Job Interview

▼ Greetings

▼ Before

▼ Common

Hello, ***. Good morning/afternoon/evening, ***. Hi, it’s nice to meet you. Good day, how are you?

Hi, thank you for taking the time to speak with me today.

1. Good morning/afternoon/evening, ***. Thank you for taking the time to speak with me today.

2. Hi, ***. It’s great to finally to talk to you. Thank you for considering me for the position.

3. Hello, ***. I hope you are doing well. I am excited to discuss the opportunity further with you.

Answer above case:

1. I’m doing well, thank you. How about yourself?

2. I’m good, thank you for asking. How’s your day going so far

3. I’m doing great, thanks. How are you today?

4. I’m doing well, thanks for asking. I’ve been looking forward to this interview and am excited to learn more about the position.

5. I’m doing fine, thanks. It’s a pleasure to meet you in person.

▼ After weekend

1. Good morning/afternoon/evening! I hope you had a great weekend.

2. Hi there! How was your weekend?

3. Hello! I hope you had a restful weekend.

4. Greetings! I trust you had a wonderful weekend.

5. Hey! How did you spend your weekend?

Answer above case:

1. It was great, just spent some time with family and friend. How are you?

2. I had a relaxing weekend. I went for a hike on Saturday morning and then caught up on some reading in the afternoon. On Sunday, I had a nice brunch with friends and went to see a movie in the evening.

3. It was pretty busy, but enjoyable. I went to a friend's birthday party on Friday night and then spent most of Saturday running errands. On Sunday, I took some time to relax and catch up on household tasks. How about yours?

4. It was awesome! I went to a concert on Saturday night and saw my favorite band perform. I also got to try to a new restaurant and went shopping for some new clothes. How was your weekend?

▼ Before weekend

1. Happy Friday! The weekend's almost here.

2. Greetings! Wishing you a relaxing and enjoyable weekend.

3. Hello, hope you're doing well. Any fun plans for the weekend?

Answer above case:

1. Yes, it's great to know the weekend is almost here! How about you, what are your week?

2. Thank you! I hope you have a great weekend as well.

3. I'm planning on attending a webinar on Saturday. There will be some great speakers, so I'm excited to learn some new things. On Sunday, I'm planning on taking a break from coding and exploring a new hiking trail in the area. What about you?

▼ After

▼ Common

1. Thank you for your time. Have a great day!

2. It was a pleasure speaking with you. Take care!

3. Thank you for the opportunity to speak with you. Have a good one!

4. I appreciate your time and consideration. Have a wonderful day!

5. Thank you for your time. I hope we can speak again soon. Bye for now!
6. Bye.
7. Take care.
8. Talk to you soon.
9. Have a great day/evening/weekend.
10. It was great speaking with you.

▼ Before weekend

1. Have a great weekend!
2. Enjoy your weekend!
3. Hope you have a relaxing weekend!
4. Take care and have a wonderful weekend!
5. Wishing you a happy and enjoyable weekend!
6. Enjoy your time off and recharge for next week!
7. Have a fun-filled weekend with your loved ones!

▼ Interview

▼ IIntro

▼ Software Engineer intro

Hello, my names is ***** and I’m a senior software engineer with 10 years of experience in proper coding, requirement analysis, designing, building, deploying and managing mission-critical web apps. I’m excited to be interviewing for this position and learning more about how I can contribute to the team.

Thanks for the opportunity to speak with you today. As a senior software engineer with a strong background in developing web apps, I’m confident in my ability to tackle complex projects and deliver high-quality results. I am looking forward to discussing how my skills and experience can benefit the company.

Sample Intro

Good morning, my name is Adam, a seasoned software engineer with over ten years of experience delivering innovative solutions in web development. Thank you for the opportunity to discuss how my expertise can contribute to the success of your organization.

My skills include extensive knowledge and experience in JS, TS, React, Vue, Jest, NodeJS, Express, GraphQL, MySQL, and AWS. A passion for creating quality products that meet all customer's needs and staying up-to-date with the latest technologies to propose innovative solutions for developing projects.

Throughout my career, I have worked on a diverse range of web development projects, including large-scale e-commerce platforms and social media platforms. At Arctouch, I gained extensive experience building and maintaining web applications using React, implementing responsive design, optimizing performance, and developing new features such as customer reviews, shopping cart, product search, and filtering options.

As a senior software engineer, I am experienced in debugging and troubleshooting issues with web applications, utilizing performance optimization techniques such as code splitting and lazy loading to ensure high performance and seamless user experience. Additionally, I am responsible for writing and executing deployment of code for our products, communicating progress, and releasing high-quality code.

In addition to technical skills, I am a strong communicator and competent problem-solver, working closely with UI and product teams to ensure seamless collaboration, participation in testing, and troubleshooting. Passionate about working in an environment where I get to help people and collaborate with team members.

My goal-oriented approach allows me to thrive on challenges, and I see that your company emphasizes excellence and relationships, which are vital to your success. Excited about the opportunity to contribute to your organization's growth, I appreciate your consideration for the senior software engineer role. I look forward to discussing my qualifications and experience in greater detail.

I appreciate the chance to share my experience as a candidate for this position.

I appreciate the chance to share my experience as a senior software engineer with you today. Please let me know if there is anything else you would like me to add.

Thank you for allowing me to introduce myself as a candidate for this position.

I believe that I have presented myself and demonstrated my passion for web development. I am looking forward to discussing my experience and learning more about this role.

Thank you for allowing me to share my background and qualifications. I have done my best to present myself and provide you with a clear understanding of my expertise.

▼ Senior Front End Engineer (React) IIntro React

Hello, my name is *****, a senior frontend engineer specializing in JS, TS, React, Angular and Vue. I've developed and maintained robust web applications for 10 years.

Hello, my name is *****, a senior frontend engineer with a passion for creating intuitive and visually appealing user interfaces. I've worked with JS, TS, Jest, Cypress, React , Vue, and Angular for 10 years and have a proven track record of developing scalable and maintainable code.

Sample Intro

Hello, my name is Adam, it's great to have the opportunity to speak with you today regarding this role (senior frontend engineer position). With 10 years of experience, I take pride in keeping up with the latest web development trends and techniques to deliver exceptional results.

Throughout my career, I have gained deep knowledge and extensive experience in developing e-commerce projects and social media platforms. My passion lies in creating high-quality products that meet all customer needs, and I thrive in collaborative environments where I can assist and work alongside fellow team members. Furthermore, I am dedicated to researching new technologies and proposing innovative solutions to develop projects.

My core strengths include strong communication skills and a proficient problem-solver. I possess extensive knowledge of JS, TS, React, Vue, Jest, NodeJS, Express, GraphQL, MySQL, and AWS. Additionally, I am experienced in Git for version control and working in a team environment following Agile development methodologies.

As a software engineer at Arctouch, I am responsible for building and maintaining large-scale web applications such as e-commerce platforms. I excel in optimizing performance and responsive design while creating and implementing new features such as customer reviews, shopping carts, product search, and filtering options. I collaborate closely with product, design, and engineering teams, participating in testing, troubleshooting, code reviews, and shipping code on a daily basis.

In terms of my experience with React, I have worked with various libraries and frameworks such as Material-UI, AntD, Next.js, and Gatsby. I have a strong understanding of functional and class-based components, as well as experience in building reusable components and using hooks such as useState, useEffect, useCallback, and useMemo. I have also worked with Higher-Order Components (HOCs) and Render Props. I am adept at using React context for state management and have experience with Redux for managing global state.

Moreover, I have experience integrating React with other technologies such as API and databases, using Axios and Fetch for API calls, and working with GraphQL. I am confident in my ability to create and maintain high-quality web applications using React and look forward to continuing to work with this technology in the future.

Thank you for considering me for the senior frontend engineer position.

I appreciate the chance to share my experience as a candidate for this position.

I appreciate the chance to share my experience as a senior front end engineer with you today. Please let me know if there is anything else you would like me to add.

Thank you for allowing me to introduce myself as a candidate for this position.

I believe that I have presented myself and demonstrated my passion for web development. I am looking forward to discussing my experience and learning more about this role.

Thank you for allowing me to share my background and qualifications. I have done my best to present myself and provide you with a clear understanding of my expertise.

▼ Introduction in an onboarding session

Open to work.

I have been working for 8 years as a software engineer and focusing on front end development and engineering such as proper coding, planning, designing, building, implementing for web applications.
I developed a lot of e-commerce web sites, companies sites and job sites.
My core skills include being strong communicator and competent problem solver. I would describe myself as having extensive knowledge of JavaScript, TypeScript, React, NodeJS, AWS, GraphQL, MySQL, MongoDB, etc.

At my company, my responsibilities include writing and executing the deployment of code for our product, writing technical documents, managing workload, communicating progress, and releasing high-quality code.
I am also responsible for working closely with design and product teams, and participating in testing, and troubleshooting.

Responsibilities

- Assist in translating business requirements into software designs, implementation schedules, and deployment strategies.
- Develop highly functional, responsive websites based on mockups, wireframes, design artifacts, product and technical requirements
- Define new features that ensure a high performing and scalable user experience for both mobile and desktop users.
- Participate in sprint planning by understanding the effort, risk, and priority of remaining work committed for each release.
- Collaborate with peers on technical design, work estimation, and feature implementation.
- Perform peer code reviews and collaborate with the team to improve overall code quality.

- Build CSS that enables as many devices and resolutions as possible while maintaining minimum load times.
- Participate in the refinement and ideation of user interface A/B tests.

At new company

Hi, my names is James Chang and I’m excited to be starting here at *** (company name) as a Software Engineer on the *** (team name). Previously I have worked at *** (last company) as a Senior Software Engineer on their engineering team and before that as a Software Engineer at Themesoft (in my case). I hope to bring my experience from those places to my team here at *** (present company).

Outside of work I like to watch sci-fi and comedy shows, play video games, play chess (poorly), and go hiking. My last trip was to Yosemite(yosemity) Valley which was beautiful and fantastic for hiking. Right now, I live in Cary, NC.

I only have one pet for now, she’s a lovely little Tabby cat named Ada.

I’m very happy to be on board and can’t wait to see where things go from here!

Summary of Role in With pulp

We're bridging the divide between art, branding, design and technology to make digital products and services that are both memorable and usable.
My mission is to build and oversee the technical design and development of our product while collaborating with leadership to provide critical insight. This is both a strategic and technical role. I have developed and guided the creation of technology while acting as engineering thought leader within the organization.
My responsibilities include writing and executing the deployment of code for our product, writing technical documents, managing workload, communicating progress, and releasing high-quality code.
As the frontend Engineer, I understood our business objectives, integrated them into our development priorities, articulate technical considerations to non-technical peers, and grow our engineering team. And I reported to the Product Lead and then evolve to report directly to the CTO or CEO.

▼ Common

▼

▼ Do you have any questions for me?

:: **All Case**

- Can you share more about the day-to-day responsibilities of this role? How would you describe the pace of a typical day? (**about the role)**
- Could you tell me about hiring process for this role?
- Is there anything about my background or resume that makes you question whether I am a good fit for this role?

:: **CTO**

- What do you see as the most challenging aspect of this job?
- Could you tell me about your team structure and culture?
- What programming languages, frameworks, and libraries do you use?
- What software development process do you use?
- How many regular meetings do software engineers attend?
- If I were hired for this role, what would you want me to achieve in my first two months?
- What kind of databases do you use and where is the data stored?

:: **HR & CEO**

- What's the company's biggest priority right now?

▼

▼ What are your strengths?

I have a solid work ethic. When I'm working on a project, I don't just want to meet deadlines. Rather, I prefer to complete the project well ahead of schedule. Last year, I even earned a bonus for completing my three most recent reports one week ahead of time.
Flexibility and Adaptability is my key strength.
I am believe I am much more flexible than most people, and in different situations , on different teams, I can adapt as required.
I am a dedicated software engineer always looking to learn and expand skill set, utilizing best development practices, implementing new technologies as well as writing well structured and optimized code. Adapting and the ability to learn are two of my greatest assets.
I have a unique ability of understanding and translating business requirements into development roadmaps.

Thanks to my proactive attitude, effective communication, and flexibility, I can adapt to any team conditions in a short amount of time.

---

leadership skills

ex: decisiveness(problem-solving, initiative, research, project evaluation, expectation setting), integrity(diplomacy, ethics, reliability, professionalism, responsibility, confidentiality, honesty), creativity(critical thinking, curiosity, diversity, innovation, collaboration, open-mindedness), flexibility(negotiation, adaptability, feedback, work-life balance), positive attitude(conflict management, social skills, rapport, empathy, positive reinforcement, respect), communication(active listening, verbal communication, written communication, nonverbal communication, public speaking, presentation skills), relationship-building(collaboration, management, interpersonal, social teamwork), problem-solving(critical thinking, analytical skills, research, decisiveness, team-building), dependability(realistic goal-setting, timeliness, initiative, detail-oriented, loyalty), ability to teach and mentor(motivation, clarity, able to recognize and reward, understanding employee differences, feedback, help, helpfulness)

---

collaboration skills

I'm very collaborative and have always preferred to work in groups. In the project teams I've directed, members work with a variety of people and are motivated by diverse creative tasks. Since I began managing the e-commerce project in our team, I've increased productivity by 15 percent and retention by 25 percent over two years.

I think I have two big strengths. First, I believe that I'm very good at managing my time. This allows me to prioritize my assignments appropriately and meet all of my project deadlines.

Second, I'm skilled at managing project and web development.

Teambuilder

like to think one of my greatest strengths is that I am a good teambuilder. In my last job, we were entering the final stages of a project one night, but feeling a bit done with the whole thing and just discouraged. Though I was also feeling similarly, I tried not to let it show. Instead, I put a smile on one evening as we were heading out and convinced the entire team to head out together for dinner. We ended up having a great time that night, a true bonding session. The next day and *ever since, we all just get along, have more energy, and <u>generally work better together</u>.*

I have serveral key strengths that I believe will add value to your company.
My discipline is a strength because it meanse I will always organize my work diligently and I will take ownership of important tasks and projects.
I am naturally a positive and enthusiastic person.
My positive nature and resilience also means that I will always seek ways to quickly overcome difficult and challenging situations. For example, in my last role I recommended to my manager that we should implement a new customer review facility on our website as this would help build trust between the business and potential customers.
Finally, I am a loyal and dedicated person. If you hire me, I will represent your business in a positive way, I will stay working here for the long term, and you will always be able to trust me with important duties and responsibilities, safe in the knowledge they will be done to exacting standards.

▼ What are your weaknesses?

---

self-critical

“I can be too critical of myself. A pattern I’ve noticed throughout my career is that I often feel I could have done more, even if objectively, I’ve done well. Earlier in my career, this led to burnout and negative self-talk.

One solution I’ve implemented over the last three years is to actively pause and celebrate my achievements. Not only has this helped my own self-esteem, but it has also helped me genuinely appreciate and recognize my team and other support systems.”

---

lacking confidence

“I’m naturally shy. From high school and into my early professional interactions, it prevented me from speaking up. After being a part of a workgroup that didn’t meet our strategic goals two quarters in a row, I knew I owed it to my team and myself to confidently share my ideas. I joined an improv acting class—it’s fun and has really helped me overcome my shyness.

I learned practical skills around leading discussions and sharing diverse perspectives. Now, in group settings, I always start conversations with the quieter *folks. I know exactly how they feel, and people can be amazing once they start talking."*

difficulty asking questions

"I default to believing that I can solve any problem on my own. This works well in some situations, but in many cases, I need the help of others to overcome factors beyond my control. In one instance last year, I was spearheading a client event that had a lot of moving parts. It wasn't until after the event that I realized how narrowly I had pulled it off.

I was trying to manage everything from the strategic plan down to the tiniest details, like table settings. I did a lot of self-reflection afterward. Since then, I've been training myself to take a step back before diving into problem-solving mode and identify people or groups that can be resources to me."

lacking experience

"I'm not familiar with the latest version of [insert name of non-critical software]. Instead, I've focused on [insert name of preferred software] because user-centric design has become a strong passion of mine. In my last few jobs, that's where I've spent time learning and growing."

procrastination

"I've always been a procrastinator. I used to think it wasn't such a bad habit because I was only creating stress for myself. But when I was working for XYZ Company several years ago, I was on a group project where I could see how my putting things off to the last minute created stress for everyone else.

It was a wake-up call. I started creating daily schedules that hold me accountable to my team, and I broke the habit. It was hard at first, but using the Agile process was a real breakthrough in my workflow and mindset."

perfectionism

"I tend to be a perfectionist and can linger on the details of a project which can threaten deadlines. Early on in my career, when I worked for ABC Inc., that very thing happened. I was laboring over the details and in turn, caused my manager to be stressed when I almost missed the deadline on my deliverables.

I learned the hard way back then, but I did learn. Today I'm always aware of how what I'm doing affects my team and management. I've learned how to find the *balance between perfect and very good and being timely.”*

---

time management skills

I guess I’d have to say one of my weaknesses relates to having less-than-optimal <u>time management skills</u>. I have this tendency to always try to find one more thing to fix before pushing something to production, which delayed the release of one of my first projects. But, that instance served as a good wake-up call, and I try to follow the mantra now that “done is better than perfect.” I have made it a point to note everything I would like to change and fix them instantly, instead of leaving them to be fixed at the end. And, if it doesn’t make it, we can schedule it for the next release (unless it’s too urgent and important now).

Well, I would have to say that I might need to work on being a bit better listener in some situations. In the past, sometimes I have been very focused on what I think is right solution, and others might think I push too hard without getting input first. so I am trying to do that whenever I work in a team.

▼ What are your salary expectations

I am looking at opportunities in the 4k range, which I understand is about the industry average.

question: Alright, and besides salary, what's important to you on the **intangible** side of things?

answer: Oh, well what's important to me is a position work environment with opportunities to explore new ideas. I am also looking for good people and **dynamic, creative** teams. And flexible working hours would be a real bonus.

What are your salary expectations?

- I’m looking for a competitive offer that includes benefits and other kinds of compensation, but I ‘d like to know more about the specifics of what this job requires first.

After I’ve delayed answering the question, I will tell recruiter like this

- Based on my experience in this field and my research on the current market, I understand that mid 90k(60/h) to low 110k(75/h) is a competitive range.

“Thank you for asking. I feel that an annual salary between 80k and 90k annually is in line with the industry average and reflects my skills and experience level well. I am, however, flexible and open to hearing about the company’s compensation expectations for this position.”

“I am seeking a position that pays between $75,000 and $80,000 annually, but I am open to negotiating salary.”

▼ What are your future career goals?

i**nappropriate examples, so reference it**

“A few of my future goals include leading a finance team in some capacity. I’m excited about the prospect of working with teams like legal and procurement on developing streamlined processes—this is a natural fit with my business administration background. One reason this job stood out to me was that it calls for a candidate with organizational expertise. I’ve had inspirational managers that I really admired, and would love to manage my own team in a few years.”

“A few of my future goals include leading a finance team in some capacity. I’m excited about the prospect of working with teams like legal and procurement on developing streamlined processes—this is a natural fit with my business administration background. One reason this job stood out to me was that it calls for a candidate with organizational expertise. I’ve had inspirational managers that I really admired, and would love to manage my own team in a few years.”

“In five years, I’d love to be a true apparel industry expert with successful end-to-end project management experience under my belt as I look to grow into a more senior market analyst role. It’s exciting that your company has a strong focus on hands-on experience and continued learning opportunities.”

▼

▼ What are you passionate about?

“As a software developer, I’m passionate about creating truly beautiful, efficient digital products to make people’s experience with technology memorable. One of the things I loved about my last job was witnessing the results of my team’s code update and watching as our months of work yielded positive user feedback.

Having the opportunity to lead projects from ideation through launch was one of the reasons I was so excited to apply for this role."

▼

▼ Why are you applying for this position?

"I've been working for several years on gaining skills in your industry. I feel I have the knowledge, skills and qualifications you're looking for, along with a unique perspective coming from a different industry. I am passionate about working in the environmental protection space, and it is time for me to make a change. I feel your company is the perfect place for me to do that."

"I read an article in the business magazine about your company CEO and was attracted to the plans that this firm has developed about new technology. Since I consider myself a technician, I would be happy to take this organization to another level through innovation."

"The reputation of this company is a factor I considered when applying for the job. I would feel proud to work with this organization that has great leadership in this industry. More so, a close friend of my family has been working in this firm for many years and he always said good things about this company. I am aware you reward hard work, encourage development at workplaces and supporting learning while working - and these are values I share."

"I know that my excellent track record at work makes me a perfect fit for the job. Also, the job position motivates me because I support the idea of improving the productivity of any company through innovation. My vast skills in software development could help me deliver results from the start."

"With the software products you possess, I reserve respect for this firm. Therefore, I would love to work with the best team in the business. More so, I have friends who work for this company and helped me to understand the respect you have for employees. You have a working environment designed to meet the needs of many employees. I feel my style of working will fit well in this organization."

▼ Why do you want to work at our company?

Example 1: I’m ready to apply my expertise.

> *“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.”*

Example 2: Your company ranks at the top of the list.

> *“While your company has been around for several decades, you’ve never wavered from your mission to give people a comfortable and memorable travel experience.*
>
> *The company has always been forward-focused and used technology to help improve customer experience. Your app is ranked as one of the top travel apps, and you’ve won some of the most prestigious awards in the industry year after year.*
>
> *When I began looking for a new position, I purposefully sought out companies that are committed to integrity, philanthropy and innovation, and your company ranks at the top of the list.”*

Example 3: I would love to grow my career.

> *“I’ve been a loyal user of your product for many years. I’m consistently impressed by the cutting-edge technology and great user experience it provides.*
>
> *A few months back I had an issue and called customer support. The representative I spoke with was helpful, personable and sharp. I remember thinking at that time that your company must be a good one to work for.*
>
> *When I saw the opening for the Product Manager position I decided to research your company values and culture further, all of which supported my intuition that this is a place I’d love to work. I would love to grow my career at a company that is passionate about user experience and innovation.”*

---

startup

The company’s commitment to inclusion and diversity is really inspiring. While many startups and corporations are vowing to become more inclusive and diverse, this one actually follows through. Based on your about page, I clearly see a diverse representation of women and people of color on your engineering team, and I’d be very honored to be a part of that!

