感謝你分享了每一週的資料夾結構截圖！這些截圖清楚地展示了從 W01 到 W06 每個專案的檔案組織方式。根據你的要求，我會詳細分析每個資料夾結構，並提供每一週每個部分的完整步驟，確保你能準備好所有必要的檔案和資料夾。因為資料夾結構已經很明確，我會直接根據這些結構來補充步驟，並針對 W01 到 W06 的每個任務提供具體的設置和程式碼範例。

W01: 基礎設置與 Tailwind CSS

W01-P1: 建立 GitHub 儲存庫

目標：創建 GitHub 儲存庫並分享給老師和助教。 **步驟：**

1. **創建本地資料夾：**
   - 在你的專案根目錄下創建 `w01_tailwind_51` 資料夾。
   - 進入資料夾：`cd w01_tailwind_51`
2. **初始化 Git：**
   - 運行 `git init`。
3. **創建 GitHub 儲存庫：**
   - 登錄 GitHub，點擊 "New Repository"，命名為 `1132-semester-w01`，選擇公開或私有。
   - 複製儲存庫的 HTTPS/SSH 連結。
4. **連結本地與遠端：**
   - 運行 `git remote add origin <儲存庫連結>`。
   - 編輯 `README.md`，內容如下：

     ```
     # W01 Tailwind Project

     Student ID: 51
     ```

   - 提交並推送：

     ```
     git add .
     git commit -m "Initial commit"
     git push -u origin main
     ```

5. **分享給老師和助教：**
   - 在 GitHub 上點擊 "Settings > Collaborators"，添加他們的 GitHub 用戶名。

W01-P2: 設置 Vite 和 Tailwind CSS，顯示姓名和學號

目標：使用 Vite 建立專案，整合 Tailwind CSS，顯示姓名和學號。 **步驟**：

1. **初始化 Vite 專案**：
   - 在 `w01_tailwind_51` 資料夾中運行：

```
npm create vite@latest . -- --template react
```

   - 安裝依賴：`npm install`
   - 這會自動生成 `vite.config.js`、`index.html`、`src/main.jsx` 等檔案。
2. **安裝 Tailwind CSS**：
   - 運行：

```
npm install -D tailwindcss postcss autoprefixer
npx tailwindcss init -p
```

   - 這會生成 `tailwind.config.js` 和 `postcss.config.js`。
   - 修改 `tailwind.config.js`：

```
/** @type {import('tailwindcss').Config} */
import { defineConfig } from 'vite';
import react from '@vitejs/plugin-react';
import tailwindcss from '@tailwindcss/vite';
export default defineConfig({
  plugins: [react(), tailwindcss()],
});
```

   - 修改 `src/index.css`：

```
@tailwind base;
@tailwind components;
@tailwind utilities;
```

   - 根據資料夾結構，另有一個 `index_51.css`，可以複製 `index.css` 的內容，或直接用 `index.css`。
3. **創建 App_51.jsx 顯示姓名和學號**：
   - 在 `src/components` 下創建 `App_51.jsx`：

```
export default function App_51() {
  return (
```

```
    <div className='flex justify-center items-center h-screen bg-gray-
100'>
      <h1 className='text-4xl font-bold text-blue-600'>
        Name: Your Name, ID: 51
      </h1>
    </div>
  );
}
```

4. 修改 **App.jsx**：
   - 編輯 `src/App.jsx`：

```
import App_51 from './components/App_51';
import './index.css';

function App() {
  return (
    <div>
      <App_51 />
    </div>
  );
}

export default App;
```

5. **檢查 main.jsx**：
   - 確保 `src/main.jsx` 正確渲染 App：

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import App from './App';
import './index.css';

ReactDOM.createRoot(document.getElementById('root')).render(
  <React.StrictMode>
    <App />
  </React.StrictMode>
);
```

6. **啟動專案**：
   - 運行 `npm run dev`，打開 `http://localhost:5173` 檢查。

W01-P3: 顯示響應式導航欄 (Navbar_51)

目標：根據螢幕大小 (< 640px 和 > 640px) 展示不同的導航欄。 **步驟**：

1. **創建導航資料：**
   - 根據資料夾結構，`src/data.jsx` 已經存在，編輯如下：

```
export const navLinks = [
  { id: 1, name: 'Home', href: '#' },
  { id: 2, name: 'About', href: '#' },
  { id: 3, name: 'Projects', href: '#' },
];
```

2. **創建 Navbar_51 組件：**
   - 在 `src/components` 下創建 `Navbar_51.jsx`：

