

上海citywalk指南

2051498储岱泽

introduction

"上海CityWalk指南"是一个基于Vue3开发的出行地图系统。该项目整合了百度地图API、高德地图天气API、ChatGPT的API、维基百科API以及来自天行数据的天气诗句API、节假日查询API和景点介绍API等多个功能模块，旨在为用户提供一个全方位的地图系统。

通过"上海CityWalk指南"，用户可以快速获取关于上海市的各种信息，包括天气预报、节假日查询、地理信息、维基百科介绍、景点介绍等。用户可以使用该系统规划路线、导航、查询天气情况，并了解上海市的文化、历史和旅游资源，同时也能够查询节假日，更方便的安排自己的出行日期。

Background

上海作为中国最具国际化的城市之一，拥有着丰富的历史文化和现代都市风光。现如今在上海"citywalk"已经成为了一个流行的词汇，该词汇的意思大抵是指：探索城市，在城市的漫步中收获乐趣。

然而，对于首次来到上海的游客来说，往往难以找到合适的旅游路线和景点，同时也不是很清楚各个景点背后的文化内涵。同时，一次美好的citywalk计划不仅仅需要优秀的路线，也需要一份美好的天气和合适的时机。因此，我们通过"上海citywalk指南"这个项目，为游客提供一份详尽的上海市区步行旅游指南，包括路线规划，天气查询，景点介绍，节假日查询等内容，让游客可以更加轻松地探索这座城市的魅力。

Goal

- 整合多个在线数据源，包括天气预报、百度地图、景点介绍、维基百科、天气诗句、节假日查询等API，以为城市探索者提供全面的信息。
- 提供用户友好的界面，允许用户通过点击景点标注的点来获取景点介绍和百科搜索结果，交互方便有趣。
- 实现出行智能小助手功能，使得用户可以根据当天的天气状况获得合理的出行建议，穿衣建议以及天气概述。
- 使用echart展示未来一周的天气，折线图方便用户看到未来天气的走势。
- 提供导航和路线规划的功能，使得使用者不再迷路。

技术路线

为了实现我的"上海citywalk指南"项目，我经过仔细考虑并选择了以下技术栈，以满足项目的需求和目标：

前端技术

- Vue3**: 我选择了Vue3作为前端框架，因为它提供了灵活性和性能，允许我构建交互式、响应式的用户界面。Vue3的Composition API使代码组织更加清晰和可维护。

