Vue.js for Jobseekers

A complete guide to learning Vue.js, building projects, and getting hired

Clive Harber

www.bpbonline.com

First published: 2024

Published by BPB Online
WeWork
119 Marylebone Road
London NW1 5PU

UK | UAE | INDIA | SINGAPORE

ISBN 978-93-55518-750

www.bpbonline.com

Dedicated to

*For **Cathy***

Love of my Life, Wife, Best Friend,
Inspiration and Anchor,
Without whom this would not have been possible

About the Author

Clive Harber has been programming computers since he was 13. He holds a Master's degree in Chemical Engineering from the University of Wales, Swansea and diplomas in Marine Biology and Art History. Having written code in a number of programming languages and different paradigms over the years for the sports and betting industries, telecommunications, health care, retail and entertainment sectors, he is now at a point where he feels he can be useful to the programming community as a whole.

Clive has previously co-written two other books – Getting MEAN with MongoDB, Express, Angular and Node 2nd ed and Implementing Cloud Design Patterns for AWS 2nd ed, and has been a reviewer and technical reviewer on more than 10 other widely available books on a variety of technical subjects (Frontend frameworks – Vue.js/React/Web Components, testing, Elixir, UI/UX, Microservices, desktop applications). He has been using Vue.js for several years on multiple projects.

Currently, Clive runs his own consultancy providing expertise in building web and mobile applications using Node, Vue.js, Quasar, NativeScript, Elm, Elixir and PhoenixFramework.

About the Reviewer

Ivan Almaši is a full-time Frontend Developer. He has been working as a contractor/freelancer for the past 6 years and the last 5 years almost exclusively in the Vue.js ecosystem during which he worked on different projects, most notably a SaaS enterprise solution for quality assurance. His experience also includes converting designs to pixel-perfect UIs utilizing Tailwind CSS.

Acknowledgements

I want to express my deepest gratitude to my family and friends for their unwavering support and encouragement throughout this book's writing, especially my wife Cathy and my children, Jacob and Rebecca.

I am also grateful to BPB Publications for their guidance and expertise in bringing this book to fruition. It was a long journey of building this book, with valuable participation and collaboration of reviewers, technical experts, and editors.

I would also like to acknowledge the valuable contributions of my colleagues and co-workers during many years working in the tech industry, who have taught me so much and provided valuable feedback on my work.

Thank you to all the readers who have taken the time to read this book. I hope that this will help you in your future endeavours.

Lastly, burn out, depression, anxiety and imposter syndrome are real things that we can all suffer from. If you are effected by any of these, speak up and ask for help, the world is a better place for you in it.

Preface

Knowing where to start with any element of building software is usually half the battle. There are a plethora of options, myriad of opinions and multiple ways of doing things, so finding the best way can prove to be very daunting. This book is a good place to begin getting acquainted with frontend development using Vue.js.

This book is designed to give insight into not only the framework, but also how the industry works, some questions you might face at interview, plus some handy pointers of the wider Vue.js ecosystem that you might need to understand, or use as a springboard for your own projects.

This is not a total beginners manual, you will need to know the basics of HTML, CSS and some JavaScript already. You might have completed some self- or college study, already have a year or two of industry experience or transitioning from a different framework. You will be covering some of the more advanced topics that you might not have come across before, that you will need in order to create maintainable code and advance your career.

We will be bringing you Vue.js from the ground up, so do not have to worry about that. Once we have the basics of Vue.js under our collective belts, we will forge ahead through standard uses such as front-side routing and state management before you finally cover some advanced, more modern, topics, such as server-side rendering and building for mobile which are not ordinarily covered in books like this.

With this book, you will gain knowledge and experience of the Vue.js framework and its ecosystem, learn how to use the framework to solve problems and build easy to maintain user interfaces that can be used across many different display devices. We hope that you find this book interesting, helpful and informative, regardless of which stage your career is at.

