铜陵学院

课程报告

学　　院：	建筑工程学院
专　　业：	22 地理信息科学
课程名称：	网络地理信息系统原理与技术
小　　组：	第一组
指导老师：	房家伟
开课学期：	2024 至 2025 学年第二学期

二〇二五年四月二十九日

小组分工

数据查找与处理	周华强、张恩杰
代码编写与调试	周涵、刘宇
报告撰写	许雪玉、操菲、刘宇
其他	贾奇奇

目录

一、系统背景与需求分析

1.1 实验背景

随着社会经济的不断进步和人民生活水平的显著提高，旅游业呈现出蓬勃发展的态势，随之而来的是人们对旅游信息的需求日益增长。在以往，人们获取旅游信息的方式主要依赖于纸质地图、导游的口头讲解等传统手段，但这些方式往往存在信息更新不够及时、携带不便、查询效率低下等明显问题。随着科技的发展，WebGIS 技术的出现为旅游信息的获取和管理带来了革命性的变化。基于 WebGIS 的旅游查询系统，能够为游客提供一个方便快捷的平台，让他们能够轻松获取最新的旅游信息，合理规划自己的行程，从而极大地提高了旅游体验的质量和满意度。

铜陵市作为安徽省内一个重要的旅游城市，它不仅拥有着丰富多彩的自然景观，还蕴藏着深厚的人文历史资源。为了进一步提升铜陵市旅游服务的智能化水平，确保游客能够快速而便捷地获取所需的旅游信息，设计并开发一个基于 OpenLayers 的铜陵市旅游查询系统显得尤为关键。这样的系统能够有效地整合铜陵市的各类旅游资源，提供一个用户友好的查询和展示平台，不仅方便游客了解和探索铜陵市的旅游景点，还能够帮助他们更好地规划旅行行程。通过这样的系统，铜陵市的旅游业有望得到进一步的促进和发展，从而为当地经济的增长贡献新的动力。

1.2 功能需求

在对铜陵市旅游市场的深入调研和细致分析之后，我们明确地确定了系统的主要需求，具体如下：

1.基础地图显示功能：系统需要提供铜陵市的基础地图显示功能，这包括但不限于地图的缩放、平移、全图显示等操作，以确保用户能够方便地浏览和探索铜陵市的各个角落；

2.信息发布与管理系统：为了方便管理员高效地发布和管理旅游景点信息，系统将配备一个功能强大的后台管理系统。在这个系统中，管理员可以轻松添加、编辑或删除景点的名称、评分、图片、简介、类型以及地址等详细信息，确保信息的准确性和及时更新；

3.用户登录功能：为了使用户能够保存个人信息，并享受个性化的旅游体验，系统将提供用户登录功能；

4.景点查询与展示功能：系统将支持多种查询方式，包括但不限于景点名称、类型、评

分、关键字搜索等，以满足不同用户的查询需求，查询结果将在地图上以标记的形式展示，用户可以直观地看到每个景点的具体位置，点击标记后，系统将展示景点的详细信息，如图片、简介和用户评价等，帮助用户做出更明智的旅行决策；

5.地图控件功能：为了提高用户体验，系统中的地图控件将提供实时反馈，这意味着，当用户在地图上移动鼠标或进行缩放操作时，坐标位置、比例尺等信息将随鼠标移动和地图缩放实时变化，此外，系统还将添加有图例控件，帮助用户更好地理解地图上的各种符号和标记。

二、系统体系结构设计

2.1 关键技术研究与实践

2.1.1 地图加载与数据管理

通过运用 OpenLayers 所提供的图层概念，我们能够轻松地将各种不同种类的地图数据进行叠加整合，其中包括了景点的矢量数据、铜陵市的行政区划矢量数据、地图的底图以及相关的注记信息。这种图层叠加的技术，使得我们能够根据需要定制和展示个性化的地图内容。举例来说，在景点的详细信息弹窗中，我们可以展示景区的评分数据，这些数据能够直观地反映出各个景区的受欢迎程度和人气高低。此外，我们还可以在弹窗中提供关于景点的详尽描述，包括但不限于景点的历史背景、文化特色、地址等信息，从而让用户在浏览地图的同时，能够更加深入地了解每一个景点的特色和详情，增强用户的互动体验和信息获取的便利性。

2.1.2 交互功能实现

交互功能在电子地图应用中扮演着至关重要的角色。通过利用 OpenLayers 所提供的丰富控件和强大的 API 接口，开发者能够轻松实现一系列的交互操作，从而极大地提升用户体验。其中，缩放和平移功能是基础且核心的交互方式，它们允许用户根据个人的需求对地图进行放大或缩小，以及在地图上进行水平或垂直的移动，这样不仅能够更精确地查看特定区域的地理信息，还能在宏观和微观之间自由切换，以获得最佳的视觉效果和信息展示。空间查询功能则进一步增强了电子地图的应用价值，它通过与数据库技术的紧密结合，使得用户能够执行基于地图的高级空间查询，从而快速定位到感兴趣的景点，并获取关于这些景点的详细信息。除此之外，电子地图应用还提供了其他多种交互功能，例如地图的缩放、全屏

