

Offline-first PWA with OPFS deployed to Github Pages

1. Set up Sveltekit project

1. Install Sveltekit

```
npx sv create B01
```

Which template would you like?

- SvelteKit minimal

Add type checking with Typescript?

- no

What would you like to add to your project?

- prettier
- eslint
- tailwindcss

tailwindcss: Which plugins would you like to add?

- don't select any

Which package manager do you want to install dependencies with?

- npm

2. Open VsCode

```
cd B01  
code .
```

3. Adjust tailwind.config.cjs

```
/** @type {import('tailwindcss').Config} */  
export default {  
  content: ['./src/**/*.html,js,svelte,ts'],  
  
  theme: {  
    extend: {  
      zIndex: {
```

```

    '100': '100',
    '1000': '1000',
    '2000': '2000',
    '3000': '3000',
    '5000': '5000',
    '10000': '10000',
    '20000': '20000',
  },
  scale: {
    '200': '2.00',
    '250': '2.50',
    '300': '3.00',
  }
}
},
plugins: []
};

```

4. Adjust app.html

```

<!doctype html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <meta name="theme-color" content="#FFFFFF">
    <!--link rel="manifest" href="%sveltekit.assets%/manifest.json" /-->
    <link rel="icon" href="%sveltekit.assets%/favicon.png" />
    <title>B01</title>
    <meta
      name="viewport"
      content="width=device-width, initial-scale=1.0, maximum-scale=1.0,
user-scalable=0, interactive-widget=resizes-visual"
    >
    %sveltekit.head%
  </head>
  <body data-sveltekit-preload-data="hover" class="overscroll-contain">
    <div style="display: contents">%sveltekit.body%</div>
  </body>
</html>

```

2. Init Git

```

git init
git branch -M main
git add .
git commit -m "initial commit"

```

3. Set up SPA

1. Install adapter-static

<https://kit.svelte.dev/docs/adapter-static>

```
npm install -D @sveltejs/adapter-static
```

2. set svelte.config.js

```
import adapter from '@sveltejs/adapter-static';

/** @type {import('@sveltejs/kit').Config} */
const config = {
  kit: {
    adapter: adapter({
      fallback: '404.html'
    }),
    /**paths: {
      base: process.argv.includes('dev') ? '' : process.env.BASE_PATH
    }*/
  }
};

export default config;
```

3. Add +layout.js to routes

```
export const prerender = true;
export const ssr = false;
```

4. commit modifications

```
git add .
git commit -m "SPA added"
```

4. Set up PWA

1. add icons folder to static folder

2. add manifest.json to static folder

```
{
  "id": "ffd-B01-pwa",
  "short_name": "B01",
  "start_url": "/B01/", // ez a jó!
  "scope": "/B01/",    // ez a jó!
  "display": "standalone",
  "orientation": "portrait",
  "theme_color": "#A3E635",
  "background_color": "#ffffff",
  "dir": "ltr",
  "lang": "en",
  "icons": [
    {
      "src": "icons/icon192.png",
      "sizes": "192x192",
      "type": "image/png"
    },
    {
      "src": "icons/icon512.png",
      "sizes": "512x512",
      "type": "image/png"
    }
  ]
}
```

3. add manifest link to app.html

```
<link rel="manifest" href="%sveltekit.assets%/manifest.json" />
```

4. add service-worker.js to scr folder

<https://kit.svelte.dev/docs/service-workers>

```
/// <reference types="@sveltejs/kit" />
import { build, files, version } from '$service-worker';

const CACHE = `cache-${version}`;

const ASSETS = [
  ...build, // the app itself
  ...files  // everything in `static`
];

self.addEventListener('install', (event) => {
  async function addFilesToCache() {
    const cache = await caches.open(CACHE);
    await cache.addAll(ASSETS);
  }
});
```

```
    }
    event.waitUntil(addFilesToCache());
  });

self.addEventListener('activate', (event) => {
  async function deleteOldCaches() {
    for (const key of await caches.keys()) {
      if (key !== CACHE) await caches.delete(key);
    }
  }
  event.waitUntil(deleteOldCaches());
});

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

  if (event.request.method !== 'GET') return;

  async function respond() {
    const url = new URL(event.request.url);
    const cache = await caches.open(CACHE);
    if (ASSETS.includes(url.pathname)) {
      const response = await cache.match(url.pathname);
      if (response) {
        return response;
      }
    }

    try {
      const response = await fetch(event.request);

      if (!(response instanceof Response)) {
        throw new Error('invalid response from fetch');
      }

      if (response.status === 200) {
        cache.put(event.request, response.clone());
      }

      /* Insert cross-origin isolation headers here and
      comment out next line*/
      return response;
    } catch (err) {
      const response = await cache.match(event.request);
      if (response) {
        return response;
      }
      throw err;
    }
  }

  event.respondWith(respond());
});
```

