

开发环境：

Visual studio code

软件工具的使用：

1. 开发前的准备：配置了 go lang 语言的环境变量。查询了 js 与 html, css 开发前的准备建议，下载了 live server 插件与本地服务器进行实时预览，学习使用 Chrome 开发者工具验证功能与故障排查。（指用 F12 看控制台报错信息和应用程序看前端历史记录保存成功没有）

2. 开发过程的使用

后端：感觉主要在学习命令行工具 cmd 的使用，用来管理环境，启动与运行项目。

前端：感觉也没啥啊就扣呗可能我水平太低了（枯

代码的逻辑结构思路

后端：

总体上我想的是从需要实现的功能出发，去反推能够实现功能的代码，模块化解决功能需求。同时尽量运用上指针，函数，结构体这种使程序更高效的工具。

需求功能里涉及单个用户的多个相关信息（如电话号码，验证码，发送时间等），于是想到用结构体来存储信息。

1. 结构体存储信息，用 map 模拟数据库。

2. 设置多个函数：验证手机号码是否有效函数，随机六位数生成验证码函数，发送验证码，并检验是否符合发送条件的函数。

3. 更新数据库。

4. main 函数里，调用函数，并用 If else if，嵌套结构，完成不同情况下的打印。

前端：

1. 分工上：

html 负责网站基础结构的搭建。定义元素和结构，时钟，输入框框，历史记录。链接外部样式表 style.css，控制页面的外观和布局，链接外部 css 与 java script 文件。

Css 负责背景图片，居中布置，时钟，搜索框，按钮颜色，大小，边距，字体，历

史记录框框。

Js 负责获取 HTML 中的元素，编写所有函数，如更新时间、处理搜索、保存历史记录，显示历史记录。为元素添加事件监听器，如点击、按键等。

2.结构上：

Html: 创建文件的所有的“容器”和“组件”，链接了 Font Awesome 图标库。通过的 `<i>` 标签来使用各种图标。还有用来显示时间的 `<div>`、输入框和图标的 `<div>`，以及用来展示历史记录的 `<div>`。用 id 和 class 属性，区分组件。引入了两个外部文件：CSS 文件 JavaScript 文件负责功能。

Css: 就不同属性分块处理上面的东西就行了，比较值得一提的是下拉式历史记录的显示，需要设置默认隐藏与打开两种样式。

Js: 获取 HTML 中的关键元素，搜索框、时钟显示区和图标，历史记录。设置搜索功能函数，保存搜索记录函数，显示搜索记录函数，时钟函数，事件监听点击案件，指针位置在一定范围内下拉历史记录。

问题及解决方案 (可多啦 QAQ)

(这里我已经筛选掉了过失类的，或者仅需普通查询资料即可解决的较为简单的语法问题，例如电话号码属于什么数据类型等等)

Q1: 后端：命名并初始化了 map 叫 Database 还有用伪随机数生成器设置一个种子。后，系统报错显示必须在函数内进行赋值行为。

解决方案：创建 init 函数，把这几行代码放入 init 函数中，达到一次性初始化的目的。

Q2: 后端：报错信息 declared and not used: VerificationCode (compiler-unusedvar)

原因：声明了 VerificationCode 这个变量，但是未使用它。VerificationCode 变量在 init 函数里创建的是**局部变量**，当 init 函数执行完毕后，这个变量就会被销毁。

解决：**生成验证码的逻辑**移动到需要使用它的地方，也就是 ValidVerificationCode 函数里。

Q3: 后端：main 函数里的运行部分操作不执行，验证验证码是否有效的功能没正常执行。(见 AI)

原因：else 块只**生成了验证码**，但没有把它保存到 **Database 这个 map 中**。当函数执行完毕后，VerificationCode 这个变量就消失了。当“输入验证码”时，系统无法找到

之前生成的验证码，因为数据库里是空的。

解决：确保无论是哪种情况（第一次发送还是重复发送），最终生成的验证码和发送时间都要被存回 Database 这个 map 中。

Q4：后端：代码不发挥作用,,, 输出验证码后没有经历下一次输入就弹出验证码不正确。(AI)

原因：fmt.Scanln 在读取了你的输入（比如 2）后，也会读取你按下的回车（换行符 \n）。当你选择了 2 并获取了验证码后，程序会再次进入循环，并执行 fmt.Scanln(&choice)，等待你的下一个输入。但是，fmt.Scanln 发现输入流中还有一个没有被读取的换行符。它会把这个空行当做你的新输入。

