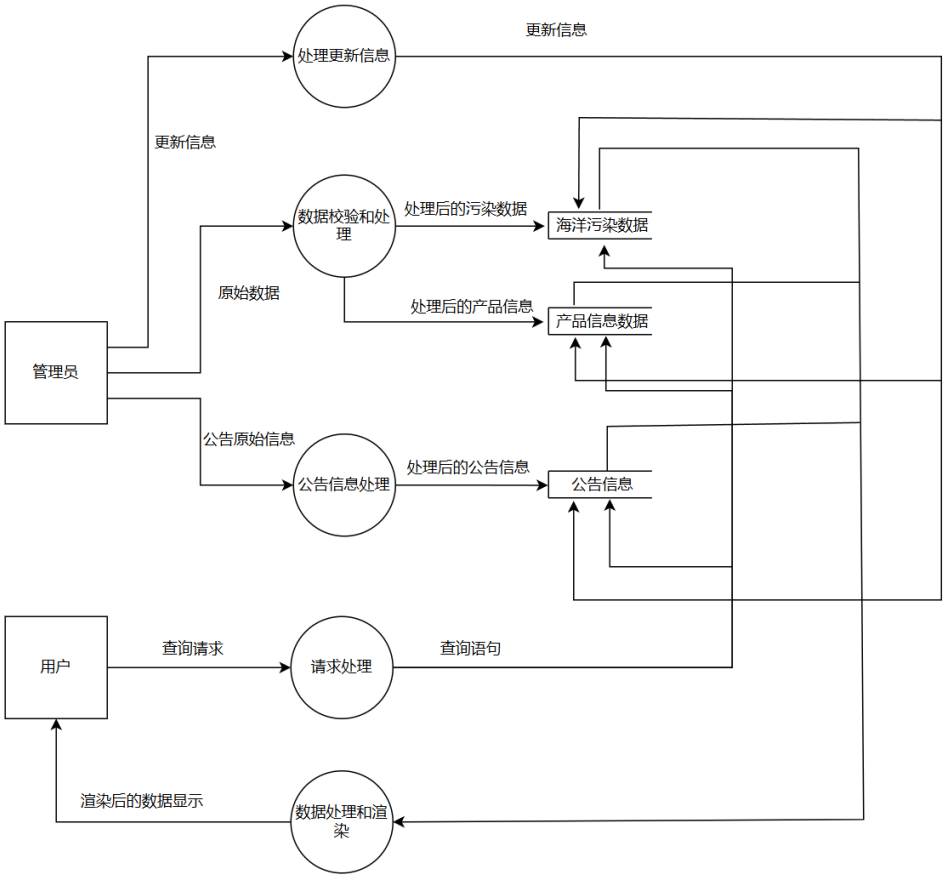
名称	内容	说明
海洋污染数据	海洋的辐射程度，位置	搜集到的海洋的污染信息数据，包含了具体经纬度的海洋的辐射程度值
产品信息数据	登记在内的海产品的信息数据	包含海产品的种类、生产者，产地坐标，量，状态，生产和过期时间等信息
公告信息	管理员发布的公告的信息	包含公告图片的 URL，公告标题，内容，优先级，展示时间等

数据处理字典：

名称	来源	处理逻辑	输出数据流
处理更新信息	管理员	将管理员的更新信息处理校验为内部格式	处理后的更新信息
数据校验和处理	管理员	将管理员输入的原始数据进行格式化，产生内部形式的数据	处理之后的内部数据
请求处理	用户	将用户的查询信息的请求转换为和与数据库打交道的 ORM 或者 sql 语句	格式化的数据库查询语句
数据渲染和处理	后端查询到的数据信息	获取查询到的信息，动态渲染组件进行显示	可视化的污染或者食品信息

2.3 数据流图

用户和管理员作为外部实体，通过信息检索和信息更新等过程与系统交互，以进行数据的录入、查询和更新。所有这些数据流动都通过一个中央数据库进行管理，确保数据的存储和访问。流图展示了从外部实体到系统内部过程的数据流向，以及在各个过程和数据库之间的互动。



三.可行性分析

3.1 技术可行性

- **技术资源**

  1）软件和开发工具

  数据库技术：使用如 Mysql 这样的数据库来存储和管理污染数据和食品安全信息，支持大数据量的高效查询。

  前端技术：使用 django 的模板语法，用 bootstarp3 的组件和样式，使用 chart.js 进行图表的绘制和分析。

  后端技术：使用 python 的后端框架 Django 实现后台的服务以及数据的管理

  2）硬件资源

  需要高性能的服务器来处理数据的存储和查询，特别是在处理大量实时数据时。

  3）第三方服务

  考虑集成第三方 API，如 Google Maps API、百度地图或者谷歌地图用于地图服务，百度或者 chatGPT 的 api 用于提供自然语言处理的服务。

- **技术能力**

  1）web 技术方面：

大二下学期选修了 web 技术，对 web 前后端开发的基本知识有一些了解，实践过一些简单的 web 项目，具有基本的前后端 web 应用开发的能力，使用过 django 框架进行 web 的开发。

