MINAB
IT SOLUTIONS

# Frontend development Class 07, Series 02

## Frontend

HahuJobs

# CLASS 07

Hahu Jobs

01 SSR

02 State management

03 TypeScript


# Server-Side Rendering (SSR)

Vue.js is a framework for building client-side applications. By default, Vue components produce and manipulate DOM in the browser as output. However, it is also possible to render the same components into HTML strings on the server, send them directly to the browser, and finally "hydrate" the static markup into a fully interactive app on the client.

A server-rendered Vue.js app can also be considered "isomorphic" or "universal", in the sense that the majority of your app's code runs on both the server and the client.

# Why SSR?

**Compared to a client-side Single-Page Application (SPA), the advantage of SSR primarily lies in:**

- **Faster time-to-content**: this is more prominent on slow internet or slow devices. Server-rendered markup doesn't need to wait until all JavaScript has been downloaded and executed to be displayed, so your user will see a fully-rendered page sooner. In addition, data fetching is done on the server-side for the initial visit, which likely has a faster connection to your database than the client. This generally results in improved Core Web Vitals metrics, better user experience, and can be critical for applications where time-to-content is directly associated with conversion rate.
- **Unified mental model**: you get to use the same language and the same declarative, component-oriented mental model for developing your entire app, instead of jumping back and forth between a backend templating system and a frontend framework.
- **Better SEO**: the search engine crawlers will directly see the fully rendered page.

# SSR Trade-offs

**There are also some trade-offs to consider when using SSR:**

- **Development constraints**. Browser-specific code can only be used inside certain lifecycle hooks; some external libraries may need special treatment to be able to run in a server-rendered app.
- **More involved build setup and deployment requirements**. Unlike a fully static SPA that can be deployed on any static file server, a server-rendered app requires an environment where a Node.js server can run.
- **More server-side load**. Rendering a full app in Node.js is going to be more CPU-intensive than just serving static files, so if you expect high traffic, be prepared for corresponding server load and wisely employ caching strategies.

# SSR Trade-offs

**There are also some trade-offs to consider when using SSR:**

- **Development constraints**. Browser-specific code can only be used inside certain lifecycle hooks; some external libraries may need special treatment to be able to run in a server-rendered app.
- **More involved build setup and deployment requirements**. Unlike a fully static SPA that can be deployed on any static file server, a server-rendered app requires an environment where a Node.js server can run.
- **More server-side load**. Rendering a full app in Node.js is going to be more CPU-intensive than just serving static files, so if you expect high traffic, be prepared for corresponding server load and wisely employ caching strategies.

# SSR

## SSR Solutions

### Nuxt

Nuxt is a higher-level framework built on top of the Vue ecosystem which provides a streamlined development experience for writing universal Vue applications. Better yet, you can also use it as a static site generator! We highly recommend giving it a try.

### Quasar

Quasar is a complete Vue-based solution that allows you to target SPA, SSR, PWA, mobile app, desktop app, and browser extension all using one codebase. It not only handles the build setup, but also provides a full collection of Material Design compliant UI components.

### Vite SSR

Vite provides built-in support for Vue server-side rendering, but it is intentionally low-level. If you wish to go directly with Vite, check out vite-plugin-ssr, a community plugin that abstracts away many challenging details for you.

# State management

## What is State Management?

Large applications can often grow in complexity, due to multiple pieces of state scattered across many components and the interactions between them and this is where state management comes into play.

**State management** is an implementation of design pattern to keep state of the application in multiple components across your application.

MINAB IT SOLUTIONS

HahuJobs

# State management

## Simple State Management with Reactivity API

If you have a piece of state that should be shared by multiple instances, you can use reactive() to create a reactive object, and then import it from multiple components:

```
// store.js
import { reactive } from 'vue'

export const store = reactive({
  count: 0
})
```

```
<!-- ComponentA.vue -->
<script setup>
import { store } from './store.js'
</script>

<template>From A: {{ store.count }}</template>
```

```
<!-- ComponentB.vue -->
<script setup>
import { store } from './store.js'
</script>

<template>From B: {{ store.count }}</template>
```

# State management

## Pinia

Pinia is a store library for Vue, it allows you to share a state across components/pages.

MINAB IT SOLUTIONS

HahuJobs

# State management

## Why should I use Pinia?

- **Devtools support**
  - A timeline to track actions, mutations
  - Stores appear in components where they are used
  - Time travel and easier debugging
- **Hot module replacement**
  - Modify your stores without reloading your page
  - Keep any existing state while developing
- **Plugins**: extend Pinia features with plugins
- Proper **TypeScript support** or **autocompletion** for JS users
- **Server Side Rendering** Support

MINAB IT SOLUTIONS

HahuJobs

# Typescript


**TypeScript** is a programming language developed and maintained by Microsoft. It is a strict syntactical superset of JavaScript and adds optional static typing to the language. It is designed for the development of large applications and **transpiles** to JavaScript.

MINAB
IT SOLUTIONS

HahuJobs

# State management


## Using Vue with TypeScript

A type system like TypeScript can detect many common errors via static analysis at build time. This reduces the chance of runtime errors in production, and also allows us to more confidently refactor code in large-scale applications. TypeScript also improves developer ergonomics via type-based auto-completion in IDEs.

Vue is written in TypeScript itself and provides first-class TypeScript support. All official Vue packages come with bundled type declarations that should work out-of-the-box.

# Typescript


**TypeScript** is a programming language developed and maintained by Microsoft. It is a strict syntactical superset of JavaScript and adds optional static typing to the language. It is designed for the development of large applications and **transpiles** to JavaScript.

MINAB
IT SOLUTIONS

HahuJobs

# Class Exercise

## Build countries list

- show list of countries
    - displayed information should include
        - name
        - flag
        - continent
        - country code
        - phone code
        - capital
        - currency
        - languages spoken
        - it should include list of states
- design with figma
- GraphQL Server:

https://countries.trevorblades.com/

MINAB IT SOLUTIONS

HahuJobs

MINAB
IT SOLUTIONS

# Thank you!

Hahu Jobs

ለሀገር ልጅ በሀገር ልጅ !

Michael Sahlu
michael.sahlu@hahu.jobs