感謝你的問題！根據你提供的資料夾結構和任務描述（W01 到 W06），我可以為你總結每一週的重點準備內容。每一週的任務都有其特定的學習目標，這些目標通常圍繞技術技能、工具使用和專案開發的進展。以下是每一週的重點準備內容，幫助你更好地理解課程的進程和需要掌握的技能。

W01: 基礎設置與 Tailwind CSS

重點準備

1. **熟悉 Git 和 GitHub 的基本操作：**
   - 學習如何初始化 Git 儲存庫（`git init`）。
   - 學習如何將本地專案連接到 GitHub 儲存庫（`git remote add origin`）。
   - 學習基本的 Git 指令：`git add`、`git commit`、`git push`。
   - 學習如何在 GitHub 上添加協作者（Collaborators），以便分享專案給老師和助教。
   - **準備工具**：確保你已經安裝 Git（從 git-scm.com 下載），並有一個 GitHub 帳號。
2. **設置 Vite 專案並學習 React 基礎：**
   - 學習如何使用 Vite 創建 React 專案（`npm create vite@latest`）。
   - 熟悉 Vite 專案的結構，例如 `src/` 資料夾、`App.jsx` 和 `main.jsx` 的作用。
   - 學習 React 的基本概念，例如組件（Component）和 JSX 語法。
   - **準備工具**：確保你已經安裝 Node.js（從 nodejs.org 下載），因為 Vite 和 React 依賴 Node.js。
3. **學習 Tailwind CSS 的基本使用：**
   - 學習如何在 Vite 專案中安裝和配置 Tailwind CSS（`npm install -D tailwindcss postcss autoprefixer` 和 `npx tailwindcss init -p`）。
   - 熟悉 Tailwind CSS 的工具類（Utility Classes），例如 `flex`、`justify-center`、`text-4xl`、`bg-gray-100`。
   - 學習如何使用 Tailwind CSS 快速設計簡單的 UI，例如居中顯示姓名和學號。
   - **準備資源**：閱讀 Tailwind CSS 的官方文件（tailwindcss.com），特別是「Utility-First」和「Responsive Design」部分。
4. **實現響應式設計：**
   - 學習 Tailwind CSS 的響應式設計前綴（例如 `sm:`），用來根據螢幕大小調整樣式。
   - 準備實現一個響應式導航欄，小螢幕（< 640px）時垂直排列，大螢幕（> 640px）時水平排列。
   - **準備知識**：了解 CSS 的 Flexbox 佈局（`flex`、`flex-col`、`flex-row`），因為 Tailwind CSS 的響應式設計依賴這些概念。

W02: 組件與響應式設計

重點準備

1. **深入學習 React 組件化：**
   - 學習如何將 UI 分割成多個可重用的組件（例如 `Hero_51`、`About_51`、`Projects_51`、`Skills_51`）。
   - 學習如何通過 props 傳遞資料給組件（例如 `SectionTitle_51` 接受 `title` 屬性）。

- 學習如何使用 `map` 函數動態渲染列表（例如渲染 `navLinks`、`projects` 和 `skills`）。
- **準備知識**：熟悉 React 的組件結構、props 傳遞和列表渲染（參考 React 官方文件 react.dev）。

2. **學習資料管理：**

- 學習如何將資料集中管理，例如在 `data.jsx` 中定義 `navLinks`、`projects` 和 `skills`。
- 準備將資料傳遞給組件，並動態渲染（例如 `Projects_51` 使用 `projects` 資料渲染專案卡片）。
- **準備知識**：了解 JavaScript 的陣列方法（`map`、`forEach`），因為這些方法在 React 中常用於渲染列表。

3. **進階響應式設計：**

- 學習 Tailwind CSS 的 Grid 佈局（`grid`、`grid-cols-2`、`md:grid-cols-3`），用來實現專案和技能的網格排列。
- 準備根據螢幕大小調整佈局，例如小螢幕每行 2 個項目，大螢幕每行 3 個項目。
- **準備知識**：了解 CSS Grid 的基本概念（`grid-template-columns`），因為 Tailwind 的 Grid 類別基於此。

4. **專案結構管理：**

- 學習如何組織 React 專案，將組件放在 `src/components` 資料夾中，主應用邏輯放在 `App_51.jsx` 中。
- 準備將多個組件整合到主應用中（例如 `Navbar_51`、`Hero_51`、`About_51`、`Projects_51`、`Skills_51`）。
- **準備知識**：了解模組化開發的概念，學習如何使用 `import` 和 `export` 管理組件。