I want to work at this company because you have a solid reputation for creating excellent products, for delivering outstanding service, and for constantly innovating so that you stay at the cutting edge of what your customers and clients need.
And I spent some time researching your company before applying for the role and I was impressed with what I saw.
Also, as a frontend software engineer, during my research into your organization, it soon became apparent that you continually look for ways to grow and develop.
I am someone who is ambitious ,and I enjoy working with other professionals.
If you hire me, I feel confident you will be impressed with my work values, ethics, and high standards.

▼ Why are you interested in this position?

Senior software engineer

"While I enjoy writing code in my current role, the work environment doesn't align with my personal and professional goals. In this position, I can code and lead a team of junior software engineers. I'm eager to use my experience with various programming languages to build exciting projects for Software Programs Inc. Can you tell me more about the leadership style you expect from an ideal applicant?"

Technical expertise

"I am interested in this job because I think it would allow me to contribute both my technical expertise and my people skills. Working for Ellis Software as the Design Manager would enable me to use both my knowledge of computer programming and my experience in customer service. I pride myself on being able to handle both computers and clients successfully, so this job feels like a great fit for me. I think because I can contribute to both the production and the sale of Ellis Software's products, I would be a valuable asset to the team."

▼ Why do I want this job? why should you hire me?

“I noticed that the parts of my previous positions I enjoyed the most were those that aligned with what’s listed in your job description, like creative writing and building relationships with stakeholders. While I am grateful for my time at my current company, I feel that it’s time to move into a role more tailored to my talents where I can continue to grow as a PR professional.”

You should hire me because my experience is almost perfectly aligned with the requirements you asked for in your job listing. I have seven years’ progressive experience in building startups (or implementing IMS), advancing from my initial role as a front-end developer to my current position there as a senior Full-stack developer. I’m well-versed in providing [world-class customer service](https://blog.hubspot.com/service/world-class-service#:~:text=World-class customer service is, loyalty and facilitate customer acquisition.) to an upscale clientele, and I pride myself on my ability to quickly (build your project) resolve problems so that our guests enjoy their time with us (you can put your idea into reality in the shortest time).

I am an expert in Python and R. So, I have the exact technical skills the company is looking for. I also hold experience as a team leader, which is an essential requirement for this job. I am always willing to learn new skills which makes me an ideal choice for different projects that might come up in the future. My ability to deliver even under pressure is what makes me stand out.

Why do you want this job?

I want this job because it is the type of role that I genuinely love doing. I usually spend a lot of time at work, and I want that time to be put to good use.
And, I feel I already have the skills and qualities to match this job description which meanse I can come into this role and start helping the team to make an immediate and positive impact.
Also, I want this job because I believe it will give me lots of job satisfaction, and I will get to develop essential workplace skills.
And this role is exactly the sort of role I am currently targeting, and I am confident I will be able to make a major contribution.

Why should you hire me?

First, I am a very fast learner. This means I will come into the role, I will learn everything quickly, and I will then contribute to my team's objectives in the fastest time possible.
Second, I will always take responsibility for my own standards of work. This means you won't have to spend your valuable time closely monitoring or supervising me.
Finally, you should hire me because I have a proven track record of achievement, and I will always take positive steps to develop professionally in the position.

▼ Why do you want to leave (or have left) your job? Why are you looking for a new job?

---

1. I want to seek new challenges and opportunities.
2. I am seeking a role that’s more closely aligned with your long-term career goals.
3. I want to experience a new work environment after spending a long time at my current company already.
4. I want an opportunity to take on more leadership work.

---

common

I’ve learned a lot here over the past 3 years and it’s been great. I just feel that to grow my career further, I need to expose myself to new challenges, and your company caught my attention because your company emphasizes excellence and relationships. I think it is the most important for an organization to succeed in this world.

---

contract will be terminated this month

Now I have been working in this company as a contractor. I have already finished the last project in the previews month and the duration of the contract will be terminated this month.

---

lead role

I’ve enjoyed my role a lot, but I’ve been here for 3 years and I think it’s time to challenge myself further. One thing I’d love to do is manage projects, and I noticed that’s mentioned in the job description for this role. I’ve asked my current boss about this and it’s just not a responsibility that I can take on in my current role or my current team, and they don’t have room to promote me right now, so that’s why I’m willing to leave my current job.

---

family problem

I left my last position in order to spend more time with an ill family member. Circumstances have changed and I am ready for full-time employment again.

---

bigger than your current company

I was very fortunate to be hired by GE Digital. They taught me a lot about how to build projects, and it’s been stimulating to work as a contributor to their creative teams. However, I’m ready for the next step.
As you know, everybody has their own goals.
Now I am looking forward to a good opportunity to improve my skills and achieve my goals.

Of course, GE Digital provides a good opportunity but it is limited for me.
I think your company can give me a bigger and better platform where I can make an impact.

▼ Why did you become a software engineer?

I've been passionate about creating quality software and solving problems since I was a kid. My father was software engineer and my mother was web designer when I was in high school. They allowed me to play with computer anytime and I begun to have an interest in computer programming and science. I knew then that I wanted to become a software engineer as my career as well as my father. I studied computer science at Northeastern Illinois University and graduated 8 years ago. I’ve been working for 7 years as a senior front end engineer.

▼

▼ How many years do you plan to work with us?

long-term opportunity

I am looking for stability in my career and I applied for this position specifically because it offers a long-term opportunity. I have gone through your company profile, and I am aware of your employee-supportive culture. I also believe that I will get the maximum opportunity to improve my skills and apply my knowledge working with your organization. So, I am considering this job as a long-term association.

▼

▼ What can you offer us that someone else can not?

▼

▼ How do you handle stress?

Emotional intelligence

Emotional intelligence is my ability to be aware of and manage my emotions — as well as the ability to perceive and deal with the emotions of others, both on an individual level and in groups. Generally, when I have higher levels of emotional intelligence, I’ll be able to manage my emotions and deal with stress easier.

As you know, emotional intelligence doesn’t stop with managing stress and emotions better, though. It also a subset of skills that include empathy, self-awareness, self-regulation, and motivation.

Planning

"Planning is an important tool in handling stress for me. Drawing up detailed plans for projects and even my daily work helps me to get ahead of stressful situations.

When stress does inevitably arise, planning helps me to tackle the situation one step at a time to prioritize what needs to be done efficiently for myself and my colleagues.

Some of my best work in streamlining processes has come from a stressful situation. I've been able to design simpler, more efficient workflows with less room for error."

Motivator

"Stress can be a big motivator for me. A healthy amount of pressure helps me produce efficient, quality work by giving me a picture of what my colleagues need from me and when. I've experienced stressful situations that bring my team together, and have seen some of our best work come from pressure."

Communication

"For me, communication is key in stressful situations, if even over-communicating to ensure everyone is on the same page. For example, I was working on a project with another team and we found there was a lot of duplicate work being done.

---

playing

Outside work and research, I like playing basketball and watching the replay of NBA games. The basketball court is where I relieve my stress. I also talk to my girlfriend when I am stressed, who is always a good listener to my thoughts.

I gradually realize that, as we gain experience through withstanding different situations and challenges, we become stronger, in turns, we can convert part of the stress and pressure into motivation.

▼ How do you handle a challenge?

▼ How would you deal with a member of your team who disagreed with the work you carried out as a software engineer?

I am aware that when I do work as part of a team, it is important to listen to other people as they may have valid contributions to make.

▼

▼ Could you tell me about your Agile, Scrum experience?

I have good knowledge of Scrum and Agile concepts and vast experience in Agile methodologies especially scrum, and scrum events such as sprint planning, daily scrum, sprint, sprint review and sprint retrospective.

Scrum is a popular framework that enables teams to work together.

Based on Agile principles, Scrum enables the development, delivery, and sustenance of complex projects. It enables teams to hypothesize how they think something works, try it out, learn and reflect from their experiences, and make appropriate changes.

When I worked at GE-Digital, I worked with Scrum methodology with my team.

I participated in daily stand-up meetings, story grooming, review, demos, and other project-related meetings.

And I supported team members in their tasks.

▼

▼ Give an example of an occasion when you used logic to solve a problem.

▼ What programming languages are you proficient in?

▼ How do you explain technical information to non-technical audiences?

▼

▼ What is your career goal? long term goal, short term goal (manager, owner, software engineer, director, CTO)

---

reference

Now, you are a front end engineer with 6 years experience. There are three types in there; front end engineer, senior front end engineer and lead front end engineer.

So, your career goal is to become a lead front end engineer. First of all to become a lead front end engineer, you need to be senior front end engineer. Look at glassdoor.com (front end engineer career path) to implement this goal. In a nutshell, how to become, what skills and education you need to succeed as a senior front end engineer, lead front end engineer.

My career goal is to eventually become a tech lead. I'd love to one day lead my own team, while mentoring my team members on what I know and have learned over the years. To get there, I definitely need to work more on my leadership skills, but I do think I have the communication skills and project management abilities necessary for a future leadership position on an engineering team.

▼

▼ What is something outside of your work that you are passionate about?

One area that I'm passionate about is self-development and self-improvement in my personal life. I'm always looking to challenge myself and learn new things. That's one reason I enjoy working in customer service; I've learned great communication skills, listening skills, and problem-solving skills that help me in my day-to-day work but also in my personal life in terms of confidence, communication, etc.

I'm really passionate about hiking and being outdoors. That's usually how I spend my weekends and my time when I'm not in the office. I feel like I have more energy when I return to the office on Mondays if I've spent some time outdoors on the weekend.

▼ What does your typical day look like? What do you do in the rest of the day?

▼ Describe an important project you've worked on

Note: Use STAR method (Situation, Task, Action, Result)

Sample 1

"*I have worked on many important projects throughout my career. What's really crucial for me when starting one is to get very clear on the goals right at the start and then create a plan with milestones. I also like dealing with the most difficult parts of the projects early on—that way in case there are any significant issues, I'll*

still have a nice amount of time to complete before the deadline. I also typically break down large tasks into smaller chunks, so that it is easier to know where to start. Detailed planning is very important to ensure an important project goes smoothly. For example, last year I was in charge of . . . .
”

Sample 2

“*In order to get project “X” completed in my previous job, I found out who the key stakeholders were and got their input on the project’s different parts. Then, I outlined the major milestones that would be involved in completing the project and worked backward to break down the work that would need to be done at each stage.*

I created a list of all possible risks that might stop us from reaching those milestones, and I then added some extra time to the schedule in case anything unexpected came up. I also made sure that my role and responsibilities in the project were as clear as possible so I knew exactly what I had to do. The project was completed on time, but looking back, I realize there were some problems that could have been avoided. For example, I would have changed “Z” in order to avoid some of the minor scheduling problems we ran into. Having said that, it’s always easier to see what the learnings are after a project has been completed, and I now know what I’d do differently the next time.”

▼ Tell me about your experience as a software developer(engineering).

I’ve been working for 8 years as a senior front end developer and I have rich experience in front end development using React.

Since graduating with a bachelor’s degree in computer science from the University of Chicago-that was in 2014. Then I worked at Chicago Technologies for a year as an intern. I had hands-on experience there and I tried to gain professional skills there. After an internship at Chicago Technologies, I was hired by Illinois Software Company and I gained a lot of experience. I have recently worked for *** company remotely. (In addition, you should tell your greatest accomplishment)

—In 2020, our team developed HR software for *** company because its original HR application was outdated. We developed HR application where *** company could handle payroll, time tracking, attendance, training, and hiring in the same place. My primary role was to integrate all APIs and optimize for Performance. Also, I took part in QA(Quality Assurance) and testing.

▼ Describe an Important Project You Worked On

I have worked on many important projects throughout my career. What's really crucial for me when starting one is to get very clear on the goals right at the start and then create a plan with milestones. I also like dealing with the most difficult parts of the projects early on—that way in case there are any significant issues, I'll still have a nice amount of time to complete before the deadline. I also typically break down large tasks into smaller chunks, so that it is easier to know where to start. Detailed planning is very important to ensure an important project goes smoothly. For example, last year I was in charge of . . . .

In order to get project "X" completed in my previous job, I found out who the key stakeholders were and got their input on the project's different parts. Then, I outlined the major milestones that would be involved in completing the project, and worked backwards to break down the work that would need to be done at each stage.

I created a list of all possible risks that might stop us from reaching those milestones, and I then added some extra time to the schedule in case anything unexpected came up. I also made sure that my role and responsibilities in the project were as clear as possible so I knew exactly what I had to do. The project was completed on time, but looking back, I realize there were some problems that could have been avoided. For example, I would have changed "Z" in order to avoid some of the minor scheduling problems we ran into. Having said that, it's always easier to see what the learnings are after a project has been completed, and I now know what I'd do differently the next time.

▼ Tell us about a recent project you worked on.

▼ Describe an important project you worked on.

"I have worked on many important projects throughout my career. What's really crucial for me when starting one is to get very clear on the goals right at the start and then create a plan with milestones. I also like dealing with the most difficult parts of the projects early on—that way in case there are any significant issues, I'll still have a nice amount of time to complete before the deadline. I also typically break down large tasks into smaller chunks, so that it is easier to know where to start. Detailed planning is very important to ensure an important project goes smoothly. For example, last year I was in charge of . . . .

In order to get project "X" completed in my previous job, I found out who the key stakeholders were and got their input on the project's different parts. Then, I outlined the major milestones that would be involved in completing the project, and worked backwards to break down the work that would need to be done at each stage.

I created a list of all possible risks that might stop us from reaching those milestones, and I then added some extra time to the schedule in case anything unexpected came up. I also made sure that my role and responsibilities in the project were as clear as possible so I knew exactly what I had to do. The project was completed on time, but looking back, I realize there were some problems that could have been avoided. For example, I would have changed "Z" in order to avoid some of the minor scheduling problems we ran into. Having said that, it's always easier to see what the learnings are after a project has been completed, and I now know what I'd do differently the next time.

▼ What were your responsibilities in your previous position?

▼ Tell us about a conflict or failure that you faced at work. (mistake)

As you know, nobody is perfect.
Actually, I have faced at conflict and failures in building project in several times.

For example, I have implemented IMS of Tobacco Product Manufacturing Companies of Chengdu. It was a really big system. It took 2 years to complete successfully. At that time the site we had built was not widely used among workers. Because it was far away from requirement of reality. So we have got input from workers but not researchers or information on the internet. We have solved the issues step by step testing in a workstation level.

At that time we realized that what is more important thing is can-do attitude without disappointment.
We all make mistakes, this happens no matter in research or in life. However, when I make mistakes, big or small, I treat them very seriously. I have been trying

my very best to learn from my mistakes and take these opportunities to grow, in order not to make the same mistake twice.

"Make mistakes, don't fake perfection. But don't make the same mistake twice." This would be my advice, indeed

---

▼ Greatest accomplishment.

As a front end lead

In my last position, our technology development team lost a colleague due to relocation. He was the lead developer for the new version of web app. Unfortunately, no one else on the team had worked with the new version of apps. Since I had experience developing the kinds of web app, I volunteered to take on the lead role of development and deployment of the app. I worked with the other team members to create and troubleshoot the new app. I was able to finish the development 60 days ahead of schedule. It's currently available in the iTunes Store and already has over 350 positive reviews and has offered an additional revenue stream for the company.

As a front end lead

In my most recent job I was responsible for managing the orientation and training programs for our new hires. Unfortunately, the content was not engaging. While it was necessary information for our new hires to have, we found that only 35% 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, 93% of participants complete the training and provide positive feedback about their experience. My manager was so pleased with the improvements that she asked me to lead a training seminar in our New York office."

▼ Describe your process for completing a project from start to finish.

▼ Please tell me about React development you have experienced? Exactly what is your accomplishment in your development?

If I describe my process of building project, it is as following.

I usually start from deciding on component. as you know deciding on component is the first step to build a project keeping the OOP principle. And then I used to decide the state and where it lives. To decide a state and manage it is the important factor to make a program more flexible and extensible and reusable etc. So in order to manage the state globally, we use a redux in React and vuex in Vue.js. And then next, we have to recognize what changes when state changes.

If I summarize what I have mentioned above,

1. decide on component.
2. decide the state and where it lives.
3. what changes when state changes.

These are essential steps that I usually used to build any project.

As a result, I would describe my accomplishment in building project as clarity and robustness.

I am very proud to say that all projects I have done is built on OOP principles.

▼ What is the most difficult thing you have ever worked on?

The most difficult thing is to estimate time line to finish the project delivered to me. Sometimes I've worked with clients non-experienced in development. At that time they asked me to estimate time line to finish project successfully.

If I estimate it as I want based on my situation, it would affects the business strategy badly. As a front end leader, I should make milestone blended well with strategy and team members' ability. So I got inputs from collaborators and client and decided it. Due to urgent launching date we should sometimes work in weekday.

After all I made the intended outcomes as they want and they always enjoyed with my ability.

▼ Tell me about the project that you are most proud of. What did you do? How did it turn out?

Currently a postdoctoral fellow in HKUST, Dr. YIN Ran began his journey at the University as an MPhil student in the Department of Civil and Environmental Engineering in 2014, and obtained his MPhil and PhD degrees in 2016 and 2020 respectively. During his PhD studies, he has published 22 papers in leading journals in the field of environmental engineering, and is also the co-inventor of two granted patents. His remarkable achievements and significant advanced water treatment research has earned him the School of Engineering PhD Research Excellence Award 2020-21.

As an engineering PhD student myself, I was delighted to have the chance to speak with Dr. Yin. While he generously shared his insights, I saw the spark in his eyes that captured his fondness for research and his career.

▼ Tell us about a time when you had to give someone difficult feedback. How did you handle it?

I Listen carefully to his feedback, to identify the real problem. And respond quickly, but not in an emotional or reactive manner. If we are in my team, pull him aside, and ask questions and offer proactive solutions to correct the issue and demonstrate my genuine care.
I always take negative feedback as an opportunity to improve my products and services for the future.

▼ Tell me how you handled a difficult situation.

▼ What do you consider to be your biggest professional failure, and what did you learn from it?

I was managing a project where a new client wanted a large number of unique product descriptions written to improve the SEO ranking of their site. Because they were a new client and I wanted to impress them with the kind of results we could produce, I assured them we could have it back to them in two weeks. I thought this was doable with multiple writers working on the project, but in the end, it took an extra week, and they were not happy.

We apologized and reassured them that the mistake wouldn't happen again. I realized that it's far better to under-promise and over-deliver. The client isn't going to be upset when you are clear about what the timeline is from the beginning. Problems arise when you can't meet promised deadlines. I used this experience to be more cautious in managing client expectations. For the next client project I worked on, I made sure to include extra time for unforeseen circumstances and told them we would deliver in four weeks. We delivered in three, and they couldn't have been more thrilled.

▼ What scaling issues have you dealt with on past projects? How did you resolve them?

▼ What are your favorite software development tools?

▼ Which design patterns do you use most often?

▼ Describe a difficult bug you fixed in a large application.

▼ What motivates you to do your best work?

Working in a team and leading one, is my motivation to do a good job. United we stand, divided we fall has been my mantra for success right along. Working in a team has taught me so much more. It enlightened me and opened up my mind to a whole new world.

The confluence of ideas, thoughts, and opinions encouraged me to know and learn from my peers. There was a time when I believed that I could do everything on my own. With time I realized the magnitude of achievement possible with teamwork. It took me some time getting used to leading a team of varied individuals.

Gradually I realised that it was more about dealing with their personalities and less about their skills sets. It was a challenge which I took up and gained valuable insights related to team building and motivation. As a leader, it is imperative that you trust your team members and support them. It is also important to create synergy from their individual capabilities.

▼ How comfortable are you reviewing others' code?

▼ Tell me about a time you collaborated with others to solve a complex problem.

▼ What are the most important factors to consider when designing scalable applications?

▼ What strategies do you use to make a program faster?

▼

▼ What is your dream job?

▼ What would be the best answer if your interviewer asks you, “Explain Agile”?

Agile seeks to deliver the most value in the least amount of time. Agile involves team work in short iterative cycles, called sprints, instead of planning everything in advance. The sprint team is small - typically 3-8 people which is empowered to self-organize and participate in the decision making on what is to be worked on next. The sprint consists 1-2 weeks in length goal of completing all the work and delivering completed and testable features by the end. Communication is critical so there is often a "stand up" everyday where the team checks in. The intent is to keep the entire team aware of what everyone is working on - to increase collaboration and avoid rework or overlapping work.

Benefits: Short iterative cycles allow for faster feedback from stakeholders and customers. The team will have more direct connection to the the planning and outcomes and therefore can derive more meaning and purpose from their work. For projects that will encounter multiple changes, the primary benefit is the flexibility of agile to adapt quickly and shift focus, hence the origin of the name.

Drawbacks: Some projects are not easily broken into smaller cycles, which can make agile implementation more challenging. While agile seeks to complete each cycle to the point of being "done" the actual project may always have more stories remaining so that projects can seem to be never ending. Many traditional stakeholders and managers can be date driven, and agile is work driven or feature driven and so conflicts can arise.

▼ How do you manage changes in the workplace?

I recently experienced a similar situation at my previous job, where a client wanted to completely change the design of their website a month into the project. As the frontend developer, it meant that I would need to completely discard all of my work and start over.

But I'm able to stay positive because I keep reminding myself that change is inevitable in the business world. As an employee, I have a part in helping the company I work for succeed, and I take that responsibility seriously. I make sure to keep thinking of the benefits of the change and that, although the change may be difficult, it will result in positive updates for the organization.

At that time my client had agreed to a new budget and timeline, but I did my best to complete the task in origin timeline, I worked 60 hours per week, so I implemented and launched it in time successfully.

▼ How to collaborate with designers as a frontend developer.

The gap between designers and developers often slows workflows and makes it harder to bring products to market. Without strong collaboration between your design and development product teams, you will likely find that your company has to correct a lot of mistakes before you can finalize a product and release it to consumers.

We don’t have to tolerate the designer-developer gap.

Creating a design system that uses the same components for designers and developers is the most important thing you can do to bridge the gap between teams. Your design system should provide everything that your people need during the designing and coding phases, including the product’s approved:

- Typography
- Colors
- Icons
- Photographs
- GIFs
- Sounds
- Interactive elements

We can improve every step of production by using the same design system for your designers and developers so that they don't waste time on deciding which font or color to use, and how to code this button and that form. Traditionally, companies struggle with this approach because they take an image-based approach to building products and you need to put a lot of effort to maintain the design system. After the design team finishes its work, developers need to find ways to make components functional. A code-based approach eliminates the gap by giving designers and developers access to a real single source of truth that can be reused anytime.

