Contents

1. Introduction

I am a senior full-stack developer with 7 years of strong background and proven record in web application development and cloud engineering.

I mainly work with JavaScript, TypeScript and its frameworks like React, Angular, Vue for front-end and Node for back-end development. But my skill isn't limited to only JavaScript.

I am passionate to learn new technologies and best practices, so this really keeps myself up-to-date and keep me growing all the time. Recent 2 years, I have been working with PHP programming language.

So far, I worked for several companies from the United States and the Europe.

As a full-time employee and often contractor, I successfully accomplished all their mission and goals so that they could get great success in their industry. That's what I am proud of in my career.

1.1 Tophap (Startup)

1.1.1 Tophap.com

The company is based in California and they provide a market intelligence platform for real estate professionals. It's using AI to provide market conditions and trends so that users can make data-driven and better decisions of purchasing and selling their properties.

As a Senior Front-End Developer, I built this application from the beginning, obviously, I set up their infrastructure and built all the pages and components using React and Redux.

I made a contract with them as a senior front-end developer, but I am quite good at server-side as well. After a certain period, they wanted me to work on their back-end development. It's using serverless architecture with Node and AWS Lambda.

As you know, in real estate platforms, the search functionality is the main focus. In order to provide seamless functionality, we used ElasticSearch, data engineer imported data into ElasticSearch Clusters and then I wrote efficient queries and implemented it on the client-side application.

As the project gets bigger, they hired more mid-level front-end developers and I worked with them to deliver quality of work, just reviewed their code, managed the tasks assignment, …

1.1.2 Challenge

When we finished MVP, let's say version 1, we found that it was a little bit slower than we expected. As you know, Single page applications have many advantages but some disadvantages as well. One of them is that it usually takes a bit longer to load compared to traditional web apps since it needs to download all the necessary files for the first time, things like html, css, javascript files.

It was really important to make the load speed up, so we had a team meeting again and decided to move into Next.js which provide server-side rendering.
Finally, we could use all the advantages of single page application and overcame the issues we faced.
I think that is one of the challenging parts in this project.

Another one is redux-observable. As I said, we used redux-saga from the beginning, but from time to time, the logic became complicated and code was messed up. It was not easy to read and not easy to understand.

So I decided to polish my codes, of course, the client didn't ask me to do this, but I thought it was necessary for future development.
Finally, I took this challenge and figured this out by integrating redux-observable instead of using redux-saga, since RxJS supports many operators which is quite good to use for complex logic.

These 2 are my challenging parts in this project.

1.4 Elite Answers

1.4.1 About role

For the last 4 months, I worked for Elite Digital agency as a frontend developer and I developed the Online Jewish Learning platform for AJU (American Jewish University).
I used the Next.js and Tailwind CSS for the frontend of platform and implemented the pixel-perfect design. Mainly I worked on frontend part, but often worked on backend stuff.
For the backend, we used the nodejs.

2. Tech Interview FAQ

2.1 React.js

2.1.1 Differentiate between Real DOM and Virtual DOM.

1. It updates slow.	1. It updates faster.
2. Can directly update HTML.	2. Can’t directly update HTML.
3. Creates a new DOM if element updates.	3. Updates the JSX if element updates.
4. DOM manipulation is very expensive.	4. DOM manipulation is very easy.

5. Too much of memory wastage.	5. No memory wastage.

2.1.2 What is React?

- React is a front-end JavaScript library developed by Facebook in 2011.
- It follows the component based approach which helps in building reusable UI components.
- It is used for developing complex and interactive web and mobile UI.
- Even though it was open-sourced only in 2015, it has one of the largest communities supporting it.

2.1.3 What are the features of React?

It uses the virtual DOM instead of the real DOM.

It uses server-side rendering.

It follows uni-directional data flow or data binding.

2.1.4 List some of the major advantages of React.

It increases the application's performance

It can be conveniently used on the client as well as server side

Because of JSX, code's readability increases

React is easy to integrate with other frameworks like Meteor, Angular, etc

Using React, writing UI test cases become extremely easy

2.1.5 What are the limitations of React?

React is just a library, not a full-blown framework

Its library is very large and takes time to understand

It can be little difficult for the novice programmers to understand

Coding gets complex as it uses inline templating and JSX

2.1.6 What is JSX?

JSX is a shorthand for JavaScript XML. This is a type of file used by React which utilizes the expressiveness of JavaScript along with HTML like template syntax. This makes the HTML file really easy to understand. This file makes applications robust and boosts its performance.

2.1.7 What do you understand by Virtual DOM? Explain its working.

A virtual DOM is a lightweight JavaScript object which originally is just the copy of the real DOM. It is a node tree that lists the elements, their attributes and content as Objects and their properties. React's render function creates a node tree out of the React components. It then updates this tree in response to the mutations in the data model which is caused by various actions done by the user or by the system.