2）数据库应用方面：

学习了数据库的理论以及应用方面的相关知识，能够进行数据库和数据表的设计和创建，能够使用 sql 语句对数据表进行操作，能够创建用户并进行权限的分发和管理等。

3）自然语言处理方面：

对 chatGPT 和文心一言等大模型有过一定的了解与接触，知道如何调取和使用其 api，用于完成自己项目内的任务。

- **技术解决方案的可行性**

1）数据处理

系统设计需要能够处理和分析大规模的数据集，以便实时更新污染和食品安全信息。

系统集成

确保不同技术组件之间可以有效集成，例如前端界面与后端服务器的交互，以及数据库的调用。

2）用户界面

用户界面需要是用户友好的，尤其是地图界面和自然语言查询界面，以提供流畅的用户体验。

- **安全性和合规性**

1）数据安全

实施加密和安全协议来保护敏感数据，特别是用户的查询记录和个人信息。

2）合规性

确保所有数据处理活动符合数据保护法规，如 GDPR 或其他地区的法规。

- **结论**

基于上述分析，该项目在技术上是可行的，但需要确保有足够的技术资源，特别是自然语言处理和数据管理领域的专业知识是该项目成功的关键。同时，系统的性能和安全性也是需要特别关注的领域。

3.2 应用可行性

对于“海洋核污染与食品安全情况查询平台”项目，应用可行性分析主要考虑项目在实际应用场景中的实用性、市场需求、用户接受度以及潜在的社会和经济影响。以下是针对这些方面的分析：

- **市场需求与用户接受度**

市场需求：全球对环境保护和食品安全的关注持续增加，特别是在日本福岛核电站处理水排放引发的公众担忧之后，对于相关信息的需求显著提高。因此，市场对此类信息查询平台的需求较大。

用户接受度：目标用户群可能包括普通消费者、环保组织、政府监管机构以及食品安全监测机构。对于这些用户来说，一个能提供实时数据、方便查询的平台将具有很高的吸引力。用户可能对使用新技术（如自然语言查询）感到新奇和兴奋，这有助于提高用户接受度和平台的使用频率。

- **实用性和便利性**

功能实用性：通过提供实时的核污染数据、食品安全信息和动态公告，平台能够满足用户在日常生活和专业工作中对准确信息的需求。自然语言查询功能可以极大提升用户体验，

使非技术用户也能轻松获取所需信息。

操作便利性：界面设计的简洁性和直观性是决定用户体验的关键因素。一个易于导航且响应迅速的界面将使用户更愿意频繁使用该平台。

- **社会和经济影响**

社会影响：该平台通过提供关键的环境和食品安全信息，有助于提高公众的环境意识和食品安全意识，促进健康生活。对于政府和环保机构来说，这样的平台可以作为监测和响应环境事件的工具，帮助制定更有效的环保政策和应急措施。

经济影响：对于相关产业，如渔业和海产品市场，提供污染信息和食品安全数据可以帮助企业及时调整生产和销售策略，减少经济损失。该平台的开发和维护将创造技术和管理职位，对经济有直接的正面影响。

- **可持续性和扩展性**

项目的可持续性需要考虑长期的数据更新、维护成本以及技术升级。

平台设计需要考虑未来扩展功能或集成新技术的可能性，以适应不断变化的市场需求。

结论

综合上述分析，该项目在应用层面是可行的，具有较高的市场需求和社会价值。但成功实施该项目需要关注用户体验的优化、持续的市场推广以及长期的技术支持和更新。如果能够确保这些条件，该项目将对公众、企业和政府等多方面产生积极的影响。