See examples of design systems.

Improve communications between web designers and front-end developers

Your designer and development teams need easy ways to communicate with each other. Not too long ago, you had limited ways to improve communication between teams. You could have them share a workspace or add everyone to a social media group.

Today, you have ample tools to improve communication and collaboration, even when your team members work remotely. Some of your best options include applications like:

- Slack
- Trello
- Google Workspace

When you need a replacement for face-to-face conversations, you can use video conferencing apps like:

- Zoom
- GoToMeeting
- Microsoft Teams

Teammates get left out of critical brainstorm sessions far too often. The project manager might think to invite the development team's leader for insight into the challenges of adding certain features to a product.

Get more designers and developers involved in brainstorming sessions to build camaraderie, share ideas, and discover innovative concepts that you might have missed without everyone's participation.

▼ Tell me about your last company.

▼ What was your bosses’ strengths/weaknesses?

▼

▼ Would you work holidays/weekends?

▼ Are you willing to travel? *(Post Covid-19)

▼

▼ What are advantages of working from home remotely?

- A better work-life balance:
- No commute stress
- location independence: (One of the biggest advantages of working from home is having access to a much broader scope of job opportunities, because I am no longer limited by my geographic location.

▼

▼ My work experience

▼ React Performance Experience, React Performance Experience

project explanation

At GE Digital, I've been working on various projects and tasks for more than 3 years as a senior software/front end engineer.

Sure, I’d be happy to discuss a recent project I worked on. It was a project for a large e-commerce provider, and I was a senior software/front end engineer.

www.net-a-porter.com

project description

This site is a luxury fashion and lifestyle e-commerce website. It sells designer clothing, shoes, handbags, and accessories for women from a wide range of well-known luxury brands, including Gucci, Chanel, Saint Laurent, and many others. It also offers a range of beauty products and home decor items. The website is aimed at fashion-conscious women who are looking for the latest styles and high-quality products from the world of luxury fashion.

I had the opportunity to work on this web site that used React for UI, NodeJS for server-side logic, and AWS for hosting and deploying. As a senior software/front end engineer, I played a key role in developing the website's features and ensuring its smooth operation.

This included creating and implementing fixing bugs, responsive design and optimizing performance, and new features such as customer reviews, shopping cart, product search, and filtering options.

I was also responsible for debugging and troubleshooting issues with the application, and have experience with performance optimization techniques such as code splitting and lazy loading.

I would like to expand on my point about my experience with this project.

React, React
I used React for UI.
The significance of this was to architect React structure and optimize performance.
So what I mean by this, I had to decide how many components I need to break down and how I have to build reusable components and I implemented them on different pages of the application.
And I had to decide to place the state in an appropriate component.
Also, I implemented Memorizing React components to prevent unnecessary re-renders.
(dynamic data rendering, component-based development, working with third-party libraries, Redux manage component state, React hooks, React Router, how I've used it to handle client-side routing in my projects., component structure, optimize performance, SEO,)
*****
In addition, Memorization is an optimization strategy that caches a component-rendered operation, saves the result in memory, and returns the cached result for the same input.
In essence, if a child component receives a prop, a memorized component shallowly compares the prop by default and skips re-rendering the child

component if the prop hasn't changed.
To do this, I used useCallback() and useMemo().
Also, I used Code-splitting in React using dynamic import. As an app grows, the file size increase, thus increasing the bundle file. This increase slows the initial page load.
With code-splitting, I could split a large bundle file into multiple chunks.
And next, loading time took much longer when the page loads items because our site loaded many images over the network. So, I had to handle this problem. Customers can't see all items in under a minute before the data changes. Not only that, as rendering DOM elements occupies browser memory, there was a chance that our site will take a performance hit and slow down. To do this, I used the Virtualization concept. To put it simply, virtualization is a front-end concept that asks us to render only the amount of data visible in the customer's viewport rather than adding all DOM elements at once. If we render the exact amount of data visible in the user's viewport, the user might have to wait a couple of seconds when scrolling for the following data set to become available in the DOM.
To avoid such a user experience, I fetched additional buffer data to account for the user's scroll behavior. This formula ensured that our site and users are happy when rendering and viewing large data lists.
*****

I faced some challenges and they were...
I had to handle these situations...

Second, I used NodeJS for server-side logic.
I handled large amounts of data and debugged with NodeJS and profiling tools such as the built-in debugger and the V8 inspector.
I implemented APIs, data storage, and retrieval, and worked with third-party modules using NodeJS.
I had some challenges and they were...
I had to overcome these situations...

Third, I also used AWS for the could platform.
I leveraged AWS such as S3 for storage and EC2 for hosting, to scale and deploy the application.
To put it simply, I used EC2 to deploy and scale this project and built serverless functions with AWS Lambda.

I had experience in using AWS Cognito, S3, and RDS.
But, I had no experience with EC2 and Lambda.
Because of that, I faced scalability issues with AWS...
I resolved them by doing ...

Additionally, I maintained my best practices while developing this app like code versioning, testing, security, monitoring, and scaling.

Ultimately, we were able to deliver the project on time and within budget. I believe that my experience on this project is directly relevant to this position. I have developed a strong understanding of the software development lifecycle and have experience working on complex projects with multiple stakeholders. I am confident that I can use what I've learned to make a valuable contribution to your team.

▼ React Performance Experience, React Performance Experience

https://equalparts.com

At GE Digital, I've been working on various projects and tasks for 3 years as a software engineer.
Our company provides software and IIoT (Industrial Internet of Things) services to industrial companies and helps the industry work better.

project explanation

Sure, I'd be happy to discuss a recent project I worked on. It was a project for a large e-commerce provider, and I was a senior software/front end engineer.
It is to sell kitchen accessories and cookware kits.
On this site, customers can find the best seller, shared plats, famous slang, and designs for cookware and kitchen accessories.

We used React for building the UI, NodeJS for server-side logic, and AWS for hosting and deploying the application.
As a software engineer, I was responsible for front-end (full-stack) development and working with closely the design and backend teams.

This included creating and implementing fixing bugs, responsive design and optimizing performance, and new features such as customer reviews, shopping cart, product search, and filtering options.

I was also responsible for debugging and troubleshooting issues with the application, and have experience with performance optimization techniques such as code splitting and lazy loading.

I would like to expand on my point about my experience with this project.

React, React
I used React for UI.
The significance of this was to architect React structure and optimize performance.
So what I mean by this, I had to decide how many components I need to break down and how I have to build reusable components and I implemented them on different pages of the application.
And I had to decide to place the state in an appropriate component.
Also, I implemented Memorizing React components to prevent unnecessary re-renders.
(dynamic data rendering, component-based development, working with third-party libraries, Redux manage component state, React hooks, React Router, how I've used it to handle client-side routing in my projects., component structure, optimize performance, SEO,)
*****
In addition, Memorization is an optimization strategy that caches a component-rendered operation, saves the result in memory, and returns the cached result for the same input.
In essence, if a child component receives a prop, a memorized component shallowly compares the prop by default and skips re-rendering the child component if the prop hasn't changed.
To do this, I used useCallback() and useMemo().
Also, I used Code-splitting in React using dynamic import. As an app grows, the file size increase, thus increasing the bundle file. This increase slows the initial page load.
With code-splitting, I could split a large bundle file into multiple chunks.
And next, loading time took much longer when the page loads items because our site loaded many images over the network. So, I had to handle this problem. Customers can't see all items in under a minute before the data changes. Not only that, as rendering DOM elements occupies browser memory, there was a chance that our site will take a performance hit and slow down. To do this, I used the Virtualization concept. To put it simply, virtualization is a front-end concept that

asks us to render only the amount of data visible in the customer’s viewport rather than adding all DOM elements at once. If we render the exact amount of data visible in the user’s viewport, the user might have to wait a couple of seconds when scrolling for the following data set to become available in the DOM.
To avoid such a user experience, I fetched additional buffer data to account for the user’s scroll behavior. This formula ensured that our site and users are happy when rendering and viewing large data lists.
*****

I faced some challenges and they were...
I had to handle these situations...

Second, I used NodeJS for server-side logic.
I handled large amounts of data and debugged with NodeJS and profiling tools such as the built-in debugger and the V8 inspector.
I implemented APIs, data storage, and retrieval, and worked with third-party modules using NodeJS.
I had some challenges and they were...
I had to overcome these situations...

Third, I also used AWS for the could platform.
I leveraged AWS such as S3 for storage and EC2 for hosting, to scale and deploy the application.
To put it simply, I used EC2 to deploy and scale this project and built serverless functions with AWS Lambda.
I had experience in using AWS Cognito, S3, and RDS.
But, I had no experience with EC2 and Lambda.
Because of that, I faced scalability issues with AWS...
I resolved them by doing ...

Additionally, I maintained my best practices while developing this app like code versioning, testing, security, monitoring, and scaling.

Ultimately, we were able to deliver the project on time and within budget. I believe that my experience on this project is directly relevant to this position. I have developed a strong understanding of the software development lifecycle and have experience working on complex projects with multiple stakeholders. I am confident that I can use what I’ve learned to make a valuable contribution to your team.

▼ SEO, SEO experience

https://www.debenhams.com/

When I developed an e-commerce website last year, I built it using Next.js and focused on SEO.
As you know, SEO is very important for e-commerce websites and is the most vital digital marketing tool. When people are looking for something, they search for it on Google. It makes optimizing websites for search engines more important than ever.
As a front end engineer, I got it.
In simple words, SEO is Google's way to determine the rank of sites for the query entered into the search engine. To gain higher SEO ranks, websites must appeal to their visitors along with meeting all other criteria.
The ranking was very crucial as it would fetch more clicks and traffic for our site.
But, there were challenges with optimization for search engines.
In a SPA, the page first loads on the client side which acts as the empty container.
This is the empty container is then filled by the content integrated by JavaScript.
After requiring a browser for running the script into the SPA, it will be able to load the web pages dynamically.
And, when the search engine bots visit any SPA website, they won't be able to crawl the page. They can only crawl it if the entire content is already updated in the user's browsers. In case, if bots don't find any relevant content then they will regard our website as blank and poorly constructed. If this happens, then the search engine won't index our website.

So I had to overcome these challenges and make SEO-friendly our web site.
To solve this problem, as I said before, I built our site with Next.js.
Next.js offers two options for offering crawlable content to search engines. These options are server-side rendering or prerendering.
I selected the SSR option.
I added a lot of key metadata lives for describing our content in the Head component. By doing this, I could make sure all of our pages include important details that needed to get rendered into our page.
And then, I added a sitemap to our project.

Also, I needed to do many important things like the following.

First, I researched keywords. By doing some investigating, and using professional **keyword research tools** like Moz Keyword Explorer, and Google Keyword Planner, I could understand how people around the world tend to search for things related to our product.

Second, I considered page speed and loading times. How fast a website load is an important element of UX. Therefore, I paid close attention to this, I mean optimizing website speed.
I handled some problems like overloading it with media or features, optimizing image sizes, etc.

Third, I considered meta titles, title tags, and meta descriptions.

Next, as you know, alt text is also an important accessibility feature. So I focused on this.

Also, I created internal links and worked on external links.

The other important thing was to ensure that our site was mobile-friendly.

Finally, I analyzed the results and checked our page rankings and performance with SEO tools like Google Analytics and Google Search Console.

▼ Gatsby, Gatsby experience

++++++++++++E-commerce website (focus on Styled component & Gatsby)
https://equalparts.com, danielwellington.com
When I worked for Capgemini, our team improved performance on the previous e-commerce website with Gatsby. It is a site to shop Cookware Kits and Kitchen Accessories. In this site, customers can find the bestsellers, shared plats, famous slang, blogs, and designs for cookware and kitchen accessories. By doing this, we are branding and advertising goods.
In regard to the technical part, I used Redux for state management, Redux-Saga for middleware, styled Component.
I worked as a frontend React engineer for this project.
Our migration to Gatsby was not a work from scratch. we relied on the existing backend structure to provide me with features such as inventory and cart management. The whole payment flow was also not part of this migration.

As we started to migrate a few markets to Gatsby, we quickly realized that the build time would easily skyrocket into a range that wouldn't work for our team. Each market was taking around 5 minutes to build, so at 40 markets we would reach hours of build time.

We dug into the issue, and realized that our GraphQL queries were several orders of magnitude slower than the benchmarks provided by Gatsby.
One reason for this issue being our need for customization: every page (frontpage, category page or campaign page) can be fully customized, which results in our GraphQL queries being complex - and slow.
We started optimizing them by running them only when needed. For instance, our footer is the same on every single page (for a market), so it's better to run the query for that component in Gatsby-node's create Pages and inject the result via the context of the page, rather than having the query for the footer in the pages' GraphQL query.
It was a good improvement.

▼ Healthcare, Healthcare experience

https://***.com **Healthcare project**

My recent project was a booking application for patients to book appointments with doctors. They can choose specific doctors, times, and clinics with Google Maps. Also, by filling in the information in this application, they can introduce themselves and their medical history. By doing this, the doctor can prepare the meeting with the patient, and let them know more about the patient.

In regard to the **technical part**, I used React, Redux for global state management, AWS amplify and Lambda for the backend, AWS SNS, SES for sending SMS and email, IVS for video streaming, S3 bucket for storing image files, Twilio for send MMS, and used bootstrap and styled component for styling components.

I worked in the React frontend team for this project and mainly worked on select Appointments, card scans and OCR, Medical history, and review appointment parts. My team has 12 team members and one team leader.

The biggest challenge in this project was implementing OCR. We have to implement OCR functionality to get the information from ID cards, Insurance cards, and Driving license card. There are many services to scan cards, and we decided to use **micro blink for this**, but there's no one for an insurance card. Even if there's one, we couldn't spend money on this service, because we need 3 different OCR functionalities. Because of this, I had to implement the OCR for the Insurance card, and succeed. I am really proud of this.

To do this, I used tesseract-OCR NPM to read card data and tried to detect card data manually. It was really difficult to scan several Insurance cards. Anyway, I've succeeded to detect all data from the Insurance card.

▼

▼

▼ **Leader, Leader experience,**

▼

▼ **At GE, Themesoft**

At GE Digital

In my role as a software engineer at GE, I have the opportunity to work extensively with React. I am responsible for building and maintaining a large-scale web app such as an e-commerce platform using React.
This includes creating and implementing fixing bugs, responsive design and optimizing performance, and new features such as customer reviews, shopping cart, product search, and filtering options.

I have also been responsible for debugging and troubleshooting issues with the application, and have experience with performance optimization techniques such as code splitting and lazy loading.

I have also worked with various React libraries and frameworks such as Material-UI, AntD, Next.js, and Gatsby by building responsive and performant user interfaces.
I am gaining experience in creating reusable components and implementing them on different pages of the application.
I have experience with both functional and class-based components, and am well-versed in the use of hooks, such as useState, useEffect, useCallback, and useMemo. I also have expereience working with Redux, as well as other state management libraries such as MobX.
And I have experience with building reusable components and have a strong understanding of the component lifecycle. I have also worked on projects that required the use of Higher-Order Components (HOCs) and Render Props.
I am familiar with using React context for state management, which allowed me to easily share states across different components without the need for prop drilling.
I also have experience with Redux, which I used to manage the global state of the application.

In addition to building and maintaining the web application, I also have experience with integrating React with other technologies, such as the API and database. I also used Axios and Fetch to make API calls, and worked with GraphQL.
I am familiar with working with Git for version control and had experience in working in a team environment following Agile development methodologies.
Overall, my experience with React has been extensive and has given me a strong understanding of React concepts and best practices. I am also confident in my ability to create and maintain high-quality web apps using React and am excited to continue working with this technology in the future.

I also built a social media platform using React and Firebase, where users can create profiles, post updates, and interact with other users. I implemented various features such as user authentication, real-time chat, and a news feed.

At Themesoft

At Themesoft, I had extensive experience working with React and I was responsible for building and maintaining the company's web application, which was built using React as the primary framework.
I also had experience in using React hooks and working with state management using the Redux library. And I had experience in working with other libraries such as React Router for client-side routing and Axios for making API calls and had experience in working with CSS preprocessors such as SASS and LESS to style the components.

And I worked with other front-end technologies such as HTML, CSS, and JavaScript and integrated them with the React application.
I also had the opportunity to work on a project where we migrated an existing Angular application to React. This involved analyzing the existing application's architecture and identifying areas that could be improved, re-writing the application's components using React and Redux.
I also implemented new features and functionality using React hooks and functional components.

In addition, I had experience working with React Native for developing mobile applications and implementing push notifications and in-app purchases.

▼ React Projects

React projects

A web application for a retail company where I implemented a front-end using React, allowing users to browse and purchase products. I also implemented various features such as a shopping cart, product search, and filtering options.

A project management tool built using React and a Node.js backend. I implemented a variety of features such as task creation, task assignments, and project timelines, as well as user authentication and role-based access controls.

A real-time dashboard for monitoring and analyzing data streams, built using React and D3.js. I implemented various visualizations, such as line charts, bar charts, and pie charts, to display the data in an interactive and easy-to-understand manner.

▼

▼ Technical Round

▼ Front end

▼ React

▼ Links

https://www.javatpoint.com/react-interview-questions

▼ React?

React is a JavaScript library for building user interfaces.

React is used to build single-page applications.

React allows us to create reusable UI components.

▼ What are the advantages of using React?

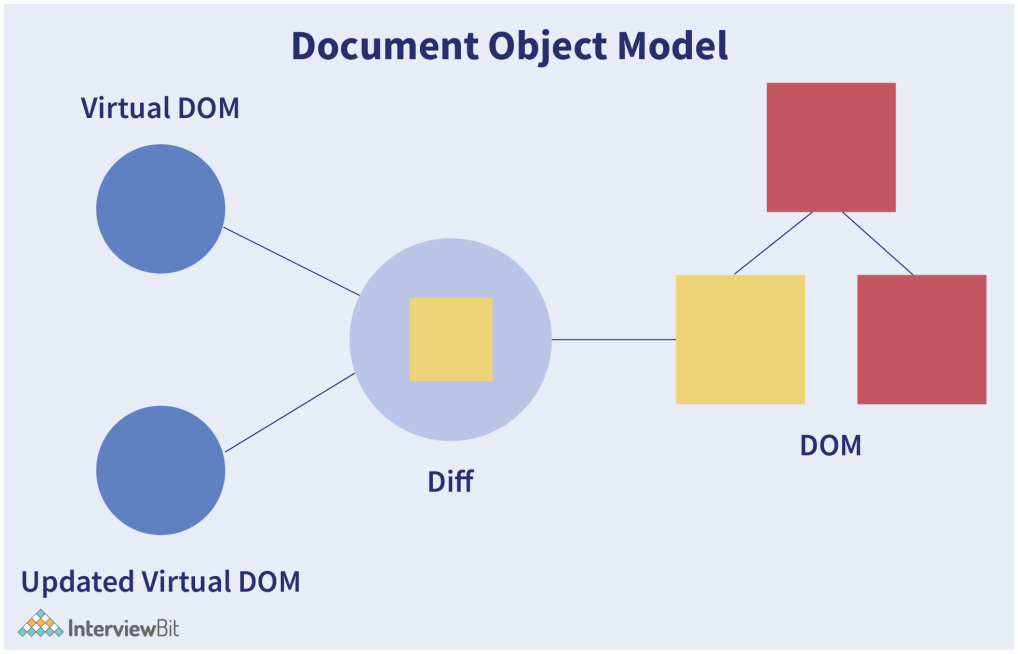
MVC is generally abbreviated as Model View Controller.

- **Use of Virtual DOM to improve efficiency:** React uses virtual DOM to render the view. As the name suggests, virtual DOM is a virtual representation of the real DOM. Each time the data changes in a react app, a new virtual DOM gets created. Creating a virtual DOM is much faster than rendering the UI inside the browser. Therefore, with the use of virtual DOM, the efficiency of the app improves.

- **Gentle learning curve:** React has a gentle learning curve when compared to frameworks like Angular. Anyone with little knowledge of javascript can start building web applications using React.
- **SEO friendly:** React allows developers to develop engaging user interfaces that can be easily navigated in various search engines. It also allows server-side rendering, which boosts the SEO of an app.
- **Reusable components:** React uses component-based architecture for developing applications. Components are independent and reusable bits of code. These components can be shared across various applications having similar functionality. The re-use of components increases the pace of development.
- **Huge ecosystem of libraries to choose from:** React provides you with the freedom to choose the tools, libraries, and architecture for developing an application based on your requirement.

▼ What is V-DOM? What is difference between V-DOM and Real DOM? What is DOM?

As stated by the react team, virtual DOM is a concept where a virtual representation of the real DOM is kept inside the memory and is synced with the real DOM by a library such as ReactDOM.



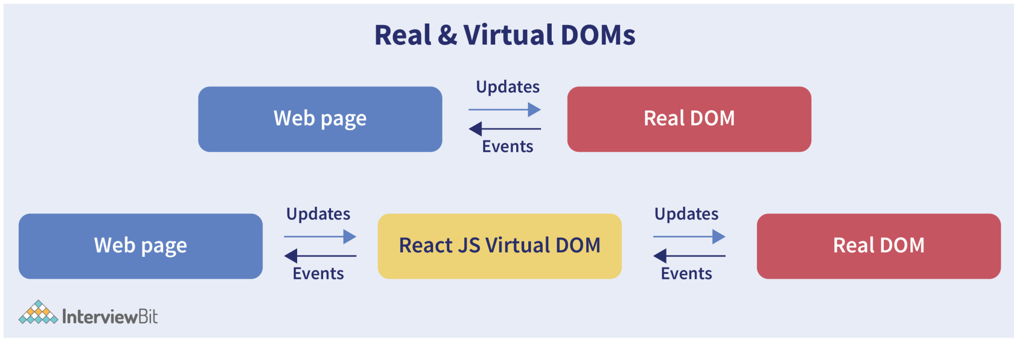
Why was virtual DOM introduced?

DOM manipulation is an integral part of any web application, but DOM manipulation is quite slow when compared to other operations in JavaScript. The efficiency of the application gets affected when several DOM manipulations are being done. Most JavaScript frameworks update the entire DOM even when a small part of the DOM changes.

For example, consider a list that is being rendered inside the DOM. If one of the items in the list changes, the entire list gets rendered again instead of just rendering the item that was changed/updated. This is called inefficient updating.

