

互動式碳足跡計算器

生成式 AI 創新應用企劃案

組員:企管 2A 林邑達、企管 2B 洪紹軒、企管 2B 蔡博承

壹、 問題說明

隨著全球氣候變化問題的日益嚴重，減少碳排放已成為各國和個人共同面對的重大挑戰。互動式碳足跡計算器在這一過程中扮演了關鍵角色，以下是需要這種工具的多個原因：

1. 提升環保意識：許多人對自己的日常活動如何影響環境缺乏具體了解。計算器提供了直觀的數據，讓用戶了解他們的碳排放量，從而提高環保意識。
2. 具體減排建議：計算器根據用戶輸入的數據提供個性化的減排建議，如減少開車次數、改用可再生能源等。這些建議能幫助用戶採取實際行動，減少碳足跡。
3. 數據驅動決策：計算器收集並分析大量用戶數據，可以幫助政府和組織制定更精確的環保政策和策略。這些數據能夠揭示碳排放的主要來源和趨勢，支持科學決策。
4. 促進企業社會責任：企業可以使用計算器來評估和減少其運營中的碳排放，從而提升企業的社會責任形象。這不僅有助於環保，還能提高企業在市場中的競爭力。
5. 個人行動的累積效應：當更多人使用計算器並採取減排行動，這些個人的小變化將會累積成顯著的環境效益，減緩氣候變化，保護生態系統。

6. 教育功能：計算器可以用作教育工具，在學校、社區和企業中推廣使用，教育更多人了解碳排放的影響及減排的重要性。

貳、問題解決方案說明

互動式碳足跡計算器通過簡單、直觀的界面讓用戶輸入他們的日常活動數據，如交通工具使用、家庭能源消耗、飲食習慣等。計算器會生成詳細的碳排放報告，讓用戶清楚了解每項活動的碳足跡。這種可視化數據能有效提高用戶的環保意識，幫助他們認識到日常行為對環境的影響，從而激發他們改變生活方式的動力。

互動式碳足跡計算器可以收集大量來自不同用戶的數據，這些數據包括個人、家庭和企業的碳排放狀況。通過對這些數據的分析，可以揭示出不同地區、不同群體的碳排放特徵和趨勢，幫助決策者更科學地制定環保政策。例如，某地區數據顯示交通是主要碳排放來源，政府可以優先改善公共交通系統，鼓勵使用低碳交通工具。

企業可以使用碳足跡計算器評估其運營中的碳排放，找出主要排放來源並採取相應的減排措施。這不僅能幫助企業達到環保目標，還能提升其社會責任形象。在當今消費者越來越關注環保的背景下，展示企業在減少碳排放方面的努力可以提高品牌聲譽和市場競爭力。

參、運用生成式 AI 協助解決問題的過程

本企劃透過以下步驟與生成式 AI 共同來規劃與執行

生成式 AI 創新應用企劃案 說明如下：

利用生成式 AI: 輸入正確指令與要求 輸出結果

流程如下:

一、輸入指令:

假設你是網頁設計與永續設計專家 現在請發想 關於網頁設計與永續發展結合的重要主題，輸出十個創意點子 繁體中文回答。

chatgpt 產出:當前，將網頁設計與永續發展結合已成為一種重要的趨勢，既能提升使用者體驗，也能減少環境影響。以下是十個創意點子：

1. 低碳網頁設計：設計減少網頁負載的網站，使用更少的圖片和動畫，優化代碼，從而減少伺服器能耗和碳排放。
2. 綠色主機服務：選擇使用可再生能源供電的網頁主機，並在網站上展示主機的環保認證，增加透明度。
3. 無紙化推廣：提供數位版的公司報告、產品手冊和宣傳資料，減少印刷和紙張使用。
4. 可持續性教育頁面：設立專門的頁面或部落格，教育訪客關於企業的環保措施和永續發展策略，增加公眾的環保意識。
5. 環保購物選項：在電商網站中，標示環保產品，提供低碳配送選項，並鼓勵使用環保包裝材料。
6. 資源節約模式：引入深色模式（**Dark Mode**），不僅能減少能源消耗，還能延長裝置的電池壽命。
7. 社會責任報告：定期發布公司的環境和社會責任報告，展示在永續發展方面取得的成就和未來目標。
8. 虛擬會議支持：設計和優化支持虛擬會議的平台，減少差旅需求，從而降低碳足跡。