四.概念设计

4.1 实体

- **用户实体**

  存储用户身份信息的相关数据
- **污染信息实体**

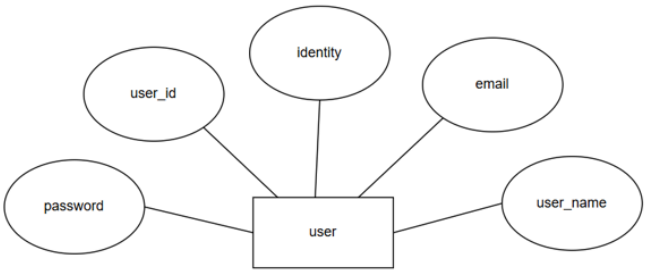
  存储污染信息的位置和辐射程度等相关信息
- **水产品信息实体**

  存储水产品信息的类型，质量，产出位置生产者，生产时间和过期时间的实体
- **公告信息实体**

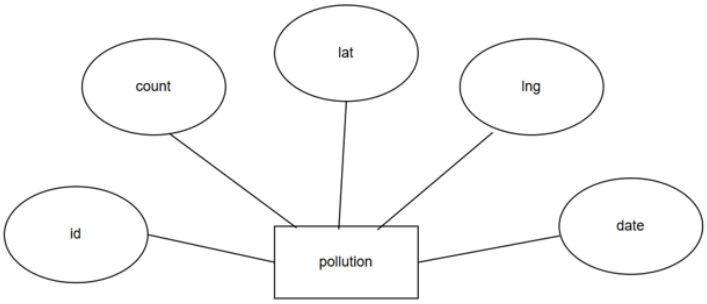
  存储管理员发布的公告信息的实体

4.2 实体属性局部 E-R 图

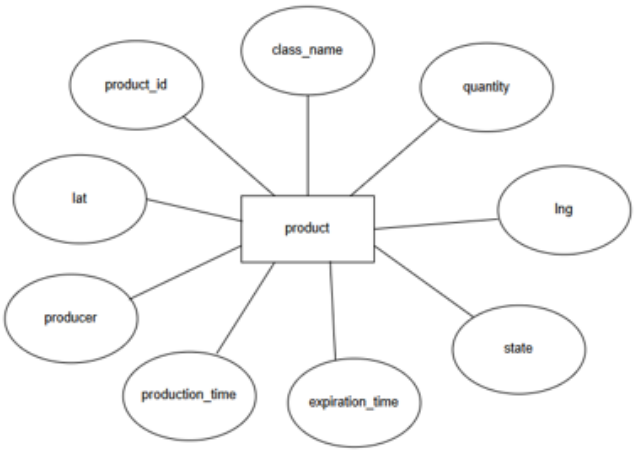
- **用户实体 E-R 图**



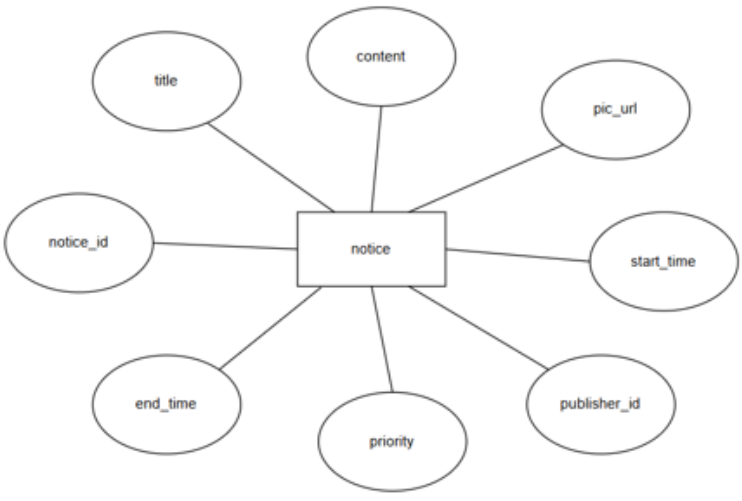
- **污染信息实体 E-R 图**



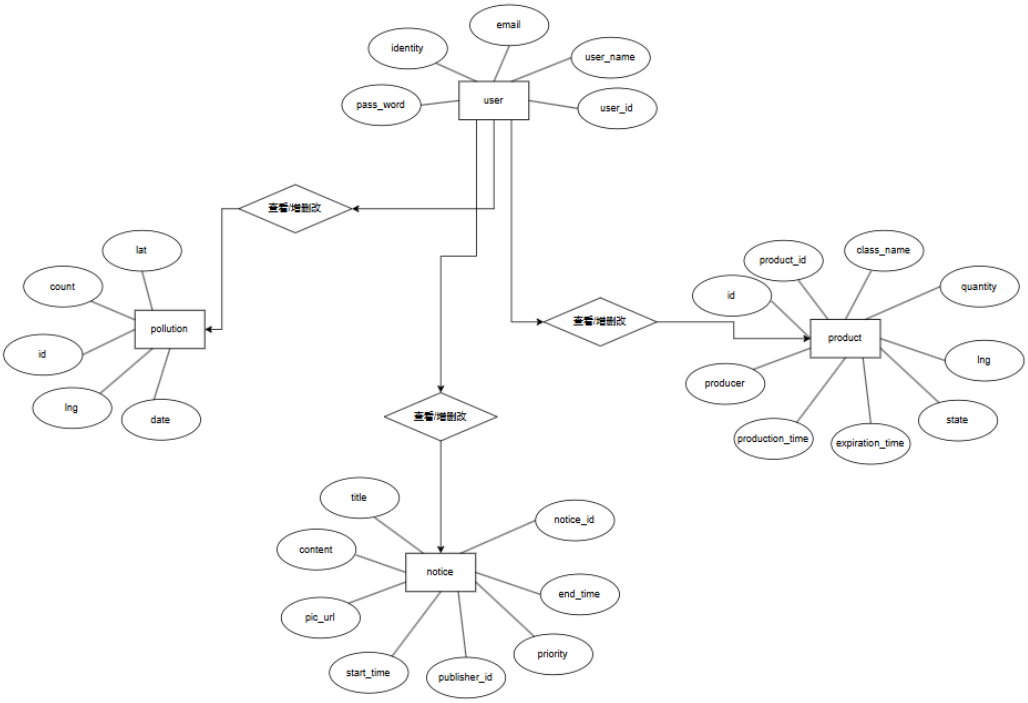
- **水产品信息实体 E-R 图**



- **公告信息实体 E-R 图**



4.3 全局 E-R 图



五.逻辑设计

5.1 E-R 图向关系模型的转变

- **水产品** = （产品 id、质量、经度、纬度、日期、种类名、生产者、生产时间、过期时间）
- **公告** = （公告 id、标题、内容、图片 URL、开始时间、结束时间、优先级、发布者 id）
- **污染** = （数据 id、污染值、经度、纬度、日期）
- **用户** = （用户 id、用户身份、用户名、密码、电子邮箱）

将一对多关系中的一加入到了多中作为外键，如公告的发布者属性

5.2 数据模型的优化及规范化设计

该关系模型中没有多值属性，本身就为 1NF 范式；由于在现在的关系模型中所有的关系表都有且仅有一个对应的 id 属性作为主键，因此不存在部分依赖，是 2NF 范式；分析上面的关系表，发现每个表中除了主键之外任何两个属性之间不存在依赖关系，因此也就不存在传递依赖，故上面的关系模型本身就为 3NF 范式。