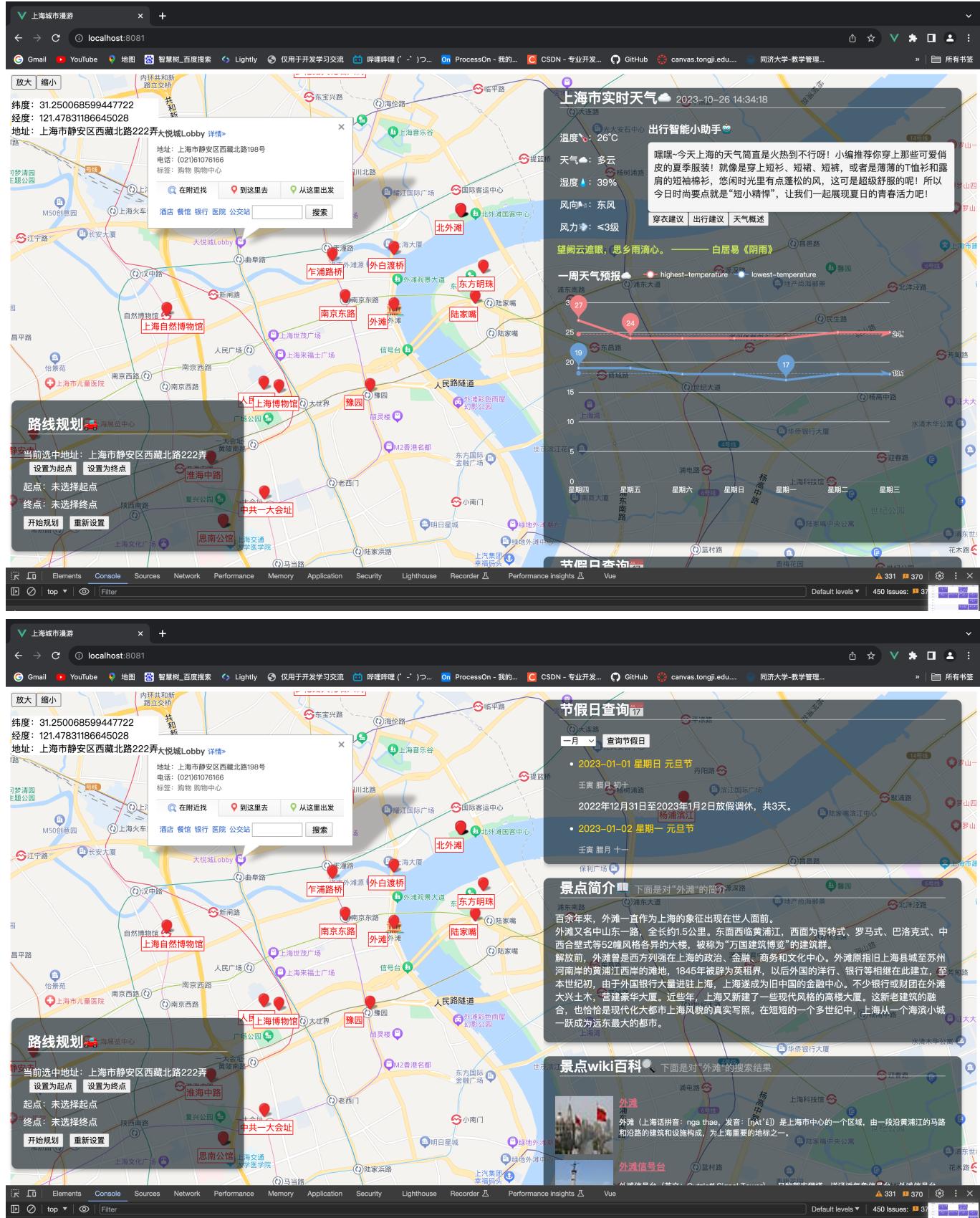
- **Axios**: 我使用Axios来进行网络请求，包括从各个在线数据源获取景点信息和天气状况所需的数据。Axios提供了易于使用的方式来处理HTTP请求，并处理响应数据。
- **Eslint**: 我选择Eslint作为代码风格统一的管理工具，使得前端代码变得规范。
- **echarts**: 我选择echarts来展示上海一周的天气状况以及走势，清晰明了，可以让用户很清晰的规划未来一周的出行。
- **vue-baidu-map-3x**: 我选择vue-baidu-map-3x组件来实现百度地图的调用以及使用，该插件针对百度地图开发，集成了百度地图路线规划，地点搜索等等功能，使用便捷。

部署和托管

- **Nginx**: 使用Nginx来进行项目的部署。
 - **git**: 通过git来实现项目的版本控制和管理。
-

实现细节

界面总览



前端页面结构

1. 实时天气与一周天气预报

- 展示当前实时天气状况。
- 右侧出行智能小助手可以给出一些出行建议。
- 显示符合当前天气状况应景的诗句，增添应用的趣味性。

- 显示接下来一周的天气预报，将最高气温和最低气温做成折线图。同时，当用户把鼠标悬浮在图表上的时候可以显示当天最高最低气温，以及日出日落时间，同时获得当天的天气状况。