```
import { navLinks } from '../data';

export default function Navbar_51() {
  return (
    <nav className='bg-blue-600 p-4'>
      <ul className='flex flex-col sm:flex-row space-y-2 sm:space-y-0
sm:space-x-4 text-white'>
        {navLinks.map((link) => (
          <li key={link.id}>
            <a href={link.href} className='hover:underline'>
              {link.name}
            </a>
          </li>
        ))}
      </ul>
    </nav>
  );
}
```

3. **整合到 App_51：**
   - 修改 `src/components/App_51.jsx`：

```
import Navbar_51 from './Navbar_51';

export default function App_51() {
  return (
    <div>
      <Navbar_51 />
      <div className='flex justify-center items-center h-screen bg-
gray-100'>
        <h1 className='text-4xl font-bold text-blue-600'>
          Name: Your Name, ID: 51
        </h1>
      </div>
```

```
    </div>
  );
}
```

W02: 組件與響應式設計

W02-P1: 展示 Hero_51 組件

目標：在 768px 寬度下展示 Hero 區塊。 **步驟**：

1. **創建 Hero_51 組件**：
   - 在 `src/components` 下創建 `Hero_51.jsx`：

```
export default function Hero_51() {
  return (
    <section className='bg-gray-800 text-white p-8 text-center md:h-96
flex items-center justify-center'>
      <div>
        <h1 className='text-4xl md:text-5xl font-bold'>
          Welcome to My Portfolio
        </h1>
        <p className='mt-4 text-lg'>Student ID: 51</p>
      </div>
    </section>
  );
}
```

2. **整合到 App_51**：
   - 修改 `src/App_51.jsx`：

```
import Navbar_51 from './components/Navbar_51';
import Hero_51 from './components/Hero_51';

export default function App_51() {
  return (
    <div>
      <Navbar_51 />
      <Hero_51 />
    </div>
  );
}
```

W02-P2: 展示 About_51 和 SectionTitle_51

目標：展示「關於我」頁面和標題組件。 **步驟**：

1. **創建 SectionTitle_51**：
   - 在 `src/components` 下創建 `SectionTitle_51.jsx`：

```
export default function SectionTitle_51({ title }) {
  return <h2 className='text-3xl font-bold text-center mb-6'>{title}
</h2>;
}
```

2. **創建 About_51**：
   - 在 `src/components` 下創建 `About_51.jsx`：

```
import SectionTitle_51 from './SectionTitle_51';

export default function About_51() {
  return (
    <section className='p-8'>
      <SectionTitle_51 title='About Me' />
      <p className='text-lg'>
        I am a student with ID 51 learning web development.
      </p>
    </section>
  );
}
```

3. **更新 App_51.jsx**：
   - 修改 `src/App_51.jsx`：

```
import Navbar_51 from './components/Navbar_51';
import Hero_51 from './components/Hero_51';
import About_51 from './components/About_51';

export default function App_51() {
  return (
    <div>
      <Navbar_51 />
      <Hero_51 />
      <About_51 />
    </div>
  );
}
```

W02-P3: 展示 Projects_51 的響應式設計

目標：展示專案列表，支援 2 或 3 個項目一行。 **步驟**：

1. **創建專案資料**：
   - 編輯 `src/data.jsx`：

```
export const navLinks = [
  { id: 1, name: 'Home', href: '#' },
  { id: 2, name: 'About', href: '#' },
  { id: 3, name: 'Projects', href: '#' },
];

export const projects = [
  { id: 1, name: 'Project 1', desc: 'Description 1' },
  { id: 2, name: 'Project 2', desc: 'Description 2' },
  { id: 3, name: 'Project 3', desc: 'Description 3' },
];
```

2. **創建 ProjectCard_51**：
   - 在 `src/components` 下創建 `ProjectCard_51.jsx`：

```
export default function ProjectCard_51({ project }) {
  return (
    <div className='bg-gray-200 p-4 rounded'>
      <h3 className='font-bold'>{project.name}</h3>
      <p>{project.desc}</p>
    </div>
  );
}
```

3. **創建 Projects_51**：
   - 在 `src/components` 下創建 `Projects_51.jsx`：

```
import { projects } from '../data';
import SectionTitle_51 from './SectionTitle_51';
import ProjectCard_51 from './ProjectCard_51';

export default function Projects_51() {
  return (
    <section className='p-8'>
      <SectionTitle_51 title='Projects' />
      <div className='grid grid-cols-2 md:grid-cols-3 gap-4'>
        {projects.map((project) => (
          <ProjectCard_51 key={project.id} project={project} />
        ))}
```