显示模式、实时坐标显示以及对景点数据进行增加、删除、修改和查询等操作，这些功能的加入确保了电子地图能够满足不同用户群体的多样化需求，无论是普通用户还是专业人员，都能在使用过程中获得便捷和高效的服务。

2.1.3 性能优化

在当今的电子地图应用领域，性能优化扮演着一个至关重要的角色。

1.数据量控制：为了确保应用的流畅运行，合理选择地图数据的精度和范围显得尤为重要，通过减少不必要的细节，我们可以有效降低数据量，从而提高地图的加载速度，确保用户能够快速获取所需信息。

2.缓存技术：为了进一步提升性能，电子地图应用广泛采用瓦片缓存技术，通过缓存那些经常被访问的地图瓦片，我们可以避免重复的请求和计算，这样不仅能够提高地图的加载速度，还能减少服务器的负载。

3.多线程处理：为了进一步提高应用的响应速度，电子地图应用通常会采用多线程技术，通过将地图渲染与数据处理这两个任务分开，并利用多线程同时进行处理，可以显著提升应用的性能，确保用户在使用过程中能够获得更加流畅的体验。

2.2 技术架构

2.2.1 前端

OpenLayers：功能强大的开源地图引擎，它允许开发者在网页上嵌入交互式的地图，并提供了丰富的地图操作功能，如缩放、平移等。

Vite：现代化的前端构建工具，它通过利用现代浏览器的原生 ES 模块导入功能，实现了快速的热模块替换和模块热更新，大大提升了开发效率。

Axios：基于 Promise 的 HTTP 客户端，用于浏览器和 node.js 环境，它支持拦截请求和响应，能够轻松处理 JSON 数据，并且具备防止 XSRF 攻击的能力。

2.2.2 后端

Node.js：基于 Chrome V8 引擎的 JavaScript 运行环境，它使得 JavaScript 能够脱离浏览器在服务器端运行，非常适合构建高性能的网络应用。

Express：灵活且简洁的 Web 应用框架，它是基于 Node.js 平台的，提供了强大的路由功能和中间件机制，使得 Web 应用的开发更加高效和模块化。

PostgreSQL：对象关系型数据库系统，它支持复杂查询、外键、事务完整性等特性，并且具有良好的扩展性和可靠性，是构建复杂后端应用的理想选择。

2.3 系统架构图

在本节中，我们将详细探讨系统的架构布局，这将有助于理解整个系统的结构和组件之间的关系。系统架构图是展示系统组件如何相互连接以及它们如何协同工作的图形表示，通过这张图，我们可以清晰地看到各个模块的功能划分，以及它们是如何通过网络、接口或直接通信来实现数据交换和处理的，此外，系统架构图还能够揭示系统的可扩展性、安全性和维护性等关键特性。在本节的后续部分，我们将进一步分析每个组件的具体功能和它们在整体架构中的作用。

