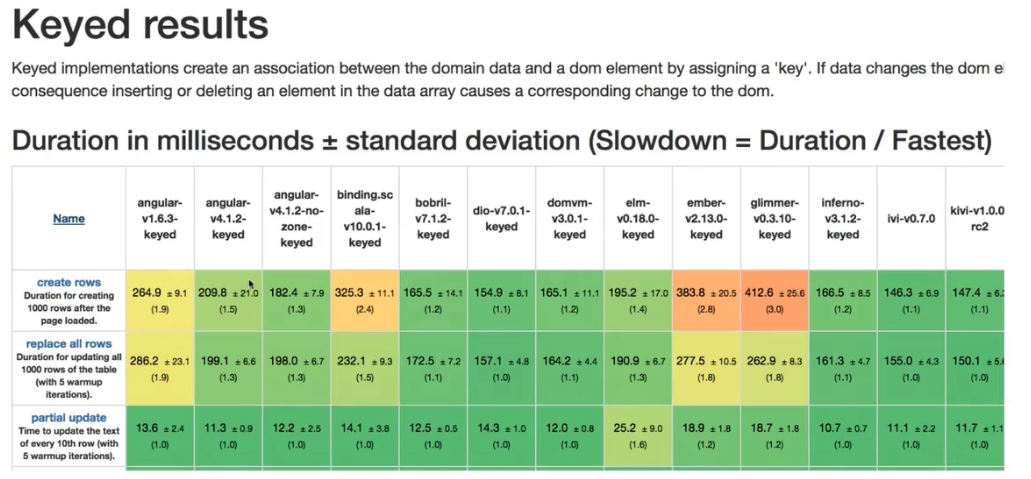
**Section 27 done: 9/9 Next Steps and Course Roundup**  
**Introduction**  
\* What to do next?  
\* Well obviously the next steps are what I mentioned right at the start of the course - practice, practice, practice, you learned a lot about React and React as you learned has a lot of different concepts and things you can do with it. Now this course covered them, you now need to practice that.  
\* So definitely dive back into some of the sections of this course, pick the things which you might’ve already forgotten by now, and practice them in projects you set on your own. Set your own challenges, come up with demo projects you wanna build, with popular apps you wanna clone and then just do it and practice the things you learned. You’ll hit some walls but I’m sure you can do it.  
\* Besides practicing, what else can you do what makes sense?  
\* Let’s take a closer look over the next lectures.  
**React Rocks! Where to find more Examples**  
\* Here’s my 1st advice for you: <https://react.rocks> - this is a web page which HOSTS a lot of React projects, you can also search for certain things there, for keywords like: “redux” for example, and there you’ll find some React projects you can check out, these are all open-source. You can click on them and then you find a demo click where you should be able to preview the project, though some of the projects are down, but you should always find a code link, most of the time leading to GitHub. There you can check out this project and that’s the cool thing - you can check out these projects and dive into the code written by other React developers.  
\* Now 1 important thing: that code doesn’t have to be 100% correct or best practice but it should give you some ideas about how you may optimize your code or how you can solve certain problems.  
\* Or you simply take one of these projects, don’t look into the code, instead try to build it on your own and then you compare your solution to the solution of the original author.  
\* So definitely dig around on React Rocks, it’s a great resource to find some nice projects to learn more about React.  
**More Inspiration: Real-World Projects Built with React**  
\* Google search: “built with react”.  
  