解决：AI 大人教我 use strings.TrimSpace 来清理所有输入。

Qn：前端：代码全都打好了但是未显示历史记录框框。怀疑是竞争条件问题，检查 F12 开启的开发者工具看应用程序内本地储存发现没有保存记录。

分析可能的原因：搜索函数执行的太快了导致保存历史搜索记录函数来不及保存，系统认为没有历史记录因此历史记录框框不下拉。

解决过程：问 AI（对话详见后续 AI 的使用中）

第一步：将 window.location.href 放入 setTimeout 中，创建了一个短暂的“等待期”。在 100 毫秒的等待期里，让 saveSearchHistory() 函数完成它的所有工作，将数据安全地存入 localStorage。然后延迟到期，页面再执行跳转。

结果：还是不行

第二步：使用调试版本的 java script，在关键部位都加上了 console.log 语句。观察控制台报错信息发现 Html 中<body>部分结构错了导致 search-wrapper 部分的结构无法被 js 识别。修改后成功辣。

与 AI 大师的交流 Gemini2.5flash/Gemini2.5pro

1.后端改错。

我的这个函数根本就没发挥作用是什么原因

显示思路

你好，我看到了你的代码。你问为什么函数没有发挥作用，这是因为你有一个关键的逻辑错误：你的函数生成了验证码，但没有把它存起来。

错误原因

让我们来看看 `ValidVerificationCode` 函数的逻辑：

```
Go
```

```
func ValidVerificationCode(number string) {  
    if data, ok := Database[number]; ok {  
        // ... 如果找到用户数据，检查60秒限制 ...  
    } else {  
        // ... 如果没找到用户数据，生成验证码 ...  
        randomNumber := ...  
        var VerificationCode ...  
        fmt.Println(...)  
    }  
}
```

// 函数在这里结束了

新增代码：

```
// 更新用户数据  
newData := UserData{  
    PhoneNumber:    number,  
    VerificationCode: verificationCode,  
    SendTime:       time.Now(),  
    DailyCount:     data.DailyCount + 1,  
    LastSendDate:   time.Now(),  
}  
Database[number] = newData  
  
fmt.Printf("验证码为: %s\n", verificationCode)
```

2.

```
验证码不正确，请重试。
1: 请输入验证码 2: 校验验证码
4: 输入
验证码不正确，请重试。
1: 请输入验证码 2: 校验验证码
验证码，请输入验证码
with struct {
0: 验证码不正确，请重试。
请输入验证码。
1: 请输入验证码 2: 校验验证码
验证码为: 142139
1: 请输入验证码 2: 校验验证码
验证码不正确，请重试。
1: 请输入验证码 2: 校验验证码
验证码为: 142139
1: 请输入验证码 2: 校验验证码
验证码不正确，请重试。
1: 请输入验证码 2: 校验验证码
```

你看输出结果还是有问题，而且在输出验证码后马上就出现了一个验证码不正确

显示思路

你观察得非常仔细，这是Go语言中一个非常难发现的陷阱。

你看到的“验证码不正确”的提示，并不是因为你的验证码真的不正确，而是因为程序在没有等待你输入的情况下，自己就进入了验证验证码的逻辑。

问题根源：看不见的“回车”

在你的代码中，`fmt.Scanln` 在读取了你的输入（比如 `2`）后，也会读取你按下的回车（换行符 `\n`）。

当你选择了 `2` 并获取了验证码后，程序会再次进入循环，并执行 `fmt.Scanln(&choice)`，等待你的下一个输入。

但是，`fmt.Scanln` 发现输入流中还有一个没有被读取的换行符。它会把这个空行当做你的新输入，然后：

解决方案

要解决这个看不见的“回车”问题，我们需要对代码的逻辑流程进行一次重构，并使用

`strings.TrimSpace` 来清理所有输入。

我将把获取验证码和验证验证码的逻辑放在不同的循环里。这样，只有当用户获取了验证码之后，程序才会进入等待用户输入验证码的循环。

修正后的完整代码

```
Go

package main

import (
    "fmt"
    "math/rand"
    "strings"
    "time"
    "unicode"
)

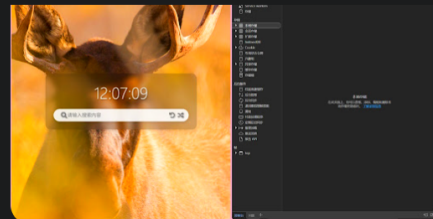
type UserData struct {
    PhoneNumber      string
    VerificationCode string
    SendTime         time.Time
    DailyCount       int
    LastSendDate     time.Time
}
```

3.

```
func main() {
    var phoneNumber string
    for {
        fmt.Println("请输入电话号码。")
        _, err := fmt.Scanln(&phoneNumber)
        if err != nil {
            fmt.Println("输入错误，请重试")
            continue
        }
        phoneNumber = strings.TrimSpace(phoneNumber)

        if !validatePhoneNumber(phoneNumber) {
            fmt.Println("登录失败，手机号格式不正确")
            continue
        }
        break
    }
}
```

3.