To address the problem of inefficient updating, the react team introduced the concept of virtual DOM.

How does it work?



For every DOM object, there is a corresponding virtual DOM object(copy), which has the same properties. The main difference between the real DOM object and the virtual DOM object is that any changes in the virtual DOM object will not reflect on the screen directly. Consider a virtual DOM object as a blueprint of the real DOM object. Whenever a JSX element gets rendered, every virtual DOM object gets updated.

> **Note- One may think updating every virtual DOM object might be inefficient, but that's not the case. Updating the virtual DOM is much faster than updating the real DOM since we are just updating the blueprint of the real DOM.

React uses two virtual DOMs to render the user interface. One of them is used to store the current state of the objects and the other to store the previous state of the objects. Whenever the virtual DOM gets updated, react compares the two virtual DOMs and gets to know about which virtual DOM objects were updated. After knowing which objects were updated, react renders only those objects inside the real DOM instead of rendering the complete real DOM. This way, with the use of virtual DOM, react solves the problem of inefficient updating.

Real/Browser DOM:

DOM stands for **'Document Object Model'.** It is a structured representation of HTML in the webpage or application. It represents the entire UI**(User Interface)** of the *web application as the tree data structure.*

Whenever there is any change in the *state of the application UI,* DOM is updated and represents the change. The DOM is rendered and manipulated with every change for updating the application User Interface, which affects the performance and slows it down.

Therefore, with many UI components and the complex structure of **DOM,** It will update in more expensive as it needs to be re-rendered with each change.

The DOM is constituted as a tree data structure. It consists of the node for each **UI element** present in the web document.

▼ **Shadow DOM**

Shadow DOM allows hidden DOM trees to be attached to elements in the regular DOM tree

▼ **What is useState() in React?**

The useState() is a built-in React Hook that allows you for having state variables in functional components. It should be used when the DOM has something that is dynamically manipulating/controlling.

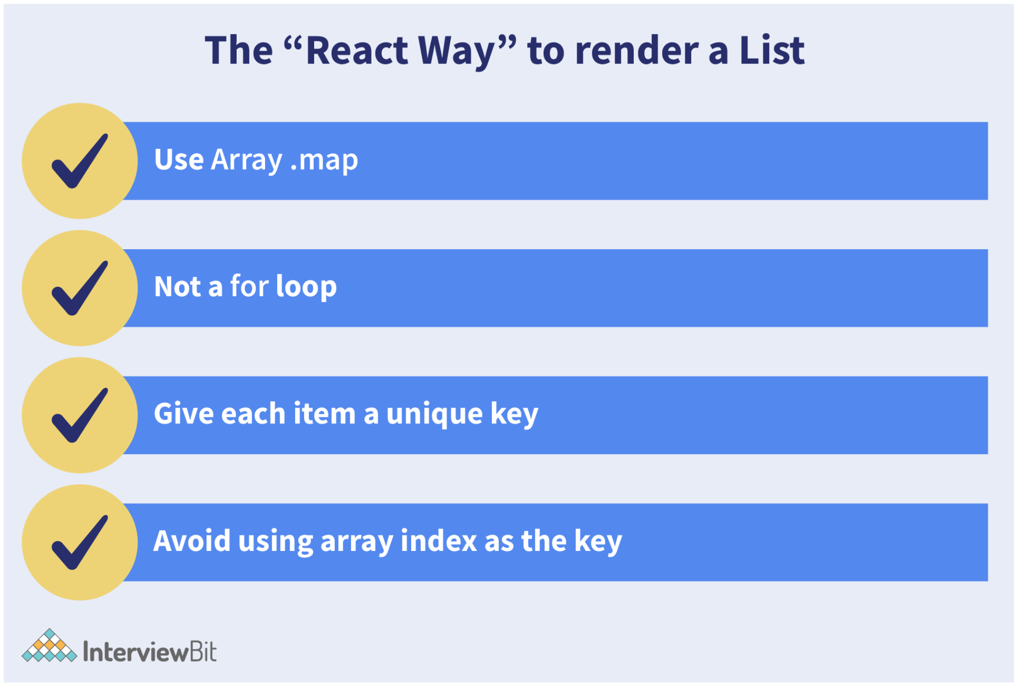
In the below-given example code, The useState(0) will return a tuple where the count is the first parameter that represents the counter’s current state and the second parameter setCounter method will allow us to update the state of the counter.

```
...
const [count, setCounter] = useState(0);
const [otherStuffs, setOtherStuffs] = useState(...);
...
const setCount = () => {
   setCounter(count + 1);
   setOtherStuffs(...);
   ...
};
```

We can make use of setCounter() method for updating the state of count anywhere. In this example, we are using setCounter() inside the setCount function where various other things can also be done. The idea with the usage of hooks is that we will be able to keep our code more functional and avoid class-based components if they are not required.

▼ **What are keys in React?**

A key is a special string attribute that needs to be included when using lists of elements.


The "React Way" to render a List
Use Array .map
Not a for loop
Give each item a unique key
Avoid using array index as the key
InterviewBit

Example of a list using key -

```
const ids = [1,2,3,4,5];
const listElements = ids.map((id)=>{
```

```
return(
<li key={id.toString()}>
  {id}
</li>
)
})
```

Importance of keys -

- Keys help react identify which elements were added, changed or removed.
- Keys should be given to array elements for providing a unique identity for each element.
- Without keys, React does not understand the order or uniqueness of each element.
- With keys, React has an idea of which particular element was deleted, edited, and added.
- Keys are generally used for displaying a list of data coming from an API.

> ***Note- Keys used within arrays should be unique among siblings. They need not be globally unique.

▼ **ESLint**

ESlint is an open-source library that’s commonly used by React developers to enforce rules about maintaining the code standard across the project. It’s fully customizable so you can configure the desired rules yourself.

Linting your code can help you spot any would be bugs. Given that JavaScript is dynamically typed, it’s really easy for developers to introduce hard to debug runtime errors. ESLint can help spot those dangerous patterns and show warnings about them ahead of execution.

▼ **JSX**

What is JSX?

JSX is a syntax extension to JavaScript. So JSX may remind you of a template language, but it comes with the full power of JavaScript. Also, we can use JSX in React and it produces React “elements”.

Why do you use JSX in React?

As you know, JSX produces React “elements” which we use to explore rendering to the DOM.

React embraces the fact that rendering logic is inherently coupled with other UI logic: how events are handled, how the state changes over time, and how the data is prepared for display.

And JSX allows us to add elements inside the JavaScript code. It also allows React to show more useful error and warning messages.

▼ **Refs**

What is Refs and does it do?

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

In the typical React dataflow, props are the only way that parent components interact with their children. To modify a child, we re-render it with new props. However, there are a few cases where we need to imperatively modify a child outside of the typical dataflow. The child to be modified could be an instance of a React component, or it could be a DOM element. For both of these cases, React provides an escape hatch.

When to Use Refs

- Managing focus, text selection, or media playback.
- Triggering imperative animations.
- Integrating with third-party DOM libraries.

▼ **Lifting State Up**

Why

When several components need to reflect the same changing data, we can use lifting the shared state up to their closest common ancestor.

How

For example, let’s imagine there are 2 components that they have two props values, here one is same. We want to share state between these two components with one same value, for example temperature value. Parent component has these two child components. And these child components will share temperature state.

To do this, we need to have temperature value as state object in parent component, and we have to write temperature value as props in child components. And in child components, we can reflect change temperature value via onChangeTemperature() as props function.

This is called “Lifting State Up”. In a nutshell, we can share state object in parent component between child components.

▼ **React elements**

What is React element?

React element is just a description of an HTML element, a React component, or a mix of these. In a nutshell, a React Element is what gets returned from components. And it’s an object that virtually describes the DOM nodes that a component represents. With a function component, this element is the object that the function returns. And with a class component, the element is the object that the component’s render function returns. React elements are not what we see in the browser. They are just objects in memory, and we can’t change anything about them.

What is difference between React component and React element?

React Element is a simple object that describes a DOM node and its attributes or properties we can say, and it is an immutable description object, and you cannot apply any methods on it.

But React Component is a function or class that accepts an input and returns a React Element. It has to keep references to its DOM nodes and to the instances of the child components. Also, it is independent and reusable. It returns the virtual DOM of the element.

How do you create a React element?

We can create by using React.createElement() and ReactDom.render() methods.

▼ **Components**

What is React Component?

Components in React basically return a piece of JSX code that tells what should be rendered on the screen. In React, we mainly have two types of components: Functional Components and Class Components.

- A component used in one area of the application can be reused in another area. This helps speed up the development process.
- A component can contain several other components.
- A component must define a render method that specifies how the component renders to the DOM in its minimal form.
- A component can also receive props. These are properties passed by its parent to specify values.

What are Functional Components?

Functional Component is a simple JavaScript function that accepts props and returns a React element. After the introduction of Hooks, functional components are equivalent to class components. It has become the standard way of writing React components in modern applications.

Although functional components are the new trend, the react team insists on keeping class components in React. Therefore, it is important to know how these components differ.

Functional components can be declared using an arrow function or the function keyword:

```
function card(props){
return(
<div className="main-container">
<h2>Title of the card</h2>
</div>
)
}
const card = (props) =>{
return(
<div className="main-container">
<h2>Title of the card</h2>
</div>
)
}
```

Why should we use functional components in React?

In my opinion, there are some benefits us get by using functional components in React.

First, functional components are much easier to read and test because they are plain JavaScript functions without state or lifecycle-hooks.

Also, we end up with less code.

In addition, they help us to use best practices. It will get easier to separate container and presentational components because we need to think more about our component’s state if we don’t have access to setState() method in your component.

Finally, the React team mentioned that there may be a performance boost for functional component in future React versions.

How do you use Functional Components in React?

We can create a functional component to React by writing a JavaScript function.

What are Class Components?

React class components have a built-in state object.

Class components, on the other hand, are declared using the ES6 class:

```
class Card extends React.Component{
constructor(props){
super(props);
}
render(){
return(
<div className="main-container">
<h2>Title of the card</h2>
</div>
)
}
}
```

Why use Class Components in React?

We use to track state and lifecycle on a React component.

What are differences between Functional Components and Class Components?

Well, I think the most obvious difference is the syntax. A functional component is just a plain JavaScript function which accepts props as an argument and returns a React element. And a class component requires you to extend from React.Component and create a render function which returns a React element.

When should we use a Functional Component and a Class Component?

In class components, the render method will be called, whenever the state of the components changes. On the other hand, the functional components render the UI based on the props.

Class Components	**Function Components**
Class components need to extend the component from “React.Component” and create a render function that returns the required element.	Functional components are like normal functions which take “props” as the argument and return the required element.
They are also known as stateful components.	They are also known as stateless components.
They implement logic and the state of the component.	They accept some kind of data and display it in the UI.
Lifecycle methods can be used inside them.	Lifecycle methods cannot be used inside them.
Format of Hooks inside class components: `constructor(props) { super(props); this.state = { color: ‘ ‘ } }`	Format of Hooks inside function components: `const [color,SetColor]= React.useState('')`
It needs to store state therefore constructors are used.	Constructors are not used in it.
It has to have a “render()” method inside that.	It does not require a render method.

Which is faster Functional Component or Class Component?

In my opinion, nothing is better, because both have pros and cons. But class components are important to understand React flow and lifecycle methods. The new learner should practice React using class components.

Once they are familiar with class components, they can learn and functional components.

Why Functional Components are better than Class Components?

Personally, nothing is better. but, in addition, I found the functional component easier to understand compared to the class component, although this might be different for a developer from object-oriented programming like Java. And it is easier to test, read and write than class component.

▼ Context API

Concept

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

In a typical React application, data is passed top-down (parent to child) via props, but such usage can be cumbersome for certain types of props (e.g. locale preference, UI theme) that are required by many components within an application. Context provides a way to share values like these between components without having to explicitly pass a prop through every level of the tree.

What is the purpose of React Context API?

The main purpose of Context API is to avoid props drilling.

React context API is an interface for sharing information with parent and nested child components without explicitly passing the data as props.
The purpose of the context system is all about sharing data between parents and some nested child components and in core context is just like props.

If we used props to send data from top to bottom of the nested child component. To do this without Context, we will need to pass the state as “props” through each nested component until it reaches the consumer component of props. This is called “prop drilling”.

+++ It is a part of React library and no need to install any third-party library and built specifically for React.

Also, it is easy to share data between parents and nested children but difficult to build store components.

Specifically designed for static data, that is not often refreshed or updated.

How to use Context API?

First, we can assign a export function in exampleContext.js while creating our context, above example, we can

```
import { createContext } from ‘react’;

export const ExampleContext = createContext();
```

Next, We can import contextEample.js in the Provider component.

And then, we need to wrap our consumer component inside the Provider component. As you know, the Provider component has value props, we can use these special props to set data to our consumer component.

```
import { ExampleContext } from ‘./context/exampleContext’;

import ConsumerComponent from ‘./components/Consumer’;

function App(){

  const example = “testContextValue";

  return (

     <ExampleContext.Provider value = {{ example }}>

          <ConsumerComponent />

     </ExampleContext.Provider>

  )

}

export default App;
```

Here, value props can have values and methods.

Finally, we can use value props in ConsumerComponent above exmple.

```
import React, { useContext } from ‘react’;

import { ExampleContext } from ‘../context/exampleContext’;

functions ConsumerComponent() {

     const { example } = useContext(ExampleContext);
```

```
    return (
        <div>
            <h3>Context Example Value: {example} </h3>


    )
export default ConsumerComponent;
```

When to use React Context API?

That’s good question, I think it is important to know when and where to use Context API. In some situations, we may need to access the same data at different levels of the nesting children components.

The context is easy to use and it lets us broadcast such data, and changes it, to all components below. We can use React context in these situations like UI Theme, Locale or Language preference, Data cache and Currently authenticated user.

On the other hand, Context makes component difficult to reuse and Context consumer component can’t be used outside the provider component hierarchy.

In addition, if you only want to avoid the Props Drilling problem. the component composition is often a simpler solution than context.

React context vs redux

The context API is not a replacement for redux, because Redux provides a way to share data throughout our component hierarchy and gives us a way to centralize all our data inside the common store. Also, provide a mechanism for changing the data store.

In addition, if you are using Redux only to avoid passing props down to deeply nested components, then you could replace Redux with the Context.

As you know, the Redux is a state management library and it makes complex applications easier to write and manage. Keep in mind each of these two has different usage and we can use both based on our needs.

▼ **Redux**

Concept

The Redux is a state management library and it makes complex applications easier to write and manage.

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

Why we use Redux?

When using Redux with React, states will no longer need to be lifted up. This makes it easier for you to trace which action causes any change. The component does not need to provide any state or method for its children components to share data among themselves.

What is Redux used for

Redux is used to maintain and update data across our applications for multiple components to share, all while remaining independent of the components.

Without Redux, we have to pass data through different components to where it's needed.

But with Redux, we can make state data independent of the components, and when needed, a component can access or update through the Redux store.

What is state management in Redux?

As you know, React are built with a way for components to internally manage their state without any need for an external library or tool. It does well for applications with few components, but as the application grows bigger, managing states shared across components becomes a chore.

For instance, in React, to share data among siblings, a state has to live in the parent component. A method for updating this state is provided by the parent component and passed as props to these sibling components.

State management is essentially a way to facilitate communication and sharing of data across components. It creates a tangible data structure to represent the state of our app that we can read from and write to.

▼ **What is reducer**

A reducer is a pure function that takes the previous state and an action as arguments and returns a new state. Actions are an object with a type and an optional payload:

Reducers specify how the application's state changes in response to actions that are dispatched to the store.

Since reducers are pure functions, we **don't** mutate the arguments given to it, perform API calls, routing transitions or call non-pure functions like Math.random() or Date.now().

If your app has multiple pieces of state, then you can have multiple reducers. For example, each major feature inside your app can have its own reducer. Reducers are concerned only with the value of the state.

▼ What's an Action?

Actions are plain JavaScript objects that represent payloads of information that send data from your application to your store. Actions have a `type` and an optional `payload`.

Most changes in an application that uses Redux start off with an event that is triggered by a user either directly or indirectly. Events such as clicking on a button, selecting an item from a dropdown menu, hovering on a particular element or an AJAX request that just returned some data. Even the initial loading of a page can be an occasion to **dispatch** an action. Actions are often dispatched using an action creator.

▼ What is Store

In Redux, the `store` refers to the object that brings actions (that represent what happened) and reducers (that update the state according to those actions) together. There is only a **single** store in a Redux application.

The store has several duties:

- Allow access to state via `getState()`.
- Allow state to be updated via `dispatch(action)`.
- Registers listeners using `subscribe(listener)`.

- Unregisters listeners via the function returned by `subscribe(listener)`.
- Holds the whole application state.

▼ **What’s an Action Creator?**

In Redux, an action creator is a function that returns an action object. Action creators can seem like a superfluous step, but they make things more portable and easy to test. The action object returned from an action creator is sent to all the different reducers in the app.

Depending on what the action is, reducers can choose to return a new version of their piece of state. The newly returned piece of state then gets piped into the application state, which then gets piped back into our React app, which then causes all of our components to re-render.

So let's say a user clicks on a button, we then call an action creator, which is a function that returns an action. That action has a `type` that describes the type of action that was just triggered.

▼ **Redux-Saga**

Redux-Saga is a library that aims to make application side effects (i.e. asynchronous things like data fetching and impure things like accessing the browser cache) easier to manage, more efficient to execute, easy to test and better at handling failures. Redux-Saga basically is a middleware for Redux.

▼ **Redux-Thunk**

Thunk ia a pattern of writing functions with logic inside that can interact with a Redux store’s dispatch and getState() methods. Uning thunk requires the redux-thunk middleware to be added to the Redux sotre as part of its configuration.

The most common use case for Redux Thunk is for communicating asynchronously with an external API to retrieve or save data. Redux Thunk makes it easy to dispatch actions that follow the lifecycle of a request to an external API.

▼

▼ **Styled-components**

Introduction

The component-driven approach to building software applications majorly influenced by modern libraries and frameworks begs for better, more scalable and more usable styling methods. This has given rise to new and modularized methods of structuring and managing our styled components-scoped rather than involves generating CSS styles from JavaScript.

Also, it allows us to use all CSS features, including media queries, keyframe animation, nesting, pseudo-classes, etc.

What is styled-components?

It is a React-specific CSS-in-JS styling solution that creates a platform for developers to write actual CSS code to style React components, as well as in React Native. In a nutshell, it makes presentational components very simple.

Using styled-components makes it simple to use CSS to style components by writing JavaScript code. These components are known as “styled components” which are, in actuality, a React component with styles.

???It removes the mapping between components and styles. This means that when you’re defining your styles, you’re actually creating a normal React component, that has your styles attached to it.

What does it do?

First, it only applies the styles of the components that are rendered on a page. Also, it enables us to add styles to a component depending on props or a global theme, and is easy to delete CSS. Finally, it is easy and accessible maintenance.

Advantages of Styled-Components

First, Styled-components generate unique class names for our styles. This means that we do not have to worry about class names colliding or styles conflicting, since each style is tied to an individual component.

Second, it is easier to maintain. We can debug our code easily no matter how big the codebase is.

Finally, Styled-components makes it possible to nest our CSS selectors just as we would in Sass. Only in this case, we do not have to install Sass.

Downsides of Using Styled-component.

First, a third-party JavaScript library will be added to our application, and this means more load to our project. It is easier for the browser to run CSS than tons of JS.

And then, it may take a while for developers to get used to the styled-components syntax as it differs from the traditional way of writing CSS.

Styled-Components vs functional component using the style parameter?

With styled-components you can leverage the full power of CSS, which means you can:

First, we can overwrite the CSS of child tags and classes

Also, we can use pseudo classes such as :visited, :first etc.

And, we can use hover and other UI states. Finally, we can use CSS variables easily.

---

createGlobalStyle

Styled-components provides a createGlobalStyle helper function that allows we to create a special styled component where you can declare your global/base styles.

Normally, styled-components are scoped to a particular class or component but the createGlobalStyle function removes that limitation and allows whatever style we declare within it to be applied globally across your app. This serves as an opportunity for you to declare your CSS resets and base styles.

making style from styles

const styles = `border-color: red, padding-left: 15px`;

```
const lines = styles
                  .split(';')
                  .map(line => line.trim())
                  .filter(line => line !== "")

  // Function courtesy of mck89 on StackOverflow
  const convertToCamelCase = (key) =>
    key.replace(/-([a-z])/g, (x, up) => up.toUpperCase())
  const style = lines.reduce((acc, line) => {
    const lineParsed = line.split(':');
    const key = convertToCamelCase(lineParsed[0]);
    const val = lineParsed[1];
    return { ...acc, [key]: val };
  }, {});
```

style = { borderColor: red, paddingLeft: 15px };

CSS-in-JS

CSS-in-JS is a technique for writing CSS style directly into JavaScript file. It helps to define styles in JS in a more component-approach manner, this means that it scopes each style to an individual component.

Styled components vs inline styles

1. In inline styles, you write Object-like styles vs. actual CSS
2. Writing CSS allows all the benefits of CSS (like selectors).
3. Re-use and maintaining CSS is much easier than inline styles.
4. The real debate is between CSS-in-JS vs. CSS modules vs. SASS vs. Normal CSS (for that you can google or read related questions in SO).
5. Inline styles considered bad-practice as in addition they override all defined styles.
6. Inline-styles are limited in their functionality (try animating something in inline style or even implementing a media query).

▼ **What is difference between Pure component and function, programming?**

▼ **Pure components?**

Higher Order Components

- It is a function that takes a component and returns a new component.
- It facilitates reusing of component logic.

Pure Components

- React.Component is the base class for React components. React.PureComponent is a variation of React.Component class and does a shallow comparison of props and state.

- A React component can be considered pure if it renders the same output for the same state and props.

▼ **Higher-Order Components (HOC)**

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.

Concretely, **a higher-order component is a function that takes a component and returns a new component.**

▼ **State, Lifecycle and Props**

State

The state is a built-in React object that is used to contain data or information about the component.

And it represents parts of an Application that can change. It is also mutable and local to the component only.

Why state and not a local variable?

First, local variables don't persist(retain data) between renders. When React renders a component for the second time, it renders it from scratch. It doesn't consider any changes to the local variables.

