# Tell Me About Yourself

# Describe a situation when you had a conflict at work and how you handled it

# How do you handle disagreements when working as part of a team?

In my previous experience, I was working with a group to prepare a presentation for leadership. The group was responsible for conceptualizing an idea and presenting a project plan to implement the idea. We selected an idea and developed an action plan for completing the presentation. Two of the team members wanted to hold daily meetings to discuss the plan and progress. Other team members, including myself, disagreed with this approach because the team had a log to report task progress and completion.

As long as everyone knew their assigned tasks, we did not think we needed to divert time for meetings that would not move the project forward. We also thought these meetings would use valuable time needed to complete the project. Instead of starting an argument about the meetings, we gathered everyone together to explain their reasoning for their side. Eventually, we decided to hold a daily meeting that would last no more than ten minutes to provide short updates on task progress

# Explain a situation in which you disagreed with your manager and how you handled it

When preparing data about the success of past projects to request funding for a new project in our department, a supervisor asked me to delete data records for projects that were less successful. I knew that deleting these records would skew the results and representation of our past performance. I expressed this concern to my supervisor, but she insisted that I remove the data.

After careful consideration, I decided to discuss the issue with the next level of management, which dismissed my concerns and supported the decision to remove the data. Ultimately, I removed data as directed, but I also changed the language in the report to clearly communicate that the reported data represented successful projects to ensure I accurately represented the information. I also documented the situation, including data records removed from reported results, in the event of questioning or an audit.

I understood my supervisor’s decision for the request, but I could not compromise my values. I tried to create a compromise in removing the data and adding the language. We did eventually receive funding for our project based on the reports.

# What is Code Review?

Code review is a software quality assurance process in which software’s source code is analyzed manually by a team or by using an automated code review tool. The motive is purely, to **find bugs**, **resolve errors**, and for most times, improving **code quality**. Reviewing the codebase makes sure that every software or new feature developed within the company is of high quality.

## ****Top 9 Practices for Reviewing Code****

### 1. Set goals and standards

Before implementing a code review process, it is imperative to decide on important metrics and define unambiguous goals. Goals include acceptable coding standards in the company. Having set standards makes sure that each software product developed in the company meets the company’s standards.

### 2. Communicate Goals and Expectations

Communication of goals and expectations is essential. Not conveying goals and expectations to everyone in the team can lead to ambiguity regarding the result. Knowing what is expected from a developer makes it easier for them to complete a task.

### 3. Define a Code Review Process

Now that the goals and expectations are set and communicated with everyone it is time to define a code review process. Having a defined code review process will help everyone stay on track and decrease the time spent on technical debt

### 4. Use a code review checklist

A good code review requires a well-defined checklist. This checklist can be used by the reviewer to make sure nothing is missed out.

### 5. Authors should annotate source code before the review

Annotation can be a useful tool throughout the software development cycle and also for the reviewer. It helps the code reviewer understand the code better and what each block of code does. We encourage developers to add annotations to their code but do not overdo it.

### 6. Review for no more than 60 minutes at a time

It is a well-known fact that a person’s efficiency can deplete if they try to work continuously for hours without taking a break. The same concept applies when trying to review code. It is not recommended to perform code reviews for more than 60 minutes. Research has stated that post the 60 minutes mark the reviewer’s efficiency can deplete can some of the defects that may go unnoticed.

### 7. Establish a process for fixing defects found

Fixing the defects post a code review process is an ultimate goal. Having a defined process to fix the defects will make sure that it is done in the most effective way and with the least technical debt.

### 8. Foster a positive code review culture

In many companies, code reviews are used to evaluate the developer’s performance. But code reviews should be used for more than that. It should be used to develop an environment of learning. Rather than just telling them their mistakes it should be seen that they learn how to solve them and make sure these are not repeated.

### 9. Automate to Save Time

Automated Code Review tools like Codegrip are great assets for every software company. Tools like these can help reduce code review times to a few seconds. They can scan the entire codebase in less than a minute and find defects and also provide solutions for them.

## ****Best Code Review Technique****

### 1. Instant Code Reviewing Technique

The most direct form of reviewing code is the Instant Code review technique. In this, the developer is writing code while the reviewer sits beside reading the code simultaneously and correcting it on the go. Also known as pair programming, this process is best suited for highly complex programs where two minds can solve the problem much quicker and efficiently.

While this process looks favorable for companies but in reality, the time and workforce needed by this technique make it unfavorable. Two or more people working on code together means fewer average lines per developer. Interruption for corrections also halts the flow of work for the author of the code and the learning curve for a developer hinders if constant support or solution is presented right away by a reviewer for a complex problem.

### 2. Ad-hoc (synchronous) Code Reviewing Technique

Also known as the “Over the Shoulder” code-review process. It is the most commonly used process with around 75% of companies participating in ad-hoc reviews. In this type of synchronous method, the coder produces the code and then asks the reviewer to review the code. The reviewer joins the coder at the screen and reviews the code while discussing it, over the shoulder. It is implemented wisely because it is informal and spontaneous. The process is successful only if the reviewer is available at the time or it disrupts the coder’s speed.

### 3. Meeting-Based Code Reviewing Technique

This is the least commonly used process with only 44% using it once a month. In meeting-based code review, coders complete their work, and a meeting is called. The whole tech team sits, commenting, and attempting to improve the code together. It is a temporary process as it is highly unlikely to perform constantly considering the amount of time, loss of workforce for the time, decreased efficiency, and inability to get the whole team together.

### 4. Tool-Based Code Reviewing Technique

This process is not done by a team together, at least not on the same screen. It is also called an asynchronous code review. In this, once the code gets finished, the coder makes it available for others to review. The reviewer will review the code on their screen commenting, or even amending the errors in the codes. Then notifying the coder who on her agenda will improve it. When there are no changes, the code is marked with no comments for improvements, and the software gets approved.

# Tell us about a time you had to make a quick choice using limited information. Please describe the situation, what actions you took, and the result.

# Tell us what you would do if you were asked to join a group of people to help complete a project.

# The rest of the group has been working together for weeks and you have new ideas on how to approach the project.

# Your ideas are different from the group’s current approach. How would you handle the situation?

# Please describe a time you were asked to perform skills or meet requirements that were outside of your comfort zone. Describe the situation, the steps you took, and the outcome.

# Please describe a time you had to cope with a high-pressure or stressful situation at work or in school. Describe the situation, how you responded, and your approach. What was the outcome?

# Please describe a time when you were faced with conflicting interests between what was right for a customer or coworker versus what was right for your company as a whole. Describe the situation, your actions, and the outcome.

# Most Recent Project

# Finally, ask the interviewer questions to find out if the job and the company is the right one for you. You could ask:

Where do you see the company in 5 years?

Can you describe the working culture of the organisation?

Can you tell me more about the team I would be working in?

What training programmes are available to your employees?

What do you enjoy most about working here?

When are you hoping for someone to fill this position?

What are the next steps in the interview process?

# About My Team project building process

* **Agile Development**

There were team leader, QA engineer, data engineers, UX designers, developers in my team.

Team leader had broad responsibilities that might include estimating stories and planning implementation details with the team.

The analysts created wireframes, documented user stories, and reviewed test results

Our team decided how much work it can complete during the sprint where the product owner shared priorities.

I could escalate blocks that impede the team’s progress and discussed the status of user stories, teammates shared our daily goals.

We used Jira for managing our issues for our project and decided on how to assign the task to someone

I resolved the issue that the Project Manager assigned me and send Pull Request to him.

Then he checked my code and merge my updated code to Github.

In this way, we built the product according to the requirements of every step.

We showed the functionality to the product owner to gain acceptance of the completed work.

After that, we discussed what went well and what needs improvement in our process in Retrospective meetings.

* **CI / CD pipeline**

I had to update the server whenever the source code was updated

Here’s a manual way to update the server

Compose SSH into the server.

Pull from VCS.

Check configs before starting the app.

Start app in containers.

Check logs and make sure app is running.

End SSH session.

I used Jenkins to automate this process.

At first, I installed Jenkins, choosed Freestyle project type there

Then I used github as a vcs. So new changes were encountered then Jenkins started the project build steps

So I implemented CI / CD pipeline completely in my project

1. I developed the code and commits the changes to a centralized code repository.
2. When the repot detects a change, it triggers the Jenkins server.
3. Jenkins gets the new code and carries out the automated build and testing. If any issues are detected while building or testing, Jenkins automatically informs the development team via a preconfigured method, like email or Slack.
4. The final package is uploaded to AWS Elastic Beanstalk, an application orchestration service, for production deployment.
5. The elastic beanstalk manages the provisioning of infrastructure, [load balancing](https://www.bmc.com/blogs/aws-elastic-load-balancing/), and scaling of the required resource type, such as EC2, RDS, or others.

# What is the biggest challenge you’ve faced in work?

In my most recent role, a former coworker left her job unexpectedly. During this time, he was the team leader on an important new project. When he left, my manager asked me to take it over. While I had never performed his job duties in the past, I had previous leadership experience. Using my leadership and problem-solving skills, I pushed the project forward, encouraged the rest of the team and completed his responsibilities as well as mine. We were able to complete that project successfully and on deadline

# Why Did You Leave Your Job

Unfortunately, in my last job, I wasn't able to use my training and experience fully.

I wanted to get more exciting opportunity and ideal match for my qualifications.

# What Are Your Career Goals

My current short-term goal is to develop and use my frontend and communications skills in a job like this one. However, I eventually want to develop into a position that allows me to continue to use these skills while also managing a team group. I will prepare for this goal by taking on leadership positions in team projects, and by developing my professional career through attending leadership conferences such as the one put on annually by your company.

# Do you have any questions?

- What does it take to do the job really well?

- What are the greatest challenges?

- It is important to me to get into my area of responsibility quickly. Will there be someone who will familiarize me with everything and show me everything in the first few weeks?

- Could you talk about company culture?

- What is the greatest challenge facing the company?

- What are the company's goals for the upcoming year?

# What motivates you

As a developer, I’ve always been motivated by creative projects, teamwork and being able to draw a connection between my efforts and the organization’s bottom line. One of the things I loved about my last job was witnessing the results of our team’s campaigns and watching as the leads we nurtured became customers. Having the opportunity to lead campaigns from ideation through launch was one of the reasons I was so excited to apply for this role

What motivating me to choose a developer career was time to have an opportunity to read the book "How to become a Genius".

I wanted to become Genius like others, and I have a role model in deep mind.

It was versatile according to specific events or stories that I have ever encountered.

While reading this book, I was moved about the expression, "Where I am today is my starting point. Who I am today is my starting point. My failures and successes of the past, my fears and hopes of the future are all shadows. Today is my reality, and I’ll use it to create my world"

Based on this query, I reminded myself what my strength is and what to do.

That lead me to here, over 7 years of developer career.

# What did you like most and least about your last job?

-Least

The company was growing quickly, so it was a bit disorganized and I often had multiple supervisors and senior team members giving me different instructions. However, I think this was a great opportunity to improve my ability to work under pressure and my communication skills. For example, I had to learn to communicate clearly and quickly to clarify with my managers and team if there was a conflict in what I had been asked to do. I enjoyed most aspects of the job and I ended up liking working in such a fast-paced environment in the end. It just took some adjustment. It was a great learning experience for me.

-Most

I love how the company offered learning opportunities and seemed to care about helping their employees develop and grow professionally. They had weekly lunch meetings where we’d learn a new topic, they offered tuition reimbursement, and all of the senior leadership team was very approachable and supportive.

# What sets you apart from other candidates?

I have a great willingness to learn. I think many prospects may expect to be hired for this position and not continue to cultivate their skills, but that is a priority.

# Do you have other interviews happening?

Honestly, I’m interviewing with a few companies for a range of positions, but they all come down to delivering an excellent experience.

But this company is really valuable to work for me and great fit in my interest.

