const CACHE\_NAME = 'rama-ems-v2.0';

const STATIC\_CACHE = 'rama-ems-static-v2.0';

const DYNAMIC\_CACHE = 'rama-ems-dynamic-v2.0';

const urlsToCache = [

'/',

'/index.html',

'/manifest.json',

'https://cdn.tailwindcss.com',

];

// Install event - cache static assets

self.addEventListener('install', (event) => {

event.waitUntil(

Promise.all([

caches.open(STATIC\_CACHE).then((cache) => cache.addAll(urlsToCache)),

caches.open(DYNAMIC\_CACHE)

]).then(() => {

console.log('Service Worker: Multi-year EMS assets cached');

self.skipWaiting();

})

);

});

// Activate event - clean up old caches

self.addEventListener('activate', (event) => {

event.waitUntil(

caches.keys().then((cacheNames) => {

return Promise.all(

cacheNames.map((cacheName) => {

if (cacheName !== STATIC\_CACHE && cacheName !== DYNAMIC\_CACHE) {

console.log('Service Worker: Deleting old cache', cacheName);

return caches.delete(cacheName);

}

})

);

}).then(() => {

console.log('Service Worker: Multi-year EMS activated');

self.clients.claim();

})

);

});

// Fetch event - serve from cache, fallback to network

self.addEventListener('fetch', (event) => {

event.respondWith(

caches.match(event.request).then((response) => {

if (response) {

return response;

}

return fetch(event.request).then((response) => {

if (!response || response.status !== 200 || response.type !== 'basic') {

return response;

}

const responseToCache = response.clone();

caches.open(DYNAMIC\_CACHE).then((cache) => {

cache.put(event.request, responseToCache);

});

return response;

}).catch(() => {

if (event.request.destination === 'document') {

return caches.match('/index.html');

}

});

})

);

});

// Background sync for multi-year data

self.addEventListener('sync', (event) => {

if (event.tag === 'background-sync-multiyear') {

event.waitUntil(doBackgroundSync());

}

});

async function doBackgroundSync() {

console.log('Service Worker: Multi-year background sync triggered');

// Sync academic year data when online

}