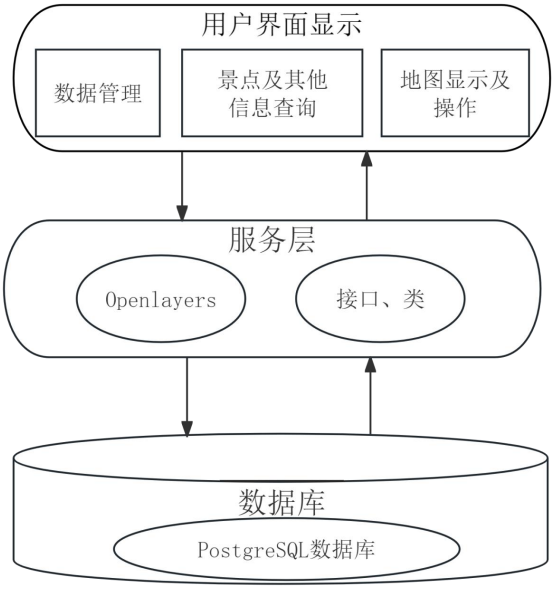

图 1 系统架构图

三、界面与功能模块设计

3.1 界面设计

3.1.1 登录界面

登录界面背景采用铜陵市铜官山的无人机俯瞰照片，中央设有用户登录输入框，供用户输入用户名和密码以进入铜陵市旅游地图系统

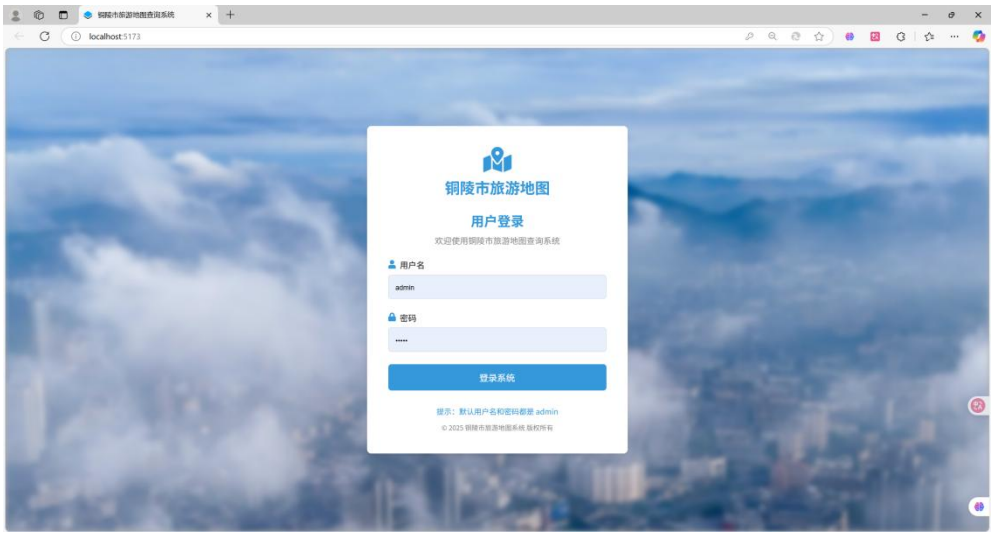

图 2 登录界面

3.1.2 主界面

在完成登录后，进入系统的主要操作界面，界面的顶部配置了功能菜单栏，而界面的左右两侧则设置了常用的地图控件和景点概览数据框

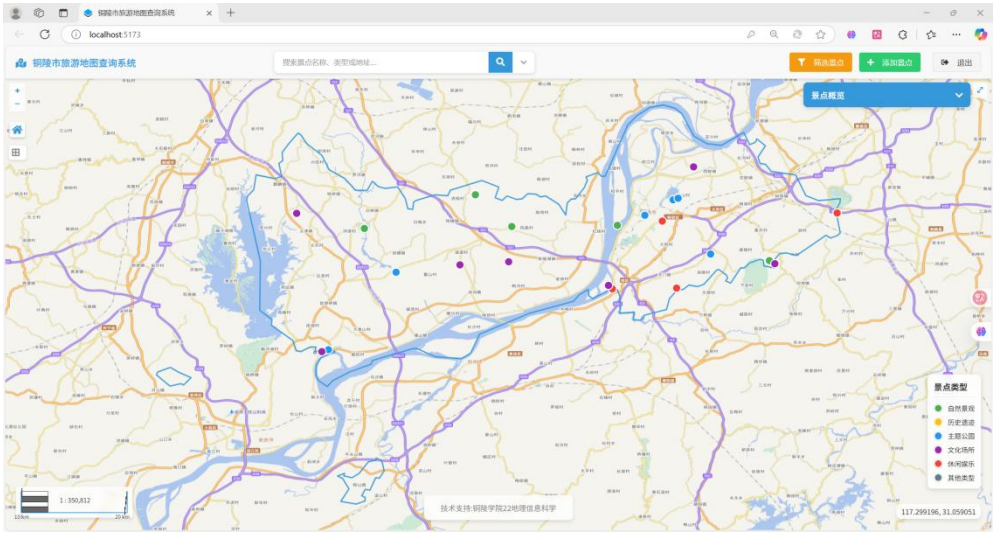

图 3 主界面

3.2 功能模块设计

3.2.1 用户认证模块

（1）登录：输入用户名和密码，登录系统（默认用户名和密码都是 admin）

图 4 登录

（2）退出：点击右上角的“退出”按钮，退出到登录界面

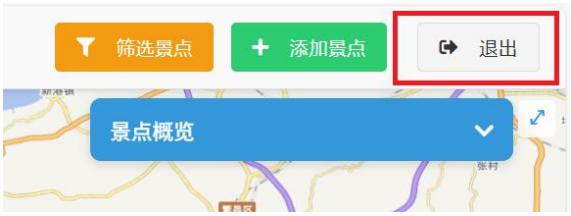

图 5 退出

3.2.2 地图操作模块

（1）地图显示：地图底图采用天地图

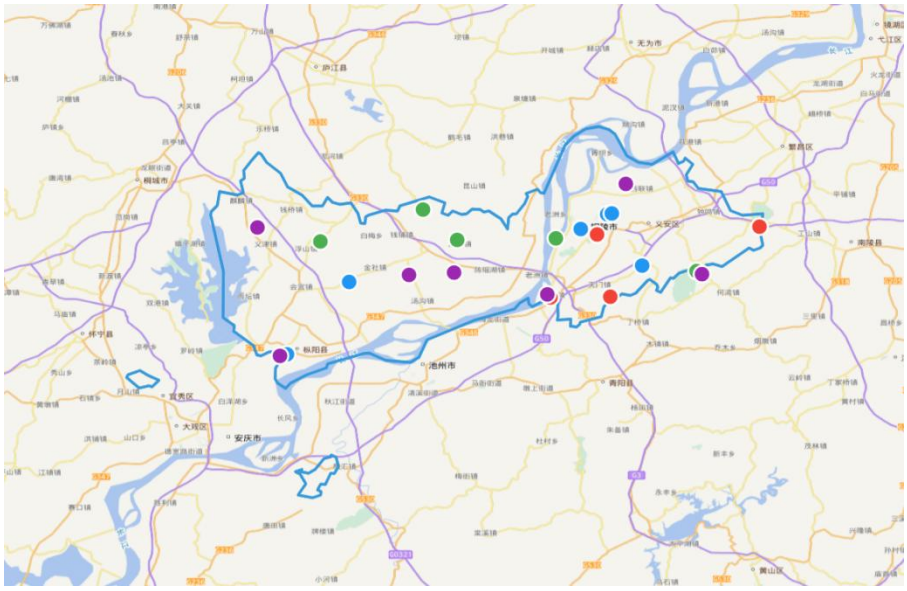

图 6 地图显示

（2）缩放控制：滑动鼠标滚轮或点击界面左上方的“+”、“-”按钮进行放大缩小

图 7 “+”、“-”按钮

图 8 缩放控制

（3）全屏显示：点击右上角的“全屏显示”按钮，进入全屏模式，点击全屏模式右上角的“×”按钮或键盘“esc”按钮，退出全屏模式

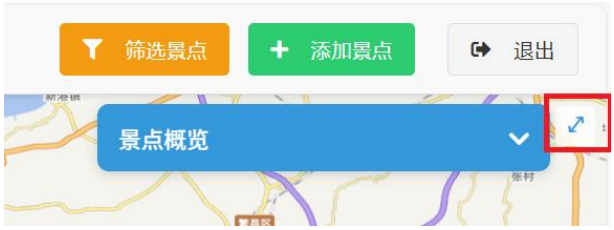

图 9 “全屏显示”按钮

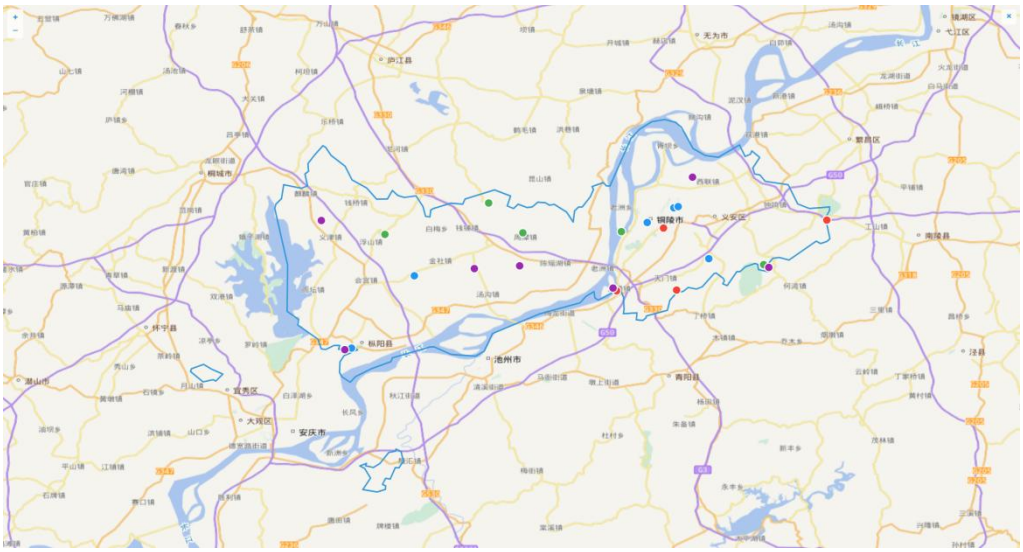

图 10 全屏模式

图 11 退出全屏模式

（4）坐标显示：主界面的右下角有坐标显示控件，坐标数值随鼠标移动而实时改变

图 12 坐标显示

（5）比例尺：主界面的左下角有比例尺控件，比例尺大小随鼠标缩放而实时改变

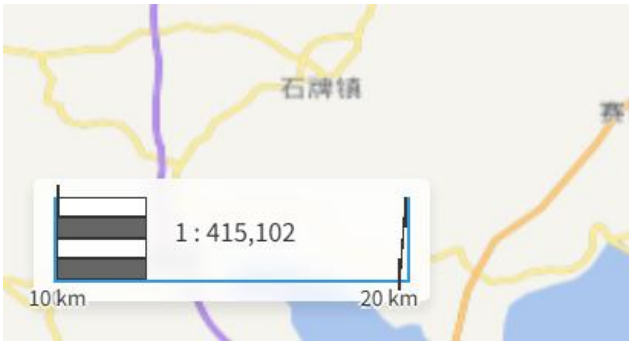

图 13 比例尺

（6）图例：主界面的右下角添加有景点类型图例，将景点分为自然景观、历史遗迹、主题公园、文化场所、休闲娱乐和其他类型，并用不同颜色表示

图 14 图例