忘掉上一次对话，继续上上次的历史记录问题，为什么搜索过一次还是没有搜索记录

显示思路

好的，我们来重新聚焦于搜索记录无法保存的问题。

您提供的这张带有开发者工具的截图非常关键，它直接指出了问题的核心。

问题诊断

您的截图右侧显示，浏览器的“本地存储 (Local Storage)”中“未存储任何内容”。

这证实了问题的原因：在您进行搜索后，记录根本没有被成功地保存下来。因此，当您返回页面并点击输入框时，JavaScript 检查历史记录，发现是空的，所以自然不会显示下拉框。

根本原因

```
// 这是修改后的正确代码
function performSearch() {
  let query = searchInput.value.trim();
  if (!query) {
    query = searchInput.placeholder;
  }

  // 1. 先调用保存函数，让它开始工作
  saveSearchHistory(query);

  const searchURL = `https://www.google.com/search?q=${encodeURIComponent(query)}`;

  // 2. 将页面跳转放入一个微小的延迟执行
  // 这给了 saveSearchHistory() 足够的时间去完成本地存储操作
  setTimeout(() => {
    window.location.href = searchURL;
  }, 100); // 延迟100毫秒，对于用户来说几乎无感，但足以完成保存
}
```

这个问题我没看出来我哪里输错了。

您好，感谢您提供截图！啊哈，这次的问题非常细微，是一个非常容易忽略的大小写拼写错误。您的 `setTimeout` 确实是加上了，但它前面的代码出了错，导致它没有机会执行。

问题出在您的 `performSearch` 函数的第 22 行。

问题定位

请仔细看这一行代码：

```
const searchURL = `https://www.google.com/search?q=${encodeURIComponent(query)}`;
```

错误点：您写的是 `encodeURIComponent`，其中字母 **C** 是大写的。

正确的写法：JavaScript 内置的这个函数是 `encodeURIComponent`，其中字母 **c** 必须是小写的。

JavaScript 语言是严格区分大小写的，所以 `encodeURIComponent` 和 `encodeURIComponent` 在它看来是两个完全不同的东西。

错误导致的结果

1. 当 `performSearch` 函数被调用时，代码执行到第 22 行。
2. 浏览器尝试寻找一个名为 `encodeURIComponent` 的函数或变量，但找不到（因为正确的名字是 `encodeURIComponent`）。
3. 此时，浏览器会抛出一个 `ReferenceError`（引用错误），并立即停止执行 `performSearch` 这个函数。
4. 因此，函数中排在错误代码之后的所有代码，包括 `saveSearchHistory()` 和 `setTimeout`，

```
<link rel="icon" href="/images/奶衣.gif" type="image/gif">
<link rel="stylesheet" href="/css/style.css">
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.8.0/css/all.min.css">
```

我想做一个favicon这段代码能实现吗

显示思路

是的，这段代码在语法上是完全正确的，可以用来实现 favicon。

我们来分析一下您写的这行关键代码：

HTML



```
<link rel="icon" href="images/奶衣.gif" type="image/gif">
```

- `rel="icon"`：这个属性告诉浏览器，这个 `<link>` 标签链接的是一个图标文件，用作网站的 favicon。这是最关键的部分。
- `href="images/奶衣.gif"`：这个属性指定了图标文件的路径。它告诉浏览器，你的图标文件（名为 `奶衣.gif`）存放在与你的 HTML 文件同目录下、名为 `images` 的文件夹里。
- `type="image/gif"`：这个属性指明了图标文件的类型。对于 `.gif` 文件，`image/gif` 是正确的 MIME 类型。