```
      </div>
    </section>
  );
}
```

4. **更新 App_51.jsx**：
   - 添加 `<Projects_51 />`：

```
import Navbar_51 from './components/Navbar_51';
import Hero_51 from './components/Hero_51';
import About_51 from './components/About_51';
import Projects_51 from './components/Projects_51';

export default function App_51() {
  return (
    <div>
      <Navbar_51 />
      <Hero_51 />
      <About_51 />
      <Projects_51 />
    </div>
  );
}
```

W02-P4: 展示 Skills_51 的響應式設計

目標：展示技能列表，支援 2 或 3 個項目一行。 **步驟**：

1. **創建技能資料**：
   - 編輯 `src/data.jsx`：

```
export const navLinks = [...]; // 已有
export const projects = [...]; // 已有

export const skills = ["HTML", "CSS", "JavaScript", "React",
"Tailwind"];
```

2. **創建 SkillsCard_51**：
   - 在 `src/components` 下創建 `SkillsCard_51.jsx`：

```
export default function SkillsCard_51({ skill }) {
  return (
    <div className='bg-blue-100 p-4 rounded text-center'>{skill}</div>
```

```
  );
}
```

3. **創建 Skills_51**：

   - 在 `src/components` 下創建 `Skills_51.jsx`：

```jsx
import { skills } from '../data';
import SectionTitle_51 from './SectionTitle_51';
import SkillsCard_51 from './SkillsCard_51';

export default function Skills_51() {
  return (
    <section className='p-8'>
      <SectionTitle_51 title='Skills' />
      <div className='grid grid-cols-2 md:grid-cols-3 gap-4'>
        {skills.map((skill, index) => (
          <SkillsCard_51 key={index} skill={skill} />
        ))}
      </div>
    </section>
  );
}
```

4. **更新 App_51.jsx**：

   - 添加 `<Skills_51 />`：

```jsx
import Navbar_51 from './components/Navbar_51';
import Hero_51 from './components/Hero_51';
import About_51 from './components/About_51';
import Projects_51 from './components/Projects_51';
import Skills_51 from './components/Skills_51';

export default function App_51() {
  return (
    <div>
      <Navbar_51 />
      <Hero_51 />
      <About_51 />
      <Projects_51 />
      <Skills_51 />
    </div>
  );
}
```

W03: Next.js 與圖片優化

初始化 Next.js 專案

1. **創建專案：**
   - 在根目錄下運行：

```
npx create-next-app@latest w03-wild-oasis-51
```

   - 選擇預設選項，確保使用 App Router (因為資料夾結構中有 `app` 目錄)。
2. **啟動：**`cd w03-wild-oasis-51 && npm run dev`。

W03-P1: 使用圖片品質比較大小

目標：展示不同品質的圖片大小。 **步驟：**

1. **準備圖片：**
   - 根據結構，圖片位於 `public/starter/cabin-images/` 和 `public/components/others/`。
   - 確保 `about-1.jpg` 和 `about-2.jpg` 已存在，假設原始大小為 6.86MB。
2. **修改首頁：**
   - 編輯 `app/page.js`：

```
import Image from 'next/image';

export default function Home() {
  return (
    <div className='p-8'>
      <h1 className='text-2xl mb-4'>Image Quality Comparison</h1>
      <div>
        <p>Quality 100 (107KB):</p>
        <Image
          src='/components/others/about-1.jpg'
          width={500}
          height={300}
          quality={100}
          alt='About 1'
        />
      </div>
      <div>
        <p>Quality 60 (16.7KB):</p>
        <Image
          src='/components/others/about-1.jpg'
          width={500}
          height={300}
          quality={60}
          alt='About 1'
        />
      </div>
      <div>
```

```
          <p>Original (6.86MB):</p>
          <img
            src='/components/others/about-1.jpg'
            width={500}
            height={300}
            alt='About 1'
          />
        </div>
      </div>
    );
  }
```

W03-P2: 在 About 頁面展示圖片大小

目標：展示優化前後的圖片。 **步驟**：

1. **創建 About 頁面**：
   - 編輯 `app/about_51/page.js`：