1. Whenever any underlying data changes, the entire UI is re-rendered in Virtual DOM representation.
2. Then the difference between the previous DOM representation and the new one is calculated.
3. Once the calculations are done, the real DOM will be updated with only the things that have actually changed.

2.1.8 Why can't browsers read JSX?

Browsers can only read JavaScript objects but JSX in not a regular JavaScript object. Thus to enable a browser to read JSX, first, we need to transform JSX file into a JavaScript object using JSX transformers like Babel and then pass it to the browser.

2.1.9 How different is React's ES6 syntax when compared to ES5?

1. require vs import
2. export vs exports (module.exports vs export default)
3. component and function (React.createClass({ render: func() {} }))
4. props
5. state

2.1.10 How is React different from Angular?

1. ARCHITECTURE	Only the View of MVC	Complete MVC
2. RENDERING	Server-side rendering	Client-side rendering
3. DOM	Uses virtual DOM	Uses real DOM
4. DATA BINDING	One-way data binding	Two-way data binding
5. DEBUGGING	Compile time debugging	Runtime debugging
6. AUTHOR	Facebook	Google

2.1.11 What is component in React? Explain

Components are the building blocks of a React application's UI. These components split up the entire UI into small independent and reusable pieces. Then it renders each of these components independent of each other without affecting the rest of the UI.

2.1.12 What is purpose of render() in React?

Each React component must have a render() mandatorily. It returns a single React element which is the representation of the native DOM component. If more than one HTML element needs to be rendered, then they must be grouped together inside one enclosing tag such as <form>, <group>, etc. This function must be kept pure i.e., it must return the same result each time it is invoked.

2.1.13 What is props?

Props is short for properties and they are used to pass data between React components. React's data flow between components is uni-directional (from parent to child only).

2.1.14 What is State?

State is a JavaScript object that stores component's dynamic data and it enables a component to keep track of changes between renders. Because state is dynamic, it is reserved only for interactivity so you don't use it for static React projects.

2.1.15 Difference between state and props?

1. Both can receive the initial value from parent component
2. Props can be changed by parent component but State can not be
3. Both can be set the default values inside component
4. State can be changed inside component but Props can not be
5. Both are set the initial value for clild components

2.1.16 How to implement CI/CD in React?

2.1.17 What is difference between CI and CD?

2.1.18 How to do api integration?

We can use axios.

2.2 Redux

Redux is a predictable state container designed to help you write JavaScript apps that behave consistently across client, server, and native environments and are easy to test.

The way Redux works is simple. There is a central store that holds the entire state of the application. Each component can access the stored state without having to send down props from one component to another.

There are three building parts: actions, store, and reducers.

Actions in Redux

Actions are sent using the store.dispatch() method. Actions are plain JavaScript objects, and they must have a type property to indicate the type of action to be carried out. They must also have a payload that contains the information that should be worked on by the action.

Reducers in Redux

Reducers are pure functions that take the current state of an application, perform an action, and return a new state. These states are stored as objects, and they specify how the state of an application changes in response to an action sent to the store.

Store in Redux

The store holds the application state. It is highly recommended to keep only one store in any Redux application. You can access the state stored, update the state, and register or unregister listeners via helper methods.

2.3 Vue

2.3.1 What is Vue.js

VueJS is a progressive JavaScript framework used to develop interactive web interfaces. Focus is more on the view part, which is the front end. It is very easy to integrate with other projects and libraries.

2.3.2 What is virtual Dom?

Virtual DOM in Vue is a JavaScript object that represents the Document Object Model (DOM). The application updates the Virtual DOM instead of the DOM directly. So, it minimizes the updating cost of the real DOM as it is computationally expensive

2.3.3 What is Data Binding?

The data binding feature helps manipulate or assign values to HTML attributes, change the style, assign classes with the help of binding directive called v-bind available with VueJS.

2.3.4 Why is Vue.js called a progressive framework?

Vue.js called a progressive framework because it is being changed and developed continually.

2.3.5 What is basic difference between vue vs react and angular

React is a UI library, Angular is a fully-fledged front-end framework, while Vue. js is a progressive framework. They can be used almost interchangeably to build front-end applications, but they're not 100 percent the same, so it makes sense to compare them and understand their differences.

2.3.6 What are the advantages of using Vue.js?

1. One of Vue.js' biggest advantages is that it is very small in size. This exciting JavaScript plug-in is only 18-21KB, so you can download and use it very easily in no time.

2. The framework of Vue.js is very easy to understand, and it id one of the reasons for the popularity of this framework. The users can easily add Vue.js to their web project because of its simple structure and develop applications.

3. Vue.js framework can be integrated with the existing applications very easily. Vue.js has a lot of components for everything. You can integrate it with any application that is written in JavaScript.