W03: Next.js 與圖片優化

重點準備

1. **學習 Next.js 基礎：**

- 學習如何初始化 Next.js 專案（`npx create-next-app@latest`），並選擇 App Router。
- 熟悉 Next.js 的專案結構，特別是 `app/` 資料夾的作用（用於路由）和 `public/` 資料夾（用於靜態資源）。
- 學習 Next.js 的檔案系統路由（File-System Routing），例如 `app/page.js` 對應根路由，`app/about_51/page.js` 對應 `/about_51`。
- **準備資源**：閱讀 Next.js 官方文件（nextjs.org），特別是「App Router」和「Getting Started」部分。

2. **學習圖片優化：**

- 學習 Next.js 的 `<Image>` 組件，用於自動優化圖片（例如壓縮、格式轉換）。
- 準備比較不同品質的圖片大小（例如 `quality={100}`、`quality={60}` 和原始圖片）。
- 學習如何將圖片放在 `public/` 資料夾中，並通過路徑（例如 `/components/others/about-1.jpg`）引用。
- **準備知識**：了解圖片優化的基本概念（例如壓縮、WebP 格式），以及為什麼圖片優化對網站性能重要。

3. **學習 Next.js 的路由：**

- 準備創建多個頁面（例如 `/` 和 `/about_51`），並在頁面之間切換。
- 學習如何在 `app/layout.js` 中定義全域佈局（例如添加共用的標頭或導航）。
- **準備知識**：了解 Next.js 的路由規則，特別是 `page.js` 和 `layout.js` 的作用。

4. **專案結構調整：**

- 學習為什麼 `app/` 和 `components/` 分開（`app/` 用於路由，`components/` 用於可重用組件）。
- 準備將靜態資源（例如圖片）放在 `public/` 資料夾中，並在專案中引用。
- **準備知識**：了解 Next.js 專案的標準結構，以及如何組織檔案以避免路由衝突。

W04: Supabase 整合

重點準備

1. **學習 Supabase 基礎：**

- 學習如何註冊 Supabase 帳號並創建專案（在 supabase.com 上操作）。
- 學習如何獲取 Supabase 的 URL 和 Key，並在專案中配置（`lib/supabase.js`）。
- 學習如何使用 Supabase 的 SQL Editor 創建資料表（例如 `cabins_51` 表）。
- **準備工具**：確保你有一個 Supabase 帳號，並熟悉其儀表板操作。

2. **學習資料庫操作：**

- 學習如何撰寫簡單的 SQL 語句，例如 `CREATE TABLE` 和 `INSERT INTO`（用於創建 `cabins_51` 表並插入資料）。
- 學習如何使用 Supabase 的 JavaScript 客戶端（`@supabase/supabase-js`）查詢資料（例如 `supabase.from('cabins_51').select('*')`）。
- 準備從 Supabase 獲取資料並在前端顯示（例如 `getCabins` 函數）。
- **準備知識**：了解基本的 SQL 語法（`CREATE`、`INSERT`、`SELECT`），以及如何處理非同步資料（`async/await`）。

3. **學習 Next.js 的伺服器端組件：**

- 學習如何在 Next.js 的頁面中執行伺服器端邏輯（例如在 `app/page.js` 中調用 `getCabins`）。
- 準備將從 Supabase 獲取的資料渲染到前端（例如使用 `CabinCard_51` 組件顯示 cabins）。
- **準備知識**：了解 Next.js 的伺服器組件（Server Components），以及如何在頁面中處理非同步資料。

4. **專案結構進階管理：**

- 學習如何將資料服務邏輯放在 `lib/` 資料夾中（例如 `supabase.js` 和 `data-service.js`）。
- 準備將可重用組件（例如 `Header_51`、`CabinCard_51`）放在 `components/` 資料夾中，並在多個頁面中使用。
- **準備知識**：了解如何組織後端邏輯（`lib/`）和前端組件（`components/`），以保持專案的模組化。

W05: 靜態與動態渲染

重點準備

1. **學習 Next.js 的渲染方式：**
    - 學習 Next.js 的靜態網站生成（SSG），使用 `generateStaticParams` 預生成頁面（例如 `/cabins_51/[cabinId]`）。
    - 學習 Next.js 的動態渲染（Dynamic Rendering），使用 `export const dynamic = 'force-dynamic'` 強制每次請求時重新獲取資料。
    - 準備比較 SSG 和動態渲染的差異（例如資料更新的即時性）。
    - **準備資源**：閱讀 Next.js 官方文件的「Data Fetching」和「Rendering」部分，特別是 SSG 和 SSR 的區別。