So I am really focusing on the interview with this company.

# Do you want to tell us anything else about you?

I don’t have anything else to add. I feel like we covered the important topics, and I was able to share the key pieces of my background and how they’d help me perform well in the role. It was great learning about the role.

# What is my strength and weakness?

- strength

As mentioned, I have experience working as a frontend, backend side and full-stack developer.

Of course, I don’t know exactly if this experience will be helpful or not for your work.

However, in this period I have so many experiences in NodeJS, their frameworks or libraries including Nuxt, Next, React, Vue.

If you accept me, I will try to give you great satisfaction and success.

You probably will be satisfied.

My strength is creativity and tenacity. I love failure, learn from failure and how to deal with failure next time.

thus being optimistic, never give up to realize dream.

- weakness

I always try to avoid confrontation, in both my personal and professional life. This caused me to compromise sometimes on the quality of my work or what I needed to complete a project just to keep the peace. This became a real problem when I became a manager. One of the most critical aspects of managing people is telling them what they need to hear and not what they want to hear. I recognized this weakness and had been actively working to voice my opinions constructively and helpfully for the betterment of the team.

# What is your desired salary?

My current salary is $130,000, which was the maximum compensation that my employer allowed for this position. As this job has a greater number of responsibilities, I want you increase salary a bit.

# Where do you see yourself in 5 years?

In five years, I would like to become an engineering lead or team manager. As a web developer for your business, I know I would be able to take on a number of wide-ranging responsibilities related to the company’s web efforts. This would help me refine my knowledge of all areas of the company’s stack, which could then be applied in a managerial role further down the line.

# Why Are You the Best Person for the Job?

I am a self-motivated person who is willing to go above and beyond on any project, and to learn valuable skills on my own time. For example, I taught myself four computer programming languages in college, simply out of a passion to learn how to code.

My previous job working as a frontend expert provided me with the ideal experience for this position. For 9 years, I developed many of the skills required for this job, including performance optimization, data visualizing, Api integration, good communication skill, increasing site speed.

# What Is Your Greatest Strength?

I believe that my greatest strength is the ability to solve problems quickly and efficiently. I can see any given situation from multiple perspectives, which makes me uniquely qualified to complete my work even under challenging conditions. That problem solving allows me to be a better communicator. I am just as comfortable speaking to senior executives as I am junior team members. I think my ability to see all sides of an issue will make me a great asset to the team.

# Are you comfortable working with a team of engineers and programmers to develop new products or features?

I am very comfortable working with a team of engineers and programmers to develop new products or features. I have always enjoyed learning from my colleagues and using their expertise to help me solve problems. At my last job, we had a tight deadline to create a new feature for our software program. I was able to use the knowledge of my teammates to come up with a solution quickly

# What are some of the most important qualities that a technical lead should have

I believe that one of the most important qualities of a technical lead is communication. As a technical lead, I would need to communicate with my team members about their projects and how we can best meet our clients’ needs. Another quality that I think is important is problem-solving. In this role, I would need to solve any issues that arise during the project development process. Finally, I think leadership is another essential quality because I would need to motivate my team members and ensure that they feel comfortable asking me questions

# How do you stay up-to-date with the latest developments in your field

I subscribe to several newsletters and blogs related to my field. I also attend conferences and seminars where experts speak on the newest technologies. These resources help me learn more about what’s happening in the world of tech. They also give me ideas for how we can improve our processes at work.

# What is your experience with developing software applications?

In my last role, I led a team of developers who developed a new application for our client’s business. We had a tight deadline but managed to create a functional app within two months. The client was very happy with the final product and even asked us to develop another version of their app

# Provide an example of a time when you had to manage a team of engineers and programmers and how you handled the situation

In my last role as a senior engineer, I had to oversee a large-scale project involving many different departments. We had to create an entirely new system for our company’s website, which meant we needed to collaborate with other engineers, designers and programmers. I met with each member of my team individually to discuss their roles and responsibilities on the project. Then, I held weekly meetings with my entire team to check in on everyone’s progress and offer help if anyone was struggling.

# What is your greatest achievement?

In my most recent role, I was responsible for managing the orientation and training programs for our new hires. Unfortunately, the content was not so good. While it was necessary information for our new hires to have, we found that only 40% of new hires did not complete the training. We were also receiving poor feedback on the course evaluation forms. I decided to rework the training program to make it more relevant and interesting, based on industry best practices and feedback on the evaluation forms. Today, 90% of participants complete the training and provide positive feedback about their experience. My manager was so pleased with the improvements that he asked me to lead a training seminar.

# If hired for the position, what would be your priorities and goals as a technical lead?

My first priority as technical lead would be to ensure all projects I oversee meet their deadlines and stay within budget. To do this, I would create a plan for monitoring project progress and communicate regularly with my team members about any challenges they’re facing so we can find solutions together. Another priority would be to develop new training programs for employees who want to advance in their careers. I believe it’s important to provide opportunities for growth and development to everyone on staff.

# What would you do if you and your team members had different ideas about how to implement a new feature

I would first ask my team members why they think the feature should be implemented in a certain way. I’d also explain why I think it’s best to do things a different way. If we still couldn’t agree on one solution, I would take some time to research other ways of implementing the feature. After researching, I would present both ideas to my team again and see if anyone had any additional thoughts or feedback.

# How well do you communicate with non-technical employees, such as business analysts and product managers?

I find it important to understand the perspectives of all team members. I try to ask questions to learn more about what other employees think about the project. This helps me make decisions that benefit everyone involved. In my last role, I worked with both business analysts and product managers. I asked them for their opinions on certain aspects of the project so I could get a better idea of what they thought was most important

# Do you have experience working with legacy code? If so, can you provide an example

In my last role, I was tasked with updating an old system that had many bugs in its coding. The company wanted to use this system for more than just basic functions, but it couldn’t because of all the issues. So, I worked with my team to rewrite the entire program using modern coding techniques. We were able to successfully update the system and make it compatible with other programs

# When reviewing code written by other engineers, how do you provide constructive feedback

I find that providing constructive feedback is one of my favorite parts of being a technical lead. I enjoy helping others learn new coding languages or update existing ones. In my last role as a software engineer, I worked with an engineer who was learning Python for the first time. He struggled at first but eventually learned it well enough to apply it to his work. When reviewing his code, I provided him with specific feedback on what he did well and areas he could improve.

# We want to improve our use of automation in our development process. Are there any automation tools that you use regularly

I’ve used several different automation tools in my past roles, including Selenium, Jenkins CI, Maven, Ant, CruiseControl and TeamCity. I find that these tools are useful when working on large projects because they allow me to automate repetitive tasks so I can focus on more important aspects of development.

# What makes you stand out from other candidates for this position

I am an excellent communicator who is always willing to help others. I also have five years of experience as a technical lead, which means I know how to manage projects and teams effectively. In my last position, I helped my team complete our project ahead of schedule by delegating tasks and providing guidance when needed.

# Describe your experience with Agile development

I have extensive experience using Agile development. I find this method of software development to be highly beneficial because it allows me to collaborate with my team members while also testing new features as they’re created. When working on projects using Agile development, I typically start by creating user stories for each feature we want to add to our product. Then, I break down those stories into smaller tasks so my team can complete them more easily.

# Which programming languages do you have the most experience with?

I’ve worked mostly with Java and C++ throughout my career. These two languages are very similar but also different enough that I can use one when working on projects that require speed and efficiency and the other when working on projects that require more flexibility. I’m also familiar with JavaScript, React, Python.”

# What do you think is the most important thing that a technical lead can do to help their team succeed

I believe that one of the most important things a technical lead can do to help their team is create a culture of learning. When I first started as a technical lead, my team was struggling with some projects because they didn’t understand all of the elements involved in completing them. I scheduled time for us to learn more about our work so we could apply what we learned to our daily tasks. This strategy helped us improve our performance and develop new skills.

# How often do you make mistakes when programming and how do you go about fixing them

I make mistakes all the time when programming because it’s an iterative process where I’m constantly making changes and testing my code. However, I try to minimize the number of mistakes I make by thoroughly reading through the requirements before starting any project. If I do make a mistake, I immediately stop what I’m doing and go back to the original coding to find out what went wrong. Then, I fix the problem and test my code again

# There is a bug in the code and the deadline is fast approaching. What do you do?

If there was a bug in the code and the deadline was fast approaching, I would first try to find out what caused the bug. If I could fix the bug myself, I would do so immediately. If not, I would have my team members help me look for the cause of the bug. Once we found the cause, we would work together to fix the bug before the deadline.

# What are you passionate about?

My biggest passion is learning about software and technology. I was fascinated with computers from a young age, so I decided to study Computer Science in college and I’ve continued in that direction since graduating. It’s been a great choice for me! As a full stack software developer, I get to shape how the internet is progressing and how websites are evolving, and that’s exciting to me.

# Why Do You Want This Job?

Before the recruiter called me about this opportunity, I must admit that I didn't know much about your company. I've been doing some research in preparation for this discussion, and I need to say that I'm impressed by your work and your vision for the future. I want to be a part of this business as it grows, and I know my experience in product development would help your company

This opportunity is really exciting for me as I will be able to share my great experiences of making the architectural decisions I learned from my past enterprise-scale projects.

I see the role as a way of developing my career in a forward-thinking/well-established company/industry as it is a newly started Enterprise team.

I feel I will succeed in the role because I have lots of experience in React and TypeScript.

I believe my skills are well-suited to this job because I have much experience of functional programming skills and AWS experiences as well as React/Redux.

# Why do you want to work for us?

I have great respect for the quality products this company sells and would be proud to help maintain its positive reputation in the industry. I recently saw an article stating this company is a top-rated employer for encouraging innovation, and as a creative person I believe I would be an asset to the team

# Why do you want to work here?

I’ve been working in the travel industry for more than five years. I’ve expanded my experience in customer service, and I’m ready to apply that expertise with a global company that is committed to its employees and is consistently ranked one of the city’s best places to work. I love working with people, and I am particularly excited that this role will allow me to use my bilingual skills to speak to customers all over the world. My ultimate objective is to rise to a leadership position in customer service, and I know having the opportunity to grow my experience and improve upon my skills will help bring me closer to achieving that goal

# Do you prefer to work alone or on a team?

I have always thrived in positions where I can collaborate with others. I strongly believe that great work is achieved when more opinions and ideas are shared. However, I am fully capable of doing heads-down work when it’s needed.

# How would your coworkers describe you

My previous company had annual peer reviews that were very helpful in understanding how my coworkers viewed me. In their feedback, they consistently shared that I was reliable, a natural developer and someone who works well under pressure.

# How do you handle stress

My first position out of college had strict deadlines and a large workload. Although it was stressful at times, it allowed me to learn how to handle stress in the workplace. Organization and attention-to-detail help me to navigate these stressful situations. I create daily, weekly and monthly to-do lists to ensure I stay prioritized and on track.

# How important is work-life balance to you

A healthy work-life balance is important to me. I believe it’s normal to put in extra hours to finish tasks that are time-sensitive from time to time, however, I believe it’s also important to have time outside of work to relax and refresh so I can continue to be as productive as possible while at work

# Which of our company values resonates most with you

I identify with your company’s value of putting the customer first. I am a strong believer in great customer service. When I receive great customer service from a company I am much more likely to return. I would love to work for a company that promotes putting yourself in the shoes of a customer before making any decisions

# What is your Weakness?

1. The mocking API and full end-to-end test is always tricky for me and it takes much time. That’s why I spend my spare time practicing mocking API and full end-to-end tests by Cypress on my side project. I am still learning them.

I am joining Udemy classes related to the end-to-end tests.

Also I have been trying to volunteer work to build unit tests and end-to-end tests if possible to improve my skills.

1. Another thing is impatience with too big structure and frequent calls. I agree the elegant management flow and fair calls is really helpful for the work quality but sometimes I am impatient with frequent calls and meetings. So, recently I always try to share my opinion on how to organize our calls more efficiently with my team members.