（7）返回初始视图：点击界面左上角的“回到初始视图”按钮，定位到铜官区的位置

图 15 返回初始视图

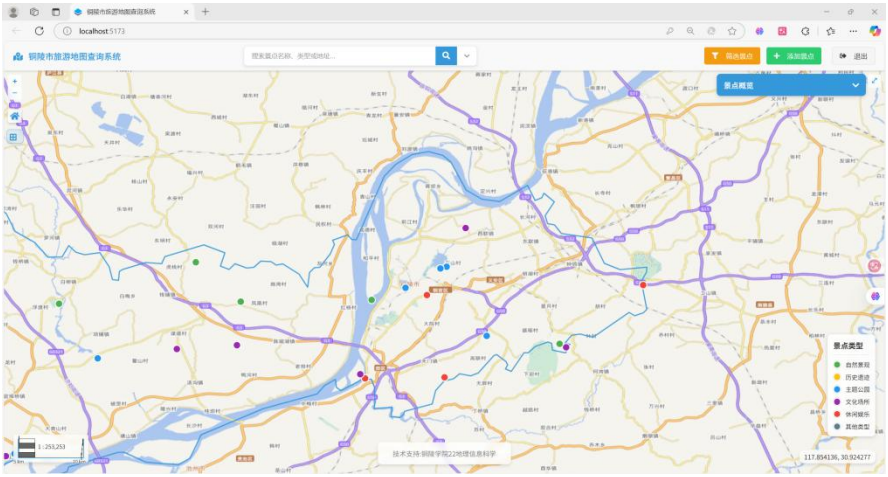

图 16 定位到铜官区

（8）边界显示：点击界面左上角的“切换边界”按钮，控制铜陵市边界数据显示与否

图 17 切换边界

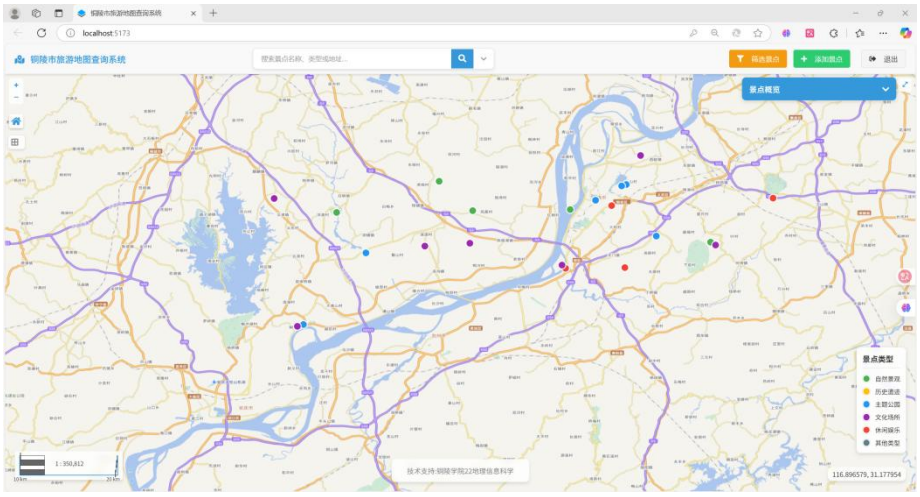

图 18 不显示铜陵市边界

3.2.3 景点管理模块

（1）查看景点：点击地图上的景点，弹出对话框，显示景点的详细信息，分别有景点的名称、评分、图片、简介、类型以及地址

图 19 查看景点

（2）编辑景点：点击“编辑”按钮，可以对该景点的信息进行编辑，编辑完成后点击更新，弹出“景点编辑成功”弹窗，这时数据库的数据同时被更新

图 20 编辑景点

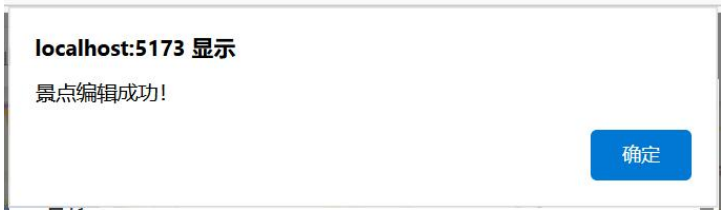

图 21 “景点编辑成功”弹窗

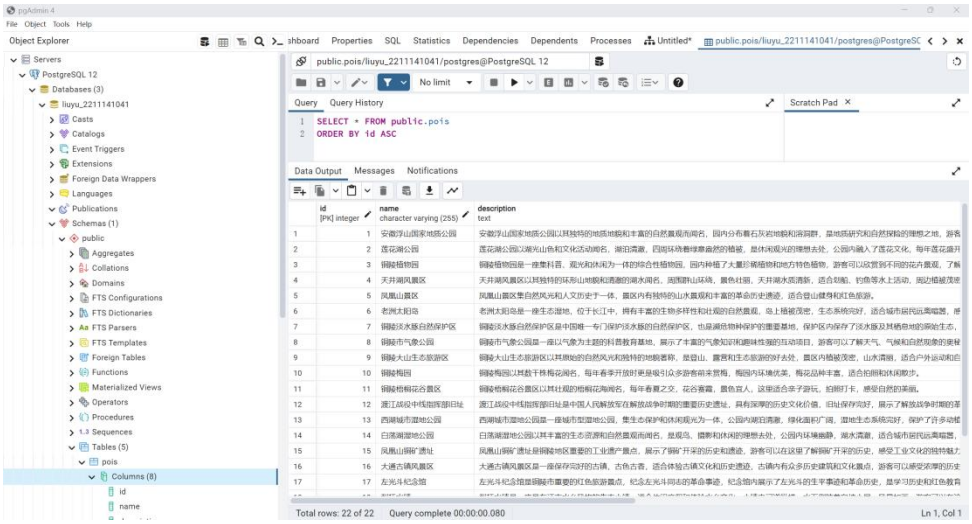

图 22 数据库

（3）添加景点：点击菜单栏右侧的“添加景点”按钮，在弹出的对话框填写新景点的信息并在图上选择位置，点击保存，添加的景点会同时存入数据库中

图 23 添加景点

图 24 “添加景点”对话框

（4）删除景点：点击景点，点击“编辑”按钮，点击删除，景点被删除，数据库中的数据同时会被删除

图 25 编辑

图 26 删除景点

（5）景点概览：点击菜单栏右侧的“景点概览”下拉框，该模块显示有类型分布饼状图及类型评分雷达图，鼠标靠近会显示具体占比及平均评分

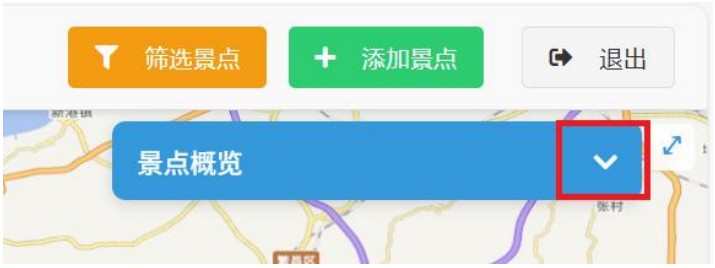

图 27 景点概览

图 28 类型分布饼状图

图 29 类型评分雷达图

3.2.4 搜索筛选模块

（1）景点搜索：在菜单栏的中间部分设有搜索框，可以进行景点的搜索，点击搜索结果，会定位到该景点的位置，并显示该景点的信息

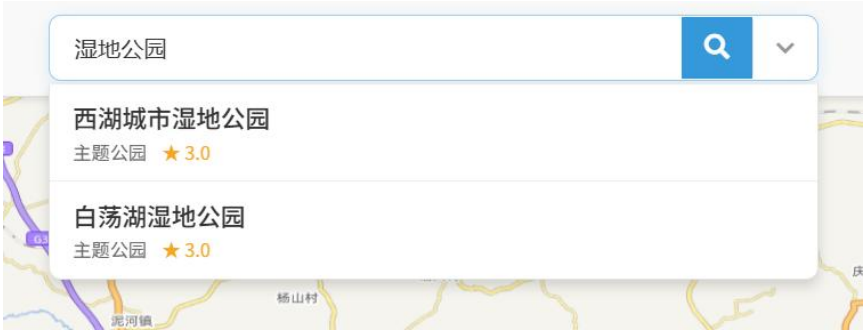

图 30 景点搜索

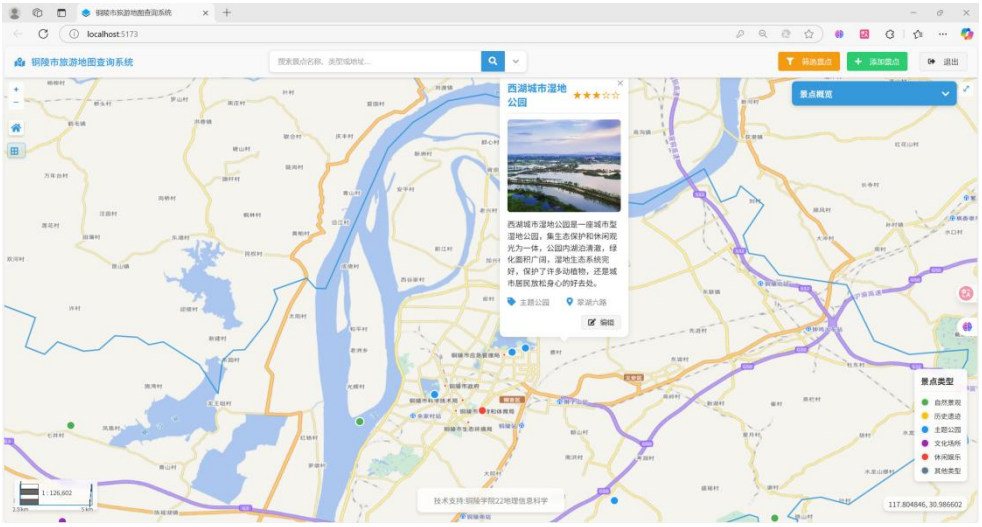

图 31 搜索结果

（2）景点筛选：点击菜单栏右侧的“筛选景点”按钮，在弹出的对话框选择以景点类型、最低评分或是两者组合进行筛选，设置好筛选条件，点击筛选

图 32 筛选景点