Second, changes to local variables won't trigger renders. React doesn't realize it needs to render the component again with the new data.

Props

Props are known as properties it can be used to pass data from one component to another. Props can't be modified, read-only and immutable.

Difference between Props and State

First, Props are used to pass data from one component to another. But, State is a local data storage that is local to the component only and can't be passed to other components.

Second, Props can't be modified, but State can be modified and state is both read and write.

props	***state***

Can get initial value from parent Component?	Yes
Can be changed by parent Component?	Yes
Can set default values inside Component?*	Yes
Can change inside Component?	No
Can set initial value for child Components?	Yes
Can change in child Components?	Yes

PROPS	**STATE**
The Data is passed from one component to another.	The Data is passed within the component only.
It is Immutable (cannot be modified).	It is Mutable (can be modified).
Props can be used with state and functional components.	State can be used only with the state components/class component (Before 16.0).
Props are read-only.	State is both read and write.

Lifecycle

In React, each component has a lifecycle which we can monitor and manipulate during its three main phases. The three phases are Mounting, Updating, and Unmounting.

- Mounting means putting elements into the DOM.

  constructor(), render(), componentDidMount()

- Updating is when a component is updated.

  A component is updated whenever there is a change in the component’s state or props.

  shouldComponentUpdate(), render(), componentDidUpdate()

- Unmounting is when a component is removed from the DOM, or unmounting as React likes to call it.

  componentWillUnmount()

▼ **React.lazy**

The React.lazy function lets us render a dynamic import as a regular component.

`const OtherComponent = React.lazy(() => import('./OtherComponent'));`

This will automatically load the bundle containing the `OtherComponent` when this component is first rendered.

`React.lazy` takes a function that must call a dynamic `import()`. This must return a `Promise` which resolves to a module with a `default` export containing a React component.

The lazy component should then be rendered inside a `Suspense` component, which allows us to show some fallback content (such as a loading indicator) while we're waiting for the lazy component to load.

The `fallback` prop accepts any React elements that you want to render while waiting for the component to load. You can place the `Suspense` component anywhere above the lazy component. You can even wrap multiple lazy components with a single `Suspense` component.

▼ 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.

Error boundaries do **not** catch errors for:

- Event handlers (learn more)
- Asynchronous code (e.g. `setTimeout` or `requestAnimationFrame` callbacks)
- Server side rendering
- Errors thrown in the error boundary itself (rather than its children)

▼ React Fragment

Fragments let us group a list of children without adding extra nodes to the DOM.

▼ What is difference between React and ReactJS?

▼ React Hooks

What is React Hooks?

React Hooks are the built-in functions that permit developers for using the state and lifecycle methods within React components. These are newly added features made available in React 16.8 version. Each lifecycle of a component is having 3 phases which include mount, unmount, and update. Along with that, components have properties and states. Hooks will allow using these methods by developers for improving the reuse of code with higher flexibility navigating the component tree.

Using Hook, all features of React can be used without writing class components. ***For example***, before React version 16.8, it required a class component for managing the state of a component. But now using the useState hook, we can keep the state in a functional component.

Explain React Hooks.

What are Hooks? Hooks are functions that let us “hook into” React state and lifecycle features from a **functional component.**

React Hooks **cannot** be used in class components. They let us write components without class.

Why were Hooks introduced in React?

React hooks were introduced in the 16.8 version of React. Previously, functional components were called stateless components. Only class components were used for state management and lifecycle methods. The need to change a functional component to a class component, whenever state management or lifecycle methods were to be used, led to the development of Hooks.

Example of a hook: **useState hook:**

In functional components, the useState hook lets us define a state for a component:

```
function Person(props) {
// We are declaring a state variable called name.
// setName is a function to update/change the value of name
let [name, setName] = useState('');
}
```

The state variable “name” can be directly used inside the HTML.

What are the rules that must be followed while using React Hooks?

There are 2 rules which must be followed while you code with Hooks:

- React Hooks must be called only at the top level. It is not allowed to call them inside the nested functions, loops, or conditions.
- It is allowed to call the Hooks only from the React Function Components.

---

useEffect Hooks

The `useEffect` Hook allows you to perform side effects in your components.

Some examples of side effects are: fetching data, directly updating the DOM, and timers.

`useEffect` accepts two arguments. The second argument is optional.

`useEffect(<function>, <dependency>)`

1. No dependency passed:

```
useEffect(() => {
  //Runs on every render
});
```

2. An empty array:

```
useEffect(() => {
  //Runs only on the first render
}, []);
```

3. Props or state values:

```
useEffect(() => {
  //Runs on the first render
  //And any time any dependency value changes
}, [prop, state]);
```

---

useState()

To use the `useState` Hook, we first need to `import` it into our component.

Example:

At the top of your component, `import` the `useState` Hook.

```
import { useState } from "react";
```

Initialize state at the top of the function component.

```
import { useState } from "react";

function FavoriteColor() {
  const [color, setColor] = useState("");
}
```

The useState() is a built-in React Hook that allows you for having state variables in functional components. It should be used when the DOM has something that is dynamically manipulating/controlling.

In the below-given example code, The useState(0) will return a tuple where the count is the first parameter that represents the counter's current state and the second parameter setCounter method will allow us to update the state of the counter.

```
...
const [count, setCounter] = useState(0);
const [otherStuffs, setOtherStuffs] = useState(...);
...
const setCount = () => {
   setCounter(count + 1);
   setOtherStuffs(...);
   ...
};
```

We can make use of setCounter() method for updating the state of count anywhere. In this example, we are using setCounter() inside the setCount function where various other things can also be done. The idea with the usage of hooks is that we will be able to keep our code more functional and avoid class-based components if they are not required.

---

useContext()

React Context is a way to manage state globally.

It can be used together with the `useState` Hook to share state between deeply nested components more easily than with `useState` alone.

The Problem

State should be held by the highest parent component in the stack that requires access to the state.

To illustrate, we have many nested components. The component at the top and bottom of the stack need access to the state.

To do this without Context, we will need to pass the state as "props" through each nested component. This is called "prop drilling".

The Solution

The solution is to create context.

To create context, you must Import `createContext` and initialize it:

```
import { useState, createContext } from "react";
import ReactDOM from "react-dom/client";

const UserContext = createContext()
```

Next we'll use the Context Provider to wrap the tree of components that need the state Context.

Wrap child components in the Context Provider and supply the state value.

```
function Component1() {
  const [user, setUser] = useState("Jesse Hall");

  return (
    <UserContext.Provider value={user}>
      <h1>{`Hello ${user}!`}</h1>
      <Component2 user={user} />
    </UserContext.Provider>);
}
```

Now, all components in this tree will have access to the user Context.

In order to use the Context in a child component, we need to access it using the `useContext` Hook.

First, include the `useContext` in the import statement:

```
import { useState, createContext, useContext } from "react";
```

Then you can access the user Context in all components:

```
function Component5() {
  const user = useContext(UserContext);

  return (
    <>
      <h1>Component 5</h1>
      <h2>{`Hello ${user} again!`}</h2>
    </>);
}
```

---

useRef()

The `useRef` Hook allows you to persist values between renders.

It can be used to store a mutable value that does not cause a re-render when updated.

It can be used to access a DOM element directly.

Does Not Cause Re-renders

If we tried to count how many times our application renders using the `useState` Hook, we would be caught in an infinite loop since this Hook itself causes a re-render.

To avoid this, we can use the `useRef` Hook.

Example:

Use `useRef` to track application renders.

```
import { useState, useEffect, useRef } from "react";
import ReactDOM from "react-dom/client";

function App() {
  const [inputValue, setInputValue] = useState("");
  const count = useRef(0);

  useEffect(() => {
    count.current = count.current + 1;
  });

  return (
    <>
      <input
        type="text"value={inputValue}onChange={(e) => setInputValue(e.targe
t.value)}/>
      <h1>Render Count: {count.current}</h1>
    </>);
}

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(<App />);
```

It's like doing this: `const count = {current: 0}`
. We can access the count by using `count.current`
.

`useRef()` only returns one item. It returns an Object called `current`.

When we initialize `useRef` we set the initial value: `useRef(0)`.

Accessing DOM Elements

In general, we want to let React handle all DOM manipulation.

But there are some instances where `useRef` can be used without causing issues.

In React, we can add a `ref` attribute to an element to access it directly in the DOM.

Example:

Use `useRef` to focus the input:

```
import { useRef } from "react";
import ReactDOM from "react-dom/client";

function App() {
  const inputElement = useRef();

  const focusInput = () => {
    inputElement.current.focus();
  };

  return (
    <>
      <input type="text" ref={inputElement} />
      <button onClick={focusInput}>Focus Input</button>
    </>);
}

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(<App />);
Tracking State Changes
The useRef Hook can also be used to keep track of previous state values.

This is because we are able to persist useRef values between renders.

Example:
Use useRef to keep track of previous state values:

import { useState, useEffect, useRef } from "react";
import ReactDOM from "react-dom/client";

function App() {
  const [inputValue, setInputValue] = useState("");
  const previousInputValue = useRef("");

  useEffect(() => {
    previousInputValue.current = inputValue;
  }, [inputValue]);

  return (
    <>
      <input
        type="text"
        value={inputValue}
        onChange={(e) => setInputValue(e.target.value)}
      />
      <h2>Current Value: {inputValue}</h2>
      <h2>Previous Value: {previousInputValue.current}</h2>
    </>
  );
}

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(<App />);
```

What are Custom Hooks?

A Custom Hook is a function in Javascript whose name begins with ‘use’ and which calls other hooks. It is a part of React v16.8 hook update and permits you for reusing the stateful logic without any need for component hierarchy restructuring.

In almost all of the cases, custom hooks are considered to be sufficient for replacing render props and HoCs (Higher-Order components) and reducing the amount of nesting required. Custom Hooks will allow you for avoiding multiple layers of abstraction or wrapper hell that might come along with Render Props and HoCs.

The **disadvantage** of Custom Hooks is it cannot be used inside of the classes.

▼ **What is difference between functional programming and functional component?**

▼ **SSR vs CSR**

===**SSR===**

What is SSR (Server-side Rendering)

SSR (Server-side Rendering) is method or concept when HTML is build on server rather than in the browser (client).

Server-side rendering is one of the easiest ways to create React web applications that are SEO-friendly. Although, if you need to build a single page application that can render on the server, then you'll require to add Next.js.

Why do we use SSR?

There are two primary reasons why you should consider using Server-side rendering:

- Search engine optimization (SEO).
- Perceived performance.

SSR (Server-side Rendering)

- Browser requests a page
- **Content visible to the user** (because HTML is build on server)
- Browser requests linked JS bundle
- React app loads
- **User can interact with app**
- Request API data
- Re-render React app with new data

Reasons we should move to the server side

Server-side rendering initially means every page is rendered and loaded from the server. The initial page is rendered from the server, meaning the subsequent pages load directly from the client.

```
import express from "express";
import ReactDOMServer from "react-dom/server";
import { StaticRouter } from "react-router-dom/server";
import App from "./App";

let app = express();
```

```
app.get("*", (req, res) => {
  let html = ReactDOMServer.renderToString(
    <StaticRouter location={req.url}>
      <App />
    </StaticRouter>
  );
  res.send("<!DOCTYPE html>" + html);
});

app.listen(3000);
```

===CSR===

What is CSR (Client-side Rendering)

CSR (Client-side rendering), happens when a empty HTML file is first served to the browser and then HTML is build on the browser and the painting is started.

Client-side rendering means that the Google bot and browser get HTML files or files that have very little content. After that, JavaScript code downloads the content from the server which enables the users to view it on their screens.

CSR has also many advantages over SSR:

Based on the architecture, CSR has also many advantages over SSR:

- CSR must be considered for static pages/content to reduce time taken First Meaningful Paint.
- CSR, can be chosen when there is a strong tendency to use mobile web.

CSR (Client-side Rendering)

- Browser requests a page
- Blank index.html loaded
- Browser requests linked JS bundle
- React app loads
- **Content visible to the user** (because HTML is build in browser)
- **User can interact with app**

- Request API data
- Re-render React app with new data

▼ **PWA**

It stands for progressive web app.

▼ **What is pre-rendering?**

▼

▼

▼

▼ **Controlled and Uncontrolled component**

Controlled component is component that get the changed value from the callback function and uncontrolled component is component that have the one from the DOM. For example, When input value is changed, we can use onChange function in Controlled Component and also we can get the value using DOM like ref.

Uncontrolled Components:
Uncontrolled Components are the components that are not controlled by the React state and are handled by the **DOM**
 (Document Object Model). So in order to access any value that has been entered we take the help of refs.

Controlled Components:
In React, Controlled Components are those in which form's data is handled by the component's state. It takes its current value through props and makes changes through callbacks like onClick, onChange, etc. A parent component manages its own state and passes the new values as props to the controlled component.

▼ **Side effects**

Side effects are not predictable because they are actions which are performed with the "outside world."

We perform a side effect when we need to reach outside of our React components to do something.

Common side effects include:

- Making a request to an API for data from a backend server
- To interact with browser APIs (that is, to use `document` or `window` directly)
- Using unpredictable timing functions like `setTimeout` or `setInterval`

This is why useEffect exists: to provide a way to handle performing these side effects in what are otherwise pure React components.

For example, if we wanted to change the title meta tag to display the user's name in their browser tab, we *could* do it within the component itself, but we shouldn't.

```
function User({ name }) {
  document.title = name;
  // This is a side effect. Don't do this in the component body!

  return <h1>{name}</h1>;
}
```

▼ **React event**

An event is an action triggered by the user or any system event, like pressing a key, a mouse key, etc.

▼

▼ What is Hoisting?

▼ What is Hoisting?

▼ **React Interview Questions for freshers(https://www.interviewbit.com/react-interview-questions/)**

▼ HTML

▼ HTML Form Validation

HTML form validation is a process of examining the HTML form page's contents to avoid errored-out data being set to the server. This process is a significant step in developing HTML-based web applications, as it can easily improve the quality of the web page or the web application.

▼ What is an Attribute in HTML?

- Attributes are the properties that can be added to an HTML tag that change the way the tag behaves or is displayed.
- It adds attributes right after the name of the HTML tag, inside the brackets.

▼ What is Marquee in HTML?

- The marquee is used to scroll the text on the webpage.
- It automatically scrolls the image or text up, down, left, or right.
- You must use <marquee> tags to apply for a marquee.

▼ What is SVG in HTML?

```
<svg width="100" height="100">
 <circle cx="50" cy="50" r="40" stroke="yellow" stroke-width="4" fill="red" />
</svg>
```

- HTML SVG is a markup language that describes vector and raster graphics. XML text files define SVG pictures and associated behaviors.
- It's typically used for X, Y coordinate system diagrams like pie charts and 2-Dimensional graphs.

▼ In HTML, how do you separate a section of text?

In HTML, you use the following tags to divide a chunk of text:

<br> tag–It's a character that's used to break up a line of text. It transfers the text flow to a new line by breaking the existing line.

<p> tag–This tag is used to create a text paragraph.

<blockquote> This tag is used to indicate big quoted passages.

▼ **How do you Create Nested Web Pages in HTML?**

<iframe src="https://simplilearn.com/" height="600" width="800"></iframe>

Using HTML's built-in iframe tag, you can create nested web pages.

▼

▼ CSS

▼ Advantage

- **Separation of content from presentation -** CSS provides a way to present the same content in multiple presentation formats in mobile or desktop or laptop.
- **Easy to maintain -** CSS, built effectively can be used to change the look and feel complete by making small changes. To make a global change, simply change the style, and all elements in all the web pages will be updated automatically.
- **Bandwidth -** Used effectively, the style sheets will be stored in the browser cache and they can be used on multiple pages, without having to download again.

▼ **What is the best way to include CSS Styling in HTML?**

There are three methods for incorporating CSS into HTML:

- You use inline CSS when only a single element needs to be styled or when a little quantity of styling is required.
- External Style Sheet: When a style is applied to many elements or HTML pages, it employs an external style sheet.
- Internal Style Sheet: An internal style sheet is employed when a single HTML document has a distinct style and numerous elements.

External styles are the preferred and cleanest way of writing CSS.

▼ **What are Sass, Less and Stylus?**

Sass - Sass is the acronym for “Syntactically Awesome Style Sheets”. A ‘$’ sign commonly precedes it.

$font-color: #fff

$bg-color: #00f   #box

color: $font-color

background: $bg-color

Less - LESS is an acronym for “Leaner Stylesheets”. Less uses ‘@’ to define the variables.

@font-color: #fff;

@bg-color: #00f

#box{

color: @font-color;

background: @bg-color;

}

Stylus - Stylus offers a great deal of flexibility in writing syntax. It doesn’t use @ or $ for defining variables.

font-color= #fff;

bg-color = #00f;   #box {

color: font-color;

background: bg-color; }

▼ **Explain Box Sizing Property**

Box Sizing property defines how the height and width of a box are calculated.

Content Box - The default width and height apply only to the element’s content. Padding and border are added outside the box.

Padding Box - You add the dimensions to both element’s content and padding. It adds the border outside the box.

Border Box - The dimensions are added to the content, padding, and border.

▼ **What are the different ways to hide an Element using CSS?**

display: none

Hides the content and doesn't store it in the DOM

visibility: hidden

It adds the element to the DOM and takes up space. However, it is not visible to the user

position: absolute

You can make the element appear outside the screen

▼ **What does 'Important' in CSS mean?**

The 'important' keyword indicates the highest precedence, and it overrides the cascaded property.

p {

color:blue !important;

}

#thing {

color: green;

}

<p id="thing">Will be BLUE.</p>

▼ **What are CSS Sprites?**

- Since each image sends out an HTTP request separately, a web page with a high number of photos takes longer to load.
- CSS sprites are used to minimize the loading time of a web page by combining multiple small pictures into a single image.
- It decreases the number of HTTP requests and, as a result, the time it takes for pages to load.

▼ **Which Property is used to Underline, Strikethrough, and Overline Text?**

text-style

Text-type

text-decoration

Text-transform

▼

▼

▼ Box model

The CSS box model is essentially **a box that wraps around every HTML element**. It consists of: margins, borders, padding, and the actual content.

- **Content:** Actual Content of the box where the text or image is placed.
- **Padding:** Area surrounding the content (Space between the border and content).
- **Border:** Area surrounding the padding.
- **Margin:** Area surrounding the border.

▼ CSS vs **CSS3**

- The main difference between CSS and CSS3 is that CSS3 has modules.CSS is the basic version, and it does not support responsive design. CSS3, on the other hand, is the latest version and supports responsive design.
- CSS cannot be split into modules, but CSS3 can be split into modules. Being an older version of CSS is slower than CSS3.
- In addition to these, CSS3 has many alignment features. CSS3 provides a box-sizing tool that allows the user to get the correct size of any element without making any changes in the element's dimensions or padding. CSS does not have any box-sizing tool, and hence the user needs to use the standard procedures defined to align text.
- The animations and 3D transformations are better in CSS3. Elements can be moved on the screen with the help of flash and JavaScript. Using this, the elements will also be able to change their size and color. All kinds of transitions, transformations, and animations can be done using CSS3. CSS does not provide 3D animation and transformations.
- CSS provides basic color schema and standard colors. CSS3 supports RGBA, HSLA, HSL and gradient colours. It also supports rounded image corners for text boxes.

- Multi-column text blocks can be defined in CSS3. CSS only supports single text blocks.

▼ ellipsis

```
.ellipsis {
  overflow: hidden;
  text-overflow: ellipsis;
  display: block;
  white-space: nowrap;
}
```

▼ Grid and flex box

The basic difference between CSS Grid Layout and CSS Flexbox Layout is that **flexbox was designed for layout in one dimension - either a row or a column.**
Grid was designed for two-dimensional layout - rows, and columns at the same time.

▼ HTML vs CSS

1. HTML is basically a standard markup language for describing the structure of web pages, whereas CSS is the style sheet language for describing the presentation and design of web pages
2. HTML is easy to learn and has clear syntax, whereas CSS can sometimes get messy and can create complications in codes.
3. CSS is independent of HTML, and it can be used with any XML-based markup language, whereas this is not the same case with HTML
4. The HTML file can contain CSS codes, but on the other hand, CSS can never contain HTML codes in it.
5. HTML provides tags which are surrounding the content of any web page elements, whereas CSS consists of selectors which are surrounded by a declaration block
6. CSS has fragmentation, but HTML doesn’t produce any such problems.
7. CSS uses a much lesser code and thus produce much lesser web page loading time than HTML

▼ How does CSS styling work?

As a styling language, **CSS specifies how users view documents, from the layout to the style**. The documents affected are typically text files that get structure from a markup language, the most common of which is HTML, however, XML and SVG are also popular.

▼ Importance

CSS (Cascading Style Sheets) is used **to style and layout web pages** — for example, to alter the font, color, size, and spacing of your content, split it into multiple columns, or add animations and other decorative features

CSS is important because **it controls all design-related aspects of your website**. Typography, colours, page layouts and any other visual aspects of your website are all controlled by CSS

▼ Include CSS in the webpage?

There are different ways to include a CSS in a webpage,

1 - External Style Sheet: An external file linked to your HTML document: Using link tag, we can link the style sheet to the HTML page.

```
<link rel="stylesheet" type="text/css" href="mystyles.css" />
```

2 - Embed CSS with a style tag: A set of CSS styles included within your HTML page.

```
<style type="text/css">
/*Add style rules here*/
</style>
```

Add your CSS rules between the opening and closing style tags and write your CSS exactly the same way as you do in stand-alone stylesheet files.

3 - Add inline styles to HTML elements(CSS rules applied directly within an HTML tag.): Style can be added directly to the HTML element using a style tag.

```
<h2 style="color:red;background:black">Inline Style</h2>
```

4 - Import a stylesheet file (An external file imported into another CSS file): Another way to add CSS is by using the @import rule. This is to add a new CSS file within CSS itself.

```
@import "path/to/style.css";
```

▼ Inline, inline-block, and block?

Block Element: The block elements always start on a new line. They will also take space for an entire row or width. List of block elements are <div>, <p>.

Inline Elements: Inline elements don't start on a new line, they appear on the same line as the content and tags beside them. Some examples of inline elements are <a>, <span> , <strong>, and <img> tags.

Inline Block Elements: Inline-block elements are similar to inline elements, except they can have padding and margins and set height and width values

▼ Preprocessor

A **CSS preprocessor** is a program that lets you generate CSS from the preprocessor's own unique syntax.