```
import Image from 'next/image';

export default function About() {
  return (
    <div className='p-8'>
      <h1 className='text-2xl mb-4'>About Page</h1>
      <div>
        <p>Original (1.1MB):</p>
        <img
          src='/components/others/about-1.jpg'
          width={500}
          height={300}
          alt='About 1'
        />
      </div>
      <div>
        <p>Optimized:</p>
        <Image
          src='/components/others/about-1.jpg'
          width={500}
          height={300}
          alt='About 1'
        />
      </div>
      <div>
        <p>Quality 50:</p>
        <Image
          src='/components/others/about-1.jpg'
          width={500}
          height={300}
          quality={50}
```

```
            alt='About 1'
          />
        </div>
      </div>
    );
  }
```

W04: Supabase 整合

W04-P1: 展示 Header_51 組件

目標：展示帶有 Logo 和導航的標頭。 **步驟**：

1. **創建 Logo_51**：
   - 編輯 `components/Logo_51.js`：

```
export default function Logo_51() {
  return <div className='text-xl font-bold'>Logo</div>;
}
```

2. **創建 Navigation_51**：
   - 編輯 `components/Navigation_51.js`：

```
export default function Navigation_51() {
  return (
    <nav>
      <ul className='flex space-x-4'>
        <li>
          <a href='#' className='hover:underline'>
            Home
          </a>
        </li>
        <li>
          <a href='#' className='hover:underline'>
            About
          </a>
        </li>
      </ul>
    </nav>
  );
}
```

3. **創建 Header_51**：
   - 編輯 `components/Header_51.js`：

```
import Logo_51 from './Logo_51';
import Navigation_51 from './Navigation_51';

export default function Header_51() {
  return (
    <header className='bg-blue-600 text-white p-4 flex justify-
between'>
      <Logo_51 />
      <Navigation_51 />
    </header>
  );
}
```

4. **更新 layout.js**：
    - 編輯 `app/layout.js`：

```
import Header_51 from '../components/Header_51';
import './globals.css';

export default function RootLayout({ children }) {
  return (
    <html lang='en'>
      <body>
        <Header_51 />
        {children}
      </body>
    </html>
  );
}
```

W04-P2: 從 Supabase 創建並獲取 cabins 資料

目標：設置 Supabase 並獲取資料。 **步驟**：

1. **設置 Supabase**：
    - 註冊 Supabase 帳號，創建新專案，獲取 URL 和 Key。
    - 安裝客戶端：`npm install @supabase/supabase-js`
2. **創建資料表**：
    - 編輯 `lib/cabins_51.sql`：

```
CREATE TABLE cabins_51 (
  id SERIAL PRIMARY KEY,
  name TEXT,
  price INT,
```

```
  maxCapacity INT
);
INSERT INTO cabins_51 (name, price, maxCapacity) VALUES
  ('Cabin 1', 250, 4),
  ('Cabin 2', 300, 6),
  ('Cabin 3', 200, 2),
  ('Cabin 4', 400, 8),
  ('Cabin 5', 350, 5),
  ('Cabin 6', 280, 3),
  ('Cabin 7', 320, 7),
  ('Cabin 8', 270, 4);
```

- 在 Supabase 儀表板執行此 SQL。

3. **設置 Supabase 客戶端**：

- 編輯 `lib/supabase.js`：

```
import { createClient } from '@supabase/supabase-js';

const supabaseUrl = 'YOUR_URL';
const supabaseKey = 'YOUR_KEY';
export const supabase = createClient(supabaseUrl, supabaseKey);
```

4. **獲取資料**：

- 編輯 `lib/data-service.js`：

```
import { supabase } from './supabase';

export async function getCabins() {
  const { data, error } = await supabase.from('cabins_51').select('*');
  if (error) throw new Error(error.message);
  return data;
}
```

5. **在控制台顯示資料**：

- 編輯 `app/page.js`：

```
import { getCabins } from '../lib/data-service';

export default async function Home() {
  const cabins = await getCabins();
  console.log(cabins);
  return (
    <div className='p-8'>
      <h1 className='text-2xl'>Home Page</h1>
```

```
      <pre>{JSON.stringify(cabins, null, 2)}</pre>
    </div>
  );
}
```

W04-P3: 在瀏覽器中展示 8 個 cabins

目標：將資料渲染到前端。 **步驟**：

1. **創建 CabinCard_51**：
   - 編輯 components/CabinCard_51.js：

```
export default function CabinCard_51({ cabin }) {
  return (
    <div className='bg-gray-200 p-4 rounded'>
      <h3>{cabin.name}</h3>
      <p>Price: ${cabin.price}</p>
      <p>Capacity: {cabin.maxCapacity}</p>
    </div>
  );
}
```

2. **修改 cabins_51 頁面**：
   - 編輯 app/cabins_51/page.js：