9. 互動式碳足跡計算器：在網站上添加計算工具，讓用戶能夠計算他們的碳足跡，並提供減排建議。
10. 零垃圾挑戰：設立挑戰活動，鼓勵用戶分享和參與零垃圾生活方式，並提供相關的資源和指導。

這些創意點子既能提高網站的功能性和使用者滿意度，也能幫助企業達到永續發展的目標。

二、選擇主題輸入指令：現在針對「互動式碳足跡計算器」，設計完整的執行方案，包含詳細步驟及效益，輸出 500 字，繁體中文回答。

chatgpt 產出：

互動式碳足跡計算器執行方案

一、設計目標

建立一個易於使用的互動式碳足跡計算器，幫助用戶計算他們的碳足跡，並提供減排建議。此計算器將成為教育工具，提升公眾對環保的意識，並推動行動以減少個人和企業的碳排放。

二、詳細步驟

1. 需求分析與調研

- 目標用戶確定：確定目標用戶群體（如個人、家庭、企業）。
- 市場調查：研究現有的碳足跡計算工具，分析其優缺點，確定計算器應包含的功能和特點。

2. 功能設計

- 界面設計：設計簡潔直觀的用戶界面，確保易於操作。包括輸入部分、結果展示部分和建議部分。
- 計算模型：與環保專家合作，制定精確的碳足跡計算模型，涵蓋日常活動（如交通、能源消耗、食物消費等）。
- 數據來源：確定可靠的數據來源，確保計算結果的準確性。

3. 開發與測試

- 前端開發：使用現代網頁技術（如 HTML、CSS、JavaScript）開發用戶界面，確保計算器在各種設備上都能正常使用。
- 後端開發：開發伺服器端邏輯，處理數據輸入和計算。選擇合適的編程語言和框架（如 Python、Node.js）。
- 測試：進行單元測試和整合測試，確保計算器的準確性和穩定性。

4. 上線與推廣

- 網站整合：將計算器嵌入公司網站的相關頁面，確保訪客能夠方便地找到並使用。
- **SEO 優化**：優化搜索引擎，以提高計算器的可見性。
- 社交媒體推廣：利用社交媒體平台進行宣傳，吸引更多用戶使用計算器。

5. 用戶反饋與改進

- 收集反饋：設置反饋機制，收集用戶意見和建議，了解用戶體驗和需求。
- 持續改進：根據反饋和使用數據，持續優化計算器的功能和界面，提高使用者滿意度。

三、選擇「開發與測試」輸入指令: 針對互動式碳足跡計算器 網頁設計與開發，每一個步驟進行詳細流程說明 輸出 300 字 繁體中文回答

chatgpt 產出:

