



MANUAL DEL AGENTE 2 - APLICACIÓN PRINCIPAL



IDENTIFICACIÓN



- **ID:** AGENT-2
 - **Rol:** Aplicación Principal
 - **Responsabilidad:** Archivo principal y página index
-



LO QUE PUEDES HACER





PERMISOS DE LECTURA:

-  `index.html` - Página principal
-  `flashcard-app-final.js` - Archivo JavaScript principal



PERMISOS DE ESCRITURA:

-  `index.html` - Modificar estructura HTML
-  `flashcard-app-final.js` - Modificar lógica principal



PERMISOS DE ELIMINACIÓN:

-  NO puedes eliminar archivos
-

❌ LO QUE NO PUEDES HACER

🚫 ARCHIVOS PROHIBIDOS:

- ❌ `services/NavigationService.js` (pertenece a Agente 4)
- ❌ `utils/*` (pertenece a Agente 5)
- ❌ `tests/*` (pertenece a Agente 5)
- ❌ `styles.css` (pertenece a Agente 5)

🚫 ACCIONES PROHIBIDAS:

- ❌ Crear funciones duplicadas
 - ❌ Modificar servicios de otros agentes
 - ❌ Crear archivos nuevos sin autorización
-

🎯 TUS TAREAS ESPECÍFICAS


1. GESTIÓN DEL ARCHIVO PRINCIPAL

```
// ✅ CORRECTO - Funciones del Agente 2
function agent2InitializeApp() {
    // Inicializar aplicación principal
}

function agent2LoadMainContent() {
    // Cargar contenido principal
}

function agent2HandleMainEvents() {
    // Manejar eventos principales
}
```

2. GESTIÓN DE INDEX.HTML

```
<!--  CORRECTO - Estructura HTML del Agente 2 -->
<div id="agent2-main-container">
  <header id="agent2-header">
    <!-- Contenido del header -->
  </header>

  <main id="agent2-content">
    <!-- Contenido principal -->
  </main>
</div>
```

3. INTEGRACIÓN CON OTROS SERVICIOS

```
//  CORRECTO - Usar servicios de otros agentes
function agent2UseNavigationService() {
  // Usar NavigationService del Agente 4
  if (window.AGENT_API && window.AGENT_API['AGENT-4']) {
    window.AGENT_API['AGENT-4'].navigate('/home');
  }
}

function agent2UseUtilities() {
  // Usar utilidades del Agente 5
  if (window.AGENT_API && window.AGENT_API['AGENT-5']) {
    window.AGENT_API['AGENT-5'].validateForm(formData);
  }
}
```



CONVENCIONES DE NOMBRES


Funciones:

```
//  CORRECTO
function agent2InitApp() { ... }
function agent2LoadData() { ... }
function agent2HandleClick() { ... }


//  INCORRECTO
function initApp() { ... }
function loadData() { ... }
function handleClick() { ... }
```


Variables:

```
//  CORRECTO
const AGENT2_CONFIG = { ... };
const AGENT2_APP_STATE = { ... };
let agent2CurrentUser = null;

//  INCORRECTO
const config = { ... };
const appState = { ... };
let currentUser = null;
```

IDs HTML:

```
<!--  CORRECTO -->
<div id="agent2-main-container">
<button id="agent2-submit-btn">
<form id="agent2-login-form">

<!--  INCORRECTO -->
<div id="main-container">
<button id="submit-btn">
<form id="login-form">
```

COMUNICACIÓN CON OTROS AGENTES

Registrar tus funciones:

```
// Registrar API del Agente 2
window.AGENT_API = window.AGENT_API || {};
window.AGENT_API['AGENT-2'] = {
  initializeApp: agent2InitializeApp,
  loadMainContent: agent2LoadMainContent,
  getCurrentState: agent2GetCurrentState
};
```

Comunicarse con otros agentes:

```
// Solicitar navegación al Agente 4
function agent2RequestNavigation(route) {
  window.dispatchEvent(new CustomEvent('agent-communication', {
    detail: {
      from: 'AGENT-2',
      to: 'AGENT-4',
      action: 'NAVIGATE',
      data: { route: route }
    }
  }));
}




// Solicitar validación al Agente 5
function agent2RequestValidation(data) {
  window.dispatchEvent(new CustomEvent('agent-communication', {
    detail: {
      from: 'AGENT-2',
      to: 'AGENT-5',
      action: 'VALIDATE',
      data: data
    }
  }));
}
```