\* Now this might sound stupid but if you look at it, you will find official links in the official GitHub Repo where you can find projects that are using React.  
\* If you click on one of these projects, you’ll be redirected to the page, not to the source code.   
\* Still, when I talked about practicing that, this can give you some ideas about what you can build.  
\* It also finds a Made with React site. You can inspect and also find source code for the projects.  
\* So that’s another way of really practicing React and digging into other people’s projects to learn how you may improve your code, how other people solved certain problems and so on.  
**Static React Apps with Gatsby.js**  
\* Now let’s move away from demo projects and so on.  
\* Gatsby.js is actually a library building up on React, just like Next.js did.  
\* Gatsby.js serves a different purpose though and it’s a pretty exciting library I’ll be honest with you.  
\* Gatsby.js is a library that allows you to build Static Websites with React.  
\* A Static Website is a page which uses only HTML, JavaScript and CSS.   
\* Well that’s true for a normal React SPA too but a Static Website also has 1 special thing going for it - you normally don’t use React Router there, you don’t use it directly at least, instead what you have is: you have a folder structure where each folder has an index.html file and the according assets belonging to that file in it.  
=> Now the cool thing with that is that you really build websites like you did in the past, but you’re using React, you’re building this as a React app and Gatsby.js then actually does the rest for you - splits everything up and upon a certain command generates this folder structure where you got a couple of folders with HTML files in there, based on the React app you built.  
\* You can start with the docs at <https://www.gatsbyjs.org/docs/> to learn how to use it.  
\* Comparable alternatives, mentioned on the “features” page, would be Jekyll but there you don’t use React, or Content Management Systems (CMSs) like Wordpress - there, however, you’re stuck to writing some server-side code too, you need a Sequel database and all that stuff.  
\* Gatsby.js works with React and that offers quite a lot of advantages.  
\* Check out the docs and the features page on their page to learn if it’s the right choice for your project.  
**Introducing React Native**  
\* React Native is another project created by Facebook.  
\* React Native essentially is React but for mobile apps and with that I mean Native Apps - Native iOS or Android apps.  
\* With React.js taught in this course, you of course learned how to build Web Apps with JavaScript.  
\* With React Native you use JavaScript and React and another library: React Native to build Mobile Apps.  
\* You still write JavaScript code, but then React Native will compile that to code that runs on iOS or Android devices.  
\* It does so by exposing some building blocks to you like this **View** component, which is your alternative to a DIV. DIVs only work in Web Browsers of course, now the View behind the scenes is an element, an element running on iOS or Android.  
\* That’s the idea behind React Native: you write Native code but with JavaScript that gets compiled and with the same React logic you learned in this code.  
\* So you still need to learn some new things when diving into React Native because of all these new Components and some other things that change, BUT it’s still extremely simple to get started.  
\* There even is a create-react-app alternative for React Native.  
=> It’s called **create-react-native-app**.   
\* Feel free to take your React knowledge to start building Native Apps with it.  
\* It takes some adjustments, it takes some new things to learn, but it’s an amazing opportunity to build real Native Apps and not just Web Apps.  
\* If you want to get into Native App development, with that you have a chance of doing so without learning Android or Swift Objective C first.  
**Component Libraries (Example: Material UI)**  
\* Now let’s move back to React.js.  
\* Let’s have a look at Material UI: [www.material-ui.com](http://www.material-ui.com).  
\* Material UI is another library you can use together with React but unlike Next.js or Gatsby.js, it’s not building up on React, it’s a Component Library instead. And it’s not the only one.  
\* There are a lot of Component Libraries you can import into your React project.  
\* Component Libraries are libraries that give you pre-built and pre-styled Components.   
\* You can still write your own ones but for example Material UI gives you beautiful, nice-working, Material Design Components.  
\* On their web page you can click on **Component Demos** and actually see the many components you can easily import and use in your projects. You can always find the source code for a given demo if you click on that source code icon in the top right corner of the example.  
\* And there you see what you need to import from that library - if you have it installed - and then you can quickly add nicely styled and looking components without having to build them on your own.  
\* That’s a powerful feature if you’re building very big application where you want to have a uniform look. You can of course sketch out and build all the core components on your own, but if you know that you want to follow the Material look, definitely have a look at Material UI.  
\* And if you want another look, have a look at other popular Component Libraries.  
\* Like for example: React-Bootstrap. There you can find Components that take the Bootstrap styling but work nicely in React and are easily integrated into React apps.  
**Smaller Apps with Preact**  
\* Here’s 1 other nice library you might want to look into: <https://preactjs.com>.  
\* Preact is a lightweight alternative to React.  
\* It’s only 3kB and it offers pretty much the same - with some deviations but not that many - API React does.  
\* There are differences though.  
\* If you click the “REPL”, you can see an example of Preact in action.  
  
=> Down there we have a normal component, the Functional form, using {{ }} to pull out 1 property of the props argument.  
=> And above it we have a Class-based one which also extends Component.  
\* So this syntax looks a lot like React syntax and it is to a majority or to a very big degree.  
\* Preact uses pretty much the same API, but uses a leaner “diffing?” algorithm behind the scenes, so for finding out if it needs to update the real DOM, it has a leaner algorithm for that.  
\* Lean doesn’t mean better, but of course it leads to a much smaller file size.  
\* The important thing is that Preact also offers some differences to React, even some new features.  
\* You start working with Preact if you google: preact-cli, GitHub page. There you find a project which is a lot like create-react-app, which allows you to create new Preact apps, based on some of the templates that you can find here.   
\* So you can quickly scaffold out a new app where you can use the majority of the things you learned in this course and simply use this much smaller bundle, this much smaller library.  
**Comparing React with Preact**  
\* Preact > Guide > Differences to React.  
\* You see “What’s Included?”.  
=> These are things like HOCs, creating Components, Functional Components and all that stuff.  
=> Also React.createElement() becomes h().  
\* You see “What’s Added?”.  
=> For example you get this.props and this.state passed as an argument to render() function so that inside there you can simply call props and state without `this` but you can also still use the `this` keyword.  
=> You can use the `class` keyword instead of `className`.  
\* You see “What’s Missing?”.  
=> PropType Validation.  
=> The most things that are missing are things behind the scenes.   
=> Here these are really mostly cosmetic things though they also strip out quite a lot of source code.  
\* We can imagine PropType was put into a separate library by React to also reduce the size.  
\* Now the biggest difference behind the scenes probably is how DOM Diffing works - so how Preact finds out whether it needs to update something and how React does it.  
\* To answer this question you can search for js-framework-benchmark - there a couple of different JavaScript frameworks are benchmarked against each other, you can also download this and run it on your machine though I will warn you that this takes quite a lot of time.  
\* But conveniently you can find latest results of these benchmarks in the GitHub Repo if you click on this image here:  
  
\* You should always be careful when analyzing such numbers because the tests here might not be important for the kind of app you’re building.  
\* For example the startup time is noticably better with Preact because of the size.  
\* We can also see that despite the smaller size, for operations like clearing a lot of rows, React is faster than Preact because file size of the library doesn’t matter for that, what matters is how performant the diffing algorithm and the updating algorithm is.  
\* So if the initial download time matters a lot and where you want to send as little code as possible across the wire, Preact might be an awesome alternative to React.  
**Congratulations**  
\* Great job!   
\* You made it through this course and therefore you learned a lot about React and the whole React ecosystem.  
\* You now definitely have all the core fundamentals and way more than that it takes to become a successful React developer.  
\* I can of course only wish you all the best for your future and I definitely want to encourage you here to do 1 important thing: practice, practice, practice.  
\* You practiced a lot throughout this course but to really excel at something, you have to keep that going and I can only encourage you to set your own goals, your own challenges, build Web Apps with React, set your own Demo App you want to build and then build it with React. This is how you learn the most about it and how you really become better and better using it and how you can build your successful React future.  
\* I wish you all the best for that and definitely share anything you build with React, I can’t wait to see it and I really mean it like that.  
\* So all the best to you, would of course be great to welcome you again in any of my other courses or any future course and I can only encourage you to keep on going.