Final Project 專案目錄結構規範 v1.0

這份文件定義了我們專案的檔案組織方式。請大家在開發時，將檔案放在對應的資料夾中，以避免合併衝突。

📂 根目錄結構 (Root Directory)

```
/cg_final_project/
│
├── 📁 assets/              <-- 靜態資源 (圖片、CSS)
│   ├── 📁 css/
│   │   └── style.css       <-- 全站共用的樣式 (Navbar, Form design)
│   └── 📁 img/
│       └── logo.png
│
├── 📁 config/              <-- 設定檔
│   └── db_connect.php      <-- 資料庫連線設定 (不要重複寫連線程式碼！)
│
├── 📁 includes/            <-- PHP 共用模組 (頁首、頁尾)
│   ├── header.php          <-- 導覽列 (Navbar)
│   └── footer.php          <-- 頁腳
│
├── 📁 js/                  <-- 前端 JavaScript (核心重鎮)
│   ├── 📁 libs/            <-- 第三方函式庫 (Face-api, Three.js 若不透過 CDN 抓)
│   │
│   ├── 📁 graphics/        <-- [組員 A 專區] 圖學渲染邏輯
│   │   ├── scene_init.js   <-- Three.js 場景初始化、光照設定
│   │   ├── loader.js       <-- 負責載入 .gltf 模型
│   │   └── shader.js       <-- 自定義 Blinn-Phong Shader 字串放這裡
│   │
│   ├── 📁 tracking/        <-- [PM 專區] 視覺追蹤邏輯
│   │   └── face_tracker.js <-- 處理 Webcam 與 Face-api 邏輯，回傳 (x,y)
│   │
│   └── main.js             <-- [整合專區] 負責把 tracking 數據傳給 graphics
│
├── 📁 uploads/             <-- [系統生成] 使用者上傳的模型存放處
│   └── (user_id)/          <-- 依使用者 ID 分資料夾 (避免檔名衝突)
│       └── model.gltf
│
├── 📁 api/                 <-- [組員 B 專區] 純後端處理 (沒有 HTML 的 PHP 檔)
│   ├── login_action.php    <-- 處理登入表單 POST
│   ├── register_action.php <-- 處理註冊表單 POST
│   ├── upload_action.php   <-- 處理檔案上傳邏輯、搬移檔案
│   └── logout.php          <-- 執行登出並清除 Session
│
├── index.php               <-- 首頁 (也是登入頁)
├── register.php            <-- 註冊頁面 (HTML Form)
├── gallery.php             <-- 藝廊頁面 (顯示所有模型縮圖)
├── viewer.php              <-- [核心戰場] 展示頁面 (整合 Canvas 與 Webcam)
└── README.md               <-- 專案說明文件
```

📝 詳細檔案職責說明 (File Responsibilities)

1. 公用設定與元件 (Config & Includes)

- `config/db_connect.php`：

- 負責建立 `$conn` 物件連接 MySQL。
- **重要：** 所有人要連資料庫時，直接 `require_once 'config/db_connect.php';` ，不要自己在頁面裡重寫 `mysqli_connect` 。

- `includes/header.php`：
  - 包含 HTML `<head>` 、引入 Bootstrap/Tailwind、以及上方的導覽列 (Navbar)。
  - 利用 `session_start()` 判斷使用者是否登入，顯示「登入」或「登出」按鈕。

2. 前端 JavaScript 分工 (JS Logic)

這是最容易打架的地方，所以我們拆成三個檔案：

- `js/graphics/scene_init.js` **(組員 A)**:
  - 這裡面寫 `Three.js` 的 `init()` , `animate()` 。
  - **介面規定：** 必須提供一個全域變數或函式 `updateCameraPosition(x, y)` ，讓外部可以控制相機。
- `js/tracking/face_tracker.js` **(PM)**:
  - 這裡面寫 `face-api.js` 或 `MediaPipe` 的初始化。
  - 不涉及渲染，只負責算出人臉在畫面中的正規化座標 (-1.0 到 1.0)。
- `js/main.js` **(整合)**:
  - 負責在 `viewer.php` 載入時，把上面兩者串起來。
  - 例如： `FaceTracker.onUpdate((x,y) => Scene.updateCameraPosition(x,y));`

3. 後端邏輯 (Backend Logic - 組員 B)

- `gallery.php`：
  - 從資料庫 `SELECT * FROM models` 。
  - 用 `foreach` 迴圈把模型印成卡片 (Card) 形式。
  - 點擊卡片連結要帶參數： `<a href="viewer.php?id=5">` 。
- `viewer.php`：
  - 這是最重要的頁面。
  - PHP 部分：接收 `$_GET['id']` ，去資料庫查出該模型的 `filepath` 。
  - JS 部分：將 PHP 查到的路徑傳給 `scene_init.js` 去載入。

4. 檔案上傳處理 (api/upload_action.php)

- 這部分最容易出錯，請注意：
  1. 檢查檔案大小 (例如限制 50MB)。
  2. 檢查副檔名 (只允許 `.gltf` , `.glb` , `.bin`)。
  3. 使用 `move_uploaded_file()` 將檔案從暫存區移到 `uploads/` 。
  4. 成功後，才 INSERT 一筆資料到 MySQL。

💡 PM 的特別叮嚀 (開發守則)

1. **路徑問題**：PHP 的 `include` 和 HTML 的 `<script src>` 路徑寫法不同。建議統一使用相對路徑，或者定義一個 `BASE_URL` 常數。
2. **Git 忽略檔 (.gitignore)**：
   - 請務必在專案根目錄建立 `.gitignore` 檔案。
   - 內容加入 `/uploads/` 。我們**不要**把使用者上傳的大型 3D 檔案推送到 GitHub/GitLab，這會讓儲存庫爆炸。
3. **除錯模式**：開發期間，請在 PHP 檔案最上方加入以下兩行，這樣有錯才會噴出來，不用猜：

```
ini_set('display_errors', 1);
error_reporting(E_ALL);
```