And I am trying to use efficient tools to help me track my time and handle the public meetings efficiently.

1. I’ve always been on the React/Vue UI side and haven’t had much experience with database management, so I’d say that’s a weakness for me. However, I’m a quick learner, and I believe I could improve my SQL skills if I ever needed to for my job

# Why should we hire you?

I believe that my experience with technology, specifically in the web site and web application space, make me the best match for this position. In my previous job, I was responsible for maintaining and updating our company website. This required keeping employee profiles updated and continuously posting information regarding upcoming events. I truly enjoyed what I was doing, which is what drew me to this position with your company. I would love to bring the coding and content skills I learned there to this position.

# Describe the environment in which you work best.

I enjoy working as part of a team to achieve goals. My previous jobs have had open offices without cubicle walls, which encouraged us to get feedback from others about our ideas and equally share in the workload

# What Is Your Ideal Work Environment

My ideal work environment is one where I'm able to work as part of a team and that allows everyone's talents to grow.

As I researched your company, I noticed its devotion to cultivating each employee's skills and abilities.

I've found that this type of environment is most conducive to my productivity, especially in a position that requires me to constantly improve my full stack skills. It allows me to remain passionate about my job and helps me express my creativity to the best of my ability

# Describe your ideal boss or supervisor

My ideal boss is someone who allows employees to have freedom in daily operations, but is always available to answer questions and help when needed.

# Do you prefer to get feedback about your performance through formal reviews or informal meetings

I prefer regular performance evaluations at least once a year, but I also appreciate receiving informal feedback whenever it is appropriate

# Mentoring

I tell juniors how to accomplish what they are trying to do, but do not do it for them.

I regularly hold code reviews and use code reviews as an opportunity to point out best practices.

I get start early with planning and documentation instead of letting junior developers jump right into code.

I help a junior developer learn from his or her mistakes. There will be mistakes. I show them that mistakes are part of improving.

# What are the 3 most important things in your next role?

* Company vision/product

The company’s product should be innovative and promising.

That’s why I applied for this particular position in your company.

To me, your company’s vision and product is really promising and has good future plans.

* Culture

I look for companies who have a positive and adaptive culture, and trust.

It should be part of a positive culture where contributions are appreciated

The team should encourage each employee to suggest any good idea or solution for improving the product or innovating the team management.

And if it sounds good to the team’s vision, then the team should have a mind to accept it.

* Career

Growth potential is the most important thing to me with the company’s overall growth. I’d love to work for a company that is innovative and always looking for new opportunities to expand.

Able to learn new things and develop your skill set

# 7 JavaScript Design Patterns Every developer should know

Design patterns in JavaScript are reusable solutions applied to commonly occurring problems in writing JavaScript web applications.

It is quite appropriate to refer JavaScript design patterns as templates to provide solutions to problems but not quite to say that these patterns can replace the developers.

1. Constructor Design Pattern.

This is a special method that is used to initialize the newly created objects once a memory is allocated. Since JavaScript is typically object-oriented, it deals with objects most, therefore I intend to delve in to object constructors. There are three ways to create new objects in JavaScript

2. Prototype Pattern

The prototype pattern is based on prototypical inheritance whereby objects created to act as prototypes for other objects. In reality, prototypes act as a blueprint for each object constructor created

3. Module Design Pattern

In the module design pattern, there is an improvement from the prototype pattern. The different types of modifiers (both private and public) are set in the module pattern. You can create similar functions or properties without conflicts. There is the flexibility of renaming functions publicly. The daunting part of this is the inability to override the created functions from the outside environment

4. Singleton Pattern

It is essential in a scenario where only one instance needs to be created, for example, a database connection. It is only possible to create an instance when the connection is closed or you make sure to close the open instance before opening a new one. This pattern is also referred to as strict pattern, one drawback associated with this pattern is its daunting experience in testing because of its hidden dependencies objects which are not easily singled out for testing

5. Factory Pattern.

It is a creational concerned with the creation of objects without the need for a constructor. It provides a generic interface for creating objects, where we can specify the type of factory objects to be created.

6. Observer Pattern

The observer design pattern is handy in a place where objects communicate with other sets of objects simultaneously. In this observer pattern, there is no unnecessary push and pull of events across the states, rather the modules involved only modify the current state of data.

7. Command Pattern

The command design pattern encapsulates method invocation, operations, or requests into a single object so that we can pass method calls at our discretion. The command design pattern gives us an opportunity to issue commands from anything executing commands and delegates responsibility to different objects instead. These commands are presented in run() and execute() format.

# Why should we not update the state directly?

One should never update the state directly because of the following reasons.

* If you update it directly, calling the setState() afterward may just replace the update you made.
* When you directly update the state, it does not change this.state immediately. Instead, it creates a pending state transition, and accessing it after calling this method will only return the present value.
* You will lose control of the state across all components.

# Advantages of Laravel

Laravel benefit #1: Laravel makes implementing authentication very simple. Almost everything is configured out-of-the-box. Laravel also provides a simple way to organize authorization logic and control access to resources.

Laravel benefit #2. Laravel provides a clean, simple API over the popular SwiftMailer library. Laravel also provides drivers for SMTP, Mailgun, Mandrill, SparkPost, Amazon SES, PHP's “mail” function, and “sendmail”, allowing an application to quickly get started sending mail through a local or cloud-based service. In addition to support for sending email, Laravel provides support for sending notifications across a variety of delivery channels, including SMS (via Nexmo) and Slack.

Laravel benefit #3. Laravel supports popular cache backends like Memcached and Redis out-of-the-box. By default, Laravel is configured to use the file cache driver, which stores cached objects in the file system. For larger applications, it is better to use an in-memory cache such as Memcached or APC. However, with Laravel it is even possible to configure multiple cache configurations

Laravel benefit #4. Laravel helps to secure the web application by protecting it against the most serious security risks: SQL injection, cross-site request forgery, and cross-site scripting. Laravel itself is secure. We can tell you first hand that the codebase is fanatically guarded, and that the code has been vetted by several people.

Laravel benefit #5. Error and exception handling is already configured for any new Laravel-based project. In addition, Laravel is integrated with the Monolog logging library, which provides support for a variety of powerful log handlers.

Laravel benefit #6. Laravel is built with testing in mind. In fact, support for testing with PHPUnit is included out-of-the-box and a phpunit.xml file is already setup for the application. The framework also ships with convenient helper methods allowing for expressive testing of the applications. It provides easy ways for simulating basic behavior of users (making requests to the application and examining the output, for example, clicking links, filling out forms).

Laravel benefit #7. All Laravel routes are defined in the app/Http/routes.php file, which is automatically loaded by the framework. The most basic Laravel routes simply accept a URI and a Closure, providing a very simple and expressive method of defining routes.

Laravel benefit #8. Laravel is an MVC framework, so separation is already done. See the figure: the full MVC request cycle in a Laravel 5 application.

Laravel benefit #9. The Laravel queue service provides a unified API across a variety of different queue backends. Queues allow you to defer the processing of a time-consuming task, such as sending an e-mail, until a later time, which drastically speeds up web requests to your application.

Laravel benefit #10. In the past, developers have generated a Cron entry for each task they need to schedule. However, this is a headache. Such task schedule is no longer in source control, and developers must SSH into the server to add the Cron entries. The Laravel command scheduler allows for the fluent and expressive defining of command schedule within Laravel itself, and only a single Cron entry is needed on the server.

# Higher order component

A higher-order component acts as a container for other components.

This helps to keep components simple and enables re-usability.

They are generally used when multiple components have to use a common logic.

# Higher order component with Routes

When using React Router in React, one can use the Navigate component to navigate a user away from a page in case of a certain condition.

For example, the following example does not render a list if there is no data, but redirects a user to the home page instead:

import { Navigate } from 'react-router-dom';

const List = ({ data }) => {

if (!data.length) {

return <Navigate replace to='/home' />;

}

return (

<ul>

{data.map((item) => {

return <li key={item}>{item}</li>;

})}

</ul>

);

};

export default List;

In this case the redirect is well placed. However, if there is much logic happening before of the conditional, e.g. by using React Hooks, then the logic has to execute even though there may be a redirect.

import { Navigate } from 'react-router-dom';

const List = ({ data }) => {

// lots of hooks here

// which is bad, because they execute

// even though there may be a redirect

// and all the hooks logic may not be used after all

if (!data.length) {

return <Navigate replace to='/home' />;

}

return (

<ul>

{data.map((item) => {

return <li key={item}>{item}</li>;

})}

</ul>

);

};

export default List;

Therefore, you can use a higher-order component (HOC) for the redirect, because when wrapping the component into a HOC, the logic of the HOC would occur before the hooks from the wrapped component:

import { withRedirectIfBlank } from './withRedirect'

const List = ({ data }) => {

// lots of hooks here

return (

<ul>

{data.map((item) => {

return <li key={item}>{item}</li>;

})}

</ul>

);

};

export default withRedirectIfBlank({

redirectCondition: (props) => !props.data.length,

redirectTo: '/home',

})(List);

The HOC implementation could look like the following then:

import { Navigate } from 'react-router-dom';

const withRedirectIfBlank = (config) => (Component) => (props) => {

const { redirectCondition, redirectTo } = config;

if (redirectCondition(props)) {

return <Navigate replace to={redirectTo} />;

}

return <Component {...props} />;

};

export { withRedirectIfBlank };

Higher-Order Components are still useful these days, even though many React developers take them as legacy, because they are from a time when React Class Components where used. Especially when they are used to render conditional JSX. However, if not using any conditional JSX, using a Hook instead of a HOC is often a better design choice in modern React.

# How about pure component

A React component is considered pure if it renders the same output for the same state and props. For this type of class component, React provides the PureComponent base class. Class components that extend the React.PureComponent class are treated as pure components.

In practice, a React pure component looks like the following code.

class PercentageStat extends React.PureComponent {

render() {

const { label, score = 0, total = Math.max(1, score) } = this.props;

return (

<div>

<h6>{ label }</h6>

<span>{ Math.round(score / total \* 100) }%</span>

</div>

)

}

}

export default PercentageStat;

# Usememo and UseCallback, UseRef

useCallback

We can use the useCallback hook to return a memoized callback.

It takes a callback function as the first argument, and an array of value that changes for the callback in the first argument to be called.

This is useful when passing callbacks to optimized child components that rely on reference equality to prevent unnecessary renders.

useMemo

useMemo caches values that are computed from a function. The first argument is a function that computes the value, and the second argument is an array with the dependencies that are used to compute the returned value.

useMemo runs during rendering. It’s used as a performance optimization. React may forget previously memoized value and recalculate them on the next render in cases it needs to free memory for offscreen components for example.

Therefore, we should write code that works with useMemo and then add it to optimize performance.

useRef

The useRef hook returns a mutable ref whose current property is initialized to the passed argument. The returned object will persist for the full lifetime of the component.

# How do you pass data from one component to one other component and what are those

For passing the data from the child component to the parent component, we have to create a callback function in the parent component and then pass the callback function to the child component as a prop. This callback function will retrieve the data from the child component. The child component calls the parent callback function using props and passes the data to the parent component.

# Bundling

Bundling is the process of following imported files and merging them into a single file

# Code Splitting

Code-Splitting is a feature supported by bundlers like [Webpack](https://webpack.js.org/guides/code-splitting/), [Rollup](https://rollupjs.org/guide/en/#code-splitting) and Browserify (via [factor-bundle](https://github.com/browserify/factor-bundle)) which can create multiple bundles that can be dynamically loaded at runtime.

# Context

Context is designed to share data that can be considered “global” for a tree of React components, such as the current authenticated user, theme, or preferred language

Context is primarily used when some data needs to be accessible by many components at different nesting levels. Apply it sparingly because it makes component reuse more difficult.