- **水产品** = （产品 id、质量、经度、纬度、日期、种类名、生产者、生产时间、过期时间）
- **公告** = （公告 id、标题、内容、图片 URL、开始时间、结束时间、优先级、发布者 id）
- **污染** = （数据 id、污染值、经度、纬度、日期）

• **用户** = （用户 id、用户身份、用户名、密码、电子邮箱）

六.项目管理

6.1 框架选择

由于本次课程设计为单人作业，全部工作由一人完成，因此我选择的是前后端不分离的开发方式，使用 django 框架，利用其模板语法进行前后端的开发。前端使用 bootstrap3 和 jquery 等来辅助开发，加快进程。

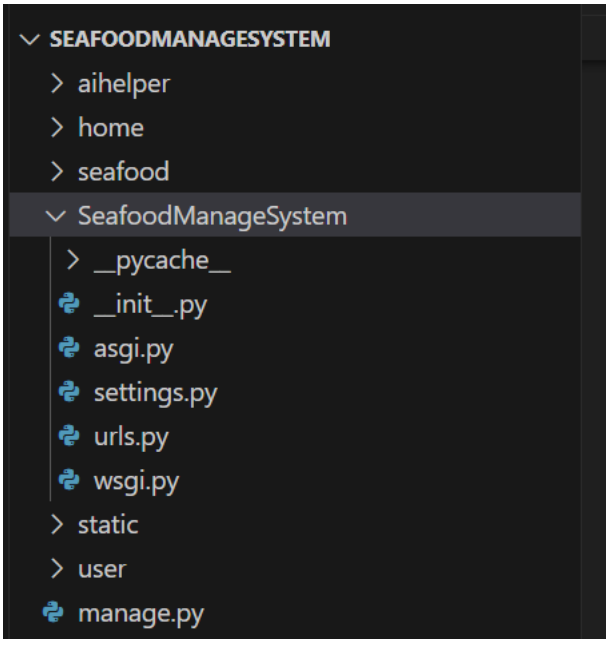
6.2 开发平台

1）操作系统：Windows11
2）开发语言：python3.12
3）集成开发环境：VSCODE
4）APIs：使用百度地图 API 作为查询和显示的交互，使用文心一言 API 作为自然语言处理的工具

七.系统实现

7.1 系统架构搭建

1）项目结构



项目根目录下具有如下几个文件或者文件夹：

manage.py：使用 django 进行开发和管理的工具脚本

SeafoodManageSystem：该文件夹下包含了一些全局的文件，包括全局的路由分发，应用注册，配置信息等

static：该文件夹下面包含了一些静态的资源文件，包含 css 文件、js 文件、插件文件、图片等

user：该文件夹为登录注册和核验身份相关功能的 app，其中包含相应的路由管理、模板定义、视图函数以及模型（数据表）定义。

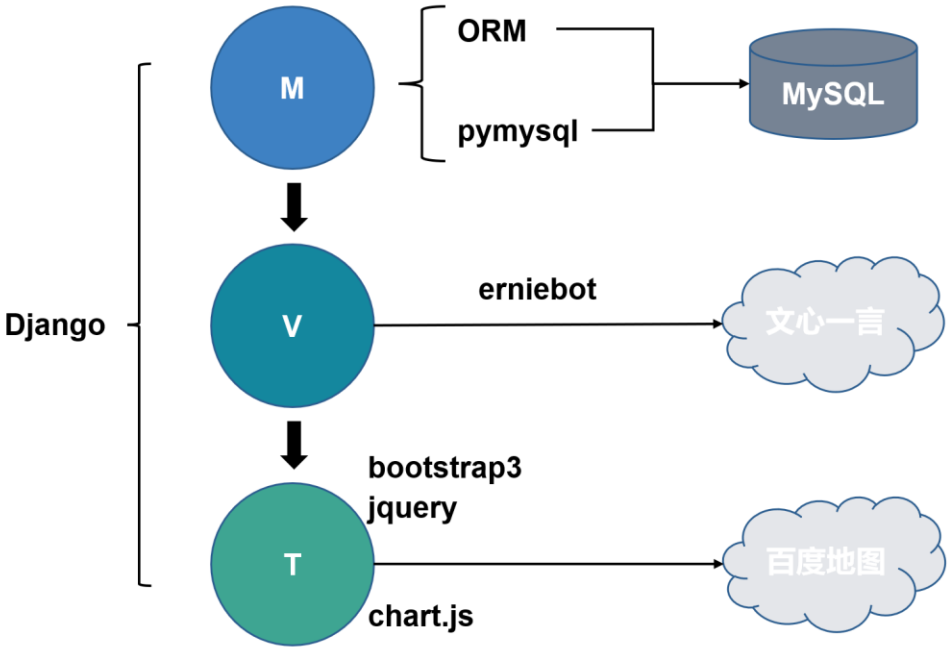
home：该文件夹为网站主页信息显示和查询相关功能的 app，其中包含相应的路由管理、模板定义、视图函数以及模型（数据表）定义。

seafood：该文件夹为水产品显示和查询和筛选相关功能的 app，其中包含相应的路由管理、模板定义、视图函数以及模型（数据表）定义。

aihelper：该文件夹为智能助手相关功能的 app，其中包含相应的路由管理、模板定义、视图函数以及模型（数据表）定义。

2）整体模型

整体的架构采用 MVT 模型，M 即模型层，负责数据表的定义和数据的增删改查，在本系统中使用了 Djano 封装的 ORM 和原生的 pymysql 两种方式进行数据库的交互；V 即视图层，负责应用逻辑的实现；T 即模板层，负责和用户进行交互，展示相应的信息。