Chapter 1: Introducing Front-End Development with Vue.js – this is the start of your journey. This chapter covers web development as a paradigm, where it came from, and where it is going. You will then get a 10,000 ft view of the framework, where it fits in and how it can improve your working life.

Chapter 2: Working in Vue.js Roles – this chapter gives you an overview of the job roles that might be on offer and the kinds of work that you can expect to do

and how you might progress in your career. We will also cover different types of company and how developers fit into these organizations, and how you might progress your learning.

Chapter 3: HTML, CSS, and JS/TS for Extra Credit – presents more advanced topics that underpin the use of Vue.js. We will cover virtual DOM, CSS rules and specificity, before digging into JavaScript's many different capabilities and covering the basics of TypeScript. This chapter gives you many tools that you might not have in order to become a successful developer. You will also go over various API's that JavaScript enables you to access so that you have an understanding of what capabilities the browser provides.

Chapter 4: Understanding the Vue.js Instance - you will start your Vue.js journey by understanding the base technology and how to structure your components. In this chapter you will get exposed to the Options API variant of Vue.js components as a basic starting point. You will also gain understanding of the life cycle and how to hook into it to get better control of your component state.

Chapter 5: Designing Component-Based UIs and Single Page Applications – throughout this chapter you will gain an understanding of how UIs can be constructed. This is not a chapter on design tools, but provides you, the developer, the tools and knowledge required to break apart a design and determine the interactions that you might need to build. Here you will be starting a prototype project from a given design.

Chapter 6: Using the Composition API to Manage Component Logic – gives attention to the modern way of building Vue.js components. You will understand how to segment your components in order to reuse functionality across many different components. This way of building components was an optional add-on in Vue2 and baked into Vue3. It is fast becoming the default way of constructing user interfaces with the Vue framework.

Chapter 7: Creating and Setting Up a Vue.js Application with Vue CLI and Vite – so far we have seen how to build components in different ways and how to make them fit together. This chapter looks, in detail, at how to bootstrap Vue.js projects quickly in order to get the most out of the framework. Of the various ways that are available for starting Vue.js projects, we will discuss using the Vue CLI, the old way, and Vite, the new way, in detail. You will start building the prototype project in this chapter.

Chapter 8: Adding a CSS Framework to the Mix – TailwindCSS, Bootstrap or Foundation – when working at a professional level, it is unlikely that you will be creating your own CSS for standard display elements. It is more likely that you will be using a CSS framework , hence, you will need to know how to integrate these frameworks into your application stack. This chapter covers integrating the three most widely used CSS frameworks. This chapter will equip you to expand the prototype project with TailwindCSS.

Chapter 9: Building User Interfaces with Components – expands the readers' knowledge of building components through building the prototype project. We will be pulling together all the knowledge we have gained and putting it into practise. After working through setting up a working directory structure that suits our needs, we will set up the basic component templates and construct the various layouts that the project needs.

Chapter 10: Routing Between Pages with VueRouter – single page applications are very limited without being able to navigate between the various "pages". After the last chapter, we'll be expanding the prototype with more pages using VueRouter. We'll also discuss how VueRouter helps you direct and manage navigation based on state – authentication we're looking at you…

Chapter 11: Interacting with the UI Using Events – at the moment, the UI is fairly boring. You can't have an event driven UI without events and in this chapter you'll understand the Vue.js events system, from standard DOM events to custom events and basic component to component state management. Events are one of the cornerstones of building effective UIs, and interacting with your user, but also having components interact with each other. With this understanding, we'll expand the prototype application to respond to users.

Chapter 12: Building Forms and Handling User Input – the next vital element that we need to learn is how to obtain information from your users and manage this data effectively so that it doesn't pollute your database, or expose your application to attack. Not only will you learn how to effectively build forms and manage the data within them, you will also get necessary experience in validating this data to ensure that it conforms to expected inputs, informing the user if it does not. The prototype application will also not be forgotten, we'll add forms to collect user data

Chapter 13: Managing State with Pinia and API Communication – when building component based applications, as the application grows, it becomes necesssary to

bring all the available state together into a *store*. This is where Pinia comes into play – the standard state management tool for the Vue.js ecosystem. Once we have understood how to utilise Pinia, we will cover the various ways we can get data into the application from a remote server. For this we will integrate data obtained via JSON ReST APIs and GraphQL.