# Error Boundaries

Error boundaries are React components that catch JavaScript errors anywhere in their child component tree, log those errors, and display a fallback UI instead of the component tree that crashed. Error boundaries catch errors during rendering, in lifecycle methods, and in constructors of the whole tree below them

# Higher-Order Components

A higher-order component (HOC) is an advanced technique in React for reusing component logic. HOCs are not part of the React API, per se. They are a pattern that emerges from React’s compositional nature.

# Refs and the DOM

Refs provide a way to access DOM nodes or React elements created in the render method.

# Profiler API

The Profiler measures how often a React application renders and what the “cost” of rendering is. Its purpose is to help identify parts of an application that are slow and may benefit from optimizations such as memorization

# Redux and Context API

Context API is a built-in React tool that does not influence the final bundle size, and is integrated by design.

Context provides a way to pass data through the component tree without having to pass props down manually at every level

Redux is an Open Source Library which provides a central store, and actions to modify the store. It can be used with any project using JavaScript or TypeScript,

Both are excellent tools for their own specific niche, Redux is overkill just to pass data from parent to child & Context API truly shines in this case.

Thus, Context API: Resourceful and ideal for small applications where state changes are minimal

Redux: Perfect for larger applications where there are high-frequency state updates

# Create the context

1. Create a context using createContext() and pass the initial state as arguments. Context can also be defined without passing any arguments.

2. Define a function that will deliver the data through the Provider.

3. Using useReducer() hook accepts a Reducer with the default state, then returns the updated state and dispatches a function.

4. Inside the provider function, use useReducer() and pass the Reducer and the initial state as arguments. The state returned and dispatch are then passed as values in the Provider.

Redux requires the following blocks to function:

Actions, Reducers, Store

- Lifecycle methods - Exp of Mount fact, Unmount face

Mounting means putting elements into the DOM.

React has four built-in methods that gets called, in this order, when mounting a component:

constructor()

getDerivedStateFromProps()

render()

componentDidMount()

The render() method is required and will always be called, the others are optional and will be called if you define them

The constructor() method is called before anything else, when the component is initiated, and it is the natural place to set up the initial state and other initial values.

The constructor() method is called with the props, as arguments, and you should always start by calling the super(props) before anything else, this will initiate the parent's constructor method and allows the component to inherit methods from its parent (React.Component).

The getDerivedStateFromProps() method is called right before rendering the element(s) in the DOM.

This is the natural place to set the state object based on the initial props.

It takes state as an argument, and returns an object with changes to the state.

The componentDidMount() method is called after the component is rendered.

This is where you run statements that requires that the component is already placed in the DOM.

Unmounting

Unmounting is when a component is removed from the DOM

React has only one built-in method that gets called when a component is unmounted:

componentWillUnmount()

The componentWillUnmount method is called when the component is about to be removed from the DOM.

# **Asynchronization**

React-async provides a declarative API to perform any REST API calls using a single React component, allowing declarative programming to be used throughout the application. It takes care of handling errors, promise resolution, and retrying promises, and deals with local asynchronous state.

you can write the very same component as a function with the useEffect hook.

Also we can implement aynchronization by using async/await

# Virtual Dom

React uses Virtual DOM exists which is like a lightweight copy of the actual DOM(a virtual representation of the DOM).

So for every object that exists in the original DOM, there is an object for that in React Virtual DOM.

It is exactly the same, but it does not have the power to directly change the layout of the document.

Manipulating DOM is slow, but manipulating Virtual DOM is fast as nothing gets drawn on the screen.

So each time there is a change in the state of our application, virtual DOM gets updated first instead of the real DOM.

# How to improve the performance of the application?

1. Using Immutable Data Structures

2. Function/Stateless Components and React.PureComponent

3. Multiple Chunk Files

4. Using Production Mode Flag in Webpack

5. Dependency optimization

6. Use React.Fragments to Avoid Additional HTML Element Wrappers

7. Avoid Inline Function Definition in the Render Function

8. Throttling and Debouncing Event Action in JavaScript

9. Avoid using Index as Key for map

10. Avoiding Props in Initial States

11. Spreading props on DOM elements

12. Use Reselect in Redux to Avoid Frequent Re-render

13. Avoid Async Initialization in componentWillMount()

14. Memoize React Components

15. CSS Animations Instead of JS Animations

16. Using a CDN

17. Using Web Workers for CPU Extensive Tasks

18. Analyzing and Optimizing Your Webpack Bundle Bloat

19. Consider Server-side Rendering

20. Enable Gzip Compression on Web Server

# Did you write any API Services? What type of Authentication did you use?

1. I have ever used REST API and Fetch API.

A REST API is an API that follows what is structured in accordance with the REST Structure for APIs. REST stands for “Representational State Transfer”. It consists of various rules that developers follow when creating APIs.

The way a REST API is structured depends on the product it’s been made for — but the rules of REST must be followed.

The fetch() API is an inbuilt JavaScript method for getting resources from a server or an API endpoint. It’s similar to XMLHttpRequest, but the fetch API provides a more powerful and flexible feature set.

The fetch() API method always takes in a compulsory argument, which is the path or URL to the resource you want to fetch. It returns a promise that points to the response from the request, whether the request is successful or not. You can also optionally pass in an init options object as the second argument.

Once a response has been fetched, there are several inbuilt methods available to define what the body content is and how it should be handled.

2. I used Oauth, JWT in my React project

# JWT? how it works?

JWT is JSON Web Token

The React app requests a JWT from the authentication server whenever the user wants to sign on.

The authentication server generates a JWT using a private key and then sends the JWT back to React app.

My React app stores this JWT and sends it to your backend server whenever user needs to make a request.

The backend server verifies the JWT using a public key and then reads the payload to determine which user is making the request

# JScript - Debouncing or throttling?

Throttling

In a nutshell, throttling means delaying function execution. So instead of executing the event handler/function immediately, you’ll be adding a few milliseconds of delay when an event is triggered. This can be used when implementing infinite scrolling, for example. Rather than fetching the next result set as the user is scrolling, you can delay the XHR call.

Another good example of this is Ajax-based instant search. You might not want to hit the server for every key press, so it’s better to throttle until the input field is dormant for a few milliseconds

Throttling can be implemented a number of ways. I can throttle by the number of events triggered or by the delay event handler being executed.

Debouncing

Unlike throttling, debouncing is a technique to prevent the event trigger from being fired too often. If you are using lodash, I can wrap the function you want to call in lodash’s debounce function

# Closures in application?

Closures are inner functions that have access to the outer function’s variables and parameters. Even after the outer function’s execution is finished, the inner functions have access to the variables in the outer function. Closures are everywhere in JavaScript

# **What are the features of React?**

JSX, Components, Virtual DOM, One-way data-binding, High performance

# **What is JSX?**

JSX is a syntax extension of JavaScript. It is used with React to describe what the user interface should look like. By using JSX, we can write HTML structures in the same file that contains JavaScript code

# **What is the virtual DOM?**

DOM stands for Document Object Model. The DOM represents an HTML document with a logical tree structure. Each branch of the tree ends in a node, and each node contains objects.

React keeps a lightweight representation of the real DOM in the memory, and that is known as the virtual DOM. When the state of an object changes, the virtual DOM changes only that object in the real DOM, rather than updating all the objects.

# Why use React instead of other frameworks, like Angular?

Easy creation of dynamic applications: React makes it easier to create dynamic web applications because it provides less coding and provides more functionality, whereas, with JavaScript applications, code tends to get complex very quickly.

Improved performance: React uses virtual DOM, which makes web applications perform faster. Virtual DOM compares its previous state and updates only those components in the real DOM, whose states have changed, rather than updating all the components — like conventional web applications.

Reusable components: Components are the building blocks of any React application, and a single app usually consists of multiple components. These components have their own logic and controls, and they can be reused through the application, which, in turn, dramatically reduces the development time of an application.

Unidirectional data flow: React follows a unidirectional data flow. This means that when designing a React app, we often nest child components within parent components. And since the data flows in a single direction, it becomes easier to debug errors and know where the problem occurs in an application at the moment.

Dedicated tools for easy debugging: Facebook has released a chrome extension that we can use to debug React applications. This makes the process of debugging React to web applications faster and easier.

# What is the difference between the ES6 and ES5 standards?

These are the few instances where ES6 syntax has changed from ES5 syntax:

Components and Function, exports vs export, require vs import

# What is an event in React?

An event is an action that a user or system may trigger, such as pressing a key, a mouse click, etc.

React events are named using camelCase, rather than lowercase in HTML.

With JSX, you pass a function as the event handler, rather than a string in HTML.

# What is an arrow function and how is it used in React?

An arrow function is a short way of writing a function to React.

It is unnecessary to bind ‘this’ inside the constructor when using an arrow function. This prevents bugs caused by the use of ‘this’ in React callbacks.

# What are the components in React?

Components are the building blocks of any React application, and a single app usually consists of multiple components. A component is essentially a piece of the user interface. It splits the user interface into independent, reusable parts that can be processed separately.

There are two types of components in React:

Functional Components: These types of components have no state of their own and only contain render methods, and therefore are also called stateless components. They may derive data from other components as props (properties).

Class Components: These types of components can hold and manage their own state and have a separate render method to return JSX on the screen. They are also called Stateful components as they can have a state.

# What is a state in React?

The state is a built-in React object that is used to contain data or information about the component. The state in a component can change over time, and whenever it changes, the component re-renders.

The change in state can happen as a response to user action or system-generated events. It determines the behavior of the component and how it will render.

# What are props in React?

Props are short for Properties. It is a React built-in object that stores the value of attributes of a tag and works similarly to HTML attributes.

Props provide a way to pass data from one component to another component. Props are passed to the component in the same way as arguments are passed in a function.

# Explain the lifecycle methods of components

getInitialState(): This is executed before the creation of the component.

componentDidMount(): Is executed when the component gets rendered and placed on the DOM.

shouldComponentUpdate(): Is invoked when a component determines changes to the DOM and returns a “true” or “false” value based on certain conditions.

componentDidUpdate(): Is invoked immediately after rendering takes place.

componentWillUnmount(): Is invoked immediately before a component is destroyed and unmounted permanently.

So far, if you have any doubts about the above React interview questions and answers, please ask your questions in the section below

# What is Redux?

Redux is an open-source, JavaScript library used to manage the application state. React uses Redux to build the user interface. It is a predictable state container for JavaScript applications and is used for the entire application’s state management.

# What are the components of Redux?

Store: Holds the state of the application.

Action: The source information for the store.

Reducer: Specifies how the application's state changes in response to actions sent to the store.

# What is the Flux

Flux is the application architecture that Facebook uses for building web applications. It is a method of handling complex data inside a client-side application and manages how data flows in a React application.

There is a single source of data (the store) and triggering certain actions is the only way way to update them.The actions call the dispatcher, and then the store is triggered and updated with their own data accordingly.

When a dispatch has been triggered, and the store updates, it will emit a change event that the views can rerender accordingly.

# What is React Router?

React Router is a routing library built on top of React, which is used to create routes in a React application.

# Why do we need to React Router?

It maintains consistent structure and behavior and is used to develop single-page web applications.

Enables multiple views in a single application by defining multiple routes in the React application.

# List some of the major advantages of React.

Some of the major advantages of React are:

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

# What are the limitations of React?

Limitations of React are listed below:

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

# NextJS

Advantages:

Easy installation, project build, modification, and required package found.

Optimal application performance due to the availability of automatic code splitting

Next JS allows optimized code bundles to be loaded lazily behind the scenes with the help of prefetching

It allows application code to use SSR or Server Side Rendering, thus offering SEO friendly flexibility, initial render to application view, and elimination of code download

Effective Hot-Module Replacement and powerful error reporting

# Javascript

Advantages:

1. Speed

Since JavaScript is an ‘interpreted’ language, it reduces the time required by other programming languages like Java for compilation. JavaScript is also a client-side script, speeding up the execution of the program as it saves the time required to connect to the server.