7.2 系统逻辑设计

本系统的主要功能有以下内容：

1）污染信息的检索和查询：

使用百度地图 api 提供基本的交互，前端获取用户的交互请求，如日期的选择、地点的选择，传输到后台对应的视图函数，进行查询筛选后将信息返还到前端进行动态的渲染组件并显示。

2）海产品的查询和筛选：

用户在前端进行条件的筛选，发送筛选请求后对应的后端视图函数进行相应的查询，

将查询到的结果返还到前端进行动态的渲染和显示。并且根据海产的产地经纬度联查污染表，给出该产品产地的污染信息变化趋势。

3）公告信息的显示：

用户进入主页之后，后台根据公告的优先级，展示时间动态的选取至多 3 个公告在前端进行渲染和展示。

3）使用 ai 助手进行自然语言查询：

用户使用自然语言提出想要查询的信息，点击提交后在后台对内容进行包装，添加提示词，要求返回的格式，包装完成之后发送给文心一言大模型进行处理，文心一言根据提示词和要求的返回限制返回回复，回复为严格的 json 格式，视图函数收到回应之后根据 json 数据进行 sql 语句的构造和查询，最后将查询结果返还前端进行渲染。

4）后台管理：

将数据表注册在 Django 的 admin 系统中，登录管理员，即可使用 djano 的管理原系统进行数据的发布和删除和管理。

7.3 具体功能编写

1）登录注册

使用 django 的 captcha 添加验证功能

```
INSTALLED_APPS = [
    "django.contrib.admin",
    "django.contrib.auth",
    "django.contrib.contenttypes",
    "django.contrib.sessions",
    "django.contrib.messages",
    "django.contrib.staticfiles",
    "user",
    "home",
    "seafood",
    "aihelper",
    "captcha",
```

将用户的密码进行哈希，避免用户密码直接暴露可能出现的安全隐患

```
def hash_code(s, salt='Covid-19'):
    h = hashlib.sha256()
    s += salt
    h.update(s.encode())
    return h.hexdigest()
```

2）公告展示

根据优先级和展示时间动态的选取公告进行展示，逻辑如下：

```
now = datetime.now()
active_notices = Notice.objects.filter(end_time__gt=now).order_by('-priority')
if active_notices.count() < 3:
    remaining_count = 3 - active_notices.count()
    additional_notices = Notice.objects.order_by('end_time')[:remaining_count]
    active_notices = list(active_notices) + list(additional_notices)
```

3）根据时间查询污染

后端接收前端提交的时间信息进行查询，返回查询信息：

```
# 根据提供的日期查询辐射信息
radiation = OceanNuclearPollution.objects.filter(date=query_date)
radiation_json = serialize('json', radiation, fields=('count', 'lat', 'lng'))
radiation_data = [item['fields'] for item in json.loads(radiation_json)]

# 创建HTTP响应，返回JSON数据
response_data = {
    'identity': identity,
    'radiation': radiation_data
}
return HttpResponse(json.dumps(response_data), content_type="application/json")
```

4）根据地点查询污染

后台接收到前台的地点信息，筛选出对应地点的污染信息，排序后返回到前端进行渲染。

```
def changelocation(request):
    if request.method != 'POST':
        return HttpResponse("This endpoint accepts only POST requests.", status=405)
    lat = request.POST.get('latitude', 0)
    lng = request.POST.get('longitude', 0)
    lat = int(float(lat) / 2) * 2
    lng = int(float(lng) / 2) * 2
    response_data = OceanNuclearPollution.objects.filter(lat=lat, lng=lng).order_by('-date')[:10].values('count', 'date')
    response_data = list(response_data)
    response_data.reverse()
    return JsonResponse(response_data, safe=False)
```

5）水产列表展示和筛选

后台接收到前台的筛选信息之后进行水产的筛选，返回前端进行渲染，逻辑如下：

```
response_data = Product.objects.filter(class_name__icontains=product_type).order_by('-production_time')

if statusFilter == '全部':
    pass
elif statusFilter == '已售出':
    response_data = response_data.filter(state=1)
else:
    response_data = response_data.filter(state=0)

today = timezone.now().date()
if timeFilter == '全部':
    pass
elif timeFilter == '今天':
    response_data = response_data.filter(production_time__date=today)
elif timeFilter == '本周':
    start_of_week = today - timedelta(days=today.weekday())
    end_of_week = start_of_week + timedelta(days=6)
    response_data = response_data.filter(production_time__date__range=[start_of_week, end_of_week])
elif timeFilter == '本月':
    start_of_month = today.replace(day=1)
    end_of_month = today.replace(day=28) + timedelta(days=4)
    end_of_month = end_of_month - timedelta(days=end_of_month.day)
    response_data = response_data.filter(production_time__date__range=[start_of_month, end_of_month])
else:
    start_of_year = today.replace(month=1, day=1)
    end_of_year = today.replace(month=12, day=31)
    response_data = response_data.filter(production_time__date__range=[start_of_year, end_of_year])

response_data = response_data.filter()[:300]
response_data = serialize('json', response_data, fields=('class_name', 'quantity', 'lat', 'lng', 'production_time', 'expiration_time', 'state'))
response_data = json.loads(response_data)
response_data = [{'product_id': item['pk'], **item['fields']} for item in response_data]
response_data = json.dumps(response_data)

return render(request, 'foodpage.html', {'response_data': response_data, 'identity': identity})
```