There are many CSS preprocessors to choose from, however most CSS preprocessors will add some features that don't exist in pure CSS, such as mixin, nesting selector, inheritance selector, and so on. These features make the CSS structure more readable and easier to maintain.

- Sass
- LESS
- Stylus
- PostCSS

▼ Semantic markup

▼ SASS

```
.services {
    @include grid-row;

    article {
        @include grid-column(12);

        @media #{$medium-up} {
            @include grid-column(6);
        }

        @media #{$large-up} {
            @include grid-column(3);
        }
    }
}
```

▼ CSS

```
.services {
    width: 100%;
    margin-left: auto;
    margin-right: auto;
    max-width: 62.5rem;
}

.services article {
    padding-left: 0.9375rem;
    padding-right: 0.9375rem;
    width: 100%;
    float: left;
}

@media only screen and (min-width: 40.063em) {
    .services article {
        width: 50%;
        float: left;
    }
}

@media only screen and (min-width: 64.063em) {
    .services article {
        width: 25%;
        float: left;
    }
}
```

▼ Pseudo-elements and Pseudo classes?

Pseudo-elements allows us to create items that do not normally exist in the document tree

- ::before
- ::after
- ::first-letter
- ::first-line
- ::selection

In the below example, the color will appear only on the first line of the paragraph.

```
p: :first-line {
  color: #ff0000;
font-variant: small-caps;
}
```

Pseudo-classes select regular elements but under certain conditions like when the user is hovering over the link.

- :link
- :visited
- :hover
- :active
- :focus

Example of the pseudo-class, In the below example, the color applies to the anchor tag when it’s hovered.

```
/* mouse over link */
a:hover {
color: #FF00FF;
}
```

▼ Reset vs Normalize CSS? How do they differ?

Reset CSS: CSS resets aim to remove all built-in browser styling. For example, margins, paddings, font-sizes of all elements are reset to be the same.

Normalize CSS: Normalize CSS aims to make built-in browser styling consistent across browsers. It also corrects bugs for common browser dependencies

▼ SCSS

- **CSS : Cascading Style Sheet** is basically the scripting language. CSS is used for designing web pages.CSS is the most important web technologies that are widely used along with HTML and JavaScript. CSS have file extension of **.css**.
- **SCSS : Syntactically Awesome Style Sheet** is the superset of CSS. SCSS is the more advanced version of CSS. Due to its advanced features, it is often termed as Sassy CSS. SCSS have file extension of **.scss**.

Differences: SCSS contains all the features of CSS and contains more features that are not present in CSS which makes it a good choice for developers to use it.

1. SCSS offers variables, you can shorten your code by using variables. It is a great advantage over conventional CSS
2. SASS adds the feature of @import which lets you import your customized SCSS files.

   ```
   @import "my theme";
   @import "framework/bootstrap";
   ```

3. SASS allows us to use nested syntax. Let’s say if you have to style a specific ‘paragraph’ in ‘div’ in ‘footer’ you can definitely do that by SASS.
4. At last, SASS is well documented than CSS.

▼ Selectors

A CSS selector is the first part of a CSS Rule. It is a pattern of elements and other terms that tell the browser which HTML elements should be selected to have the CSS property values inside the rule applied to them.

▼ **Universal Selector**

The universal selector works like a wildcard character, selecting all elements on a page. In the given example, the provided styles will get applied to all the elements on the page.

```
* {
color: "green";
font-size: 20px;
line-height: 25px;
}
```

▼ Element Type Selector

This selector matches one or more HTML elements of the same name. In the given example, the provided styles will get applied to all the ul elements on the page.

```
ul {
  line-style: none;
border: solid 1px #ccc;
}
```

▼ ID Selector

This selector matches any HTML element that has an ID attribute with the same value as that of the selector. In the given example, the provided styles will get applied to all the elements having ID as a container on the page.

```
#container {
width: 960px;
margin: 0 auto;
}

<div id="container"></div>
```

▼ Class Selector

The class selector also matches all elements on the page that have their class attribute set to the same value as the class. In the given example, the provided styles will get applied to all the elements having ID as the box on the page.

```
.box {
padding: 10px;
margin: 10px;
width: 240px;
}

<div class="box"></div>
```

▼ Descendant Combinator

The descendant selector or, more accurately, the descendant combinator lets you combine two or more selectors so you can be more specific in your selection method.

```
#container .box {
float: left;
padding-bottom: 15px;
}

<div id="container">
  <div class="box"></div>

  <div class="box-2"></div>
</div>

<div class=”box”></div>
```

This declaration block will apply to all elements that have a class of box that is inside an element with an ID of the container. It’s worth noting that the `.box` element doesn’t have to be an immediate child: there could be another element wrapping `.box`, and the styles would still apply.

▼ Child Combinator

A selector that uses the child combinator is similar to a selector that uses a descendant combinator, except it only targets immediate child elements.

```
#container> .box {
float: left;
padding-bottom: 15px;
}

<div id="container">
```

```
    <div class="box"></div>

    <div>
      <div class="box"></div>
    </div>
  </div>
```

The selector will match all elements that have a class of `box` and that are immediate children of the `#container` element. That means, unlike the descendant combinator, there can't be another element wrapping `.box` it has to be a direct child element.

General Sibling Combinator

A selector that uses a general sibling combinator to match elements based on sibling relationships. The selected elements are beside each other in the HTML.

```
h2 ~p {
margin-bottom: 20px;
}

<h2>Title</h2>
<p>Paragraph example.</p>
<p>Paragraph example.</p>
<p>Paragraph example.</p>
<div class=”box”>
  <p>Paragraph example.</p>
</div>
```

In this example, all paragraph elements (<p>) will be styled with the specified rules, but only if they are siblings of `<h2>` elements. There could be other elements in between the `<h2>` and `<p>`, and the styles would still apply.

Adjacent Sibling Combinator

A selector that uses the adjacent sibling combinator uses the plus symbol (+), and is almost the same as the general sibling selector. The difference is that the targeted element must be an immediate sibling, not just a general sibling.

```
p +p {
text-indent: 1.Sem;
margin-bottom: 0;
}

<h2>Title</h2>
<p>Paragraph example.</p>
<p>Paragraph example.</p>
<p>Paragraph example.</p>

<div class=”box”>
  <p>Paragraph example.</p>
  <p>Paragraph example.</p>
</div>
```

The above example will apply the specified styles only to paragraph elements that immediately follow other paragraph elements. This means the first paragraph element on a page would not receive these styles. Also, if another element appeared between two paragraphs, the second paragraph of the two wouldn’t have the styles applied.

▼ **Attribute Selector**

The attribute selector targets elements based on the presence and/or value of HTML attributes, and is declared using square brackets.

```
input [type=”text”] {
background-color: #444;
width: 200px;
}

<input type="text">
```

▼ Semantic HTML?

Semantic HTML or semantic markup is **HTML that introduces meaning to the web page rather than just presentation**. For example, a <p> tag indicates that the enclosed text is a paragraph. This is both semantic and presentational because people know what paragraphs are, and browsers know how to display them

- article, aside, details, figure, footer, header, main, mark, nav, section, summary, time

▼ VH/VW (viewport height/ viewport width)

It’s a CSS unit used to measure the height and width in percentage with respect to the viewport. It is used mainly in responsive design techniques.

The measure VH is equal to 1/100 of the height of the viewport. If the height of the browser is 1000px, 1vh is equal to 10px. Similarly, if the width is 1000px, then 1 vw is equal to 10px

display none, visibility hidden,

▼ JS

▼ What is JS?

JavaScript is used to add user interaction to an application.

JavaScript is a scripting language that is used to create and manage dynamic web pages, basically anything that moves on your screen without requiring you to refresh your browser. It can be anything from animated graphics to an automatically generated Facebook timeline.

▼ Why JavaScript?

JavaScript is an essential programming language, almost compulsory to learn for students or software developers that are gravitated towards web development. Wondering why? Here’s the answer:

- JavaScript is the most popular programming language in the world and that makes it a default choice for web development. There are many frameworks available which you can use to create web applications once you have learned JavaScript.
- JavaScript offers lots of flexibility. You can create stunning and fast web applications with tons of customizations to provide users with the most relevant graphical user interface.
- JavaScript is now also used in mobile app development, desktop app development, and game development. This opens many possibilities for you as a Javascript developer.
- Due to the high demand in the industry, there are tons of job growth opportunities and high pay for those who know JavaScript.
- The incredible thing about Javascript is that you can find tons of frameworks and libraries already developed, which can be used directly

in web development. That reduces the development time and enhances the graphical user interface.

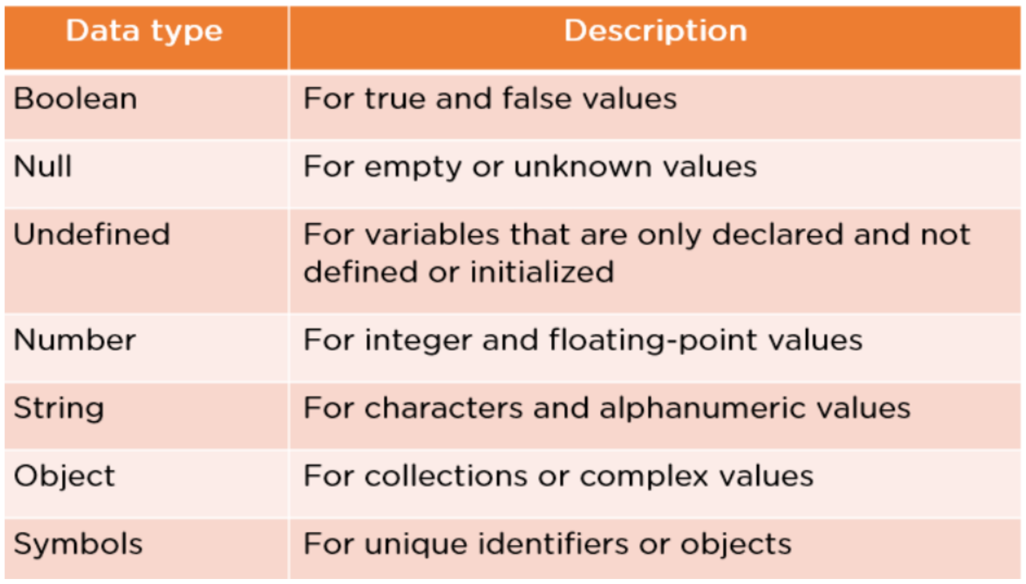
▼ What is JavaScript Used For?

JavaScript is used in various fields from the web to servers, and here's a quick list of the significant areas it's used in:

- Web Applications: JavaScript is used for adding interactivity and automation to websites. So, if you want your web application to be anything more than just a static page of contents, you'll probably need to do some "JavaScript'ing."
- Mobile Applications: JavaScript isn't just for developing web applications; it is also used for developing applications for phones and tablets. With frameworks like React Native, you can develop full-fledged mobile applications with all those fancy animations.
- Web-based Games: If you've ever played a game directly on the web browser, JavaScript was probably used to make that happen.
- Back-end Web Development: JavaScript has traditionally been used for developing the front-end parts of a web application. However, with the introduction of NodeJS, a prevalent back-end JavaScript framework, things have changed. And now, JavaScript is used for developing the back-end structure also.

▼ What are the various Data Types in JavaScript?

JavaScript has many data types to provide the basic functionality needed for a web application. They are mentioned in this table.



Data type	Description
Boolean	For true and false values
Null	For empty or unknown values
Undefined	For variables that are only declared and not defined or initialized
Number	For integer and floating-point values
String	For characters and alphanumeric values
Object	For collections or complex values
Symbols	For unique identifiers or objects

What is callback in JS?

A callback is a JavaScript function that is sent as an argument or parameter to another function.

You call this function whenever the function to which it is provided is called.

What do you understand about Cookies in JavaScript?

A cookie is a little piece of data sent by a website and kept on the user's computer by the web browser that was used to access the page.

Cookies are used to remember information for later use and to keep track of a website's browsing activities.

The simplest approach to make a cookie with JavaScript is to do it as follows:

document.cookie = "key1 = value1; key2 = value2; expires = date";

To delete a cookie, you can just set an expiration date and time; specifying the correct path of the cookie is a good practice.

```
function delete_cookie(name) {
document.cookie = name + "=; Path=/; Expires=Thu, 01 Jan 1970 00:00:01 GMT;";
}
```

▼

▼ Differences between const, let and var.

- `var` declarations are globally scoped or function scoped while `let` and `const` are block scoped.
- `var` variables can be updated and re-declared within its scope; `let` variables can be updated but not re-declared; `const` variables can neither be updated nor re-declared.
- They are all hoisted to the top of their scope. But while `var` variables are initialized with `undefined`, `let` and `const` variables are not initialized.
- While `var` and `let` can be declared without being initialized, `const` must be initialized during declaration.

▼ What is scope in JavaScript?

▼

▼ Adding JavaScript into an HTML Document

You can add JavaScript code in an HTML document **by employing the dedicated HTML tag <script> that wraps around JavaScript code**. The <script> tag can be placed in the <head> section of your HTML or in the <body> section, depending on when you want the JavaScript to load.

▼ Arrow function

ES6 arrow function is an alternative way to write a shorter syntax compared to the function expression.

Do not use - An arrow function doesn't have its own `this` value and the `arguments` object. Therefore, you should not use it as an event handler, a method of an object literal, a prototype method, or when you have a function that uses the `arguments` object.

▼ Asynchronous operation

Asynchronous operation means that **a process operates independently of other processes,** whereas synchronous operation means that the process runs only as a result of some other process being completed or handed off.

▼ Closure

Closure is the combination of a function bundled together with references to its surrounding state. In other words, a closure gives you access to an outer function’s scope from an inner function. In JavaScript, closures are created every time a function is created, at function creation time.

```
function makeAdder(x) {
  return function (y) {
    return x + y;
  };
}

const add5 = makeAdder(5);
const add10 = makeAdder(10);

console.log(add5(2)); // 7
console.log(add10(2)); // 12
```

▼ Currying

Currying is a technique of evaluating function with multiple arguments into a sequence of function with single argument.

Why it’s useful ?

- Currying helps you to avoid passing the same variable again and again.
- It helps to create a higher order function. It is extremely helpful in event handling.

▼ Data types

- Primitive values
    - Number, String, Boolean, Symbol, Object, undefined, and null.

        ▼ Undefined vs Null

        Null is a special value meaning "no value". Null is a special object because typeof null returns 'object'. On the other hand, **undefined** means that the variable has not been declared, or has not been given a value.

- Objects (collections of properties)

▼ Debug

1. Developer tools in modern web browsers. **All modern browsers have a built-in JavaScript debugger**. Built-in debuggers can be turned on and off, forcing errors to be reported to the user. With a debugger, you can also set breakpoints (places where code execution can be stopped), and examine variables while the code is executing
2. Node. Debugging the Node.js on your application's backend can be challenging.
3. **Postman** for debugging requests and responses.
4. ESLint. is a linter for JavaScript. Linters will analyze code as it is written and identify a variety of basic syntax problems. The use of ESLint will allow you to catch errors, particularly easy to resolve but annoying ones such as missing brackets or typos, before executing the code. ESLint is available as a Node package. It has also been set up as a plugin for many code editors such as Sublime Text 3 and VS Code, which will then mark the offending errors right in your editor window.

▼ Default parameter

In JavaScript, default function parameters allow you to initialize named parameters with default values if no values or `undefined` are passed into the function.

▼ Destructuring

ES6 provides a new feature called destructing assignment that allows you to destructure properties of an object or elements of an array into individual variables.

▼ ES6 module

An ES6 module is a JavaScript file that executes in strict mode only. It means that any variables or functions declared in the module won't be added automatically to the global scope.

▼ ES5 VS. ES6

1. require vs import

2. export vs exports
3. component and function
4. props
5. state

- ▼ Event loop

  JavaScript has a runtime model based on an **event loop**, which is responsible for executing the code, collecting and processing events, and executing queued sub-tasks.

  A JavaScript runtime uses a message queue, which is a list of messages to be processed. Each message has an associated function that gets called to handle the message.

  Each message is processed completely before any other message is processed.

  In web browsers, messages are added anytime an event occurs and there is an event listener attached to it. If there is no listener, the event is lost. So a click on an element with a click event handler will add a message

- ▼ Falsy value

  undefined , null , NaN , 0 , "" (empty string), and false

- ▼ Generator

  In JavaScript, a regular function is executed based on the run-to-completion model. It cannot pause midway and then continues from where it paused.

  A generator can pause midway and then continues from where it paused.

- ▼ Hash

  A ***hash function*** is any function that can be used to map data of arbitrary size to fixed-size values. The values returned by a ***hash function*** are called hash

  - ▼ time complexity

    Like arrays, hash tables provide constant-time **O(1)** lookup on average, regardless of the number of items in the table. The (hopefully rare) worst-case lookup time in most hash table schemes is O(n)

▼ HTML vs HTML5

- In HTML, video and audio are not supported, whereas in HTML5, video and audio are integrated into it.
- **HTML Semantic Elements: A semantic element clearly describes its meaning to both the browser and the developer. making the code more readable and easier to maintain**

  Ex: article, aside, details, figcaption, figure, footer, header, main, mark, nav, section, summary, time
- HTML is compatible with almost all browsers, whereas HTML5 is supported by most modern browsers such as Firefox, Mozilla, Chrome, etc.
- In HTML, JavaScript and browser interface will run in the same thread, whereas in HTML5, we can run JavaScript in the background with the help of web worker API can run in different threads.
- In HTML, vector graphics are supported with the help of other tools such as Silver light, Flash, etc., whereas in HTML5, vector graphics are supported by default; it has canvas and SVG inbuilt.
- In HTML5 applet tag was removed, which is used for displaying applets, and an object tag was introduced, whereas, in HTML, the applet tag is being used.
- In HTML, <a> tag was used as an anchor to refer to a link, whereas in HTML5, <a> tag is used as hyperlink only.
- In HTML, the <acronym> tag was used for displaying abbreviation, whereas, in HTML5, this tag is replaced by the <abbr> tag, which will be used for the same purpose.
- HTML is unable to handle inaccurate syntax and other errors, whereas HTML5 is capable of handling the errors.
- In HTML5, <table> tag can have only one attribute border, and the value needs to be zero or one, whereas, in HTML, we can have many attributes.
- In HTML, communication between client and server will happen using streaming and long pooling as it doesn’t have support for sockets,

whereas HTML5 has support for web socket through which full-duplex communication is possible between client and server.

▼ HTTP request methods

`GET`

`GET` method should only retrieve data.

`HEAD`

The `HEAD` method retrieve data like `GET` request, but without the response body.

`POST`

The `POST` method submits an entity to the specified resource, often causing a change in state or side effects on the server.

`PUT`

The `PUT` method replaces all current representations of the target resource with the request payload.

`DELETE`

The `DELETE` method deletes the specified resource.

`CONNECT`

The `CONNECT` method establishes a tunnel to the server identified by the target resource.

`OPTIONS`

The `OPTIONS` method describes the communication options for the target resource.

`TRACE`

The `TRACE` method performs a message loop-back test along the path to the target resource.

`PATCH`

The `PATCH` method applies partial modifications to a resource.

▼ Hoisting

JavaScript Hoisting refers to the process whereby the interpreter allocates memory for variable and function declarations prior to execution of the code

▼ Observable

Observable JavaScript represents **a progressive way of handling events, async the activity, and multiple values in JavaScript.** These observables are just the functions that throw values and Objects known as observers subscribe to such values that define the callback functions such as error(), next() and complete().

▼ JSON

JSON is a text-based data format following JavaScript object syntax. **Even though it closely resembles JavaScript object literal syntax**, it can be used independently from JavaScript, and many programming environments feature the ability to read and generate JSON.

JSON	JavaScript Object
The key in key/value pair should be in double quotes.	The key in key/value pair can be without double quotes.
JSON cannot contain functions.	JavaScript objects can contain functions.
JSON can be created and used by other programming languages.	JavaScript objects can only be used in JavaScript.

▼ Promise vs Async/Await

Promise is an object representing an intermediate state of operation which is guaranteed to complete its execution at some point in the future. Async/Await is a syntactic sugar for promises, a wrapper making the code execute more synchronously.

▼ Promise

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

The Promise object supports two properties: state and result
While a Promise object is "pending" (working), the result is undefined. When a Promise object is "fulfilled", the result is a value. When a Promise object is "rejected", the result is an error object.

▼ Async/await

The async keyword is used to define an asynchronous function, which returns a AsyncFunction object. The await keyword is used to pause async function execution until a Promise is fulfilled, that is resolved or rejected, and to resume execution of the async function after fulfillment.

- ▼ Prototype

  A prototype is an existing inbuilt functionality in ***JavaScript***. Whenever we create a ***JavaScript function***, JavaScript adds a prototype property to that function. A prototype is an object, where it can add new variables and methods to the existing object. i.e., Prototype is a base class for all the objects, and it helps us to achieve the inheritance.

- ▼ Pure function

  A Pure Function is **a function (a block of code) that always returns the same result if the same arguments are passed**. It does not depend on any state or data change during a program's execution. Rather, it only depends on its input arguments.

- ▼ Recursive function

  A recursive function is **a function that calls itself until it doesn't.** And this technique is called recursion.

- ▼ Rest parameters

  ES6 provides a new kind of parameter, so-called rest parameter, that has a prefix of three dots `(...)`. A rest parameter allows you to represent an indefinite number of arguments as an array.

- ▼ Spread Operator

  ES6 provides a new operator called spread operator that consists of three dots (...). The spread operator allows you to spread out elements of an iterable object such as an array, map, and set.

- ▼ Temporal death zone (TDZ)

  A variable declared by the `let` keyword has a temporal dead zone (TDZ). The TDZ is the time from the start of the block until the variable declaration is processed.

- ▼ this keyword

  The JavaScript this keyword refers **to the object it belongs to**. In a function, this refers to the global object. In an event, **this keyword** refers to the element that received the event.

▼ Triple Equal vs double equal

While these are both comparison equality operators, the **triple equal** is what's called a strict equality operator, while the double equals is an equality operator. The strict equality operator will compare both the value and type of the operands (the values on the left/right sides of the operator).

Triple Equal is superior to double equals. Whenever possible, you should use triple equals to test equality. By testing the type and value, you can be sure that you are always executing a true equality test.