2. Simplicity

JavaScript is easy to understand and learn. The structure is simple for the users as well as the developers. It is also very feasible to implement, saving developers a lot of money for developing dynamic content for the web.

3. Popularity

Since all modern browsers support JavaScript, it is seen almost everywhere. All the famous companies use JavaScript as a tool including Google, Amazon, PayPal, etc.

4. Interoperability

JavaScript works perfect with other programming languages and therefore numerous developers prefer it in developing many applications. We can embed it into any webpage or inside the script of another programming language.

5. Server Load

As JavaScript operates on the client-side, data validation is possible on the browser itself rather than sending it off to the server. In case of any discrepancy, the whole website needs not to be reloaded. The browser updates only the selected segment of the page.

6. Rich Interfaces

JavaScript provides various interfaces to developers for creating catchy webpages. Drag and drop components or sliders may give a rich interface to the webpages. This leads to improved user-interactivity on the webpage.

7. Extended Functionality

Third-party add-ons like Greasemonkey (a Mozilla Firefox extension) allow the developers to add snippets of predefined code in their code to save time and money. These add-ons help the developers build JavaScript applications a lot faster and with much more ease than other programming languages.

8. Versatility

JavaScript is now capable of front-end as well as back-end development. Back-end development uses NodeJS while many libraries help in front-end development like AngularJS, ReactJS, etc.

9. Less Overhead

JavaScript improves the performance of websites and web applications by reducing the code length. The codes contain less overhead with the use of various built-in functions for loops, DOM access, etc.

# How to use componentWillUnmount with Functional Components in React

Functional Components: Functional components are some of the more common components that will come across while working in React

Class Components: A class component requires you to extend from React. Component and create a render function which returns a React element.

There is no render method used in functional components.

import React, {useState, useEffect} from 'react';

function Example() {

const [count, setCount] = useState(0);

useEffect(() => {

document.title = `You clicked ${count} times`;

});

return (

<div>

<p> You clicked {count} times</p>

<button onClick={() => setCount(count + 1)}>

Click me

</button>

</div>

);

}

Class Component Example

import React, { Component } from "react";

class ClassComponent extends React.Component{

constructor(){

super();

this.state={

count :0

};

this.increase=this.increase.bind(this);

}

increase(){

this.setState({count : this.state.count +1});

}

render(){

return (

<div style={{margin:'50px'}}>

<h1>Welcome to Geeks for Geeks </h1>

<h3>Counter App using Class Component : </h3>

<h2> {this.state.count}</h2>

<button onClick={this.increase}> Add</button>

</div>

)

}

}

export default ClassComponent;

# What is Provider Component?

The <Provider> components makes the Redux store available to any nested components that need to access the Redux store.

Since any React component in a React Redux app can be connected to the store, most applications will render a <Provider> at the top level, with the entire app’s component tree inside of it.

The Hooks and connect APIs can then access the provided store instance via React's Context mechanism.

# HTML + CSS

No front-end dev is a stranger to HTML and CSS. The ability to work with and craft user interfaces is necessary to every organization. At a high level, React developers should be able to:

Work with and write semantic HTML tags

Work with and write CSS selectors

Implement a CSS reset

Understand the box model and how to reset to border-box

Understand flexbox

Work with and implement responsive web principles including the proper user of media queries

# **JavaScript Fundamentals + ES6**

You can’t rock React without a firm understanding of the fundamental concepts that the JavaScript language provides, but these ES6 skills are also essential:

Variables and scoping

Understanding when and where you have access to the data you need is critical. Variables are the mechanism built into JavaScript that allow us to hold onto data in memory and access that data later on within our applications.

ES6 brought with it new keywords that we can use to store variables other than the traditional `var` keyword (like `let` and `const`). You may choose to live by the principle that unless you absolutely need `var`, use `const` until your linter tells you otherwise, then default to `let`.

Arrays and objects

React suggests a pattern that your `view is a function of your state`. Arithmetically put, that’s `v = f(s)`, and something you need to remember as you revisit your skills in the foundations of the library.

State is data that we want to show to users or items in memory that we can access in order to allow our users to interact with our data. We hold all of the data that we present on an object called state and access these bits of data via properties on this state object. This is how React received its name; when state changes, the view updates (`v = f(s);`). So your view ‘reacts’ to the changes that are made in your state object.

You should brush up on how to mutate objects and change values of properties on them. Don’t worry, React takes care of the mechanism through a nifty function called `setState()` to make this work to your advantage.

Array methods

It’s one thing to be able to store data and access it within arrays and objects. It’s another to be able to manipulate that data properly. The built-in JavaScript array methods are essential tools in every developer’s toolbox. Focus in on `.map`, `.filter` and `.reduce` for maximum impact.

Functions and arrow functions

In React, every single component you build is a function in one way or another. Remember that ‘classes’ are just `constructor functions` under the hood. Regardless of the syntax you’re using, when building `functional components` or `class components` you’re using some form of a function.

Don’t underestimate the importance of these fundamentals. Many practices out there today that lend themselves to functional programming. The chance to use JavaScript functions to build out small chunks of UI is like building a Lego set without instructions. Each piece of UI is an encapsulated function that contains the state data your elements need to present, the elements themselves and the formal component logic you need to use that logic. Each component is a Lego brick, and it’s up to you to fit them all together.

DOM Manipulation and event handlers

In React, manipulating the actual DOM elements is rare. Remember we now have the JSX abstraction at our disposal. The native event object that you get with normal DOM manipulation in React is actually wrapped up in what’s called the SyntheticEvent. Make sure you can attach different types of events to HTML elements such as `onclicks`, `onchange`, `mouseenter`, etc.

The “this” keyword

The ‘this’ keyword is one of the commonly misused features in JavaScript. Think of ‘this’ as a pointer to an object. For example, you can use the ‘this’ keyword to reference an object without having to refer to that object’s name.

Higher order functions and callback functions

The idea that functions can be passed around as arguments—in the case of high order functions and callbacks—is what drives the `input/output` model of functional programming.

You pass handlers around everywhere in React. Most of the time the handlers you pass around are in the form of methods that are chained onto an object and accessed as properties, which will be bound up in the prototype chain. However, there are moments that you need to reach outside of the React component paradigm or create a few different types of components that extend some of the functionality to one another. These patterns are commonly referred to as advanced React patterns and they’re finding their way into the better/common practices realm. React will push you to be innovative and creative as you scale along with it.

Prototypal inheritance and object creation

React lends itself to a functional programming paradigm in many aspects. However, you work in the world of classes and, as a result, a world of object creation. If you understand the basics of how the prototype chain works in JavaScript, you’ll know what you need to about how we achieve inheritance in JavaScript. Remember, that there are no traditional classes in JavaScript. The class keyword is just syntactic sugar on top of the `object prototype` chain in JavaScript.

The ‘class’ keyword

JavaScript classes aren’t the same as classes in a traditional programming sense. You create classes that encapsulate your template logic, formal JavaScript logic and even styles known as components. These components are the building blocks of any React application, and there are only two ways to write the basic component: either by declaring it as a function or declaring it as a class.

Assess your proficiency with classes by making sure you can answer:

How do I set up methods on a class?

How do I bind those methods together?

How do I access properties that are found on the constructor?

How do I create objects that would be considered ‘children’ of parent objects?

4. Git

Git is essential to every developer's toolkit for storing projects on solutions like GitHub, Bitbucket and GitLab. Skills that should just be part of your day-to-day include:

Tracking changes with add, commit, push and pull

Branching and merging strategies

Handling merge conflicts

5. Node + npm

Node may be a surprise to many. Why would you need to know how to work with Node in order to be a client-side React dev? While you can pull React into any HTML file, there will be many other packages out there that will allow you to extend the React library.

React developers need to have a solid understanding of the npm registry. This is the place where software developers can go to get software to help them build software. Sounds funny, but truly that’s all the npm is: a cloud storage for packages we call dependencies.

Yarn vs npm

Yarn is a package manager that is built to utilize the npm registry. Yarn actually optimizes your npm workflows. Yarn and npm somewhat compete today, but the mission of Yarn has been to solve a lot of the problems that are accepted in the Node/npm ecosystem. npm has been doing everything it can to follow the patterns and practices that Yarn presents.

# Redux

Redux is a way to manage the “state” or we can say it is a cache or storage that can be accessed by all components with a structured way. It has to be accessed through a “Reducer” and “Actions”

5. Callbacks

In javascript, a callback is simply a function that is passed to another function as a parameter and is invoked or executed inside the other function. Here a function needs to wait for another function to execute or return value and this makes the chain of the functionalities (when X is completed, then Y executed, and it goes on.). This is the reason callback is generally used in the asynchronous operation of javascript to provide the synchronous capability.

- Promises

Promises are useful in asynchronous javascript operation when we need to execute two or more back to back operations (or chaining callback), where each subsequent function starts when the previous one is completed. A promise is an object that may produce a single value sometime in the future, either a resolved value or a reason that it’s not resolved (rejected).

- Async & Await

Stop and wait until something is resolved. Async & await just syntactic sugar on top of Promises and like promises it also provides a way to maintain asynchronous operation more synchronously. So in javascript asynchronous operation can be handled in various versions…

ES5 -> Callback

ES6 -> Promise

ES7 -> async & await

const showPosts = async () => {

const response = await fetch('https://jsonplaceholder.typicode.com/posts');

const posts = await response.json();

console.log(posts);

}

showPosts();

- Context

Context is designed to share data that can be considered “global” for a tree of React components, such as the current authenticated user, theme, or preferred language

const ThemeContext = React.createContext('light');

class App extends React.Component {

render() {

// Use a Provider to pass the current theme to the tree below.

// Any component can read it, no matter how deep it is.

// In this example, we're passing "dark" as the current value.

return (

<ThemeContext.Provider value="dark">

<Toolbar />

</ThemeContext.Provider>

);

}

}

// A component in the middle doesn't have to

// pass the theme down explicitly anymore.

function Toolbar() {

return (

<div>

<ThemedButton />

</div>

);

}

class ThemedButton extends React.Component {

// Assign a contextType to read the current theme context.

// React will find the closest theme Provider above and use its value.

// In this example, the current theme is "dark".

static contextType = ThemeContext;

render() {

return <Button theme={this.context} />;

}

}

# **Ref**

Refs are created using React.createRef() and attached to React elements via the ref attribute. Refs are commonly assigned to an instance property when a component is constructed so they can be referenced throughout the component.

class MyComponent extends React.Component {

constructor(props) {

super(props);

this.myRef = React.createRef();

}

render() {

return <div ref={this.myRef} />;

}

}

Accessing Refs

When a ref is passed to an element in render, a reference to the node becomes accessible at the current attribute of the ref.

const node = this.myRef.current;

The value of the ref differs depending on the type of the node:

. When the ref attribute is used on an HTML element, the ref created in the constructor with React.createRef() receives the underlying DOM element as its current property.

. When the ref attribute is used on a custom class component, the ref object receives the mounted instance of the component as its current.

. You may not use the ref attribute on function components because they don’t have instances.

# **PureFunction**

Pure Function is a function (a block of code ) that always returns the same result if the same arguments are passed.

function pureFunction(param) {

return param + 1

}

# TypeScript

TypeScript is a super set of JavaScript.

TypeScript builds on top of JavaScript. First, you write the TypeScript code. Then, you compile the TypeScript code into plain JavaScript code using a TypeScript compiler

There are two main reasons to use TypeScript:

TypeScript adds a type system to help you avoid many problems with dynamic types in JavaScript.

TypeScript implements the future features of JavaScript a.k.a ES Next so that you can use them today.