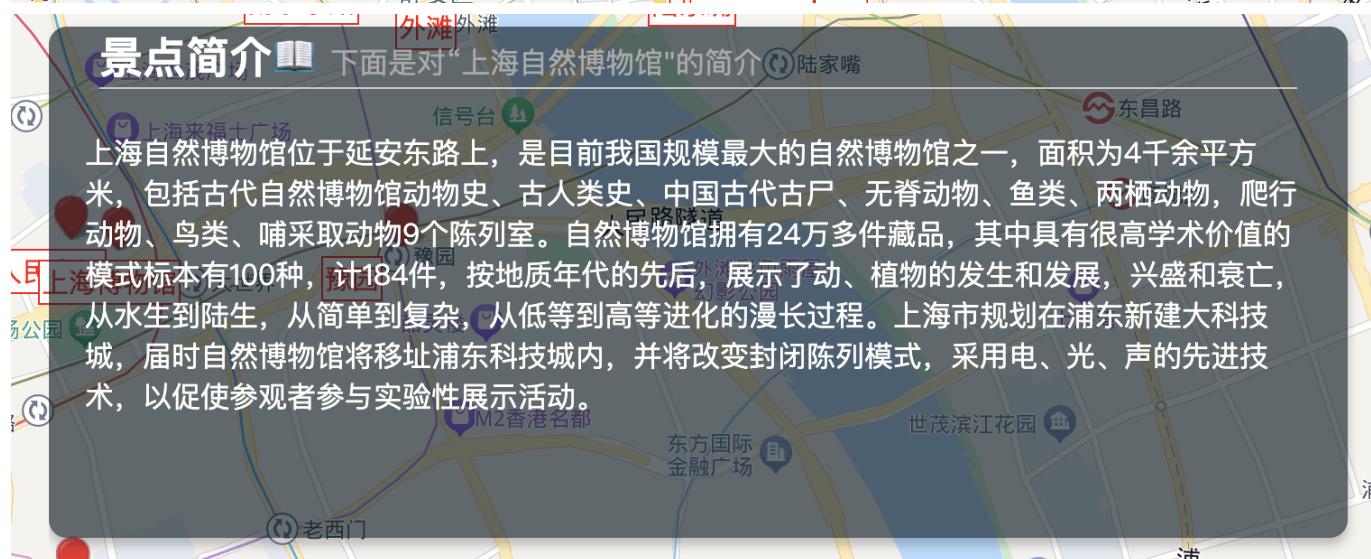
2. 节假日查询

用户可以在下拉框选择对应的月份，查询该月所有的节假日信息，从而合理的安排自己的出行。



3. 景点简介

在地图上用红色的标记标出了所有值得一看的景点，用户可以点击这些红色的标记，然后右边的“景点简介”以及“景点Wiki百科”部分就会显示该景点的介绍以及搜索结果，让用户全方位的了解这个景点。



4. 景点wiki百科



5. 路径规划

- 用户单击地图上的点就会在左上角显示当前选中的点的具体的经纬度和地名，此时如果用户在左下角的路线规划板块选择“设置为起点”，就可以将该地点设置为起点，点击“设置为终点”就可以将该地点设置为终点。
- 点击“开始规划”进行路径规划。
- 点击“重新设置”重新对起点和终点进行选择。



6. 导航功能

用户可以点击地图上的点唤醒百度地图自带的导航功能。



数据的请求与展示

天气模块

- 实时天气展示

在页面初始化渲染的时候，就采用高德地图的天气api来调用当前上海的实时天气状况。

```
//调用天气api, 已经可以调取成功
getWeatherData() {
    const key = "3f6157ccb2e4ec191a030932e211ffaa"; // 将 `your key` 替换成你的高德开发者key
    const url = `https://restapi.amap.com/v3/weather/weatherInfo?key=${key}&city=${this.city}`;
    console.log("获取天气");
    axios
        .get(url)
        .then((response) => {
            if (response.status === 200) {
                this.weatherData = response.data;
                console.log(response.data);
            }
        })
        .catch((error) => {
            console.error(error);
        });
},
}
```

在获取到当前的天气数据之后，将数据保存在weatherData当中，然后展示在页面上。

- 出行智能小助手

将页面初始化的时候调用天气api获得的实时天气数据传给chatGPT，然后让chatGPT来对当前的天气状况给出穿衣建议，出行建议，以及根据天气生成绝句。

调用chatGPT

```
async initGPT3(details, choice) {
  const API_KEY = "sk-OnuFKcXkfVm6SYpkXqDT3BlbkFJGnZAL0Zwi5zvovVuirlL"; //输入API Key
  const openai = new OpenAI({
    apiKey: API_KEY,
    dangerouslyAllowBrowser: true,
  });
  console.log("正在生成介绍" + details + "的信息");
  const chatCompletion = await openai.chat.completions.create({
    messages: [
      {
        role: "user",
        content: `你好, ${details}`,
      },
    ],
    model: "gpt-3.5-turbo",
  });
  console.log(chatCompletion);
  console.log(chatCompletion.choices[0].message.content);
  // 说明此时是在生成智能出行小助手
  if (choice == 1) {
    //显示在消息框内
    this.dialogBox = chatCompletion.choices[0].message.content;
  }
  else {
    }
},
},
```