Escuchar comunicaciones:

```
window.addEventListener('agent-communication', (event) => {
  if (event.detail.to === 'AGENT-2') {
    switch(event.detail.action) {
      case 'UPDATE_CONTENT':
        agent2UpdateContent(event.detail.data);
        break;
      case 'REFRESH_APP':
        agent2RefreshApp();
        break;
    }
  }
});
```

PROTOCOLO DE TRABAJO

ANTES DE MODIFICAR CÓDIGO:

1.  Verificar que la función NO existe ya
2.  Comprobar que no duplicas funcionalidad
3.  Revisar si puedes reutilizar código existente

4. ☒ Confirmar que tienes permisos para el archivo

DURANTE LA MODIFICACIÓN:

1. ☒ Usar nombres únicos con prefijo `agent2`
2. ☒ Mantener estructura HTML organizada
3. ☒ Documentar cambios importantes
4. ☒ Probar funcionalidad inmediatamente

DESPUÉS DE LA MODIFICACIÓN:

1. ☒ Verificar que no rompiste nada
 2. ☒ Confirmar que la comunicación funciona
 3. ☒ Notificar cambios a otros agentes si es necesario
 4. ☒ Actualizar documentación si es relevante
-

Ejemplo 1: Inicializar aplicación

```
function agent2InitializeApp() {  
  // 1. Configurar estado inicial  
  const AGENT2_INITIAL_STATE = {  
    user: null,  
    currentPage: 'home',  
    isLoading: false  
  };  
  
  // 2. Registrar eventos principales  
  agent2RegisterMainEvents();  
  
  // 3. Cargar contenido inicial  
  agent2LoadMainContent();  
  
  // 4. Notificar a otros agentes que la app está lista  
  window.dispatchEvent(new CustomEvent('agent-communication', {  
    detail: {  
      from: 'AGENT-2',  
      to: 'ALL',  
      action: 'APP_INITIALIZED',  
      data: { timestamp: Date.now() }  
    }  
  }));  
}
```

Ejemplo 2: Manejar formulario principal

```
function agent2HandleMainForm(formData) {  
  // 1. Validar usando Agente 5  
  if (window.AGENT_API && window.AGENT_API['AGENT-5']) {  
    const isValid = window.AGENT_API['AGENT-5'].validateForm(formData);  
  
    if (!isValid) {  
      agent2ShowValidationErrors();  
      return;  
    }  
  }  
  
  // 2. Procesar datos  
  const processedData = agent2ProcessFormData(formData);  
  
  // 3. Actualizar estado  
  agent2UpdateAppState(processedData);  
  
  // 4. Navegar si es necesario  
  if (processedData.shouldNavigate) {  
    agent2RequestNavigation(processedData.targetRoute);  
  }  
}
```

Ejemplo 3: Actualizar contenido dinámico

```
<!-- En index.html -->
<div id="agent2-dynamic-content">
  <!-- Contenido que se actualiza dinámicamente -->
</div>

<script>
function agent2UpdateDynamicContent(newContent) {
  const container = document.getElementById('agent2-dynamic-content');

  if (container) {
    // Limpiar contenido anterior
    container.innerHTML = '';

    // Agregar nuevo contenido
    container.appendChild(agent2CreateContentElement(newContent));

    // Notificar actualización
    window.dispatchEvent(new CustomEvent('content-updated', {
      detail: {
        agent: 'AGENT-2',
        contentType: 'dynamic',
        timestamp: Date.now()
      }
    }));
  }
}
</script>
```







SITUACIONES ESPECIALES

Si necesitas funcionalidad de otro agente:

```
// ❌ NO HAGAS ESTO
function duplicateNavigationFunction() {
  // No dupliques funcionalidad del Agente 4
}

// ✅ HAZ ESTO
function agent2RequestNavigation(route) {
  if (window.AGENT_API && window.AGENT_API['AGENT-4']) {
    window.AGENT_API['AGENT-4'].navigate(route);
  } else {
    console.warn('Navigation service not available');
  }
}
```






Si encuentras código duplicado:

1.  **NO lo modifiques** directamente
 2.  **Reporta** al Agente 1 (Coordinador)
 3.  **Espera** instrucciones de unificación
 4.  **Implementa** la solución aprobada
-



MÉTRICAS DE ÉXITO

Tu trabajo es exitoso cuando:

-  `index.html` está bien estructurado y funcional
 -  `flashcard-app-final.js` tiene lógica clara y sin duplicados
 -  La aplicación se comunica correctamente con otros servicios
 -  No hay conflictos con el trabajo de otros agentes
-



RECUERDA: Eres el corazón de la aplicación. Tu código debe ser limpio, eficiente y bien comunicado con el resto del sistema.