# Component Lifecycle

Mounting

These methods are called in the following order when an instance of a component is being created and inserted into the DOM:

constructor()

static getDerivedStateFromProps()

render()

componentDidMount()

Updating

An update can be caused by changes to props or state. These methods are called in the following order when a component is being re-rendered:

static getDerivedStateFromProps()

shouldComponentUpdate()

render()

getSnapshotBeforeUpdate()

componentDidUpdate()

Unmounting

This method is called when a component is being removed from the DOM:

componentWillUnmount()

# What is JavaScript?

JavaScript is a client-side and server-side scripting language inserted into HTML pages and is understood by web browsers. JavaScript is also an Object-based Programming language

# What are JavaScript Data Types?

Following are the JavaScript Data types:

Number

String

Boolean

Object

Undefined

10. Write the code for adding new elements dynamically?

<html>

<head>

<title>t1</title>

<script type="text/javascript">

function addNode () { var newP = document. createElement("p");

var textNode = document.createTextNode(" This is a new text node");

newP.appendChild(textNode); document.getElementById("firstP").appendChild(newP); }

</script> </head>

<body> <p id="firstP">firstP<p> </body>

</html>

A JavaScript callback is a function which is to be executed after another function has finished execution.

A more formal definition would be - Any function that is passed as an argument to another function so that it can be executed in that other function is called as a callback function.

The Promise object represents the eventual completion (or failure) of an asynchronous operation and its resulting value.

# Advantages of MongoDB over RDBMS

• Schema less − MongoDB is a document database in which one collection holds different documents. Number of fields, content and size of the document can differ from one document to another.

• Structure of a single object is clear.

• No complex joins.

• Deep query-ability. MongoDB supports dynamic queries on documents using a document-based query language that's nearly as powerful as SQL.

• Tuning.

• Ease of scale-out − MongoDB is easy to scale.

• Conversion/mapping of application objects to database objects not needed.

• Uses internal memory for storing the (windowed) working set, enabling faster access of data.

Why Use MongoDB?

• Document Oriented Storage − Data is stored in the form of JSON style documents.

• Index on any attribute

• Replication and high availability

• Auto-Sharding

• Rich queries

• Fast in-place updates

• Professional support by MongoDB

Where to Use MongoDB?

• Big Data

• Content Management and Delivery

• Mobile and Social Infrastructure

• User Data Management

• Data Hub

# Generator Function In Javascript

A generator is a function that can stop midway and then continue from where it stopped.

Generators are functions that can be exited and later re-entered.

Their context (variable bindings) will be saved across re-entrances

Calling a generator function does not execute its body immediately;

an iterator object for the function is returned instead.

When the iterator's next() method is called, the generator function's body is executed until the first yield expression, which specifies the value to be returned from the iterator or, with yield\*, delegates to another generator function.

The next() method returns an object with a value property containing the yielded value and a done property which indicates whether the generator has yielded its last value, as a boolean.

Calling the next() method with an argument will resume the generator function execution, replacing the yield expression where an execution was paused with the argument from next().

A return statement in a generator, when executed, will make the generator finish (i.e. the done property of the object returned by it will be set to true).

If a value is returned, it will be set as the value property of the object returned by the generator.

Much like a return statement, an error thrown inside the generator will make the generator finished -- unless caught within the generator's body.

When a generator is finished, subsequent next() calls will not execute any of that generator's code, they will just return an object of this form: {value: undefined, done: true}.

# What is Memoization in Javascript

Memoization is a top-down, depth-first, optimization technique of storing previously executed computations. Whenever the program needs the result of these computations, the program will not have to execute that computation again. Instead, it will reuse the result of the previously executed computation. This way the program will not have to repeat expensive computations. An expensive function is a function that takes some time to execute.

# The "use strict" Directive

The purpose of "use strict" is to indicate that the code should be executed in "strict mode".

With strict mode, you can not, for example, use undeclared variables.

# **Hoisting**

is JavaScript's default behavior of moving all declarations to the top of the current scope

# WebPack Plugins

are the backbone of webpack. Webpack itself is built on the same plugin system that you use in your webpack configuration!

|  |  |
| --- | --- |
| [BannerPlugin](https://webpack.js.org/plugins/banner-plugin) | Add a banner to the top of each generated chunk |
| [CommonsChunkPlugin](https://webpack.js.org/plugins/commons-chunk-plugin) | Extract common modules shared between chunks |
| [CompressionWebpackPlugin](https://webpack.js.org/plugins/compression-webpack-plugin) | Prepare compressed versions of assets to serve them with Content-Encoding |
| [ContextReplacementPlugin](https://webpack.js.org/plugins/context-replacement-plugin) | Override the inferred context of a require expression |
| [CopyWebpackPlugin](https://webpack.js.org/plugins/copy-webpack-plugin) | Copies individual files or entire directories to the build directory |
| [DefinePlugin](https://webpack.js.org/plugins/define-plugin) | Allow global constants configured at compile time |
| [DllPlugin](https://webpack.js.org/plugins/dll-plugin) | Split bundles in order to drastically improve build time |
| [EnvironmentPlugin](https://webpack.js.org/plugins/environment-plugin) | Shorthand for using the [DefinePlugin](https://webpack.js.org/plugins/define-plugin) on process.env keys |
| [EslintWebpackPlugin](https://webpack.js.org/plugins/eslint-webpack-plugin) | A ESLint plugin for webpack |
| [HotModuleReplacementPlugin](https://webpack.js.org/plugins/hot-module-replacement-plugin) | Enable Hot Module Replacement (HMR) |
| [HtmlWebpackPlugin](https://webpack.js.org/plugins/html-webpack-plugin) | Easily create HTML files to serve your bundles |
| [IgnorePlugin](https://webpack.js.org/plugins/ignore-plugin) | Exclude certain modules from bundles |
| [LimitChunkCountPlugin](https://webpack.js.org/plugins/limit-chunk-count-plugin) | Set min/max limits for chunking to better control chunking |
| [MinChunkSizePlugin](https://webpack.js.org/plugins/min-chunk-size-plugin) | Keep chunk size above the specified limit |
| [MiniCssExtractPlugin](https://webpack.js.org/plugins/mini-css-extract-plugin) | creates a CSS file per JS file which requires CSS |
| [NoEmitOnErrorsPlugin](https://webpack.js.org/configuration/optimization/#optimizationemitonerrors) | Skip the emitting phase when there are compilation errors |
| [NormalModuleReplacementPlugin](https://webpack.js.org/plugins/normal-module-replacement-plugin) | Replace resource(s) that matches a regexp |
| [NpmInstallWebpackPlugin](https://webpack.js.org/plugins/install-webpack-plugin) | Auto-install missing dependencies during development |
| [ProgressPlugin](https://webpack.js.org/plugins/progress-plugin) | Report compilation progress |
| [ProvidePlugin](https://webpack.js.org/plugins/provide-plugin) | Use modules without having to use import/require |
| [SourceMapDevToolPlugin](https://webpack.js.org/plugins/source-map-dev-tool-plugin) | Enables a more fine grained control of source maps |
| [EvalSourceMapDevToolPlugin](https://webpack.js.org/plugins/eval-source-map-dev-tool-plugin) | Enables a more fine grained control of eval source maps |
| [TerserPlugin](https://webpack.js.org/plugins/terser-webpack-plugin/) | Uses Terser to minify the JS in your project |

# PUT vs PATCH

PUT is a method of modifying resource where the client sends data that updates the entire resource. It is used to set an entity’s information completely. PUT is similar to POST in that it can create resources, but it does so when there is a defined URI. PUT overwrites the entire entity if it already exists, and creates a new resource if it doesn’t exist.

For example, when you want to change the first name of a person in a database, you need to send the entire resource when making a PUT request.

{“first": "John", "last": "Stone”}

To make a PUT request, you need to send the two parameters; the first and the last name.

Unlike PUT, PATCH applies a partial update to the resource.

This means that you are only required to send the data that you want to update, and it won’t affect or change anything else. So if you want to update the first name on a database, you will only be required to send the first parameter; the first name.

# What is the difference between HTTP and HTTPS?

* **Protocol**

The HTTP protocol stands for Hypertext Transfer Protocol, whereas the HTTPS stands for Hypertext Transfer Protocol Secure.

* **Security**

The HTTP protocol is not secure protocol as it does not contain SSL (Secure Sockets Layer), which means that the data can be stolen when the data is transmitted from the client to the server. Whereas, the HTTPS protocol contains the SSL certificate that converts the data into an encrypted form, so no data can be stolen in this case as outsiders do not understand the encrypted text.

* **Port numbers**

The HTTP transmits the data over port number 80, whereas the HTTPS transmits the data over 443 port number. Under the documentation issued by Tim Berners-Lee, he stated that "if the port number is not specified, then it will be considered as HTTP".

When RFC 1340 was announced, then the IETF (Internet Engineering Task Force) provided port number 80 to the HTTP. When the new RFC was released in the year 1994, the HTTPS is assigned with a port number 443.

* **Layers**

The HTTP protocol works on the application layer while the HTTPS protocol works on the transport layer. As we know that the responsibility of the transport layer is to move the data from the client to the server, and data security is a major concern. HTTPS operates in the transport layer, so it is wrapped with a security layer.

* **SSL Certificates**

When we want our websites to have an HTTPS protocol, then we need to install the signed SSL certificate. The SSL certificates can be available for both free and paid service. The service can be chosen based on business needs.

The HTTP does not contain any SSL certificates, so it does not decrypt the data, and the data is sent in the form of plain text.

* **SEO Advantages**

The SEO advantages are provided to those websites that use HTTPS as GOOGLE gives the preferences to those websites that use HTTPS rather than the websites that use HTTP.

* **Online Transactions**

If we are running an online business, then it becomes necessary to have HTTPS. If we do not use the HTTPS in an online business, then the customers would not purchase as they are scared that their data can be stolen by the outsiders.

**Why Choose Tailwind CSS?**

Tailwind CSS’s [official documentation](https://tailwindcss.com/) describes it as a “utility-first CSS framework” that comes with classes equipped to build custom UI designs directly in the users’ markup. It is handy to implement inline styling to rustle up a stunning UI without writing any CSS.

* **Highly Customizable:**Tailwind CSS comes with a default configuration, but it can be overridden with a **tailwind.config.js** file. This enables easy customization of styling, themes, spacing, palettes, etc. Use Tailwind’s utilities to enable easy project management and develop a website that delights customers.
* **Carries Commonly used Utility Pattern:** With Tailwind CSS, users can cut down on having to name too many classes. It comes with common utility patterns to deal with standard requirements: specifying and organizing classes, cascading classes, etc. In simple terms, creating custom components become so much easier. Instead of hard-coding, just use the theme() function to derive values from configuration files.
* **PurgeCSS Optimization:**[PurgeCSS](https://purgecss.com/) reduces file size by scanning HTML code and eliminating unused classes. In combination with Tailwind CSS, this is particularly useful. As a project expands, so does the size of accompanying CSS files. Optimizing via PurgeCSS reduces and cleans up CSS file size, making it infinitely more manageable, especially before deployment.
* **Responsiveness Made Easy:** By default, Tailwind CSS utilizes a mobile-first approach. To quote the [website](https://tailwindcss.com/docs/responsive-design), “*Every utility class in Tailwind can be applied conditionally at different breakpoints, which makes it a piece of cake to build complex responsive interfaces without ever leaving your HTML.*”Utility classes simplify the creation of complex responsive layouts but allowing them to be used across [multiple breakpoints](https://www.browserstack.com/guide/responsive-design-breakpoints). The result? Hassle-free implementation of responsive design.
* **Effortless Community Interaction**: Tailwind’s active community is an excellent place for users to get their questions answered, especially when dealing with stubborn issues. Get CSS-related queries solved and create excellent websites faster.

Why Choose Bootstrap?  
[Bootstrap](https://getbootstrap.com/) is an open-source framework containing CSS and JavaScript-based templates for interface components.