图 33 筛选景点对话框

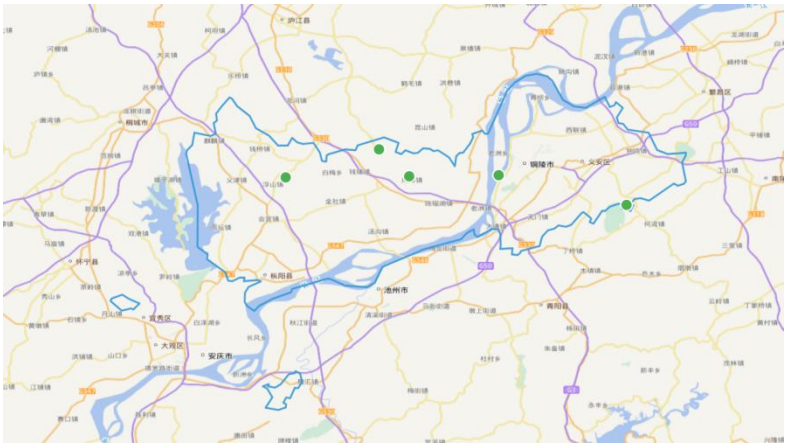

图 34 筛选出自然景观

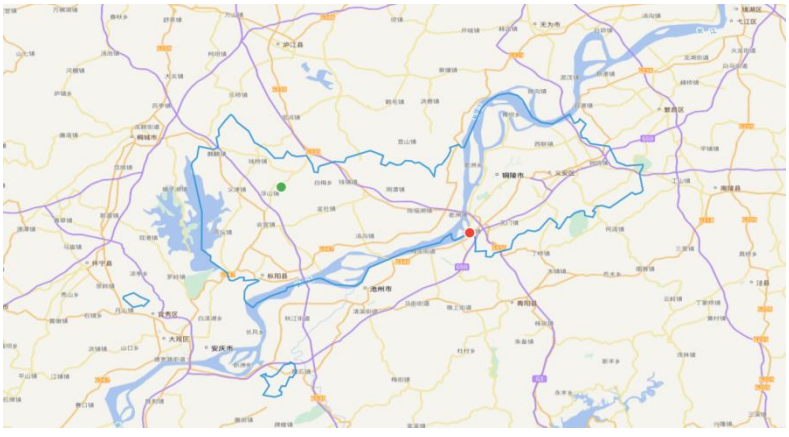

图 35 筛选出评分大于等于 5 的景点

四、核心代码分析

4.1 地图初始化

```
// 初始化地图，使用经纬度坐标系
const map = new Map({
  target: 'map',
  layers: [
    // 矢量底图 (天地图矢量)
    new TileLayer({
      title: '天地图矢量',
      source: new XYZ({
        url: `http://t0.tianditu.gov.cn/vec_w/wmts`,
      }),
      visible: true,
      type: 'base'
    }),
    //标注图层
    new TileLayer({
      title: '地名标注',
      source: new XYZ({
        url: `http://t0.tianditu.gov.cn/cva_w/wmts`,
      }),
      visible: true
    }),
    // 地形底图 (天地图地形)
    new TileLayer({
      title: '天地图地形',
      source: new XYZ({
        url: `http://t0.tianditu.gov.cn/ter_w/wmts`,
      }),
      visible: false,
      type: 'base'
    }),
    vectorLayer
  ],
  view: new View({
    center: initialView.center,
    zoom: initialView.zoom,
    maxZoom: 19,
    projection: 'EPSG:4326' // 使用经纬度坐标系
  }),
```

4.2 景点数据处理

```
// 添加新景点
app.post('/pois', async (req, res) => {