6）水产详情展示

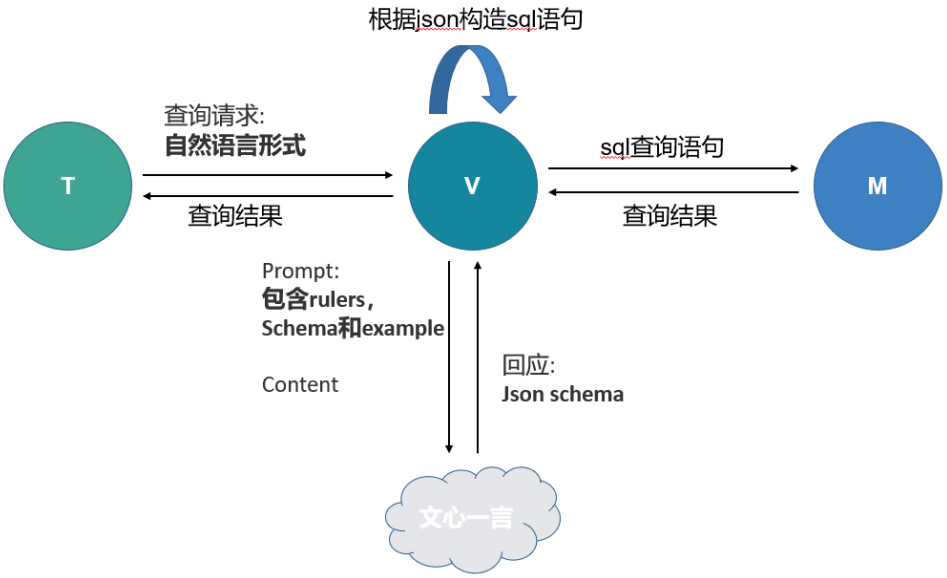
点击查看详情之后，前端路由携带水产的 id 到达后端，后端视图函数根据水产品的 id 进行查询，获取该产品的详细信息，并根据其产地的经纬度进行联查，获取到对应位置的污染程度变化趋势，返回前端进行显示：

```python
def product_detail(request, product_id):
    if request.session.get('is_login', None):
        user_name = request.session['user_name']
        user = User.objects.get(user_name=user_name)
        identity = user.identity
    else:
        identity = 0

    product = get_object_or_404(Product, pk=product_id)

    lat = product.lat
    lng = product.lng
    lat = int(float(lat) / 2) * 2
    lng = int(float(lng) / 2) * 2
    radiation = OceanNuclearPollution.objects.filter(lat=lat, lng=lng).order_by('-date')[:10].values('count', 'date')
    radiation = list(radiation)
    radiation.reverse()
    return render(request, 'product.html', {'product': product, 'identity': identity, 'radiation': radiation})
```

7）ai 助手



①定义与文心一言进行通信的包装类，使用其提供的 api 接口封装对外的简洁易用的功能，使用该类能够进行请求和响应的处理，获取到想要的格式。

定义为：

```python
class ErnieBotWrap():
```

方法包含：

def __init__(self):

def get_mes(role, dilog):

def set_promt(self, prompt_str):

def get_res(self, str_input, record=False, request_timeout=5):

def get_json_str(json_str:str):

def get_res_json(self, str_input, record=False, request_timeout=10):

以上方法包含了获取角色，设置提示词，获取回应和转变为 json 格式等方法。

②定义提示词基类，该基类描述了应对不同任务的提示词的通用属性和方法。

定义为：

```python
class PromptJson:
```

方法包含：

def __init__(self, rulers) -> None:

def json_obj(self):

def example(self):

def __call__(self, *args, **kwargs):

def set_scheame(self, json_obj):

def set_example(self, example_str:str):

def set_rulers(self, rulers):

以上方法包含了设置提示词的各种角度，包含规则、格式和样例等。

③定义用于将自然语言转化成数据库查询语句的提示词（集成 PromptJson）：

设置规则为：

```
def __init__(self) -> None:
    rulers = '''你是数据库查询程序，需要根据描述生成对应的json结果，结果是查询的一些属性和条件，如果有对应的描述就写入对应位置。
                严格按照下面的scheame描述生成给定格式json，只返回json数据，并且只返回其中的properties数据,不要其他内容：
            '''
    super().__init__(rulers)
```

设置返回格式为：

```
def json_obj(self)->dict:
    schema_attr = {'type': 'object',
            'properties':{
                'table': {'enum':['oceannuclearpollution', 'product'], 'description': '表的名称，查询污染是oceannuclea
                'pollution_select_type': {'enum':['time', 'location'], 'description': '查询的表是oceannuclearpollutic
                'pollution_date': {"type": 'string', "format": "date", 'description': '根据日期查询污染时对应的时间'},
                'lat': {"type": 'number', 'description': '根据地点查询污染的纬度'},
                'lng': {"type": 'number', 'description': '根据地点查询污染的经度'},
                'product_name': {"type": 'string', 'description': '查询产品时产品的名字'},
                'product_state': {"type": 'boolean', 'description': '水产品未售出为假，售出为真'},
                'product_date': {"type": 'string', "format": "date", 'description': '查询水产品时对应的日期'},
                'product_date_compare': {'enum':['=', '>', '<'], 'description': '查询水产品时对应的日期描述，在某一天为
                },
            "additionalProperties": False
        }
    return schema_attr
```