* **Massive Ecosystem:**Among front-end frameworks, Bootstrap’s ecosystem is unmatched. It offers, out of the box, a vast library of layouts, themes, UI elements, panels, modals, buttons, alerts, cards, etc., that devs and designers can choose from and implement. Additionally, Bootstrap is backed by best-in-industry community support.
* **Accelerated prototyping:**When using Bootstrap, designers can just write out their HTML code, include the relevant CSS classes, and achieve website responsiveness. They don’t have to spend time adjusting for browser incompatibility, CSS positioning, and the like.
* **Twitter Support**: Unsurprisingly, when a significant commercial player backs an open-source project, users can be assured that it is here to stay and carries high confidence among people who know the industry. The fact that Bootstrap grew out of, and is backed by Twitter, establishes its efficacy.
* **Supports SASS and LESS**: Although most developers don’t use LESS, significant projects rely on it. Obviously, SASS support is also highly desirable. Not too many CSS frameworks other than Bootstrap support both.

# Define SASS?

SASS means Syntactically Awesome Style sheets. It is a CSS preprocessor which is used to reduce repetition with CSS and save time. It adds power and elegance to the basic language and facilitates you to add variables, nested rules, mixins, inline imports, inheritance and more, all with fully CSS-compatible syntax.

# What are the most significant advantages of using GraphQL over REST?

A list of the most significant advantages of using GraphQL over REST:

* There is only one endpoint in GraphQL, but REST has multiple endpoints. That's why GraphQL is more cost-effective than REST. You don't have to use your resources for various endpoints.
* In REST API, you have to use multiple requests to retrieve a complex data-set, but in GraphQL, you can execute a complex query easily in just a single request.
* You can change the request data format in GraphQL, but it is not possible to do the same in REST.
* The development speed in GraphQL is faster than REST.
* GraphQL provides high consistency across all platforms, while In REST, it is hard to get consistency across all platforms.
* GraphQL doesn't support an automatic caching system, while REST uses caching automatically.

# What is Apollo in GraphQL?

Apollo is a platform for the implementation of GraphQL. As we know that GraphQL is a query language, so in order to use this query language easily, we need a platform, Apollo provides that platform.

Apollo provides two open-sourced libraries to create client and server. Here, the client is used to fetch data from a GraphQL server, and the server is used to create an API for GraphQL client.

# What do you know by Mutation in GraphQL?

Mutation is one of the most important operations in GraphQL. It is used for write operation when you want to add delete and edit data.

# What is subscription in GraphQL?

In GraphQL, the subscription is used for listening for any data changes. The server sends a notification message to the client after any data changes, if the client is subscribed to that event.

# What do you mean by a Query in GraphQL?

A GraphQL query is used to read data. It is similar to the GET request we use in REST APIs. The GraphQL queries are used to retrieve data from the GraphQL server.

# What does useEffect do?

By using this Hook, you tell React that your component needs to do something after render. React will remember the function you passed (we’ll refer to it as our “effect”), and call it later after performing the DOM updates. In this effect, we set the document title, but we could also perform data fetching or call some other imperative API.

# Why is useEffect called inside a component?

Placing useEffect inside the component lets us access the count state variable (or any props) right from the effect. We don’t need a special API to read it — it’s already in the function scope. Hooks embrace JavaScript closures and avoid introducing React-specific APIs where JavaScript already provides a solution.

# Does useEffect run after every render?

Yes! By default, it runs both after the first render and after every update. Instead of thinking in terms of “mounting” and “updating”, you might find it easier to think that effects happen “after render”. React guarantees the DOM has been updated by the time it runs the effects.

# Private Key

The private key is used to both encrypt and decrypt the data. This key is shared between the sender and receiver of the encrypted sensitive information. The private key is also called symmetric being common for both parties. Private key cryptography is faster than public-key cryptography mechanism.

# Public Key

The public key is used to encrypt and a private key is used decrypt the data. The private key is shared between the sender and receiver of the encrypted sensitive information. The public key is also called asymmetric cryptography.

Public and private keys can also be used to create a digital signature. A digital signature assures that the person sending the message is who they claim to be.

Typically, we use the recipient’s public key to encrypt the data and the recipient then uses their private key to decrypt the data. However, using the scheme of digital signatures, there’s no way to authenticate the source of the message. Mike could get a hold of Alice’s public key (since it’s public) and pretend that Bob is the person sending a message to Alice.

To create a digital signature, Bob digitally signs his email to Alice using his private key. When Alice receives the message from Bob, she can verify the digital signature on the message came from Bob by using his public key. As the digital signature uses Bob’s private key, Bob is the only person who could create the signature.

# ****What Is a REST API?****

REST is basically an architectural style of the web services that work as a channel of communication between different computers or systems on the internet. The term REST API is something else.

Those application programming interfaces that are backed by the architectural style of REST architectural system are called REST APIs. REST API compliant web services, database systems, and computer systems permit requesting systems to get robust access and redefine representations of web based resources by deploying a predefined set of stateless protocols and standard operations.

By these protocols and operations and redeploying the manageable and updatable components without causing the effect on the system, REST API systems deliver fast performance, reliability, and more progression.

# ****What Is a SOAP API?****

SOAP is a standard communication protocol system that permits processes using different operating systems like Linux and Windows to communicate via [HTTP](https://dzone.com/articles/the-http-series-part-1-overview-of-the-basic-conce) and its [XML](https://dzone.com/articles/writing-and-reading-xml-file). SOAP based APIs are designed to create, recover, update and delete records like accounts, passwords, leads, and custom objects.

These offers over twenty different kinds of calls that make it easy for the API developers to maintain their accounts, perform accurate searches and much more. These can then be used with all those languages that support web services.

SOAP APIs take the advantages of making web based protocols such as HTTP and its XML that are already operating the all operating systems that are why its developers can easily manipulate web services and get responses without caring about language and platforms at all.

**Differences:**

* REST API has no official standard at all because it is an architectural style. SOAP API, on the other hand, has an official standard because it is a protocol.
* REST APIs uses multiple standards like HTTP, JSON, URL, and XML while SOAP APIs is largely based on HTTP and XML.
* As REST API deploys multiple standards, so it takes fewer resources and bandwidth as compared to SOAP that uses XML for the creation of Payload and results in the large sized file.
* The ways both APIs exposes the business logics are also different. REST API takes advantage of URL exposure like @path("/WeatherService") while SOAP API use of services interfaces like @WebService.
* SOAP API defines too many standards, and its implementer implements the things in a standard way only. In the case of miscommunication from service, the result will be the error. REST API, on the other hand, don't make emphasis on too many standards and results in corrupt API in the end.
* REST API uses Web Application Description Language, and SOAP API used Web Services Description language for describing the functionalities being offered by web services.
* REST APIs are more convenient with JavaScript and can be implemented easily as well. SOAP APIs are also convenient with JavaScript but don't support for greater implementation.

# What is Destructuring in Javascript?

The destructuring assignment syntax is a JavaScript expression that makes it possible to unpack values from arrays, or properties from objects, into distinct variables.

# Smart component vs Dumb component

Smart components are app level components that perform functions and manage data while dumb components focus solely on the UI.

# Difference between JDK, JRE, and JVM

### JVM

JVM (Java Virtual Machine) is an abstract machine. It is called a virtual machine because it doesn't physically exist. It is a specification that provides a runtime environment in which Java bytecode can be executed. It can also run those programs which are written in other languages and compiled to Java bytecode.

JVMs are available for many hardware and software platforms. JVM, JRE, and JDK are platform dependent because the configuration of each [OS](https://www.javatpoint.com/os-tutorial) is different from each other. However, Java is platform independent.

### JRE

JRE is an acronym for Java Runtime Environment. It is also written as Java RTE. The Java Runtime Environment is a set of software tools which are used for developing Java applications. It is used to provide the runtime environment. It is the implementation of JVM. It physically exists. It contains a set of libraries + other files that JVM uses at runtime.

### JDK

JDK is an acronym for Java Development Kit. The Java Development Kit (JDK) is a software development environment which is used to develop Java applications and [applets](https://www.javatpoint.com/java-applet). It physically exists. It contains JRE + development tools.

# Spring Boot Annotations

Spring Boot Annotations is a form of metadata that provides data about a program. In other words, annotations are used to provide **supplemental** information about a program. It is not a part of the application that we develop. It does not have a direct effect on the operation of the code they annotate. It does not change the action of the compiled program.

# Difference between abstract class and interface In Java

|  |  |
| --- | --- |
| **Abstract class** | **Interface** |
| 1) Abstract class can **have abstract and non-abstract** methods. | Interface can have **only abstract** methods. Since Java 8, it can have **default and static methods** also. |
| 2) Abstract class **doesn't support multiple inheritance**. | Interface **supports multiple inheritance**. |
| 3) Abstract class **can have final, non-final, static and non-static variables**. | Interface has **only static and final variables**. |
| 4) Abstract class **can provide the implementation of interface**. | Interface **can't provide the implementation of abstract class**. |
| 5) The **abstract keyword** is used to declare abstract class. | The **interface keyword** is used to declare interface. |
| 6) An **abstract class** can extend another Java class and implement multiple Java interfaces. | An **interface** can extend another Java interface only. |
| 7) An **abstract class** can be extended using keyword "extends". | An **interface** can be implemented using keyword "implements". |
| 8) A Java **abstract class** can have class members like private, protected, etc. | Members of a Java interface are public by default. |
| 9)**Example:** public abstract class Shape{ public abstract void draw(); } | **Example:** public interface Drawable{ void draw(); } |

# CSS BEM

BEM stands for **Block, Element,**and**Modifier**. It’s a CSS naming convention for writing cleaner and more readable CSS classes.

BEM also aims to write **independent CSS blocks** in order to reuse them later in your project.

**how BEM class namings are:**

// Blocks are named as standard CSS classes  
**.block {}**// Elements declared with 2 underscores, after block  
**.block\_\_element {}**// Modifiers declared with 2 dashes, after block or after element  
**.block--modifier {}//** element and modifier together  
**.block\_\_element--modifier {}**

**What is a Block?**

Blocks are independent, reusable and usually bigger components of a webpage. They can have modifiers and contain elements.

**Elements**

Elements are children of blocks. An element can only have 1 parent Block, and can’t be used independently outside of that block.

**Modifiers**

Modifiers represent different states or styles of classes. They can be used both for blocks or elements.

# Git rebase vs merge difference

Merging is a safe option that preserves the entire history of your repository, while rebasing creates a linear history by moving your feature branch onto the tip of main

# SASS CSS

Sass stands for Syntactically Awesome Stylesheet

Sass is an extension to CSS

Sass is a CSS pre-processor

Sass is completely compatible with all versions of CSS

Sass reduces repetition of CSS and therefore saves time

**Why Use Sass?**

Stylesheets are getting larger, more complex, and harder to maintain. This is where a CSS pre-processor can help.

Sass lets you use features that do not exist in CSS, like variables, nested rules, mixins, imports, inheritance, built-in functions, and other stuff

# Flex box

The Flexible Box Layout Module, makes it easier to design flexible responsive layout structure without using float or positioning

flex-container, flex-direction, flex-wrap, flex-shrink, flex-grow, align-item, justify-content, gap

# Dependency Injection

Dependencies are services or objects that a class needs to perform its function. Dependency injection, or DI, is a design pattern in which a class requests dependencies from external sources rather than creating them.

# Difference between var, let and const keywords in Javascript

**var** keyword in JavaScript: The var is the oldest keyword to declare a variable in JavaScript.

Scope: Global scoped or function scoped.

The scope of the var keyword is the global or function scope. It means variables defined outside the function can be accessed globally, and variables defined inside a particular function can be accessed within the function.

**let** keyword in JavaScript: The let keyword is an improved version of the var keyword.

Scope: block scoped:

The scope of a let variable is only block scoped. It can’t be accessible outside the particular block ({block}). Let’s see the below example.