Chapter 14: Testing Vue.js Applications – understanding testing and how to test your application is vitally important. In a professional context, you will most likely end up writing more code in test files than you will for your actual application. This is normal, but many juniors (or those that are self-taught) do not understand the importance of testing. By the end of this chapter, you will understand this and more. We will go over component and unit tests, integration tests and end-to-end testing and show that it does not need to be complicated.

Chapter 15: Server-Side Rendering with Nuxt.js – with the core functionality now covered, it is time to strike out and look at some parts of the wider ecosystem. One of the biggest trends at the moment is server-side rendering, where for the Vue.js ecosystem, Nuxt.js is one of the bigger players. This chapter covers the basics of building a site with Nuxt.js, which combines much of what we have already covered, but in new and interesting ways.

Chapter 16: Building Multi-tenanted Apps with Quasar – in this final content-rich chapter, we will explore Quasar, one of the many ways of getting your web application available across many different platforms using Vue.js and not having to learn a mobile-native language. Quasar allows you to build different types of applications that have the same UI implementation in common, reducing time to market and engineering overhead. This chapter aims to add this one size fits all framework to your toolbox.

Chapter 17: Interview Questions – what good is learning all of this, if when you get to interview, and cannot answer any questions. This last chapter of the book goes some way to preparing you for technical questions that might be asked, plus provides pointers to other areas that you might need knowledge of.

Code Bundle and Coloured Images

Please follow the link to download the
Code Bundle and the ***Coloured Images*** of the book:

https://rebrand.ly/93x9jcr

The code bundle for the book is also hosted on GitHub at **https://github.com/bpbpublications/Vue.js-for-Jobseekers**. In case there's an update to the code, it will be updated on the existing GitHub repository.

We have code bundles from our rich catalogue of books and videos available at **https://github.com/bpbpublications**. Check them out!

Errata

We take immense pride in our work at BPB Publications and follow best practices to ensure the accuracy of our content to provide with an indulging reading experience to our subscribers. Our readers are our mirrors, and we use their inputs to reflect and improve upon human errors, if any, that may have occurred during the publishing processes involved. To let us maintain the quality and help us reach out to any readers who might be having difficulties due to any unforeseen errors, please write to us at :

errata@bpbonline.com

Your support, suggestions and feedbacks are highly appreciated by the BPB Publications' Family.

Piracy

If you come across any illegal copies of our works in any form on the internet, we would be grateful if you would provide us with the location address or website name. Please contact us at **business@bpbonline.com** with a link to the material.

If you are interested in becoming an author

If there is a topic that you have expertise in, and you are interested in either writing or contributing to a book, please visit **www.bpbonline.com**. We have worked with thousands of developers and tech professionals, just like you, to help them share their insights with the global tech community. You can make a general application, apply for a specific hot topic that we are recruiting an author for, or submit your own idea.

Reviews

Please leave a review. Once you have read and used this book, why not leave a review on the site that you purchased it from? Potential readers can then see and use your unbiased opinion to make purchase decisions. We at BPB can understand what you think about our products, and our authors can see your feedback on their book. Thank you!

For more information about BPB, please visit **www.bpbonline.com**.

Join our book's Discord space

Join the book's Discord Workspace for Latest updates, Offers, Tech happenings around the world, New Release and Sessions with the Authors:

https://discord.bpbonline.com

Table of Contents

Join our book's Discord space

Join the book's Discord Workspace for Latest updates, Offers, Tech happenings around the world, New Release and Sessions with the Authors:

https://discord.bpbonline.com

Chapter 1
Introducing Front-End Development with Vue.js

Introduction

Vue.js is the front-end framework of choice for numerous large companies and many small ones. These companies have chosen this framework over its many competitors due to its small footprint, easy-to-use semantics, predictable behavior and simple way to provide extensions. As a front-end framework, Vue.js helps you build user-friendly, functionality-rich and data-rich UI's that are easy to maintain and deploy.