5. commit modifications

```
git add .  
git commit -m "PWA added"
```

5. Set up Github Pages

1. Add deploy.yml to B01/.github/workflows

<https://kit.svelte.dev/docs/adapters-static>

```
name: Deploy to GitHub Pages  
  
on:  
  push:  
    branches: 'main'  
  
jobs:  
  build_site:  
    runs-on: ubuntu-latest  
    steps:  
      - name: Checkout  
        uses: actions/checkout@v4  
  
      - name: Install Node.js  
        uses: actions/setup-node@v4  
        with:  
          node-version: 20  
          cache: npm  
  
      - name: Install dependencies  
        run: npm install  
  
      - name: build  
        env:  
          BASE_PATH: '${{ github.event.repository.name }}'  
        run: |  
          npm run build  
  
      - name: Upload Artifacts  
        uses: actions/upload-pages-artifact@v3  
        with:  
          # this should match the `pages` option in your adapter-static options  
          path: 'build/'  
  
  deploy:  
    needs: build_site  
    runs-on: ubuntu-latest
```

```
permissions:
  pages: write
  id-token: write

environment:
  name: github-pages
  url: ${{ steps.deployment.outputs.page_url }}

steps:
  - name: Deploy
    id: deployment
    uses: actions/deploy-pages@v4
```

3. Add paths to svelte.config.js (uncomment)

```
paths: {
  base: process.argv.includes('dev') ? '' : process.env.BASE_PATH
}
```

4. Create a new repo: B01 in Github

5. Github B01/settings/pages -> Source: Github Actions

6. Add remote repo to Git and upload to Github

```
git remote add origin git@github.com:fg5002/B01.git
git add .
git commit -m "Github Pages setted up"
npm run build
git push -u origin main
```

6. Install SQLocal and set up OPFS

1. Install SQLocal

```
npm install sqllocal
```

2. set vite.config.js

```
import { sveltekit } from '@sveltejs/kit/vite';
import { defineConfig } from 'vite';
```

```
export default defineConfig({
  plugins: [
    sveltekit(),
  ],
  {
    name: 'configure-response-headers',
    configureServer: (server) => {
      server.middlewares.use((_req, res, next) => {
        res.setHeader('Cross-Origin-Embedder-Policy', 'require-corp');
        res.setHeader('Cross-Origin-Opener-Policy', 'same-origin');
        next();
      });
    },
  ],
  optimizeDeps: {
    exclude: ['sqllocal'],
  },
});
```

3. Insert cross-origin headers into service.worker

```
// cross-origin isolation using COOP and COEP headers
const newHeaders = new Headers(response.headers);
newHeaders.set("Cross-Origin-Embedder-Policy", "require-corp");
newHeaders.set("Cross-Origin-Opener-Policy", "same-origin");

const moddedResponse = new Response(response.body, {
  status: response.status,
  statusText: response.statusText,
  headers: newHeaders,
});

return moddedResponse;
```

4. Insert crossOrigin : true into every tilelayer

```
<TileLayer
  name={'OSM'}
  url={'https://tile.openstreetmap.org/{z}/{x}/{y}.png'}
  options={{
    minZoom: 7,
    maxZoom: 19,
    attribution: '&copy; OpenstreetMap',
    crossOrigin : true
  }}
  selected
/>
```


5. commit modifications

```
git add .  
git commit -m "OPFS setted up and SQLocal installed"  
npm run build  
git push
```

7. Install Calendar and time picker

```
npm install svelty-picker
```

8. Create Leaflet map

1. install Leaflet

```
npm install leaflet
```

2. install markercluster

```
npm install leaflet.markercluster
```

3. install featuregroup.subgroup

```
npm install leaflet.featuregroup.subgroup --save
```

4. install necessary turf packages

```
npm install @turf/bearing @turf/destination @turf/distance @turf/midpoint  
@turf/point-on-feature @turf/nearest-point-on-line @turf/explode @turf/helpers
```

5. add map folder to routes

6. add +page.svelte to map folder

7. add +page.js to map folder

```
export const prerender = false;
```

8. run developer server

```
npm run dev -- --open
```

if **SvelteKitError: Not found: /favicon.ico** in terminal

- create a **'public'** folder in **'static'** folder
- **'static/favicon.png'** replace to **'static/public/favicon.png'**