既然简单的代码替换没有解决问题，我们需要采取一种更主动的方法来**“侦察”代码的执行情况**。

我们将使用 `console.log`，这是一个向开发者控制台打印信息的命令，它可以像侦探一样，告诉我们代码到底执行到了哪一步，以及每一步的状态是什么。

第 1 步：使用“调试版本”的 JavaScript

请完整地复制下面这段代码，并用它完全替换您 `script.js` 文件中的所有内容。我在关键位置都添加了 `console.log` 语句。

JavaScript



```
// --- 调试版本 script.js ---

console.log("脚本已加载 (Script loaded!)");

const searchInput = document.getElementById('searchInput');
const searchIcon = document.getElementById('search-icon');
const lastSearchIcon = document.getElementById('last-search-icon');
const randomSearchIcon = document.getElementById('random-search-icon');
const searchHistoryDiv = document.getElementById('search-history');
const clockDiv = document.getElementById('clock');

// 检查元素是否被成功找到
console.log("搜索输入框元素:", searchInput);
console.log("历史记录容器元素:", searchHistoryDiv);

// 随机搜索词语列表
const randomQueries = ["南大家园", "云家园", "家园工作室", "小家园传声机"];
```

```
// --- 调试版本 ---
```


1. 保存 上述代码到您的 `script.js` 文件。
2. 在浏览器中打开您的 `index.html` 页面。
3. 按下键盘上的 `F12` 键，打开开发者工具。
4. 在开发者工具中，点击 **"Console" (控制台)** 选项卡。
5. 刷新页面 (按 `F5` 或刷新按钮)。
6. 进行一次搜索 (例如，输入 "hello" 并按回车)。
7. 页面跳转后，按“后退”返回到您的主页。
8. 用鼠标点击输入框。

第 3 步：提供线索

请将您在 **控制台 (Console)** 中看到的所有信息，特别是任何红色的错误信息，完整地截图或复制文本给我。

这些信息会告诉我们：

- 脚本是否成功加载了？
- HTML 元素（比如 `searchInput` 和 `searchHistoryDiv`）是否被正确找到了？(如果这里显示 `null`，就是问题所在！)
- `performSearch` 函数是否被调用了？
- `saveSearchHistory` 函数是否成功保存了数据？
- `focus` 事件是否被触发了？

这里直接把他要的控制台截图发给他了所以没说话。

请看您截图的控制台中，我为您圈出的这两行关键信息：

1. 历史记录容器元素：`null`
 - 含义：当我们的 JavaScript 脚本执行 `const searchHistoryDiv = document.getElementById('search-history');` 这行代码时，它在您的 HTML 文件中没有找到 `id` 属性正好是 `search-history` 的元素。因此，它返回了一个 `null`（空值）。
2. `Uncaught TypeError: Cannot set properties of null (setting 'innerHTML')`
 - 含义：紧接着，当代码执行到 `displaySearchHistory` 函数中的 `searchHistoryDiv.innerHTML = '';` 时，它试图在一个 `null`（空值）上设置 `innerHTML` 属性。这就好比让你去粉刷一栋不存在的房子，程序不知道该怎么办，于是就抛出了一个 **类型错误 (TypeError)**，整个脚本的执行也因此中断了。

结论：问题的根源不在 JavaScript，而在于您的 **HTML 文件**。JavaScript 脚本无法找到它需要操作的目标 `<div>` 元素。

解决方案

解决方案非常简单：请打开您的 `index.html` 文件，并仔细检查您用来显示历史记录的那个 `div`。

问题几乎可以肯定出在下面三种情况之一：

- **1. ID 拼写错误 (最可能)：**
 - 您写的 `id` 可能有拼写错误。请确保它一字不差地是 `id="search-history"`。
 - 常见的错误比如：`serch-history`，`search_history`，`search-histroy` 等。计算机对小写和符号非常敏感。

这里也是把代码截图发给他检查，并陈述：运行后还是没有历史记录框功能，继续排查。

您好，我们终于找到问题的关键了！您的两张截图放在一起，揭示了一个非常隐蔽但又极其常见的问题。

请您仔细对比这两张图：

1. 您的代码编辑器 (image_d46f4e.jpg)：在 `index.html` 文件的第 24 行，您已经正确地添加了 `<div id="search-history"></div>`。
2. 您的浏览器控制台 (image_d46c0c.jpg)：控制台明确地告诉我们 历史记录容器元素：null。这说明当 JavaScript 运行时，浏览器加载的那个 HTML 页面里，根本不存在 `id="search-history"` 的元素。

这两个证据是互相矛盾的。唯一的解释是：

您在 VS Code 中修改并保存了 `index.html` 文件，但是您的浏览器正在运行一个旧的、没有包含这个修改的缓存版本。

解决方案：强制刷新浏览器缓存

这是一个非常常见的问题，解决方法也很简单。我们需要强制浏览器忽略它自己的缓存，去重新加载你硬盘上最新的文件。

请按照以下步骤操作：

1. 回到您的 Chrome 浏览器。
2. 按下键盘快捷键：
 - 在 Windows/Linux 上: `Ctrl + Shift + R`

基本上这些都是涉及 AI 帮我直接修改代码的了。