In this chapter, we will be exploring where and how Vue.js fits into front-end development, how Vue.js compares with its contemporaries and finally, how to use the framework at the 10,000ft view.

Structure

We will be covering the following topics in this chapter:

- A brief history of web development
- The rise of Vue.js
- Comparing Vue.js with the competition
- Using Vue.js to solve problems – Case studies

Objectives

This chapter explores where Vue.js comes from and how it is used.

In this chapter, you will be introduced to Vue.js as a technology, learn the basics of Vue.js single file components, and understand where Vue.js fits in the development ecosystem.

You don't need to know anything special to go through this chapter; everything will be explained over the course of the book.

A brief history of web development

Over the years, front-end development has changed significantly from building complete pages using the humble triad of HTML, CSS and JavaScript. As applications became more complicated, and companies and consumers wanted more, jQuery (and similar) filled this functionality gap. There were, however, significant issue with the libraries in that the functionality created to manipulate the HTML **document object model** (**DOM**) with JavaScript was intermingled with the data that was being used to power that manipulation. This, in turn, made it more difficult to extract and maintain this functionality; large applications started taking significant effort to build and maintain, increasing development time and application development cost.

In this landscape, web development frameworks based on the **Model-View-Controller** (**MVC**) pattern were being created, initial attempts gaining small pockets of traction. The MVC pattern is a standard pattern that dictates how to manage data flow between different parts of an application.

As a brief explanation, the **Controller** receives a message/request and processes it. The Controller then passes on a request for data from the **Model**. The Model returns the data that then gets passed on to the **View** to create the UI, as shown in *Figure 1.1*:

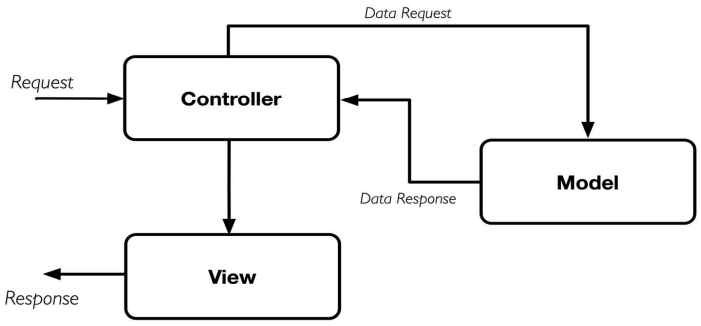

***Figure 1.1**: Model-View-Controller Pattern*

For those who have been around a while, we're talking about the frameworks Knockout.js, Backbone.js, Dojo.js and MooTools.

Then, Google released Angular and at roughly the same time, Node.js, a JavaScript engine that runs on a server rather than a browser, gained prominence in web development circles. Overnight, everything changed – the **MongoDB, Express, Angular and Node** (**MEAN**) stack was born, and an explosion of frameworks, tooling and concepts occurred, with companies wanting to utilize this change in technology. As with everything, faster ways of building web UIs came about, and Google went back to the drawing board with Angular.

While this was happening, Facebook dropped React.js on the world. With its quicker "conception to deployment" time and shallower learning curve, React.js had been set to take over the world. However, the learning curve for React isn't shallow enough for some, and it has various idiosyncrasies that are unpalatable for others.

In order to smooth out some of the perceived issues in both Angular and React, Angular core team member Evan You created Vue.js. In many ways, Vue.js and React.js are similar, but they are very different in terms of execution.

With its leaner implementation, faster execution time, and reliance on standard technologies, Vue.js has become one of the top three front-end JavaScript frameworks in use by companies today.