设置例子为：

```
def example(self)->str:
    example = '''正确的示例如下：
                查询维度为21.5，经度为144.7位置处的污染情况：```{'table': 'oceannuclearpollution', 'pollution_se
                查询2024年6月14日的污染情况：```{'table': 'oceannuclearpollution', 'pollution_select_type': 'ti
                查询叫蛏子的水产品：```{'table': 'product', 'product_name': '蛏子'}``` 。
                查询2024年6月14日生产的水产品：```{'table': 'product', 'product_state': '2024-06-14', 'product_
                查询2024年3月1日之后生产的售出水产品：```{'table': 'product', 'product_state': '2024-03-01', 'pr

            '''
    return example
```

④获取前端发送的自然语言查询请求

```
query = request.POST.get('query')
```

⑤使用上述类进行处理和包装，并获取文心一言的响应结果：

```
json_res = ernie.get_res_json(query)
```

⑥根据 json 数据中的值进行查询语句的构造：

```
if json_res['properties']['table'] == 'oceannuclearpollution':
    table_name = "home_oceannuclearpollution"
    fields = "*"
    if json_res['properties']['pollution_select_type'] == 'time':
        restriction = "date=" + "'" + json_res['properties']['pollution_date'] + "'"
        display[0] = 1
        sql = "SELECT " + fields + " FROM " + table_name + " WHERE " + restriction + ";"
    else:
        lat = int(float(json_res['properties']['lat']) / 2) * 2
        lng = int(float(json_res['properties']['lng']) / 2) * 2
        restriction = "lat=" + str(lat) + " and lng=" +  str(lng)
        display[1] = 1
        sql = "SELECT " + fields + " FROM " + table_name + " WHERE " + restriction + " ORDER BY date;"
```

```
table_name = "seafood_product"
fields = "product_id, class_name, quantity, lat, lng, production_time, expiration_time, state"
display[2] = 1
if 'product_name' in json_res['properties']:
    restriction1 = "class_name=" + "'" + json_res['properties']['product_name'] + "'"
else:
    restriction1 = ""
if 'product_state' in json_res['properties']:
    restriction2 = "state=" + str(int(json_res['properties']['product_state']))
else:
    restriction2 = ""
if 'product_date' in json_res['properties']:
    restriction3 = "DATE(production_time)" + json_res['properties']['product_date_compare'] + json_res['properties']['product_date']
else:
    restriction3 = ""
sql = "SELECT " + fields + " FROM " + table_name + " WHERE"
is_void = True
if restriction1 != "":
    sql = sql + " " + restriction1
    is_void = False
if restriction2 != "":
    if is_void == False:
        sql = sql + " AND"
    sql = sql + " " + restriction2
    is_void = False
if restriction3 != "":
    if is_void == False:
        sql = sql + " AND"
    sql = sql + " " + restriction3
sql = sql + " ORDER BY production_time DESC;"
```

⑥使用 pymysql 连接数据库进行查询

```
config = {
    'host': 'localhost',  # 数据库服务器地址
    'user': 'wenxin',  # 数据库用户名
    'password': [redacted],  # 数据库密码
    'database': 'seafoodmanagesystem',  # 数据库名
    'charset': 'utf8mb4',  # 字符编码
    'cursorclass': pymysql.cursors.DictCursor  # 使用字典游标，便于读取列名
}
connection = pymysql.connect(**config)
try:
    with connection.cursor() as cursor:
        cursor.execute(sql)
        results = cursor.fetchall()
        if display[0]:
            points = results
        elif display[1]:
            date_count = results[:10]
        elif display[2]:
            products = results[:300]
finally:
    connection.close()
```

