Código	INSTRUMENTO DE EVALUACIÓN	Revisión:
F-SGC-033		**00**

Universidad Tecnológica Metropolitana

DATOS GENERALES DEL INSTRUMENTO.

División: **TIC**

FDC*/Carrera: **Ingeniería de Gestión de Desarrollo de Software**

Asignatura: **Aplicaciones Web Progresivas**

Cuat.-Gpo(s): **10 A, B** Fecha de aplicación: **Septiembre 2025**

Unidad(es) de aprendizaje y/o tema(s) a evaluar.

Unidad I. Tipos de proyectos y normativa.

Especificar con una “X” el tipo de instrumento de evaluación a utilizar (señalar sólo uno).

Tec. evaluación para el SABER				Tec. evaluación para el SABER HACER + SER			
	Prueba oral (entrevista)		Otro (Especificar):		Proyectos		Otro (Especificar):
	Prueba escrita			x	Prácticas, ejercicios, demostraciones		
	Trabajo investigación				Rúbrica		
	Ensayo, informe				Lista de cotejo		
					Guía de observación		

Profesor(es) de la asignatura: IDS. Norma Lorena Esquivel Pech

Nombre del alumno: Hacer referencia a la lista de asistencia septiembre de 2025 Calificación (puntaje): 10%

CONTENIDO DEL INSTRUMENTO DE EVALUACIÓN

PRIMER PARCIAL

ACTIVIDAD 2. HERRAMIENTAS DE EJECUCIÓN Y DESARROLLO

Unidad de aprendizaje: El alumno establecerá entornos de desarrollo para generar aplicaciones web progresivas.
Tema 2.- Herramientas de ejecución y desarrollo.
Material: Entornos de desarrollo
Forma de trabajo: Individual.
Instrucciones:

Crear una PWA básica que use un Service Worker para cachear archivos estáticos y permitir que la app funcione de manera offline

¿Qué estructura debe tener mi proyecto?
mipwa/

- Index.html
- App.js
- Service-worker.js
- Manifest.json
- Styles.css

**FDC: familia de carreras*

¿Qué debe contener mi index?
Tu index debe contener la estructura que generalmente tiene un HTML cualquiera, donde llames a tu manifest y tu style.
Dentro de tu etiqueta `body` del HTML agrega un encabezado denominado Mis Notas, junto con un textarea que tenga un id que se llame **nota** y un botón con evento onclick que se llame **guardarNota(),** por último, llama a tu scrip denominado app.js

¿Qué debe contener mi app.js?
Aquí debe contener la función guardarNota(), el cuál va a contener al final un alert que diga que la nota fue guardada.

```
function guardarNota() {
  const texto = document.getElementById('nota').value;
  localStorage.setItem('nota', texto);
  alert('¡Nota guardada!');
}
```

Seguidamente, vas a “cargar la nota” al iniciar tu aplicación web.

```
document.addEventListener('DOMContentLoaded', () => {
  const texto = localStorage.getItem('nota');
  if (texto) {
    document.getElementById('nota').value = texto;
  }
});
```

Por último, vamos a “registrar” todos estos movimientos realizados en el Service Worker.

```
if ('serviceWorker' in navigator) {
  navigator.serviceWorker.register('service-worker.js')
    .then(() => console.log('Service Worker registrado'))
    .catch(error => console.error('Fallo en el registro', error));
}
```

¿Qué debe contener mi service-worker.js?
Para trabajar con el service worker vamos a apoyarnos de los elementos de la *caché*, y le vamos a indicar al service worker que elementos de nuestra estructura va a “guardar dentro del navegador” o va a “cachear”

```
const CACHE_NAME = 'notas-pwa-v1';
const ARCHIVOS_A_CACHEAR = [
  '/',
  '/index.html',
  '/styles.css',
  '/app.js',
  '/manifest.json'
];
```

Seguidamente vamos a instalar el service worker dentro del navegador y vamos a mantener los archivos “cacheados”

```
self.addEventListener('install', (event) => {
  event.waitUntil(
    caches.open(CACHE_NAME)
      .then(cache => {
```

```
        return cache.addAll(ARCHIVOS_A_CACHEAR);
      })
  );
});
```

Mientras nos encontremos dentro de la red, vamos a interceptar las peticiones de la misma, utilizando fetch

```
self.addEventListener('fetch', (event) => {
  event.respondWith(
    caches.match(event.request)
      .then(respuesta => {
        return respuesta || fetch(event.request);
      })
  );
});
```

¿Qué debe contener mi manifest.json?

Por último, en el manifest vamos a poner los elementos que contiene nuestra PWA.

```
{
  "name": "Notas PWA",
  "short_name": "Notas",
  "start_url": "/index.html",
  "display": "standalone",
  "background_color": "#ffffff",
  "theme_color": "#4CAF50",
  "icons": [
    {
      "src": "icon.png",
      "sizes": "192x192",
      "type": "image/png"
    }
  ]
}
```

Ya que tenemos todo esto, ¿Cómo comprobamos que realmente funciona?, podemos colocar todos estos elementos en un servidor local o si utilizamos VSCode para su desarrollo se puede quedar en una carpeta, abre tu index en Chrome, (como vimos es el navegador que más tiene compatibilidad con los services worker) puedes probar con otros navegadores y registrar como funciona, una vez que hayas hecho las pruebas con el internet, apaga tu wifi y recarga la página, debe funcionar.

CRITERIOS DE EVALUACION

La prueba escrita tiene un valor de 10 puntos de una escala del 1 al 10 que en este momento del saber equivale en su calificación al 10%.

VALIDACION DE LA ACADEMIA

Nombre de los integrantes de la academia	Firma
Ing. Norma Lorena Esquivel Pech	