  const { name, description, type, address, rating, longitude, latitude } = req.body;

  // 数据验证
  if (!name || !longitude || !latitude) {
    return res.status(400).json({ error: '名称、经度和纬度是必填项' });
  }

  // 确保数值型数据正确处理
  const parsedRating = rating ? parseFloat(rating) : null;
  const parsedLongitude = parseFloat(longitude);
  const parsedLatitude = parseFloat(latitude);

  // 验证经纬度范围
  if (isNaN(parsedLongitude) || isNaN(parsedLatitude) ||
      parsedLongitude < -180 || parsedLongitude > 180 ||
      parsedLatitude < -90 || parsedLatitude > 90) {
    return res.status(400).json({ error: '无效的经纬度坐标' });
  }

  try {
    const result = await pool.query(
      'INSERT INTO pois (name, description, type, address, rating, longitude, latitude) VALUES ($1, $2, $3, $4, $5, $6, $7) RETURNING *',
      [name, description || '', type || '', address || '', parsedRating, parsedLongitude, parsedLatitude]
    );
    res.status(201).json(result.rows[0]);
  } catch (err) {
    console.error('添加景点失败', err);
    res.status(500).json({ error: '服务器错误: ' + err.message });
  }
});

// 更新景点信息
app.put('/pois/:id', async (req, res) => {
  const { id } = req.params;
  const { name, description, type, address, rating, longitude, latitude } = req.body;
  try {
    const result = await pool.query(
      'UPDATE pois SET name = $1, description = $2, type = $3, address = $4, rating = $5, longitude = $6, latitude = $7 WHERE id = $8 RETURNING *',
      [name, description, type, address, rating, longitude, latitude, id]
    );
    if (result.rows.length === 0) {
      return res.status(404).send('景点未找到');
    }
    res.json(result.rows[0]);
  } catch (err) {
    console.error('更新景点失败', err);
    res.status(500).send('服务器错误');
  }
});

// 删除景点
app.delete('/pois/:id', async (req, res) => {
  const { id } = req.params;
  try {
    const result = await pool.query('DELETE FROM pois WHERE id = $1 RETURNING *', [id]);
    if (result.rows.length === 0) {
      return res.status(404).send('景点未找到');
    }
    res.send('景点已删除');
  } catch (err) {
    console.error('删除景点失败', err);
    res.status(500).send('服务器错误');
  }
});
```