**const** keyword in JavaScript: The const keyword has all the properties that are the same as the let keyword, except the user cannot update it.

Scope: block scoped:

When users declare a const variable, they need to initialize it, otherwise, it returns an error. The user cannot update the const variable once it is declared.

# Prototype in Javascript

JavaScript is a prototype based language, so, whenever we create a function using JavaScript, JavaScript engine adds a prototype property inside a function,

Prototype property is basically an object (also known as Prototype object), where we can attach methods and properties in a prototype object, which enables all the other objects to inherit these methods and properties.

In JavaScript, we achieve Inheritance with the help of prototype chaining.

Prototype chain

The prototype chain mechanism is simple: When you access a property p on object obj, the JavaScript engine will search this property inside obj object. If the engine fails to search, it continues searching in the prototype of obj object and so on until reaching Object.prototype. If after the search has finished, and nothing has been found the result will be undefined. For example:

var obj1 = {

a: 1,

b: 2

};

var obj2 = Object.create(obj1);

obj2.a = 2;

console.log(obj2.a); // 2

console.log(obj2.b); // 2

console.log(obj2.c); // undefined

In above snippet, the statement var obj2 = Object.create(obj1) will create obj2 object with prototype obj1 object.

In other words, obj1 becomes the prototype of obj2 instead of Object.prototype by default.

As you can see, b is not a property of obj2, you can still access it via the prototype chain.

For c property, however, you get undefined value because it can’t be found in obj1 and Object.prototype.

# How to Implement User Tracking System such as login, register

User behavior is an umbrella term for all the activities a user performs on your app or website.

User behavior analytics (UBA) involves collecting, grouping, and analyzing users’ behavior data to make decisions that enhance your overall UX.

Important user behavior metrics to track include trial signups and conversions, activation, and complete adoption.

You can understand user behavior by measuring the following: product and feature engagement analytics, session recordings and scroll maps, product and user experience analytics.

Tracking user behavior in SaaS helps you understand what brings value to users, identify friction points in the user journey, remove blockers and shorten time to value for your tool.

The best tools for tracking user behavior are Userpilot for tracking in-app behavior and how users progress through the journey, Amplitude for a wide range of analytics data and Hotjar for heatmaps and session recording.

# How to scale up server

Overloading the web server

With an increase in the number of clients simultaneously connecting to one web server, the server will eventually run out of CPU and RAM and cease performing. So what can be done to remedy this? Quite simply, server resources need to be increased in order to accommodate more clients. This can be done in a number of ways.

Vertical scaling: this is either the addition of resources to the existing server, or its replacement with another more powerful server. In this case, the architecture remains the same, but it's important to note that this isn’t a permanent fix. Why? Because even these resources will eventually run out and the vertic

al scaling would need to go on forever. This is why you need a better long-term fix. Enter horizontal scaling.

Horizontal hybrid scaling: This involves the addition of more servers which serve the same purpose as the first one. With an application’s continued popularity, the current servers runout of resources, thus we need to add more servers to serve other incoming clients.

A combination of both horizontal and vertical scaling: The above-mentioned scaling approaches are not mutually exclusive and can certainly be used in combination. Any application is capable of vertically scaling up, horizontally scaling out, neither, or both. You may well have a scenario in which parts of your application only vertically scale up, while at the same time other parts might horizontally scale out.

# Unit Test

<https://jestjs.io/docs/tutorial-react>

<https://www.smashingmagazine.com/2020/06/practical-guide-testing-react-applications-jest/>

<https://www.confluent.io/learn/distributed-systems/>

# Data Structure

https://towardsdatascience.com/8-common-data-structures-every-programmer-must-know-171acf6a1a42

UNIT TEST #

In this type of test, individual units or components of the software are tested. A unit might be an individual function, method, procedure, module, or object. A unit test isolates a section of code and verifies its correctness, in order to validate that each unit of the software’s code performs as expected.

In unit testing, individual procedures or functions are tested to guarantee that they are operating properly, and all components are tested individually. For instance, testing a function or whether a statement or loop in a program is functioning properly would fall under the scope of unit testing.

COMPONENT TEST #

Component testing verifies the functionality of an individual part of an application. Tests are performed on each component in isolation from other components. Generally, React applications are made up of several components, so component testing deals with testing these components individually.

For example, consider a website that has different web pages with many components. Every component will have its own subcomponents. Testing each module without considering integration with other components is referred to as component testing.

Testing like this in React requires more sophisticated tools. So, we would need Jest and sometimes more sophisticated tools, like Enzyme, which we will discuss briefly later.

SNAPSHOT TEST #

A snapshot test makes sure that the user interface (UI) of a web application does not change unexpectedly. It captures the code of a component at a moment in time, so that we can compare the component in one state with any other possible state it might take.

We will learn about snapshot testing in a later section.

Advantages And Disadvantages Of Testing #

Testing is great and should be done, but it has advantages and disadvantages.

ADVANTAGES #

It prevents unexpected regression.

It allows the developer to focus on the current task, rather than worrying about the past.

It allows modular construction of an application that would otherwise be too complex to build.

It reduces the need for manual verification.

DISADVANTAGES #

You need to write more code, as well as debug and maintain.

Non-critical test failures might cause the app to be rejected in terms of continuous integration.

Introduction To Jest #

Jest is a delightful JavaScript testing framework with a focus on simplicity. It can be installed with npm or Yarn. Jest fits into a broader category of utilities known as test runners. It works great for React applications, but it also works great outside of React applications.

Enzyme is a library that is used to test React applications. It’s designed to test components, and it makes it possible to write assertions that simulate actions that confirm whether the UI is working correctly.

Jest and Enzyme complement each other so well, so in this article we will be using both.

Process Of Running A Test With Jest #

In this section, we will be installing Jest and writing tests. If you are new to React, then I recommend using Create React App, because it is ready for use and ships with Jest.

# Body-parser

Body-parser is the Node.js body parsing middleware. It is responsible for parsing the incoming request bodies in a middleware before you handle it.

# Shopify liquid template language

https://shopify.github.io/liquid/filters/sort/

# Python Observer Method

The observer method is a Behavioral design Pattern which allows you to define or create a subscription mechanism to send the notification to the multiple objects about any new event that happens to the object that they are observing.

The subject is basically observed by multiple objects.

The subject needs to be monitored and whenever there is a change in the subject, the observers are being notified about the change.

This pattern defines one to Many dependencies between objects so that one object changes state, all of its dependents are notified and updated automatically.

# What is Decorator in Angular

Decorators are a design pattern that is used to separate modification or decoration of a class without modifying the original source code. In AngularJS, decorators are functions that allow a service, directive or filter to be modified prior to its usage.

# Angular Hook Lifecycle

| HOOK METHOD | PURPOSE | TIMING |
| --- | --- | --- |
| ngOnChanges() | Respond when Angular sets or resets data-bound input properties. The method receives a [SimpleChanges](https://angular.io/api/core/SimpleChanges) object of current and previous property values. | Called before ngOnInit() (if the component has bound inputs) and whenever one or more data-bound input properties change.  **NOTE**: If your component has no inputs or you use it without providing any inputs, the framework will not call ngOnChanges(). |
| ngOnInit() | Initialize the directive or component after Angular first displays the data-bound properties and sets the directive or component's input properties. | Called once, after the first ngOnChanges(). ngOnInit() is still called even when ngOnChanges() is not (which is the case when there are no template-bound inputs). |
| ngDoCheck() | Detect and act upon changes that Angular can't or won't detect on its own. | Called immediately after ngOnChanges() on every change detection run, and immediately after ngOnInit() on the first run. |
| ngAfterContentInit() | Respond after Angular projects external content into the component's view, or into the view that a directive is in. | Called once after the first ngDoCheck(). |
| ngAfterContentChecked() | Respond after Angular checks the content projected into the directive or component. | Called after ngAfterContentInit() and every subsequent ngDoCheck(). |
| ngAfterViewInit() | Respond after Angular initializes the component's views and child views, or the view that contains the directive. | Called once after the first ngAfterContentChecked(). |
| ngAfterViewChecked() | Respond after Angular checks the component's views and child views, or the view that contains the directive. | Called after the ngAfterViewInit() and every subsequent ngAfterContentChecked(). |
| ngOnDestroy() | Cleanup just before Angular destroys the directive or component. Unsubscribe Observables and detach event handlers to avoid memory leaks. See details | Called immediately before Angular destroys the directive or component |

# What Is A Pull Request In GitHub?

A pull request is a notification that a developer has made code changes locally that need to be reviewed, and if good, merged with the master branch. With some git branching strategies, the merge is with a feature, develop, or release branch. And, in Continuous Integration and Continuous Delivery/Deployment, pull requests are automated, though code reviews are still conducted.

# What’s The Difference Between A Pull Request And A Push?

The difference is actually pretty simple:

A push equates to assigning a task to a developer who must then start by making a clone of the “main branch” where changes can be freely done.

A pull takes the recent changes or commits in a developer’s local or cloned branch and, if they pass review and testing, merges them with your main branch.

# Same Criteria For Writing And Reviewing Code

There’s a lot to be considered when writing and reviewing code, including but not limited to:

Does the code comply with your team’s coding standards?

Is the code modular, logically structured and easy to read?

Are variables and functions named suitably and meaningfully?

Are functions, methods, behaviors, etc. sufficiently documented?

Does the code work?

How complex is the code and can it be simplified or is it “good enough”?

Is there any duplicate or unnecessary code?

How much of the code is covered with meaningful tests?

Are there any “untested hot spots” that should be scrutinized in greater detail?Does the code cover all relevant failure states?

Does the code meet performance requirements or have performance issues?

# Does Every Pull Request Involve A Code Review?

Yes, but… remember there are several types of code review. Code reviews for pull requests can be (formal) technical reviews or (informal) inspections. Additionally, in Continuous Integration, Delivery, and Deployment, pull requests are automated though code reviews are still done but usually at a later stage.

In formal code reviews, the author and reviewer sit down together to discuss the changes and make revisions, as needed. Formal code reviews are best for when the PR involves significant amounts of code, code with many changes, or very complex code. A general rule of thumb is that a formal code review can cover 400 lines of code in an hour and catch 70-90% of defects.

Informal reviews can be handled via emails or comments, and are best for handling many small changes in code.

# How GitHub Pull Requests Fit The Development Process

* Create a Branch – This is the push. The actual branch you’re pushing depends on your team’s branching strategy. This should always be the master branch in GitHub Flow and Trunk Based Development – It could be different (Feature, Develop, Release) if you use Git Flow, One Flow or GitLab Flow.
* Add Commits – Commits are the files and changes to files that you make. Each commit should have a clear description of the reason for the commit. Each commit is a distinct change applied to your local branch first.
* Open a Pull Request – This is a formal request for your commit to be reviewed by another developer.
* Code Review – A developer agrees to perform a code review on your commit, which may be done formally or informally. If approved, your commit takes a step forward (to #5 or #6 depending on your team’s process). If not approved, you return to #2 with comments advising you of issues that need to be fixed.
* Deploy – If your commit passes the code review, most teams will deploy it for further, more complex testing in a simulated production environment. If the changes don’t pass tests, you return to #2 with feedback from the test.
* Merge – Finally, after your code is reviewed, approved, and passes all tests, your commit is merged with the master branch – which, in GitHub Flow and TBD should always be deployment-ready.

# What is Prop Drilling?

Prop drilling is a situation where data is passed from one component through multiple interdependent components until you get to the component where the data is needed.

# What is Reusable Component?

In React, a reusable component is a piece of UI that can be used in various parts of an application to build more than one UI instance.

For instance, we can have a Button component that displays different texts on different pages.

# object.assign() vs spread operator

The difference is that Object.assign changes the object in-place, while the spread operator (...) creates a brand new object, and this will break object reference equality.