

Universidad Autónoma de Baja California
Facultad de Ingeniería

Asignatura

36297 - Ingeniería de Software

POR LA REALIZACIÓN PLENA DEL SER

Equipo

Jayson Chico Vargas

Ricardo Emmanuel Romo Ruiz

Matrículas

1183976

1182850

Profesor

Olguin Espinoza Jose Martin

Trabajo

Meta 1.1 - Crear una PWA con asistencia de la IA

Ciclo escolar

2026 – 1

Fecha: 27 de Enero del 2026

Bitácora de uso de la IA

Prompt:

Se tiene como objetivo construir una Lista de Tareas, Podrías generar los pilares técnicos (Manifest y SW)? El Service Worker su código debe ser en JavaScript

Código generado:

Web App Manifest (*manifest.json*)

```
JSON
{
  "name": "Lista de Tareas",
  "short_name": "Tareas",
  "description": "Aplicación para gestionar tareas diarias",
  "start_url": "/index.html",
  "display": "standalone",
  "background_color": "#ffffff",
  "theme_color": "#4CAF50",
  "orientation": "portrait",
  "icons": [
    {
      "src": "/icons/icon-192.png",
      "sizes": "192x192",
      "type": "image/png"
    },
    {
      "src": "/icons/icon-512.png",
      "sizes": "512x512",
      "type": "image/png"
    }
  ]
}
```

Service Worker (*sw.js*) – en JavaScript

```
JavaScript
const CACHE_NAME = 'lista-tareas-v1';
const ASSETS = [
  '/',
  '/index.html',
  '/styles.css',
  '/app.js',
  '/manifest.json'
];

// Instalación
self.addEventListener('install', event => {
  event.waitUntil(
    caches.open(CACHE_NAME).then(cache => {
      return cache.addAll(ASSETS);
    })
  )
});
```

```

    );
  });

  // Activación (limpieza de caché vieja)
  self.addEventListener('activate', event => {
    event.waitUntil(
      caches.keys().then(keys => {
        return Promise.all(
          keys
            .filter(key => key !== CACHE_NAME)
            .map(key => caches.delete(key))
        );
      })
    );
  });

  // Intercepción de peticiones
  self.addEventListener('fetch', event => {
    event.respondWith(
      caches.match(event.request).then(response => {
        return response || fetch(event.request);
      })
    );
  });
});

```

Prompt:

Siguiente paso, ahora tocaría crear el código para el .html

Código generado:

index.html

Este HTML es simple, semántico y escalable, ideal como base.

```

HTML
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Lista de Tareas</title>

  <!-- Manifest PWA -->
  <link rel="manifest" href="manifest.json">

  <!-- Color del tema (Android / navegador) -->
  <meta name="theme-color" content="#4CAF50">

  <!-- Estilos -->
  <link rel="stylesheet" href="styles.css">
</head>

```

```

<body>

  <header>
    <h1><img alt="notepad icon" data-bbox="218 158 232 172"/> Lista de Tareas</h1>
  </header>

  <main>
    <section class="task-input">
      <input
        type="text"
        id="taskInput"
        placeholder="Escribe una nueva tarea..."
        aria-label="Nueva tarea"
      >
      <button id="addTaskBtn">Agregar</button>
    </section>

    <section class="task-list">
      <ul id="taskList">
        <!-- Las tareas se insertan aquí dinámicamente -->
      </ul>
    </section>
  </main>

  <footer>
    <p>Lista de Tareas PWA © 2026</p>
  </footer>

  <!-- Lógica de la app -->
  <script src="app.js"></script>

</body>
</html>

```

Prompt:

De este listado:

- Primero la estructura
- La persistencia local
- Finalmente la conversión PWA.

Qué es lo que ya se lleva resuelto y que no?

Prompt:

Vamos por pasos, primero sigue la Implementación de la persistencia local

Código generado:

app.js – Persistencia local

JavaScript

```
const taskInput = document.getElementById('taskInput');
const addTaskBtn = document.getElementById('addTaskBtn');
const taskList = document.getElementById('taskList');

const STORAGE_KEY = 'tareas';

// -----
// Utilidades LocalStorage
// -----
function getTasks() {
  return JSON.parse(localStorage.getItem(STORAGE_KEY)) || [];
}

function saveTasks(tasks) {
  localStorage.setItem(STORAGE_KEY, JSON.stringify(tasks));
}

// -----
// Renderizado
// -----
function renderTasks() {
  const tasks = getTasks();
  taskList.innerHTML = '';

  tasks.forEach(task => {
    const li = document.createElement('li');
    li.textContent = task.texto;

    const deleteBtn = document.createElement('button');
    deleteBtn.textContent = '✖';
    deleteBtn.addEventListener('click', () => deleteTask(task.id));

    li.appendChild(deleteBtn);
    taskList.appendChild(li);
  });
}

// -----
// Acciones
// -----
function addTask() {
  const texto = taskInput.value.trim();
  if (!texto) return;

  const tasks = getTasks();
  tasks.push({
    id: Date.now(),
    texto
  });

  saveTasks(tasks);
}
```

```

    taskInput.value = '';
    renderTasks();
  }

  function deleteTask(id) {
    const tasks = getTasks().filter(task => task.id !== id);
    saveTasks(tasks);
    renderTasks();
  }

  // -----
  // Eventos
  // -----
  addTaskBtn.addEventListener('click', addTask);

  taskInput.addEventListener('keypress', e => {
    if (e.key === 'Enter') addTask();
  });

  // -----
  // Inicialización
  // -----
  renderTasks();

```

Prompt:

"Cuando confirmes que esto funciona" Cómo podría verificar que todo va en orden?

Prompt:

"Abre la app en el navegador" Cómo la abro?

Prompt:

Bien, con la extensión del Live Server ya lo pude abrir en mi navegador (Firefox) agregué una tarea, cerré la página, la volví abrir desde el VS Code y ahí seguía la tarea, la persistencia funciona.

Podrías realizar el siguiente paso?

Código generado:

-Abre tu *app.js* y **al final del archivo**, agrega esto:

```

JavaScript
// -----
// Registro del Service Worker
// -----
if ('serviceWorker' in navigator) {
  window.addEventListener('load', () => {

```

```
navigator.serviceWorker
  .register('sw.js')
  .then(() => console.log('Service Worker registrado correctamente'))
  .catch(err => console.error('Error al registrar el Service Worker:', err));
});
}
```

Prompt:

"Tu proyecto debería verse así:"

Los 4 Archivos que me has pasado el código están en la misma carpeta, pero "styles.css" o una carpeta de imagenes no la habíamos comentando antes, son necesarios para el funcionamiento?

Nota: Añadí lo que mencionaste al "app.js"

Prompt:

Cómo agrego los iconos que necesita el html (La página) para que funcione lo de la instalación?

Prompt:

Me trabé en el paso 6, cómo abro "DevTools" y lo demás para ver si ya funciona? Todo lo anterior ya estpa realizado

Prompt:

"El recurso de "http://127.0.0.1:5500/Taller1/styles.css" se bloqueó debido a la falta de coincidencia del tipo MIME ("text/html") (X-Content-Type-Options: nosniff)."

"Live reload enabled."

"Service Worker registrado correctamente"

"GET

http://127.0.0.1:5500/Taller1/manifest.json

[HTTP/1.1 404 Not Found 15ms]"

Respuesta:

O quitar la línea del HTML que lo importa:

```
JavaScript
<!-- <link rel="stylesheet" href="styles.css"> -->
```