The rise of Vue.js

Since its creation in 2014, Vue.js has had a steady rise in uptake. While many companies tended to take the decision to use React due to its association with Facebook, developers tended to like the simpler learning curve and cleaner interface provided by Vue.js.

Slowly, over time and with the backing of large companies like Alibaba, Font Awesome, Gitlab, and Apple, Vue.js has gained traction in the development community. And over recent years, Vue has joined React and Angular as one of the three most used JS front-end frameworks in use by corporations and enterprise-sized organizations today.

The short bootstrapping time and ease of development also makes Vue.js ideal for SME's and start-ups.

There's plenty of scope for jobseekers of all kinds.

Comparing Vue.js with the competition

If you take the time to look at the "Best frameworks to learn in…" or the "React vs Angular vs Vue" blog posts, more often than not you'll end up reading some contrived half-formed, badly articulated opinion of why framework X is the best as per the authors' limited experience, not that all these types of articles should be seen as equal, some are well thought out and balanced.

As a reader, you've probably already decided that there is something behind the hype of Vue.js, but I'll try to be balanced. Let's take a look:

Do you need to have specialist knowledge beyond the framework?

Well, no; not unless you want to; the core technologies used in Vue.js are HTML, CSS and JavaScript. If you want to use SCSS, Less, JSX, TypeScript or something else, then you can, but it's not essential, unlike the other frameworks. When working with React, you need to learn and understand JSX, you also need to understand some of the more advanced JS idioms. React does however give the choice of using ReasonML if you're feeling more adventurous. Angular requires TypeScript. While TypeScript is not very different from JavaScript, it does add overhead to the learning curve.

Does everything need to be rebuilt to make the project fit the framework?

Again, the answer is no. Vue.js is not overly opinionated and therefore, tends to not get in the way. You can just drop Vue.js into your project and use your existing dependencies. While this is also mostly possible with React or Elm, it's not possible with Angular or Svelte.

Is the framework large? Does it load quickly?

The speed at which the JS is made available to the browser is dependent on file size; the smaller the file, the quicker it downloads – stands to reason, doesn't it? The core framework at version 2 has been found to weigh around 24kB when it has been minified and gzipped for production deployments, and version 3 has been pinned to about 33kB. This makes it small in terms of memory footprint and therefore, quick to download. Compare this to React, which weighs 99kB (split between the core at ~6.5kB and react-dom at ~92.5kB) and Angular at a hefty ~563kB. Frameworks like Elm or Svelte have a varying package size because of the way they are built and deployed.

Does the rendering speed matter?

Quicker, snappier rendering gives your users a better experience. The better the experience, the more likely they are to return or convert. Vue.js uses a virtual DOM,

similar to React with a similar rendering and page update speed.

Can you use the framework to bootstrap an application quickly to get user feedback?

Vue.js can be injected straight into a web page. Additionally, with the CLI and UI, bootstrapping and managing dependencies is quick and easy. Single file components can be built, tested, integrated, and iterated over quickly. Reduced development time of prototypes and new features has benefits of getting these out to users quicker, which, in turn, gives compressed feedback cycles and faster iterations.

Using Vue.js to solve problems – Case studies

Vue.js lends itself to many use cases to cater for different situations, so you can dip your toe in or go whole hog – Vue.js has your back.

These case studies are just to give you an idea of what is possible, without going into too much detail.

Case Study 1: Adding Vue.js into an already created site – client-side rendering

The Scenario: There is a website that has already been built, but you've been tasked to add a new feature that Vue.js has been deemed perfect for.

Your Goal: Build out the expected feature using client-side rendered Vue.js.

The Implementation: To make this work, you will be pulling Vue.js directly into the web page and building your JavaScript as you normally would.

As a demonstration, a simple hello world example is as follows:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>My first Vue app</title>
  <script src=”https://unpkg.com/vue"></script>
</head>
<body>
  <div id="app">
```