4. (Flexible in nature) The flexible nature of Vue.js also makes it easy to understand for the developers of React.js, Angular.js, and any other new JavaScript framework. It provides a lot of flexibility to use virtual nodes to write HTML files, JavaScript files, and pure JavaScript files.

5. Vue.js provides two-way communications with its MVVM architecture that makes it very easy to handle HTML blocks.

2.3.7 What is Two-way Communication?

Two way data binding is a powerful pattern for building JavaScript forms with Vue. ... Two way data binding means: When the user types in the input , value gets updated to match the value in input . When you update value , the input element's content updates to match value

2.3.8 What is the difference between one-way data flow/ or one-way data binding and two-way data binding?

In one-way data binding or one-way data flow, the view (UI) part of the application does not update automatically. In this model, when the data Model is changed, you need to write some custom code to make it updated every time after the change. The v-bind directive is used for one-way data flow or binding in Vue.js.

On the other hand, in two-way data binding, the view (UI) part of the application is automatically updated when the data Model is changed. The v-model directive is used for two way data binding in Vue.js.

2.3.9 How can you create Two-Way Bindings in Vue.js?

The v-model directive is used to create Two-Way Bindings in Vue js. In Two-Way Bindings, data or model binds with DOM, and Dom binds back to the model.

2.3.10 What is Vuex?

VueX is a state management pattern and library for the Vue.js application. It is used as a centralized store for all the different components in the Vue.js application. Vuex provides some rules to ensure that the state can only be mutated in a predictable fashion. You can get a lot of additional features by integrating Vuex with the official devtool extension of Vue.js.

2.3.11 What is life-cycle in vue.js?

– for example, it needs to set up data observation, compile the template, and create the necessary data bindings. Along the way, it will also invoke some lifecycle hooks, which give us the opportunity to execute custom logic. For example, the created hook is called after the instance is created:

```
new Vue({
 data: {
  a: 1
 },
 created: function () {
  // `this` points to the vm instance
  console.log('a is: ' + this.a)
 }
})
// => "a is: 1"
```

There are also other hooks which will be called at different stages of the instance's lifecycle, for example compiled, ready and destroyed. All lifecycle hooks are called with their this context pointing to the Vue instance invoking it. Some users may have been wondering where the concept of "controllers" lives in the Vue.js world, and the answer is: there are no controllers in Vue.js. Your custom logic for a component would be split among these lifecycle hooks.

2.3.12 What are filters in Vue.js?

The Filters are functionality provided by Vue.js components that allow you to apply formatting and transformations to your dynamic template data. Filters are used in two places, mustache interpolations, and v-bind expressions. Filters don't change a component data or anything, but they only affect the output.

2.3.13 Can we call Rest API from Vue.js? How?

Yes, we can call Rest API from Vue.js. There are several HTTP libraries that can used to call REST Api's from Vue.js. One of the popular libraries is Axios.

2.3.14 What is single file component in Vue?

Vue Single File Components (aka *.vue files, abbreviated as SFC) is a special file format that allows us to encapsulate the template, logic, and styling of a Vue component in a single file.

This can work very well for small to medium-sized projects, where JavaScript is only used to enhance certain views. In more complex projects however, or when your frontend is entirely driven by JavaScript, these disadvantages become apparent:

1. Global definitions force unique names for every component
2. String templates lack syntax highlighting and require ugly slashes for multiline HTML
3. No CSS support means that while HTML and JavaScript are modularized into components, CSS is conspicuously left out
4. No build step restricts us to HTML and ES5 JavaScript, rather than preprocessors like Pug (formerly Jade) and Babel

2.4 Node.js/ Express

2.5 JS/ TS/ ES6+

2.6 PHP/ Laravel

2.7 AWS

3. Interview FAQ

3.1 Why are you interested with this company?

After leaving last company, I am finding the new position. I thought MIA is looking for a someone who has good experiences with Frontend technologies. As you can read from my profile, my majors is Frontend side, and especially React.js.

3.2 Are you interested with Startup or Large-scale business?

I am so much interested with Startup company because I especially want to learn the new technologies and grow with startup company by contributing my knowledge, and effort, and my best.

3.3 Do you prefer working remotely or in person?

I really enjoy the freedom that working remotely provides. I can get my day started a little earlier or work a little later since I don't have to commute. In my ideal situation, I would primarily work remotely, with some in-person team meetings to discuss how work is progressing.

3.4 Why do you think you're a good fit for our startup?

I think I would fit well into the open culture you have created so far. I believe in constant communication to share the newest information and keep tasks on track, so I think I could easily begin contributing to your team. I'm also passionate about the company's larger goal of creating more opportunities for those living in low-income areas.

3.5 What is your greatest Strength

My greatest professional strength is the ability to handle pressure and work under a tight deadline. I think that would help me succeed in this position because of the large number of clients and deadlines I’d be working with here.