

Resumen del Flujo de Datos:

- **Carga Inicial:** index.html → carga script.js → llama a obtener_usuarios.php → lee DB → devuelve JSON → JS pinta las Tarjetas + Modelos 3D.
- **Interacción Usuario (Hover):** Usuario pasa mouse → CSS detecta :hover → Oculta → Muestra <model-viewer> (que carga el .glb).
- **Interacción Usuario (Contacto):** Clic en "Contactar" → modal_contacto.js abre Modal → Rellena email oculto → Usuario envía → enviar_contacto.php usa PHPMailer → 📩 ¡Email enviado!
- id="categories-container" (Para los botones de filtro).
- id="cards-container" (Para las tarjetas de personajes).
- id="search" (Para el input del buscador).
- id="modal-contacto" (Para el contenedor del popup).
- id="form-contacto" (Para el formulario dentro del modal).
- id="email-destinatario" (El input oculto dentro del form).

Organización y gestión del código.

1. El <HEAD>:

Aquí cargamos los recursos externos que subiste o mencionaste.

- `<link rel="stylesheet" href="style.css">`
 - **Qué hace:** Carga el archivo style.css (o style2.css según tu último diseño RPG).
 - **Integración:** Define las clases como .character-card, .modal-overlay, etc., que usaremos en el cuerpo.
 - `<script type="module" ... model-viewer ...>`
 - **Qué hace:** Carga el motor 3D de Google.
 - **Por qué:** Reemplaza tu antiguo importmap de Three.js (que era muy pesado). Esto permite que la etiqueta <model-viewer> funcione más abajo.
-

2. La Navegación (Categorías): <nav id="categories-container">

Esta sección empieza vacía en el HTML.

- **El proceso de llenado:**
 1. El archivo `script.js` se despierta al cargar la página.
 2. Llama a `obtener_categorias.php` (Backend).
 3. Este PHP consulta la base de datos (`conexion.php` → Tabla `categories`) y devuelve un JSON: `[{"id":1, "nombre":"Backend"}, ...]`.
 4. `script.js` recibe el JSON y crea botones `<button class="cat-btn">` dentro de este `div`.
 - **HTML:** contenedor vacío con el ID: `id="categories-container"`.
-

3. El Grid de Personajes(Trabajadores): <div id="cards-container">

- **La Integración de Archivos:**
 1. `script.js` pide datos a `obtener_usuarios.php`.
 2. `obtener_usuarios.php` devuelve una lista de todos los usuarios, incluyendo las rutas de sus archivos:
 - `avatar2D: "uploads/avatars/foto.png"`
 - `avatar3D: "uploads/models3d/avatar_Jorge.glb"`
 3. `script.js` genera un `<article class="character-card">` por cada usuario e inyecta el HTML dentro del `div`.
- **La Lógica 2D/3D (Hover):**
 - El JS genera este código HTML dinámicamente:

HTML

```
<div class="card-media">
     <model-viewer
    src="avatar_Jorge.glb" ...> </model-viewer> </div>
```

- CSS (`style.css`) se encargan de ocultar la imagen y mostrar el `model-viewer` cuando ocurre el `:hover`.
-

4. El Modal de Contacto: <div id="modal-contacto">

Bloque "invisible" (`display: none` en CSS) hasta que alguien hace clic.

- **Conexión Frontend (JS):**
 - Al pulsar "Contactar" en una tarjeta, se dispara la función `contactar(id)` en `modal_contacto.js`.
 - Este script busca el `div` con ID `modal-contacto` y le añade la clase `.active` para mostrarlo.

- **Conexión de Datos (PHP):**
 - Antes de mostrarse, JS llama a `obtener_info.php?id=5`.
 - Este PHP devuelve el email del profesional (ej: `andrea@email.com`).
 - JS inyecta ese email en el campo oculto `<input type="hidden" id="email-destinatario">`.
 - **Conexión de Envío (PHP Mailer):**
 - Al darle a "Enviar", `modal_contacto.js` detiene el envío normal y manda los datos por AJAX a `enviar_contacto.php`.
 - `enviar_contacto.php` (archivo con logs y SMTP) lee credenciales del archivo `.env`, conecta con Gmail y despacha el correo.
-

5. El Buscador: `<input id="search">`

- **Integración:**
 - No necesita PHP.
 - `script.js` descarga *todos* los usuarios al principio y los guarda en memoria (en la variable `todosLosUsuarios`).
 - Cuando escribes en este input, JS filtra esa lista en tiempo real y vuelve a pintar el `#cards-container`.