当用户想要生成穿衣建议调用的函数

```
//点击穿衣按钮之后获取穿衣建议
getDressAdvice() {
  const details =
    "已知今天上海的天气数据如下所示: " +
    "湿度: " +
    this.weatherData.lives[0].humidity +
    ", 温度: " +
    this.weatherData.lives[0].temperature +
    ", 天气: " +
    this.weatherData.lives[0].weather +
    ", 风力: " +
    this.weatherData.lives[0].windpower +
    ", 请问可以给出穿衣建议吗? 可以把下面这段话扩写一下变得更加活泼俏皮: " +
    this.dressAdvice + ", 用可爱俏皮的语句直接生成一段话, 50到80字左右, 不要列出一点一点的, 也不要太长) ";
  console.log(details);
  this.initGPT3(details, 1);
```

```
    this.showResponse();
},
```

当用户想要生成出行建议调用的函数

```
//点击出行按钮之后获取出行建议
getTravelAdvice() {
  const details =
    "已知今天上海的天气数据如下所示：" +
    "湿度：" +
    this.weatherData.lives[0].humidity +
    ", 温度：" +
    this.weatherData.lives[0].temperature +
    ", 天气：" +
    this.weatherData.lives[0].weather +
    ", 风力：" +
    this.weatherData.lives[0].windpower +
    ", 请问可以给出出行建议吗？就根据我给你的上海目前的天气信息给出，比如说如果今天下雨就建议打车或者地铁，如果今天天晴就建议骑自行车或者走路，如果风大就尽量不要骑车等等。
    (用可爱俏皮的语句直接生成一段话，50到80字左右，不要列出一点一点的，也不要太长) ";
  console.log(details);
  this.initGPT3(details, 1);
  this.showResponse();
},
```

当用户想要生成天气概述调用的函数

```
//点击天气概述按钮之后获取天气概述
getWeatherSummary() {
  const details =
    "已知今天上海的天气数据如下所示：" +
    "湿度：" +
    this.weatherData.lives[0].humidity +
    ", 温度：" +
    this.weatherData.lives[0].temperature +
    ", 天气：" +
    this.weatherData.lives[0].weather +
    ", 风力：" +
    this.weatherData.lives[0].windpower +
    ", 请你用可爱俏皮的语句，同时写一首唐诗来描述今天的天气，这首诗需要包含今天的天气特点以及人们的活动，景色等等";
  console.log(details);
  this.initGPT3(details, 1);
  this.showResponse();
},
```

- 根据当前天气调用天气诗句api生成应景的诗句

```
//调用天气诗句api
getWeatherPoem() {

  axios({
    method: 'post',
    url: 'https://apis.tianapi.com/tianqishiju/index',
    data: Qs.stringify({ key: '1e12d5d1fb06599d02ef546b73e2c360',
    tqtype: 2 }),
    headers: { 'Content-Type': 'application/x-www-form-urlencoded' }
  }).then(res => {
    // this.tianapi_data = res.data
    console.log(res.data)
    this.poem = res.data.result
  })
},
},
},
},
```

景点介绍模块

采用如下数据结构对地图上需要标注的景点进行存储：

```
markers: [
  {
    position: { lng: 121.4912, lat: 31.2438 },
    content: "南京东路",
    show: false,
  },
  //下面还有很多景点
]
```