▼ TS vs JS

- TypeScript is known as an Object-oriented programming language, whereas JavaScript is a prototype based language.
- TypeScript has a feature known as Static typing, but JavaScript does not support this feature.
- TypeScript supports Interfaces, but JavaScript does not.

▼ Pros & Cons of TS

Advantages of using TypeScript over JavaScript

- TypeScript always points out the compilation errors at the time of development (pre-compilation). Because of this, getting runtime errors is less likely, whereas JavaScript is an interpreted language.
- TypeScript supports static/strong typing. This means that type correctness can be checked at compile time. This feature is not available in JavaScript.
- TypeScript is same as JavaScript with some additional features such as ES6 features. It may not be supported in your target browser, but the TypeScript compiler can compile the **.ts** files into ES3, ES4, and ES5 also.

Disadvantages of using TypeScript over JavaScript

- Generally, TypeScript takes time to compile the code.

▼ Undefined vs Null

Null is a special value meaning "no value". Null is a special object because typeof null returns 'object'. On the other hand, **undefined** means that the variable has not been declared, or has not been given a value.

▼ URL Shortener

https://www.enjoyalgorithms.com/blog/design-a-url-shortening-service-like-tiny-url

▼ Encoding URL

To generate a unique short URL, we can compute it using the Unique Hash(MD5, SHA256, etc.) of the original URL and then encode using base62. If we use the MD5 algorithm as our hash function, it'll produce a 128-bit hash value. After base62 encoding, we'll get a string having more than seven characters. We can take the first 7 characters for the key.

▼ **System APIs**

We can use **REST APIs** to expose the functionality of our service. Below is an example of the definitions of the APIs for creating and deleting URLs without service:

Parameters

- `api_dev_key` (string): The API developer key of a registered account. This will be used to throttle users based on their allocated quota.
- `original_url` (string): Original URL to be shortened.
- `custom_alias` (string): Optional custom key for the URL.
- `user_name` (string): Optional username to be used in the encoding.
- `expire_date` (string): Optional expiration date for the shortened URL.

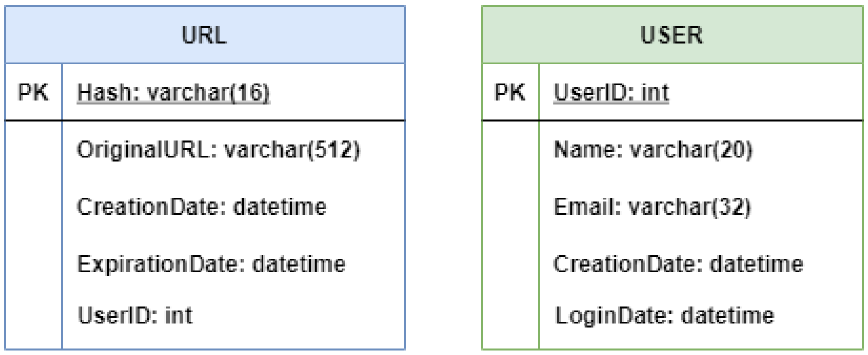
Returns (string): A successful insertion returns the shortened URL. Otherwise, it returns an error code.

The `url_key` is a string that delineates the shortened URL that needs to be deleted. A successful deletion will return. `URL Removed`.

▼ Table

While designing systems, we have two types of databases (of course, we don’t want to back to the old days where file systems were used): **Relational Databases** or **NoSQL.** For our system, relational queries will not be a large part of it/occur rarely. So, here we will go with NoSQL. **A NoSQL choice would be easier to scale.**

Database schema: We will need two main tables: 1 to store user information and 1 to store URL information.



	URL
PK	Hash: varchar(16)
	OriginalURL: varchar(512)
	CreationDate: datetime
	ExpirationDate: datetime
	UserID: int

	USER
PK	UserID: int
	Name: varchar(20)
	Email: varchar(32)
	CreationDate: datetime
	LoginDate: datetime

Scaling the service

Caching

We know that our database is going to be read heavily. We have found a way to speed up the writing process, but the reads are still slow. So we have to find some way to speed up the reading process. Let’s see.

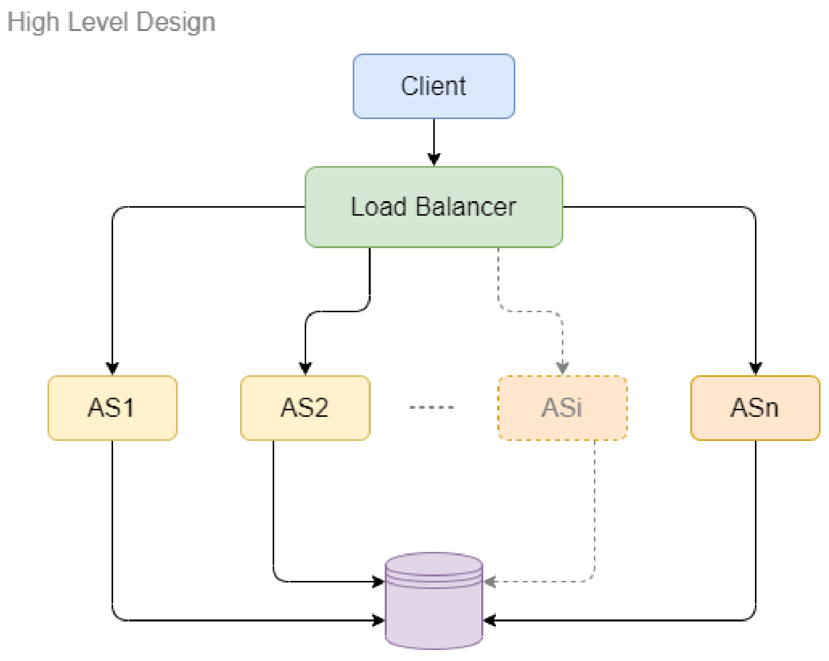
We can speed up our database reads by putting as much data in memory as possible, AKA a **cache**. This becomes important if we get massive load requests to a single link. If we have the redirect URL in memory, we can process those requests quickly. Our cache could store, let’s say, **20% of the most used URLs**. When the cache is full, we would want to replace a URL with a more popular one.

A **Least Recently Used (LRU)** cache eviction system would be a good choice.We can **shard** the cache too. This will help us store more data in memory because of the more machines. Deciding which thing goes to which shard can be done using “**Hashing”** or “**Consistent Hashing.”**

▼ Load Balancing

With multiple access systems like this one, a web server may not handle it yet. To solve this problem, I will use many web servers. Each web server will take a partial request from the user.

Since we have a large-scale system with multiple access happening simultaneously, a server may not handle the requests. To solve this problem, we can use multiple servers with a load balancer between “**the client and the server”** and “**the server and the database”** and avoid a single point of failure, and we will use multiple load balancers.



Initially, we can use a simple **Round Robin** approach that distributes incoming requests equally among backend servers. This would be the easiest to implement. A Round Robin LB does not consider server load, so our LB may still forward requests to an overloaded or slow server.

To distribute the load among servers, we will use the **Least Connection Method.** When a server is configured to use the least connection load balancing algorithm (or method), it selects the service with the fewest active connections.

▼ Var and Let

`var` and `let` are both used for variable declaration in JavaScript, but the difference between them is that `var` is function scoped and `let` is block scoped. It can be said that a variable declared with `var` is defined throughout the program, as compared to let.

`const` keyword declares blocked-scope variables. However, the block-scoped variables declared by the `const` keyword can't be **reassigned**.

▼

▼ What is the difference between Undefined, Undeclared, and Null in JavaScript?

var x

console.log(x) //Undefined variable

var y=NULL

console.log(y) //Null Variable

console.log(z) //Undeclared Variable

Undefined - Undefined means a variable has been declared but a value has not yet been assigned to that variable.

Null - Null is an assignment value that you can assign to any variable that is meant to contain no value.

Undeclared - Variables that are not declared or that do not exist in a program or application.

▼ What are Closures in JavaScript?

Closures in JavaScript are a feature where an inner function has access to the outer function's variables.

function outer_func()

{

var b =10;

```
function inner_func(){

var a =20;

console.log(a+b);

}

return inner;

}
```

A closure has three scope chains –

- Has access to the variable defined within its curly braces, which is its scope.
- Has access to the outer functions' variables.
- Has the ability to access global variables.

▼ What’s the difference between Function Declaration and Function Expression?

Function Declaration

```
function abc(){

return 5;

}
```

Within the main JavaScript code, it declares this as a separate statement. It is possible to invoke it before the function has been defined. It provides improved code readability.

Function Expression

```
var a = function abc(){

return

}
```

It is created inside an expression or some other construct. It is generally used when there is a need for a conditional declaration of a function.

▼ TS

TS is a strongly typed programming language that builds on JavaScript, giving you better tooling at any scale.

▼ Access Modifiers

access modifiers supported by TypeScript?

TypeScript supports access modifiers public, private and protected which determine the accessibility of a class member as given below:

- **Public** – All the members of the class, its child classes, and the instance of the class can access.
- **Protected** – All the members of the class and its child classes can access them. But the instance of the class can not access.
- **Private** – Only the members of the class can access them.

If an access modifier is not specified it is implicitly public as that matches the convenient nature of JavaScript.

▼ built-in data types in TypeScript.

In TypeScript, the built-in data types are also known as primitive data types and the list include:

- **Number:** This represents number type values. The numbers are stored as floating-point values in TypeScript.
- **String:** A string represents a sequence of characters stored as Unicode UTF-16 code.
- **Boolean:** This represents a logical value. When we use the Boolean type, we get the output only in true or false.
- **Null:** Null represents a variable whose value is undefined. It is not possible to directly reference the null type value itself.
- **Undefined**: The Undefined type denotes all uninitialized variables.
- **Void**: A void is the return type of the functions that do not return any type of value.

▼ Class

TypeScript introduced classes so that they can avail the benefits of object-oriented techniques like encapsulation and abstraction. The class in

TypeScript is compiled to plain JavaScript functions by the TypeScript compiler to work across platforms and browsers.

A class includes the following:

- **Constructor**
- **Properties**
- **Methods**

```
class Employee {
  empID: number;
  empName: string;

  constructor(ID: number, name: string) {
    this.empName = name;
    this.empID = ID;
  }

  getSalary() : number {
    return 40000;
  }
}
```

▼ Components

- **Language** − It comprises of the syntax, keywords, and type annotations.
- **The TypeScript Compiler** − This **compiler (tsc)** converts the instructions written in TypeScript to its JavaScript equivalent.
- **The TypeScript Language Service** − The Language Service exposes an additional layer around the core compiler pipeline, editor-like applications. The language service supports the common set of typical editor operations.

▼ Features

- **Cross-Platform:** The TypeScript compiler can be installed on any Operating System such as Windows, MacOS, and Linux.
- **Object-Oriented Language**: TypeScript provides features like Classes, Interfaces, and Modules. Thus, it can write object-oriented code for client-side as well as server-side development.

- **Static Type-Checking**: TypeScript uses static typing and helps type checking at compile time. Thus, you can find errors while writing the code without running the script.
- **Optional Static Typing**: TypeScript also allows optional static typing in case you are using the dynamic typing of JavaScript.
- **DOM Manipulation**: You can use TypeScript to manipulate the DOM for adding or removing elements.
- **ES 6 Features**: TypeScript includes most features of planned ECMAScript 2015 (ES 6, 7) such as class, interface, Arrow functions, etc.

▼ Function overloading?

Yes, TypeScript supports function overloading. But the implementation is odd. So, when you overload in TypeScript you only have one implementation with multiple signatures.

▼ Interface

The interface is a structure that defines the contract in your application. It defines the syntax for classes to follow. It contains only the declaration of the members and it is the responsibility of the deriving class to define the members. The TypeScript compiler uses interface for type-checking and checks whether the object has a specific structure or not.

▼ Internal module and the external module?

▼ Modules in TypeScript?

A module is a powerful way of creating a group of related variables, functions, classes, and interfaces, etc. It can be executed within its own scope, but not in the global scope. Basically, you cannot access the variables, functions, classes, and interfaces declared in a module outside the module directly.

A module can be created by using the **export** keyword and can be used in other modules by using the **import** keyword.

▼ Namespace in Typescript and how to declare it?

Namespace groups functionalities logically. These maintain the legacy code of typescript internally. It encapsulates the features and objects that share certain relationships. A namespace is also known as internal modules. A namespace can also include interfaces, classes, functions, and variables to support a group of related functionalities.

```
namespace <namespace_name> {
export interface I1 { }
export class c1{ }
}
```

▼ Optionally statically typed language?

TypeScript is referred to as optionally statically typed, which means you can ask the compiler to ignore the type of variable. Using any data type, we can assign any type of value to the variable. TypeScript will not give any error checking during compilation.

▼ Pros vs cons

▼ Benefit

- TypeScript is **fast, simple, easy to learn** and runs on any browser or JavaScript engine.
- It is **similar** to **JavaScript** and uses the same syntax and semantics.
- This helps backend developers write front-end **code faster**.
- You can call the TypeScript code from an **existing JavaScript code**. Also, it works with existing JavaScript frameworks and libraries without any issues.
- The Definition file, with .d.ts extension, provides support for existing JavaScript libraries like **Jquery, D3.js**, etc.
- It includes features from **ES6** and **ES7** that can run in ES5-level JavaScript engines like Node.js.

▼ Disadvantage

- TypeScript takes a **long time** to compile the code.
- It does not support **abstract classes**.
- If we run the TypeScript application in the browser, a **compilation step** is required to transform TypeScript into JavaScript.

- Web developers are using JavaScript for decades, and TypeScript doesn't bring anything new.
- To use any third party library, the **definition file** is a must.
- **Quality** of type definition files is a concern.

▼ Type vs Interface

▼ 1. **Flexibility**

Type in typescript is defined as type declaration for creating a variable name with a data type declared before the name where it can create names for a type such as primitive type declaration which includes a number, string, Boolean, null, etc and the type can also declare union, intersection and tuple type also.

Whereas interfaces are defined as a declaration of the only object type, which means the interfaces are restricted to only object type and do not support any other type for declaration. But we can say that interfaces have more capabilities than types in typescript. Hence in typescript, it is said types are more flexible than the interface.

▼ **2. Merging the declarations**

In Typescript, type does not support the feature of merging different multiple types, where the compiler cannot merge the two or more type declaration that have the same name as for global or module scope type is a unique type entity and if we try to merge the types then it would throw a compiler error saying duplicate identifier. Whereas, the interface supports this feature that is the interface can be defined various times and it can be then merged into a single interface which cannot be done with types.

▼ **3. Classes**

In typescript, type does not support the feature of implementing or extending union types in the class as we know that classes are static blueprints where the class cannot implement or extend that is it cannot exist in one or another class structure and if is done using types it will throw an error. Whereas, interface supports the implementation and extend feature in typescript and hence class can implement or extend interface using "implement" and "extend" keywords.

▼ **4. Few Aspects**

Type can describe functions, constructors, tuples but it cannot be implemented or extended in class, and also it cannot be augmented or recursive. Whereas, the interface can also describe functions, constructors, tuples, and also it can be implemented and extended in the class that also support the aspect that it can be augmented and also can be recursive when compared to types in Typescript.

▼ Variable

▼ Tailwind CSS

Tailwind CSS is a utility-first CSS library for rapidly building custom user interfaces. Tailwind enables you to write inline styling and achieve a fantastic user interface without leaving your HTML/JSX code and writing a single line of CSS.

Advantages of using Tailwind CSS

- Tailwind CSS saves time. By using tailwind we get access to several CSS classes, all we have to do is just add the classes directly into our JSX elements.
- Tailwind CSS solves the problem of naming conventions seamlessly by providing utility-based classes that can be used all the time.
- Tailwind CSS uses a default mobile-first approach and the availability of utility classes makes it easier to build complex responsive layouts freely.

Downsides of Using Tailwind CSS

- When writing Tailwind CSS, we have to write a lot of classes into our JSX and this can make our codebase very messy and hard to read and understand.
- Tailwind CSS doesn't provide default styled components like in Bootstrap or Bulma. You need to implement it from scratch.

▼ AJAX

AJAX, which stands for Asynchronous JavaScript and XML, brings together all the core front end developer technologies to communicate information between a client and a web server.

AJAX helps a client — the web browser a customer is using, for example — make a request to the server. Then, it allows the server to convey a response without forcing a reload. Because AJAX is asynchronous, the site doesn't freeze up until the server replies; instead, it works behind the scenes to process a JavaScript request and update the DOM to reflect the server's response to the user's requests.
Many of the most popular social media sites that drive web traffic today, like Facebook and Twitter, rely on simple AJAX-based interactions to show more content and updated information as users scroll up and down through their timelines. In most cases, AJAX is so firmly woven into the framework of popular websites that it illustrates how most users expect their sites to work: proactively and without requiring extra effort.

▼

▼ Common questions

▼ How are JavaScript and jQuery different?

JQuery is a library built with the JavaScript language, and JavaScript is the language itself.

▼ What is User Centered Design?

User-centered designs are those designs the designer solely focuses on which are according to the needs of users and them in every phase of designing.

▼ What is Polymorphism ?

The word Polymorphism means having many different forms. If we talk in object-oriented form, polymorphism refers to one interface, multiple functions.

▼ What is meant by the KISS principle?

KISS, a backronym of “Keep it simple, stupid”. Which was the principal design in the US Navy in 1960. The KISS principle states that the simpler the system the better it works.

▼ What does SOLID stand for?

S.O.L.I.D is an acronym of object-oriented design principles

S- single responsibility principle

O- open-closed principle

L- Liskov Substitution principle

I- interface segregation principle

D- dependency.

▼ **What is Stringify?**

To transform a JavaScript object to a string Stringify is used.

▼ **State the elements of the CSS Box Model.**

CSS Box Model consist of 4 elements

- Content
- Padding
- Border
- Margin

▼ **What is the benefit of Srcset?**

When we want to generate many new solutions of exact images on several devices, Srcset is used. This helps improve the UI.

▼ **What is MySQL?**

MySQL is a relational Database Management System which uses SQL as its standard language to manage its database. MySQL just like other databases uses a Table-like structure.

▼ **What is MongoDB?**

MongoDb is a NoSQL database which shows the data elements using JSON-like Structure. To make changes in MongoDB the programmer has to use MongoBD Query Language.

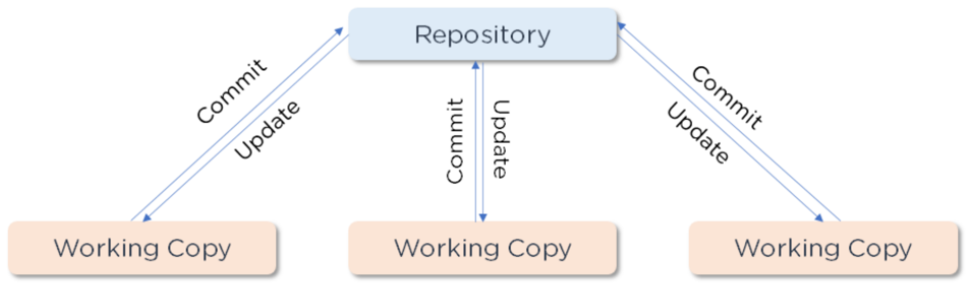
▼ **Git**

Git
is one of the most popular source-control systems that enable software development professionals in all industries, enabling multiple team members to work concurrently on projects. Since many users are

simultaneously working from different places on the same file, however, you may end up with a merge conflict.

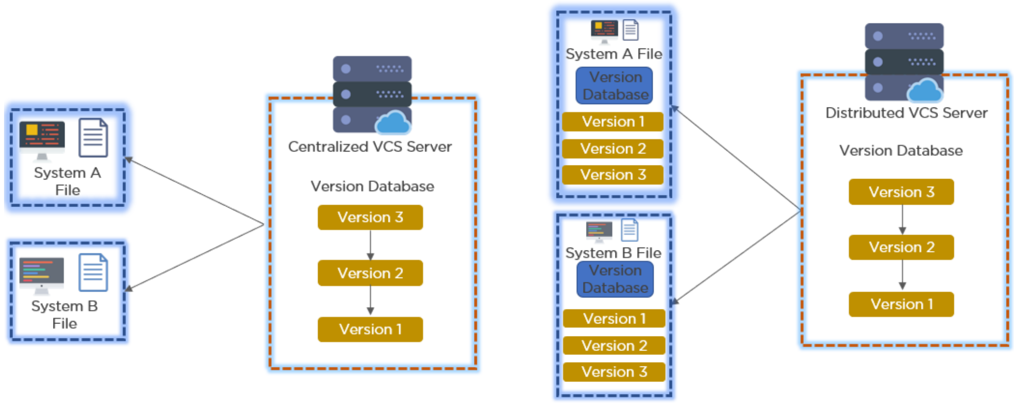
Git is an open-source, distributed version control system (VCS), which has a remote repository on the server-side and a local repository on the client-side. This means that the file or code is not present in a central server, but there is a copy of the file stored on the client's computer.

1. What is 'Version Control System'?



A version control system (VCS) is a program that records any changes to a file or set of data so that it is possible to restore it to a previous version if necessary. This guarantees that everyone on the team is working on the most up-to-date version of the file.

2. Differentiate Between Centralized and Distributed Version Control System



Centralized Version Control System *Distributed Version Control System*

In a Centralized Version Control System:

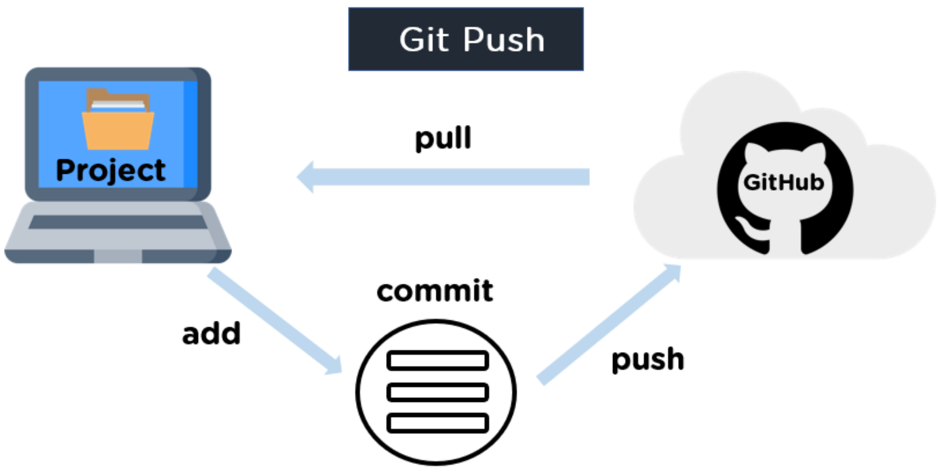
- It stores all file versions on a central server.
- No developer has a complete copy of the local system's files.
- If the project's central server fails, you will lose all the project's data.