```
import { getCabins } from '../../lib/data-service';
import CabinCard_51 from '../../components/CabinCard_51';

export default async function Cabins() {
  const cabins = await getCabins();
  return (
    <div className='p-8'>
      <h1 className='text-2xl mb-4'>Cabins</h1>
      <div className='grid grid-cols-2 gap-4'>
        {cabins.map((cabin) => (
          <CabinCard_51 key={cabin.id} cabin={cabin} />
        ))}
      </div>
    </div>
  );
}
```

W05: 靜態與動態渲染

W05-P1: 使用 /cabins_51 路由展示靜態和動態渲染

步驟：

1. **創建 cabins_51 路由 (靜態渲染)：**
   - 編輯 `app/cabins_51/page.js`：

```
import { getCabins } from '../../lib/data-service';
import CabinCard_51 from '../../components/CabinCard_51';

export async function generateStaticParams() {
  return [];
}

export default async function Cabins() {
  const cabins = await getCabins();
  return (
    <div className='p-8'>
      <h1 className='text-2xl mb-4'>Cabins (Static)</h1>
      <div className='grid grid-cols-2 gap-4'>
        {cabins.map((cabin) => (
          <CabinCard_51 key={cabin.id} cabin={cabin} />
        ))}
      </div>
    </div>
  );
}
```

2. **測試靜態渲染：**
   - 運行 `npm run build && npm start`，修改 Supabase 中 Cabin 1 的價格 (250 -> 300)，刷新頁面不會更新。
3. **切換為動態渲染：**
   - 修改 `app/cabins_51/page.js`：

```
import { getCabins } from '../../lib/data-service';
import CabinCard_51 from '../../components/CabinCard_51';

export const dynamic = 'force-dynamic';

export default async function Cabins() {
  const cabins = await getCabins();
  return (
    <div className='p-8'>
      <h1 className='text-2xl mb-4'>Cabins (Dynamic)</h1>
      <div className='grid grid-cols-2 gap-4'>
        {cabins.map((cabin) => (
          <CabinCard_51 key={cabin.id} cabin={cabin} />
```

```
          ))}
        </div>
      </div>
    );
}
```

- 再次修改價格 (300 -> 500)，刷新頁面會更新。

W05-P2: 實現 /cabins_51/cabinId 路由並使用 SSG

步驟：

1. **創建動態路由：**
   - 在 `app/cabins_51/[cabinId]` 下創建 `page.js`：

```
import { getCabins } from '../../../lib/data-service';

export async function generateStaticParams() {
  const cabins = await getCabins();
  return cabins.map((cabin) => ({ cabinId: cabin.id.toString() }));
}

export default async function Cabin({ params }) {
  const { cabinId } = params;
  const cabins = await getCabins();
  const cabin = cabins.find((c) => c.id === parseInt(cabinId));

  return (
    <div className='p-8'>
      <h1>{cabin.name}</h1>
      <p>Price: ${cabin.price}</p>
      <p>Capacity: {cabin.maxCapacity}</p>
    </div>
  );
}
```

2. **測試 SSG：**
   - 運行 `npm run build && npm start`，修改 Supabase 中 Cabin 1 的 `maxCapacity`，刷新頁面不會更新。

3. **切換為動態路由：**
   - 修改 `app/cabins_51/[cabinId]/page.js`：

```
export const dynamic = 'force-dynamic';

export default async function Cabin({ params }) {
```

```
  const { cabinId } = params;
  const cabins = await getCabins();
  const cabin = cabins.find((c) => c.id === parseInt(cabinId));

  return (
    <div className='p-8'>
      <h1>{cabin.name}</h1>
      <p>Price: ${cabin.price}</p>
      <p>Capacity: {cabin.maxCapacity}</p>
    </div>
  );
}
```

W06: 導航與測驗

W06-P1: 展示導航

步驟：

1. 修改 **Navigation_51**：
   - 編輯 `components/Navigation_51.js`：

```
export default function Navigation_51() {
  return (
    <nav>
      <ul className='flex space-x-4'>
        <li>
          <a href='/quiz1/p1_51' className='hover:underline'>
            P1
          </a>
        </li>
        <li>
          <a href='/quiz1/p2_51' className='hover:underline'>
            P2
          </a>
        </li>
        <li>
          <a href='/quiz1/p3_51' className='hover:underline'>
            P3
          </a>
        </li>
        <li>
          <a href='/quiz1/p4_51' className='hover:underline'>
            P4
          </a>
        </li>
      </ul>
    </nav>
```