用户点击地图上对应的点之后，将把该景点名称传给全国景点查询api，生成对该景点的介绍。

```
//得到景点介绍
getBriefIntro(place) {
  axios({
    method: 'post',
    url: 'https://apis.tianapi.com/scenic/index',
    data: Qs.stringify({ key: '1e12d5d1fb06599d02ef546b73e2c360',
    word: place }),
    headers: { 'Content-Type': 'application/x-www-form-urlencoded' }
  }).then((res) => {
    if (res.data.code == 200)
      this.briefIntro = res.data.result.list[0].content;
    else {
      this.briefIntro = "<p>暂时没有对该地点的简介哦～</p>"
    }
    console.log(res.data)
    console.log(res)
```

```
    });
},
```

调用wiki百科

```
//调用Wiki百科API
async searchWiki(keyword) {
  try {
    const response = await
axios.get("https://zh.wikipedia.org/w/api.php", {
  params: {
    format: "json",
    action: "query",
    generator: "search",
    gsrnamespace: 0,
    gsrlimit: 5,
    prop: "pageimages|extracts",
    pilimit: "max",
    exintro: true,
    explaintext: true,
    exsentences: 1,
    exlimit: "max",
    origin: "*",
    gsrsearch: keyword,
  },
});
const pages = response.data.query.pages;
const wikiPages = [];
for (let pageId in pages) {
  const page = pages[pageId];
  const wikiPage = {
    title: page.title,
    thumbnail: page.thumbnail,
    extract: page.extract,
    url: ""
  };
  // 将title拼接成访问中文维基百科的链接
  wikiPage.url = "http://zh.wikipedia.org/wiki/" +
encodeURIComponent(page.title);
  wikiPages.push(wikiPage);
}
this.WikiPage = wikiPages;
console.log(this.WikiPage);
} catch (error) {
  console.error("Error:", error);
}
},
},
```

节假日查询模块

调用节假日查询api

```
//调用查询节假日的接口，得到这个月的节假日
getHoliday(month) {
    // console.log(this.selectedMonth)
    // 判断月份是否为个位数，如果是则在前面添加一个零
    var formattedMonth = month < 10 ? '0' + month : month;
    var Date = '2023-' + formattedMonth;
    console.log(Date);
    axios({
        method: 'post',
        url: 'https://apis.tianapi.com/jiejiari/index',
        data: Qs.stringify({ key: '1e12d5d1fb06599d02ef546b73e2c360',
date: Date, type: 2 }),
        headers: { 'Content-Type': 'application/x-www-form-urlencoded' }
    }).then(res => {
        // this.tianapi_data = res.data;
        // console.log(res.data.result.list);
        // 提取isnotwork为1的项并存储到data的holiday变量中
        this.holiday = res.data.result.list.filter(item => item.isnotwork
==== 1);
        console.log(this.holiday)
    });
},
```

项目成果

1. 上海citywalk指南

我成功地开发了一个上海citywalk指南，让用户可以通过这个指南获得地图导航，当前天气状况，景点信息，路径规划以及节假日查询。

2. 用户友好的界面

我的项目提供了清晰、直观的用户界面，映入眼帘的首先就是大大的百度地图，用户可以通过鼠标的上下滑动来实现地图的缩放，同时各种信息整齐的排在屏幕的右侧，清晰且具有透明的设计感。

3. 数据整合和更新

项目整合了多个在线数据源，包括天气API、百度地图、维基百科等，以提供全面的信息。

4. 安全性和性能

我部分url采用了HTTPS来确保数据传输的安全性，并实施了数据验证和过滤来保护用户数据。项目的性能得到了优化，以确保快速加载和响应。

参考

Documents of the Web APIs

- 维基百科接口请求的实现

<https://www.cnblogs.com/zczhangcui/p/6251604.html>

- 百度地图接口请求的实现

<https://map.heifahaizei.com/doc/baidu-map.html>

- 高德地图天气API接口请求的实现

<https://lbs.amap.com/api/webservice/guide/api/weatherinfo/#t1>

-chatGPT API接口请求的实现

<https://www.openaidoc.com.cn/>

- 一周天气API接口请求的实现

<https://www.tianapi.com/apiview/72>

- 天气诗句API接口请求的实现

<https://www.tianapi.com/apiview/91>

- 景点介绍API接口请求的实现

<https://www.tianapi.com/apiview/93>

- 节假日API接口请求的实现

<https://www.tianapi.com/apiview/139>