五、总结

本次系统设计的核心目标是构建一个功能全面、操作简便的旅游景点管理平台。小组致力于打造一个功能强大，用户友好的系统，旨在通过精心设计的界面布局和模块化的功能架构，实现用户认证、地图交互、景点管理以及数据筛选等关键功能。系统以天地图作为基础地图服务，通过集成动态控件和丰富的可视化图表，为用户提供了一个高效的空间信息管理与分析平台。下面将从界面设计和功能实现两个方面，对系统进行介绍和分析。

在界面设计方面，我们始终遵循“用户友好”的设计原则，构建了一个清晰的视觉层级结构。为了确保用户能够轻松地访问和使用我们的平台，登录界面采用了简约的设计风格，仅保留了必要的用户名和密码输入框，并且默认设置了 admin 账号，以便于测试人员能够轻松地进行系统测试。主界面以地图作为核心展示区域，通过合理的空间分配，确保了操作控件和信息展示的互不干扰。顶部的菜单栏集中了搜索框、筛选景点、添加景点、退出登录等全局功能入口；左下角的动态比例尺和右下角的实时坐标共同构成了基础地图的导航体系；右下角的景点类型图例采用了差异化配色方案（例如，自然景观用绿色、主题公园用蓝色），这有助于用户快速建立空间认知；全屏模式的切换设计考虑到了键盘（ESC 键）和鼠标操作，以适应不同的使用场景。整体界面通过灰蓝色调和蓝白色弹窗保持了视觉上的统一性，对话框采用卡片式布局，确保了信息的聚焦。例如，在景点信息窗口中，图片预览区与表单字段分区排列，既满足了数据完整性要求，又避免了界面的拥挤感。我们还特别注意到了用户在编辑景点信息时的体验，系统会自动聚焦到首个必填字段，以减少用户的操作步骤，提高效率。

在功能模块设计方面，我们采用了分层架构来实现业务逻辑。用户认证模块通过会话机制来维持用户的登录状态，而退出功能则直接清除本地缓存并返回到初始界面，从而保障了账户的安全性。地图操作模块集成了多尺度的交互能力：滚轮缩放功能保持了 0.5 倍率的梯度变化，确保了平滑的过渡效果；坐标显示控件采用了 WGS84 坐标系，并实现了毫秒级的响应速度，当鼠标移动时，能够同步更新经纬度数值；比例尺控件能够动态计算当前比例尺，以适应不同精度的需求。景点管理模块构建了一个完整的 CRUD（创建、读取、更新、删除）闭环：查看功能通过点击地图上的标记来触发异步请求，并弹出包含评分星级组件和自适应图片容器的窗口；编辑功能采用了增量提交策略，仅传输变更字段以降低服务器的负载；添加景点时，结合地图选点和表单验证，强制校验必填项并限制评分区间（1-5 分）；删除操

作设置了二次确认弹窗，以防止误操作。数据可视化模块通过 Echarts 库来实现，饼图展示了几类景点数量的占比，雷达图对比了不同类型景点的平均评分，hover 事件触发 tooltip 显示具体数值（例如，“自然景观占比 23%，平均评分 3.8”），这有助于管理者进行决策分析。我们还特别优化了高并发场景下的响应速度，当地图加载超过 100 个景点时，采用聚类渲染策略，在缩放到 15 级后显示独立标记，从而在性能和可视化精度之间取得了平衡。

搜索筛选模块引入了复合查询逻辑，以提升检索效率。搜索框支持关键字的模糊匹配，输入时会实时调用分词算法，结果列表会根据相关性进行排序，并高亮显示匹配的文本；筛选功能提供了多条件组合，类型选择器采用了多级联设计，评分滑块支持小数点精度设置，查询结果通过地图标记重绘与列表联动展示。数据库采用了 PostgreSQL 关系型结构，景点表中包含了 geometry 空间字段，以实现地理查询，所有操作均通过事务处理来确保数据的一致性。

未来，系统还有扩展的可能性，包括接入第三方天气 API 以显示景点的实时气候信息、增加用户收藏夹功能、开发热力图分析游客分布模式等。目前，系统已经形成了一个稳定且可迭代的框架，既满足了基础的管理需求，也为后续的功能升级预留了技术接口。通过本次设计实践，我们验证了模块化开发模式在 GIS 系统中的有效性，并为同类平台的建设提供了一个可复用的解决方案模板。我们相信，随着技术的不断进步和用户需求的日益增长，我们的平台将不断进化，为旅游景点的管理提供更加智能化、便捷化的服务。