2. **學習動態路由：**
    - 學習如何在 Next.js 中創建動態路由（例如 `app/cabins_51/[cabinId]/page.js`）。
    - 準備根據 URL 參數（`params.cabinId`）動態獲取資料（例如顯示特定 cabin 的詳情）。
    - **準備知識**：了解 Next.js 的動態路由語法（`[param]`），以及如何從 `params` 中提取參數。

3. **學習資料更新的影響：**
    - 準備測試 SSG 和動態渲染的行為，例如在 Supabase 中更新資料後，SSG 頁面不會立即反映，而動態渲染頁面會更新。
    - 學習如何選擇適合的渲染方式（例如 SSG 適合靜態內容，動態渲染適合即時資料）。
    - **準備知識**：了解網站性能和資料即時性之間的權衡，以及如何根據需求選擇渲染策略。

4. **專案結構與組件重用：**
    - 準備重用現有組件（例如 `CabinCard_51`）在不同頁面（例如 `/cabins_51` 和 `/cabins_51/[cabinId]`）。
    - 學習如何在 `app/layout.js` 中定義共用佈局，確保所有頁面都有標頭（`Header_51`）。
    - **準備知識**：了解 Next.js 的佈局系統（`layout.js`），以及如何在多個頁面中重用組件。

W06: 導航與測驗

重點準備

1. **學習進階路由與導航：**
    - 學習如何在 Next.js 中創建多層路由（例如 `/quiz1/p1_51`、`/quiz1/p2_51`）。
    - 準備更新導航欄（`Navigation_51`），添加指向測驗頁面的連結（P1、P2、P3、P4）。
    - **準備知識**：了解 Next.js 的嵌套路由（Nested Routes），以及如何使用 `<a>` 標籤實現簡單導航。

2. **學習多表資料管理：**
    - 學習如何在 Supabase 中創建第二個資料表（`blogs_51`），並插入資料。
    - 準備從 Supabase 獲取多表資料（例如 `cabins_51` 和 `blogs_51`），並在不同頁面顯示（`/quiz1/p1_51` 顯示 cabins，`/quiz1/p2_51` 顯示 blogs）。
    - **準備知識**：了解如何管理多個資料來源，並在前端分別渲染。

3. **學習部署到 Vercel：**
    - 學習如何將 Next.js 專案部署到 Vercel（通過 GitHub 整合）。

- 準備配置環境變數（`NEXT_PUBLIC_SUPABASE_URL` 和 `NEXT_PUBLIC_SUPABASE_ANON_KEY`），以確保 Supabase 在 Vercel 上正常工作。
- 學習如何測試部署後的網站，確保所有功能（例如導航、資料顯示）正常運作。
- **準備工具**：確保你有一個 Vercel 帳號（在 vercel.com 註冊），並熟悉其部署流程。

4. **專案整合與測驗準備：**

- 準備將之前的組件（例如 `CabinCard_51`）重用在測驗頁面（例如 `/quiz1/p1_51` 使用 `Oasis_51` 顯示 cabins）。
- 學習如何創建新組件（例如 `Blogs_51`）來顯示博客，並整合到測驗頁面（`/quiz1/p2_51` 和 `/quiz1/p3_51`）。
- 準備確保專案結構清晰，所有功能（導航、資料顯示、部署）都能正常運作。
- **準備知識**：回顧之前的學習內容（React 組件、Next.js 路由、Supabase 整合），確保能整合所有技術完成測驗。

總結：每一週的重點準備

週數	重點準備內容	技術/工具
W01	1. Git 和 GitHub 操作 2. Vite + React 專案設置 3. Tailwind CSS 基礎 4. 響應式設計	Git, GitHub, Vite, React, Tailwind CSS
W02	1. React 組件化與 props 2. 資料管理與動態渲染 3. 進階響應式設計（Grid） 4. 專案結構管理	React, Tailwind CSS, JavaScript 陣列方法
W03	1. Next.js 基礎與 App Router 2. 圖片優化（`<Image>` 組件） 3. 路由與頁面創建 4. 專案結構調整	Next.js, App Router, 圖片優化
W04	1. Supabase 設置與資料庫操作 2. 從 Supabase 獲取資料 3. Next.js 伺服器端組件 4. 進階專案結構	Supabase, SQL, Next.js 伺服器組件
W05	1. 靜態網站生成（SSG）與動態渲染 2. 動態路由（`[cabinId]`） 3. 資料更新影響 4. 組件重用與佈局	Next.js 渲染方式, 動態路由
W06	1. 進階路由與導航 2. 多表資料管理（cabins 和 blogs） 3. 部署到 Vercel 4. 專案整合與測驗	Next.js 嵌套路由, Supabase 多表, Vercel 部署