⑦返回查询结果到前端进行渲染和显示

7.4 功能测试

1）登录注册

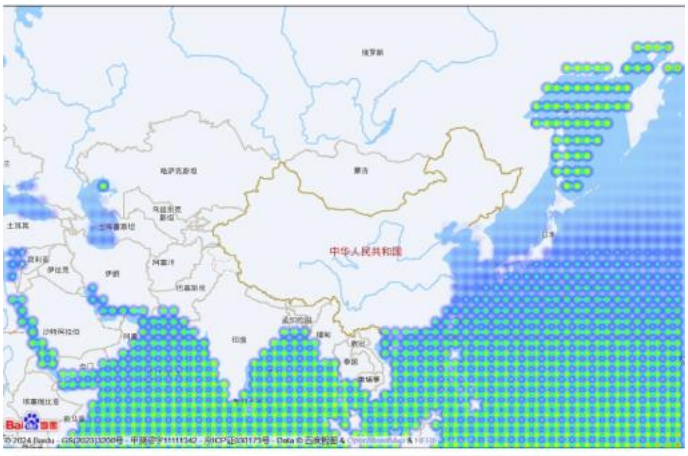
正确输入账号密码和验证码即可进行登录

2）公告展示

以轮播图的形式展示如下：

3）选择日期进行查询

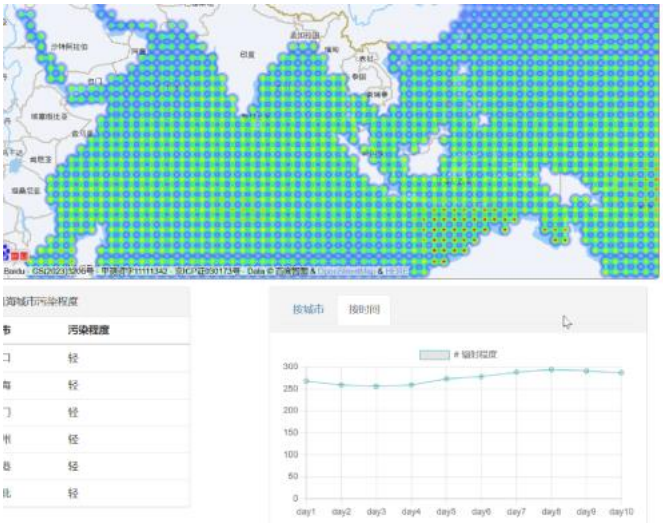
选择日期之后，污染信息展示在地图上，对应日期的城市污染情况也显示出来



城市	污染程度
海口	轻
上海	轻
厦门	轻
温州	轻
香港	轻
台北	轻

4）选择地点进行查询

点击地图上的地点，下方折线图显示对应地点一段时间之内的辐射变化值



5）海产筛选

选择条件，筛选出符合条件的海产

海产列表						
种类名称	数量 (kg)	经度	纬度	生产日期	过期日期	状态
海鲜7	62.168	40.453	110.961	2024-08-14T15:07:52Z	2033-05-17T01:55:37Z	未售出
海鲜36	9.782	21.956	112.758	2024-08-14T13:35:26Z	2031-05-21T12:47:55Z	已售出
海鲜37	32.326	19.352	128.106	2024-08-14T10:53:54Z	2028-03-30T00:32:05Z	未售出
海鲜57	82.280	25.340	124.134	2024-08-14T09:36:03Z	2025-08-04T02:45:39Z	已售出
蛏子	52.562	35.901	108.395	2024-08-14T06:03:01Z	2026-04-02T18:05:02Z	已售出
海鲜10	50.121	40.809	108.732	2024-08-14T00:52:50Z	2032-06-24T16:14:51Z	未售出
鳜鱼	78.295	29.578	124.365	2024-08-13T11:15:58Z	2029-03-10T06:57:08Z	已售出
海鲜47	96.367	15.234	124.807	2024-08-13T06:39:33Z	2028-06-27T20:55:09Z	已售出
鲋鱼	7.194	32.291	120.461	2024-08-12T23:27:30Z	2031-04-18T11:20:50Z	未售出
海鲜16	77.575	33.992	114.594	2024-08-12T21:26:27Z	2029-11-26T14:28:42Z	已售出
海鲜21	57.126	17.077	119.533	2024-08-12T20:35:14Z	2033-06-18T06:55:47Z	已售出

6）海产详情

点击列表中的某一项，显示出对应的信息以及产地辐射值的变化趋势

主 页 海 产 智能助手

产品详情

产品ID: 8474

产品类型: 海鲜32

质量（kg）： 53.49188951859

纬度: 25.23766275809224

经度: 133.663012088358

生产时间: June 10, 2024, 10:04 a.m.

过期时间: April 8, 2033, 5:12 a.m.

状态: 已售出

返回列表

7）ai 助手

输入自然语言，前端动态渲染组件显示查询到的结果，如下：

八.总结

本次数据库系统原理课程设计通过开发“海洋核污染与食品安全情况查询平台”，不仅实现了预期的功能目标，还为项目的进一步优化和扩展提供了宝贵的经验和基础。通过项目的实践，我在数据库设计、系统开发和项目管理等方面获得了以下几点心得和体会：

1）系统设计与实现的综合应用

在本项目中，我综合运用了前端和后端开发技术，结合了 Django 框架、Bootstrap 组件库、百度地图 API、文心一言 API 等工具，成功实现了从需求分析到系统上线的完整流程。这一过程不仅提高了我对 Web 开发技术的理解，也增强了我对系统设计和实现的综合能力。

2）数据库设计与优化

在数据库设计方面，通过 E-R 图和关系模型的转化，我学会了如何进行数据库的规范化设计，从 1NF 到 3NF 的优化过程，有效地避免了数据冗余和更新异常问题。这个过程让我对数据库设计的理论和实践有了更深入的理解和应用。

3）功能实现与用户体验

在功能实现过程中，特别是污染信息查询、公告展示、食品安全信息查询和智能辅助功能的实现，我深刻体会到了用户体验的重要性。通过友好的用户界面和高效的交互设计，提升了平台的易用性和用户满意度。

总体而言，本次课程设计不仅使我掌握了数据库系统的原理和应用，也锻炼了我的综合开发能力和项目管理能力，为我今后的学习和工作打下了坚实的基础。在未来的工作中，我将继续探索和应用所学知识，不断提升自己的技术水平和解决问题的能力，为实现更复杂和更具挑战性的项目做好准备。

参考文献

[1]段海燕,唐小娟,段志远,等.日本核污染水排海的生态环境损害及其赔偿机制[J].中国人口·资源与环境,2024,34(02):119-130.