互動式碳足跡計算器網頁設計與開發流程

一、界面設計

1. 需求確認

- 用戶調研：與用戶訪談，了解他們的偏好和需求。
- 功能需求：列出計算器必備功能，如數據輸入、結果展示、減排建議等。

2. 原型設計

- 線框圖：繪製線框圖，展示主要界面佈局和功能分佈。
- 用戶體驗測試：進行初步用戶測試，收集反饋，進行修改。

3. 視覺設計

- 風格確定：設計整體視覺風格，包括色彩、字體和圖標等元素。
- 高保真原型：製作高保真原型，展示最終設計效果，並進行用戶測試和調整。

二、前端開發

1. 技術選型

- 框架選擇：選擇合適的前端框架（如 React、Vue.js），提高開發效率。
- 工具配置：配置開發環境和工具（如 Webpack、Babel）。

2. 界面開發

- 靜態頁面搭建：根據視覺設計稿，搭建靜態頁面結構。
- 動態功能實現：編寫 **JavaScript** 實現用戶交互功能，如數據輸入和結果展示。

3. 優化與測試

- 性能優化：優化頁面加載速度，減少不必要的資源請求。
- 跨瀏覽器測試：確保計算器在不同瀏覽器和設備上的兼容性。

三、後端開發

1. 伺服器配置

- 伺服器選擇：選擇合適的伺服器技術（如 **Node.js**、**Django**），並進行配置。
- 數據庫設計：設計數據庫結構，存儲用戶數據和計算結果。

2. API 開發

- 數據處理：編寫 **API** 處理用戶輸入的數據，進行碳足跡計算。
- 結果返回：將計算結果通過 **API** 返回前端展示。

3. 安全性測試

- 漏洞檢查：進行安全測試，檢查常見漏洞（如 **SQL** 注入、**XSS** 攻擊）。
- 數據保護：實施數據加密和訪問控制，保護用戶隱私。

四、上線與維護

1. 部署上線

- 伺服器部署：將應用部署到生產環境，確保伺服器穩定運行。

- 域名配置：配置域名和 SSL 證書，確保網站安全和易於訪問。

2. 運行監控

- 日常監控：使用監控工具（如 Google Analytics）跟蹤計算器使用情況。
- 反饋迭代：收集用戶反饋，進行持續改進和功能更新。

通過以上詳細流程，確保互動式碳足跡計算器的設計和開發既能滿足用戶需求，又具備穩定性和安全性。

4、請 AI 幫忙架構網頁程式碼，輸入指令：針對 前端：React、Webpack 提供完整程式碼。

chatgpt 產出：

以下是一個簡單的 React 應用程式，使用 Webpack 作為打包工具。

Jsx

// App.js

import React, { useState } from 'react';

import axios from 'axios';

function App() {

const [formData, setFormData] = useState({});

const [result, setResult] = useState(null);

const handleChange = (e) => {

setFormData({

...formData,

[e.target.name]: e.target.value

});

};


```

jsx
// index.js
import React from 'react';
import ReactDOM from 'react-dom';
import App from './App';

ReactDOM.render(<App />, document.getElementById('root'));
html
<!-- index.html -->
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <title>Carbon Footprint Calculator</title>
</head>
<body>
  <div id="root"></div>
</body>
</html>

```

肆、代碼測試

將 GPT 輸出的代碼放到網路上的代碼測試器進行測試

```

// App.js import React, { useState } from 'react'; import axios from 'axios'; function App() { const [formData, setFormData] = useState({}); const [result, setResult] = useState(null); const handleChange = (e) => { setFormData({ ...formData, [e.target.name]: e.target.value }); }; const handleSubmit = async (e) => { e.preventDefault(); try { const response = await axios.post('/api/calculate', formData); setResult(response.data); } catch (error) { console.error('Error calculating footprint:', error); } }; return (
Carbon Footprint Calculator
Car Mileage:  Electricity Usage:  { /* More input fields for other categories */ } 
{result && (
Result
Total Carbon Footprint: {result.totalCarbon}
{ /* Display more details if needed */ }
)}); } export default App;

```

伍、心得

在使用 ChatGPT 的過程中，我們經歷了一段充滿探索和學習的旅程。從一開始的設計問題到議題的篩選，我們花費了相當多的時間。這段時間並非浪費，

而是為了熟悉如何最有效地利用這個工具。我們發現，提問的技巧對於獲得有用的回答至關重要。透過不斷的摸索和實踐，我們漸漸找到了一些有效的問題設計方法，使得我們能夠更快速地獲得所需的信息。

更重要的是，我們學會了如何根據我們感興趣的議題，利用生成式 AI 來創造並解決問題。這種技巧的掌握不僅擴展了我們的思維邊界，還提高了解決問題的效率。透過與 ChatGPT 的互動，我們意識到這種人工智能工具可以成為我們的良師益友，幫助我們在各種領域中取得更好的成果。

因此，這段使用 ChatGPT 的經驗不僅讓我們獲得了寶貴的知識，還培養了我們解決問題的能力。通過不斷地嘗試和學習，我們能夠更好地應用人工智能技術來解決現實生活中的各種挑戰。