

Svelte 5 + Hono 🔥

Part 1: Setup



www.youtube.com/@GetterSethya

Requirement:

- Node.js
- Code editor
- Svelte extension/lsp

Setup Hono backend

```
npm create hono@latest
```

```
create-hono version 0.14.3
? Target directory .
? Which template do you want to use? nodejs
? Do you want to install project dependencies? yes
? Which package manager do you want to use? npm
✓ Cloning the template
✓ Installing project dependencies
🎉 Copied project files
Get started with: cd .
```

Setup Hono backend

`.prettierrc.json`

```
{  
  "trailingComma": "es5",  
  "useTabs": true,  
  "tabWidth": 4,  
  "semi": false,  
  "singleQuote": true  
}
```

Setup Hono backend

Ganti jadi main.ts

```
{
  "name": "svelte5hono",
  "type": "module",
  "scripts": {
    "dev": "tsx watch src/main.ts"
  },
  "dependencies": {
    "@hono/node-server": "^1.13.7",
    "hono": "^4.6.16"
  },
  "devDependencies": {
    "@types/node": "^20.11.17",
    "tsx": "^4.7.1"
  }
}
```

Setup Sveltekit mode SPA

```
npx sv create
```

Disabling ssr

+layout.ts

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

Install adapter static

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

Setup Sveltekit mode SPA

```
import adapter from '@sveltejs/adapter-static'
import { vitePreprocess } from '@sveltejs/vite-plugin-svelte'

/** @type {import('@sveltejs/kit').Config} */
const config = {
  // Consult https://svelte.dev/docs/kit/integrations
  // for more information about preprocessors
  preprocess: vitePreprocess(),

  kit: {
    prerender: {
      handleMissingId: 'warn'
    },
    adapter: adapter({
      pages: '../src/static/',
      assets: '../src/static/',
      fallback: undefined,
      precompress: false,
      strict: true
    })
  }
}

export default config
```

Setup Hono: serve static file

```
.use('*', serveStatic({ root: './src/static' })))
```


Setup shadcn

[shadcn-svelte](#)



```
const config = {  
  // ... other config  
  kit: {  
    // ... other config  
    alias: {  
      "@/*": "./src/lib/*",  
      "@root/*": "../src/*",  
    },  
  },  
};
```

```
npx shadcn-svelte@next init
```

```
npx shadcn-svelte@next add button
```

Setup Drizzle

```
npm i drizzle-orm better-sqlite3 dotenv  
npm i -D drizzle-kit @types/better-sqlite3
```

```
src   
  lib   
    db   
       schema.ts  
       drizzle.ts
```

Setup Drizzle

```
export const db = drizzle('./src/lib/db/dev.db')
```

Setup Drizzle

```
import 'dotenv/config'
import { defineConfig } from 'drizzle-kit'

export default defineConfig({
  out: './drizzle',
  schema: './src/lib/db/schema.ts',
  dialect: 'sqlite',
  dbCredentials: {
    url: './src/lib/db/dev.db',
  },
})
```

```
"scripts": {
  "dev": "tsx watch src/main.ts",
  "generate": "npx drizzle-kit generate",
  "migrate": "npx drizzle-kit migrate"
},
```

Setup Drizzle: db schema

```
export const userTable = sqliteTable('users', {
  id: integer().primaryKey({ autoIncrement: true }),
  created_at: integer().notNull(),
  updated_at: integer().notNull(),

  email: text().notNull().unique(),
  name: text().notNull(),
})

export const authTable = sqliteTable('authentications', {
  id: integer().primaryKey({ autoIncrement: true }),
  created_at: integer().notNull(),
  updated_at: integer().notNull(),

  user: integer().references(() => userTable.id, { onDelete: 'cascade' }),
  hash_password: text().notNull(),
  refresh_token: text(),
})
```

Setup Drizzle: db schema

```
export const companyTable = sqliteTable('companies', {
  id: integer().primaryKey({ autoIncrement: true }),
  created_at: integer().notNull(),
  updated_At: integer().notNull(),

  name: text().notNull(),
  address: text(),
})

export const applicationTable = sqliteTable('applications', {
  id: integer().primaryKey({ autoIncrement: true }),
  created_at: integer().notNull(),
  updated_At: integer().notNull(),

  user: integer().references(() => userTable.id, { onDelete: 'set null' }),
  company: integer().references(() => companyTable.id, {
    onDelete: 'set null',
  }),
  position: text(),
  status: text()
    .notNull()
    .<keyof typeof APPLICATION_STATUS>()
    .default('sent'),
  notes: text(),
})
```

```
export const APPLICATION_STATUS = {
  sent: 'Ter kirim',
  reply: 'Dibalas',
  interview: 'Interview',
  ghosting: 'Ghosting',
  offering: 'Offering',
}
```