```
    );
  }
```

2. 確保 **Header_51 使用 Navigation_51**：
    - 編輯 `components/Header_51.js`：

```
import Logo_51 from './Logo_51';
import Navigation_51 from './Navigation_51';

export default function Header_51() {
  return (
    <header className='bg-blue-600 text-white p-4 flex justify-
between'>
      <Logo_51 />
      <Navigation_51 />
    </header>
  );
}
```

W06-P2: 實現 /quiz1/p1_51 展示所有 cabins

步驟：

1. **創建 db-oasis.js**：
    - 編輯 `lib/db-oasis.js`：

```
import { supabase } from './supabase';

export async function getCabins() {
  const { data, error } = await supabase.from('cabins_51').select('*');
  if (error) throw new Error(error.message);
  return data;
}
```

2. **創建 Oasis_51 組件**：
    - 編輯 `components/Oasis_51.jsx`：

```
export default function Oasis_51({ cabin }) {
  return (
    <div className='bg-gray-200 p-4 rounded'>
      <h3>{cabin.name}</h3>
      <p>Price: ${cabin.price}</p>
    </div>
```

```
  );
}
```

3. **創建 p1_51 頁面：**
   - 編輯 `app/quiz1/p1_51/page.js`：

```
import { getCabins } from '../../../lib/db-oasis';
import Oasis_51 from '../../../components/Oasis_51';

export default async function P1_51() {
  const cabins = await getCabins();
  return (
    <div className='p-8'>
      <h1>All Cabins</h1>
      <div className='grid grid-cols-2 gap-4'>
        {cabins.map((cabin) => (
          <Oasis_51 key={cabin.id} cabin={cabin} />
        ))}
      </div>
    </div>
  );
}
```

W06-P3: 實現 /quiz1/p2_51 展示靜態博客

步驟：

1. **創建博客表：**
   - 編輯 `lib/db-quiz.js`：

```
import { supabase } from './supabase';

export async function getBlogs() {
  const { data, error } = await supabase.from('blogs_51').select('*');
  if (error) throw new Error(error.message);
  return data;
}
```

   - 在 Supabase 執行 SQL (假設 `lib/db-quiz.js` 會用到)：

```
CREATE TABLE blogs_51 (
  id SERIAL PRIMARY KEY,
  title TEXT,
  img TEXT
```

```
);
INSERT INTO blogs_51 (title, img) VALUES
  ('Blog 1', '/demo/blog_51/photo-1.jpg'),
  ('Blog 2', '/demo/blog_51/photo-2.jpg');
```

2. **創建 Blogs_51 組件：**
   - 編輯 components/Blogs_51.jsx：

```
export default function Blogs_51({ blog }) {
  return (
    <div className='p-4'>
      <h2>{blog.title}</h2>
      <img
        src={blog.img}
        alt={blog.title}
        className='w-full h-48 object-cover'
      />
    </div>
  );
}
```

3. **創建 p2_51 頁面：**
   - 編輯 app/quiz1/p2_51/page.js：

```
import { getBlogs } from '../../../lib/db-quiz';
import Blogs_51 from '../../../components/Blogs_51';

export default async function P2_51() {
  const blogs = await getBlogs();
  return (
    <div className='p-8'>
      <h1>Static Blogs</h1>
      {blogs.map((blog) => (
        <Blogs_51 key={blog.id} blog={blog} />
      ))}
    </div>
  );
}
```

W06-P4: 在 Vercel 上展示所有博客

步驟：

1. **創建 p3_51 頁面：**
   - 編輯 app/quiz1/p3_51/page.js：

```
import { getBlogs } from '../../../lib/db-quiz';
import Blogs_51 from '../../../components/Blogs_51';

export default async function P3_51() {
  const blogs = await getBlogs();
  return (
    <div className='p-8'>
      <h1>All Blogs on Vercel</h1>
      {blogs.map((blog) => (
        <Blogs_51 key={blog.id} blog={blog} />
      ))}
    </div>
  );
}
```

2. **部署到 Vercel**：
    - 推送程式碼到 GitHub。
    - 在 Vercel 上導入儲存庫，設置環境變數 (Supabase URL 和 Key)。
    - 訪問 `/quiz1/p3_51` 檢查。
