The technical content of React js

Question-1 What are the front-end development guidelines and best practices?

Here are some more points to consider while developing application to boost user experience and performance of your application.

1. Minimizing HTTP Requests.
2. Adding Expires or Cache control header resulting in loading most of the contents in browser from cache rather than reloading.
3. Using Gzip/deflate compression for static and dynamic contents
4. Avoiding inline CSS. so as to load initial page faster
5. Using CSS on the top of the page
6. Avoiding inline javascript code.
7. Putting java scripts at the bottom of the pages.
8. Compressing JavaScript and CSS files.
9. Building reusable components viz. CSS, java scripts, routines, etc. to maximize the reusability and minimize the code size.
10. Cache the static contents viz. images, java scripts, CSS, etc. or deploy them on separate server(cookie less) for better performance (It reduces the waiting period of user and can start his/her work while static content gets loaded)
11. Images are taking at least 80% of the total page load time. We optimize image size and using lazy loading concepts (on demand loading) for images so that will not block required content. We are reducing number of DOM elements in pages.
12. Using minimum DOM elements in web pages.
13. Avoiding unnecessary server responses viz. “404 Page not found”.
14. URLs are meaningful & user friendly

15. Explanatory HTML page titles
16. Clear, descriptive Major headings
17. Using tags like Emphasis (bold, etc.) carefully
18. Avoiding @Import tag for CSS anywhere in website.
19. Following W3C standards for HTML and CSS.

Question-2: what is the best way to practice web development ?

But that is what everyone told you right?
So the thing is **how to practice in the most effective way**, without wasting your single bit of time.

Here is what I used to do or am doing most of the time:
To keep learning every tiny second of time, you have to challenge your capabilities. Take up a project which is well beyond your capabilities and stick to that project until you complete it. By the end of just 5-6 such projects you will found yourself almost proficient than others around you, even those professionals.

The key is - **Get yourself in trouble and you will find a way to get out of it**

It works for me at most of the time.

A step by step guide:

1. Take reference: try to make something that is already there, try to make those things. Ever wondered how the aside notifications of Facebook works?
Try something similar by your own.

2. **Gift a lot**: Now onward never gift someone a physical gift. Make something for them using your capabilities. They would be awwed..! no one gets this custom piece of stuff built specifically for them! Plus you will save some bucks for yourself.

3. Idea: Meanwhile the chances are that you may come up with an idea of your own. Don't let this idea die. Keep that alive and keep working on it. You will feel like you are never going to complete it or you will never be able to complete it, No worries you will be able to complete it.

4. Example1: In very early days of my learning I was learning how to make an objective test series through PHP and that's when I came up with an idea of making a fun test series for my friends. This test series would come up with interesting questions and scores as who knows who? and how much? There scores were displayed straight on those leader-boards. It was an instant hit right there among my friends. Can't share link due to some very personal questions listed there :)

5. Example 2: I was learning Ajax, and came up with an idea of making an instant chat application custom built for my own group. I build that. Its fast, reliable, and my friends say its great. All of us are using it from last 7 months. Reason? They can use it from almost any device that has access to internet. Again cant share you the link for privacy reasons.

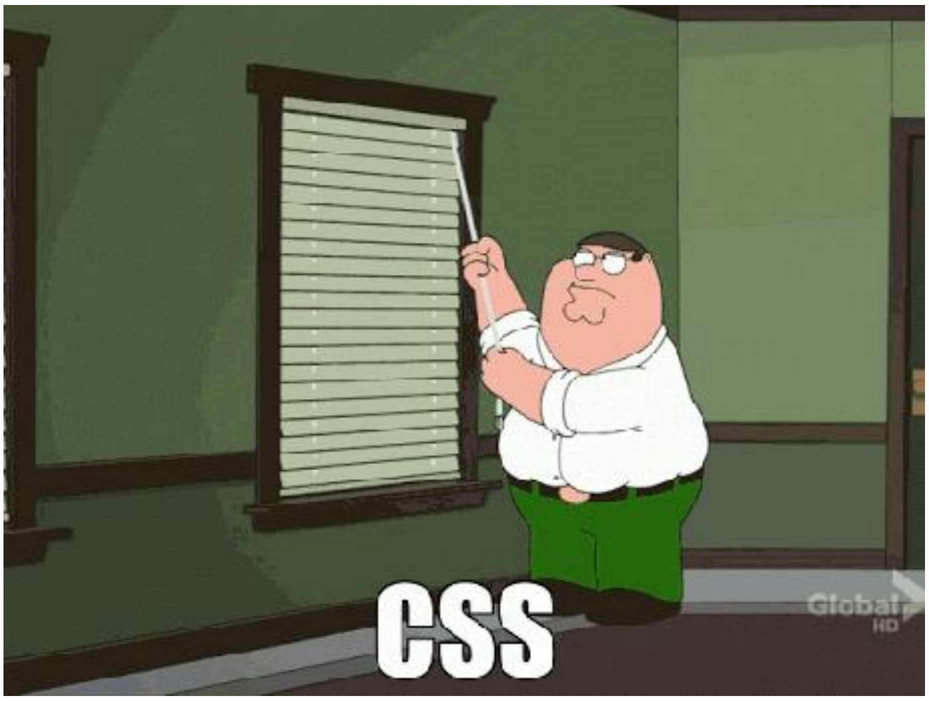
The final thing for your question:
PHP:
There are a lot of learning options available already there. You may try Udemy or something similar. There are the courses in which they make something from scratch. Take one or two of those courses and you will get an idea of how to proceed with your own projects.

Question-3 why CSS is so frustrating ?

I'm being totally blunt here.

In my opinion Front End Developers who complain about CSS being frustrating failed their job. Of course, the first one/two years can be hard, we've all been there. But after some years you shouldn't generally be complaining about CSS anymore.

Every few weeks one of my developer contacts on Twitter posts this:



And most of the time it gets around 20 likes.

All I can think of is: "Dude. Get another job. You're clearly incapable of doing Front End stuff."

CSS isn't just the stupid part of Front End Development. It is the most crucial part of Front End Develpment.

Get this in your head, many Front End Developers don't. Front End Developing means CSS, CSS and even more CSS. Being a Front End Developer means creating what the user sees. And what the user sees is defined by CSS.

Most jobs include tricky things. Like parking an 18-wheeler. Or doing a heart transplantation. Or landing a jet on an aircraft carrier. But you know what? If you're good at your job you like the tricky parts, because those make your job interesting.

Besides, CSS isn't frustrating at all as soon as you get your head around it.

The basics to master CSS:

1. Understand the difference between relative and absolute and what happens when you put something absolute into something relative.

2. Margin is spacing to the outside, padding to the inside.
3. Your best friends are box-sizing: border-box; overflow: hidden; and display: inline-block;
4. Every line has to matter. When somebody asks you why you added display:block; to a DIV, you need to know why. In the heat of the battle your CSS can get messy – but always take the time and clear out those artefacts.
5. Never ever use !important when you're angry (actually, try to never use it at all other than for testing).
6. But most importantly: think ahead. Don't just create DIVs at will. Every DIV must have a specific purpose. Don't create a DIV because you want something different. Think about how you could change that with the DIVs you already have. And as soon as you create a new one, ask yourself: "Do I really need it?"