In a Distributed Version Control System:

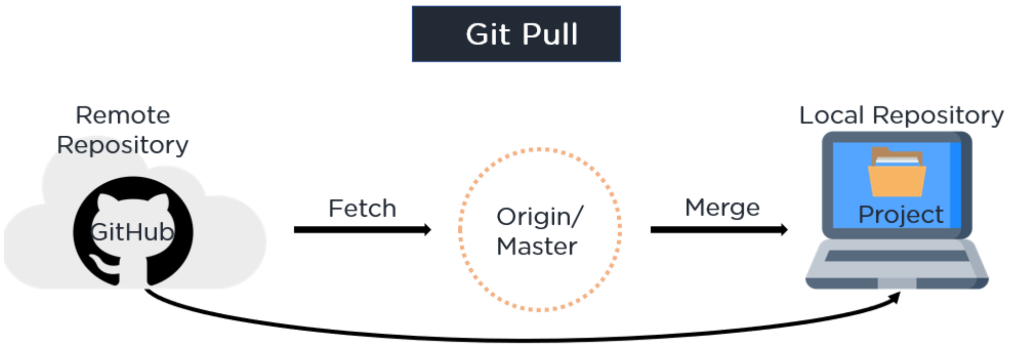
- Every developer has a copy of all the code versions on their computer.
- Improves the ability to work offline and eliminates the need for a single backup location.
- Even if the server crashes, there is no danger.

Full Stack Java Developer Course In Partnership with HIRIST and HackerEarth**EXPLORE COURSE**

3. Explain Git Push and Git Pull



Git push is a command that pushes the contents of a local repository to a remote repository. It runs a push after it has changed a local repository to share the changes with remote team members.



Git pull is a command that pulls changes from a remote repository and merges them into the local repository. It's made up of two commands: git fetch followed by git merge.

4. Name a few Git Commands and function

- Git Config - Configure the username and email address
- Git init - Initialize a local Git repository
- Git Add - Add one or more files to the staging area
- Git Diff - View the changes made to the file

- Git Commit - Commit changes to the head but not to the remote repository
- Git reset - Undo local changes to the state of a Git repo
- Git Status - Displays the state of the working directory and staging area
- Git Merge - Merge a branch into an active branch
- Git Push - Upload content from the local repository to a remote repository
- Git Pull - Fetch and download content from a remote repository

Also Read: <u>How to Become a Front End Developer?</u>

5. Explain the Difference Between Git Pull and Git Fetch

Git Fetch

- It downloads only new data from a remote repository using Git fetch
- It does not include any of this new information in your working files
- To update the remote-tracking branches, run Git fetch at any time
- Command - git fetch origin

git fetch –-all

Git Pull

- Git pulls new data and integrates it with the current working files, updating the current HEAD branch with the latest modifications from the remote server
- It attempts to combine remote modifications with those made locally
- Command - git pull origin master

FREE GIT Training Learn the basics of GIT**<u>ENROLL NOW</u>**

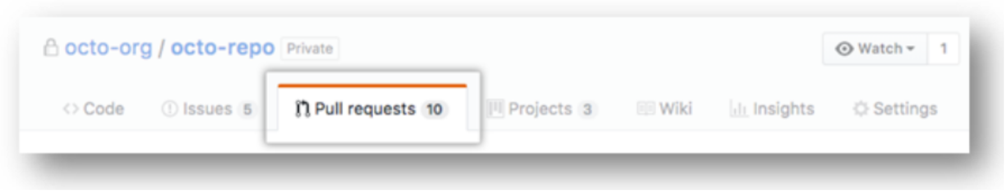
6. What is a Merge Conflict in Git and how can it be resolved?

When you have merging branches with opposing commits, a merge conflict occurs, and Git needs your help to select which changes to include in the final merge.

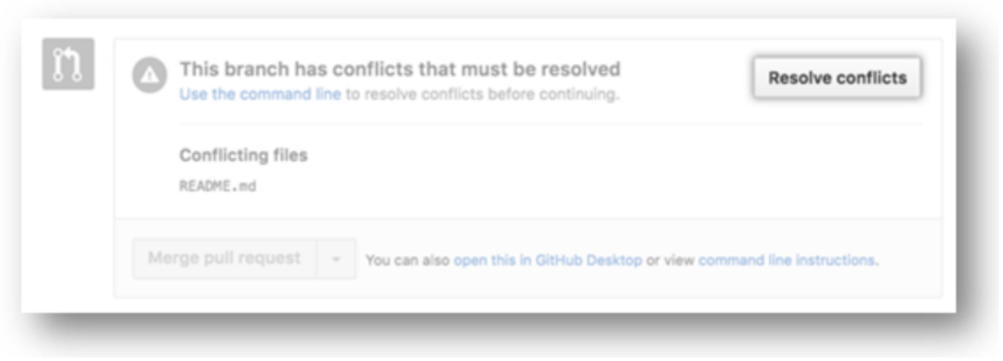
Resolve using GitHub conflict editor

When competing for line changes, i.e. when users make different modifications to the same line of the same file on different branches in your Git repository, this is done to avoid merging conflicts.

Step 1: Under your repository name, click Pull requests.



Step 2: Click the pull request with the merge conflict you'd like to resolve in the "Pull Requests" list. Click Resolve conflicts near the bottom of your pull request.



This branch has conflicts that must be resolved

Use the command line to resolve conflicts before continuing.

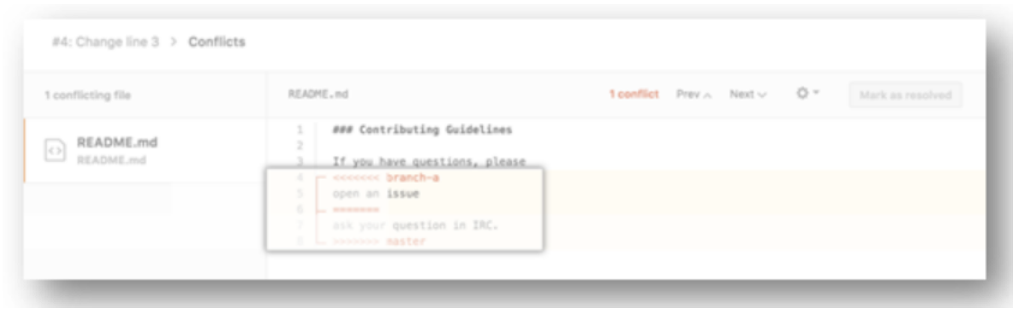
Conflicting files

README.md

You can also open this in GitHub Desktop or view command line instructions.

Step 3: Decide whether you want to maintain just your branch's changes, only the other branch's changes, or make a completely new modification that includes both branches' changes.

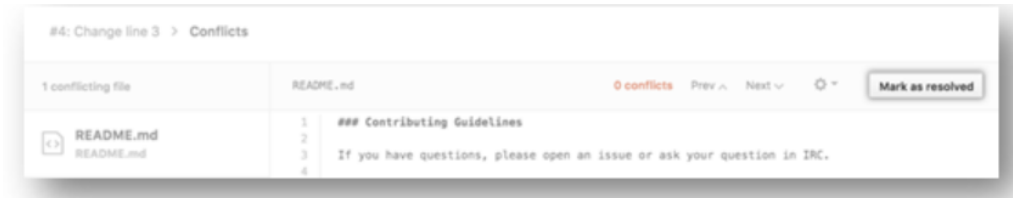
Step 4: Delete the conflict markers and make the changes you want in the final merge.



```
### Contributing Guidelines

If you have questions, please
<<<<<<< branch-a
open an issue
=======
ask your question in IRC.
>>>>>>> master
```

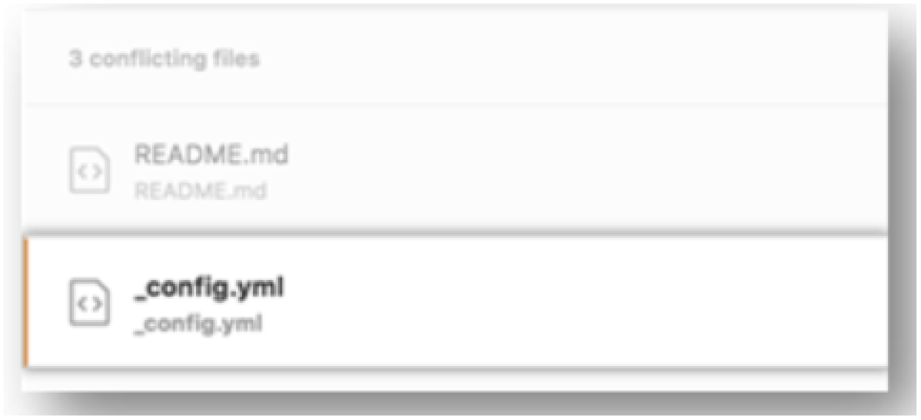
Step 5: If your file has over one merge conflict, scroll down to the next set of conflict markers and repeat steps four and five to resolve the issue. Mark the file as resolved once you've resolved all the conflicts.



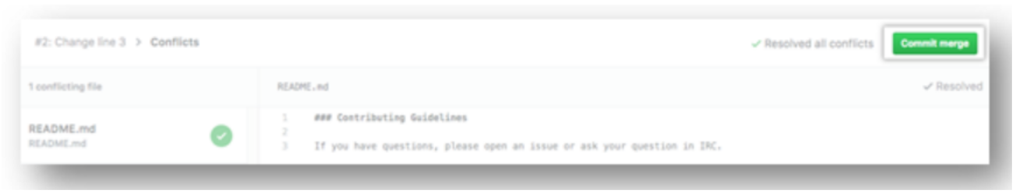
```
### Contributing Guidelines

If you have questions, please open an issue or ask your question in IRC.
```

Step 6: If you have more than one file with a conflict, go to the left side of the screen and select the next file you wish to edit under "conflicting files". Then repeat the above steps until you've resolved all the merge conflicts in your pull request.



Step 7: Click Commit merge once you've resolved all of your merge conflicts. It merges the entire base branch into your head branch as a result of this. Click Merge pull request to combine your pull requests.



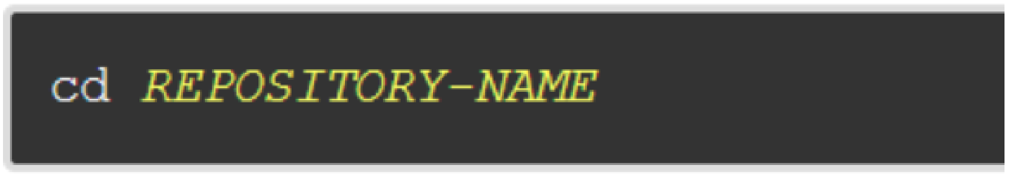
```
### Contributing Guidelines

If you have questions, please open an issue or ask your question in IRC.
```

Resolve the issue with a local clone of the repository and commit the update to your GitHub branch.

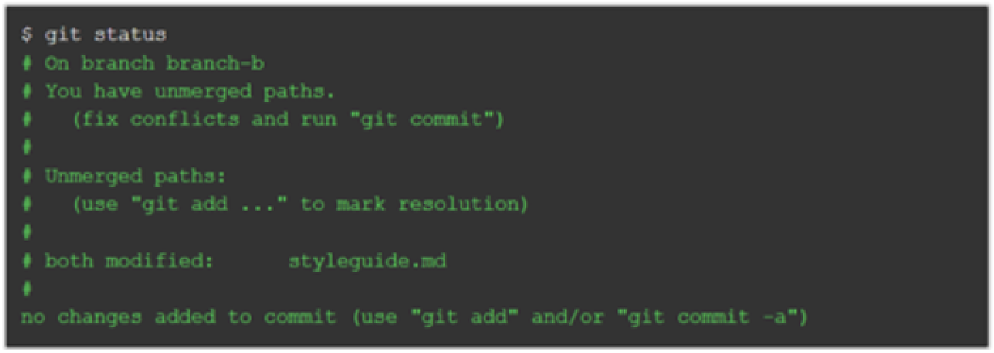
It resolves all other forms of merge conflicts using this method. To push the update, you can use the command line or a program like GitHub desktop.

Step1: Open Git Bash. Go to the local Git repository where the merge conflict exists.



```
cd REPOSITORY-NAME
```

Step2: Make a list of the files that have been affected by the merge dispute. In this case, there is a merge conflict in the file styleguide.md.



```
$ git status
# On branch branch-b
# You have unmerged paths.
#   (fix conflicts and run "git commit")
#
# Unmerged paths:
#   (use "git add ..." to mark resolution)
#
# both modified:      styleguide.md
#
no changes added to commit (use "git add" and/or "git commit -a")
```

Step 3: Navigate to the file with merge conflicts in any text editor, such as Sublime Text or Atom. Look for the conflict marker "<<<<<<" if you want to see where the merging conflict started in your file.

After the line "<<<<<<HEAD", you'll see the changes from the base branch.

Step 4: Next you'll see =======, which divides your changes from the changes in the other branch, followed by >>>>>>> BRANCH-NAME

Step 5: Decide whether you want to simplt maintain your branch's changes, only the other branch's changes, or make a completely new modification that includes both branches' changes.

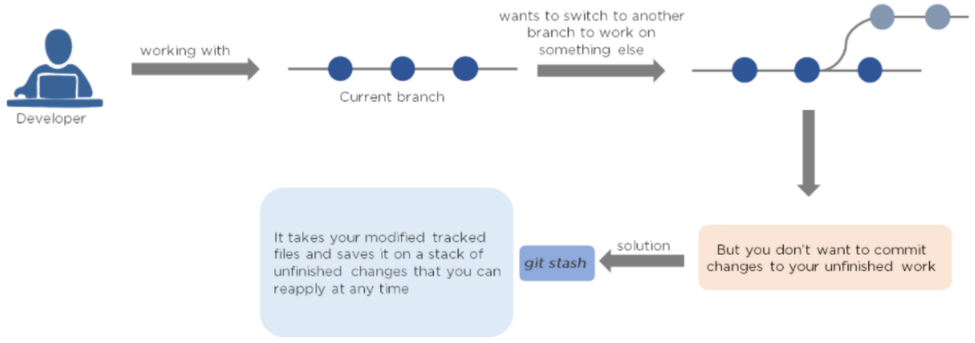
Step 6: Delete the conflict markers, <<<<<, =====, >>>>> and make changes you want in the final merge.

Step 7: Add or stage your changes. Commit your changes with a comment.

You may now combine the branches using the command line, or you can upload your changes to your GitHub remote repository and merge them in a pull request.

Also Read: Git Rebase vs. Merge

7. What is Git Stash?

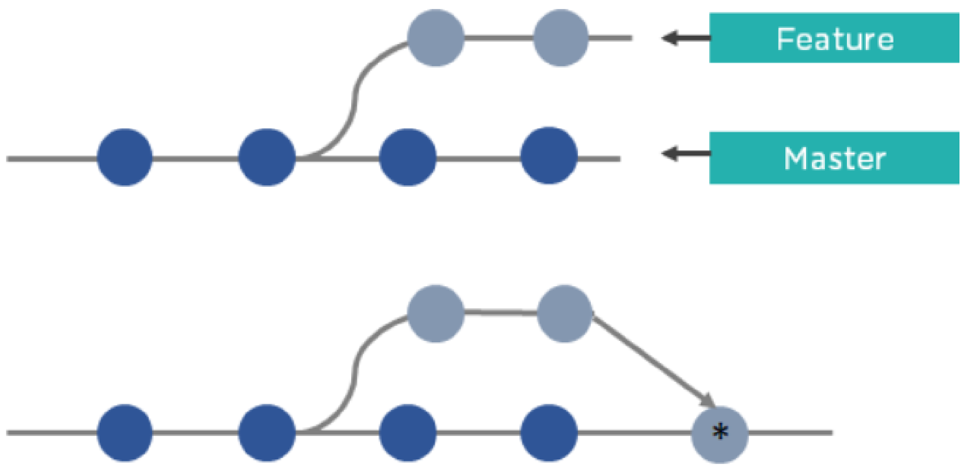

Developer
working with
Current branch
wants to switch to another branch to work on something else
But you don't want to commit changes to your unfinished work
solution
git stash
It takes your modified tracked files and saves it on a stack of unfinished changes that you can reapply at any time

FREE Java Certification Training Learn A-Z of Java like never before**ENROLL NOW**

8. Differentiate Between Git Merge and Git Rebase

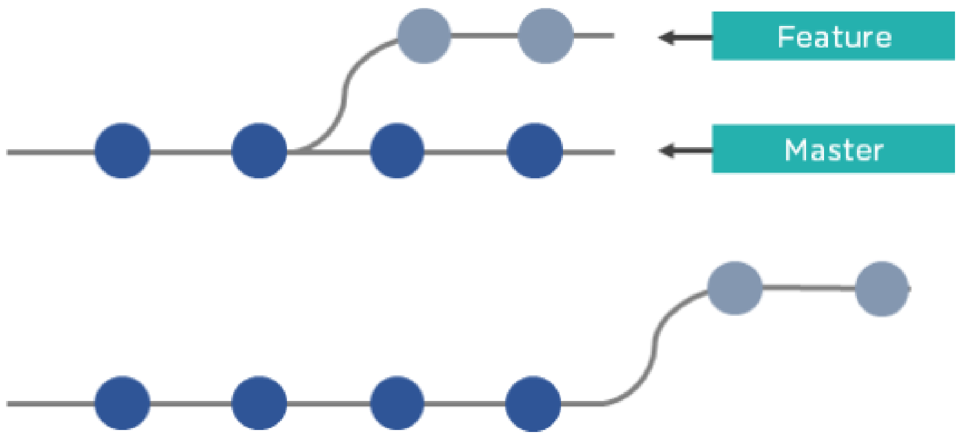
Assume you're working on a new feature in a dedicated branch, and another team member pushes new commits to the master branch.

Merge



Merge is used to incorporate new commits into your feature branch. Every time you need to include modifications, this requires an extra merging commit. It taints the history of your feature branch.

Rebase



You can rebase the feature branch into master instead of merging it. This incorporates all the new commits in the master branch. It also re-writes the project history by creating brand new commits for each commit in the original branch.

This brings us to the end of Git Questions. These are some of the most important Front End Interview Questions related to Version Control.

▼

▼

▼

▼

▼

▼

▼

▼ Angular

▼ Angular filter types

Currency;

Filter;

Date;

Lowercase;

Uppercase;

orderBy;

JSON;

Number;

LimitTo.

▼ AngularJS to Angular 2

Both are front-end frameworks maintained by Google, but Angular 2 is not a simple update of AngularJS, it is a new framework written from scratch. Updating an app from AngularJS to Angular 2 would require a complete rewrite of the code.

▼ AOT Compilation

AOT refers to Ahead-of-time compilation. In Angular, it means that the code you write for your application is compiled at build time before the application is run in a browser. It‘s an alternative to Just-in-time compilation, where code is compiled just before it is run in the browser. AOT compilation can lead to better application performance.

▼ **AsyncPipe**

You can use “AsyncPipe” to subscribe to an observable or promise. It can then return the latest value emitted by the promise or observable. “AsyncPipe” marks an Angular component that has been checked for new values emitted.

▼ Benefits of Angular

- It supports the popular “Model-View-Controller” (MVC) architecture. You can isolate the application logic from the UI layer, which is important for ensuring the separation of concerns.
- The architecture of Angular helps developers to locate the code that requires changes, which helps in development.
- Angular supports modules, which helps developers.
- Thanks to “Angular Injector”, Angular supports services and dependency injection. This is another way in which it supports developers.
- Angular uses “TypeScript”, a superset of JavaScript. It eliminates common programming errors, resulting in better-quality code.
- Programmers can use the comprehensive documentation of Angular.

▼ **basic parts of an Angular application**

Modules, Component, Property Binding, Template, Structural Directives, Dependency Injection, Services, Routing.

▼ **Change Detection**

Change Detection is the process of synchronizing a model with a view. In Angular, the flow of information is unidirectional, even when using the ng Model to implement two-way binding, which is syntactic sugar on top of a unidirectional flow.

Change Detection Mechanism-moves only forward and never looks back, starting from the root (root) component to the last. This is the meaning of one-way data flow. The architecture of an Angular application is very simple — the tree of components. Each component points to a child, but the child does not point to a parent. One-way flow eliminates the need for a $digest loop.

▼ CLI

You can use the Angular Command Line Interface (Angular CLI) for activities like adding components, deploying components, testing, and many more activities.

▼ **constructor vs. ngOnInit**

The constructor is a feature of the class itself, not Angular. The main difference is that Angular will launch ngOnInit after it has finished configuring the component. Meaning, it is a signal through which the @Input() and other banding properties and decorated properties are available in ngOnInit, but are not defined within the constructor by design

▼ **Core and Shared modules**

A *Shared module* serves as a generic module for all modules, components, directives, pipes, etc., which are not required to be in a single copy for the application but need to be imported into many different modules.

A *Core module* is a place to store services that you need to have in the form of singleton for the entire application (for example, a user authorization service with data storage about it).

▼ **data binding types categories**

From-the-source-to-view: This uses one-way data binding.From-view-to-source: It utilizes one-way data binding.View-to-source-to-view: This category uses two-way data binding.

▼ **digest cycle process**

The digest cycle process in Angular helps to monitor the watch list. This process tracks changes in the watch variable value. You can use it to compare multiple values. The process includes a comparison between the previous and present versions of the scope model values.

▼ **directive create steps**

1. Use the following command to start a new project:

‘ng new [application-name]’

2. Change the directory into a new directory by using the following command:

‘cd [application-name]’

3. Generate a new directive by issuing the following command:

‘ng generate directive [path-to-directives/test-directive]’.

▼ **HTTP Interceptors**

Interceptor is just a fancy word for a function that receives requests/responses before they are processed/sent to the server. You should use interceptors if you want to pre-process many types of requests in one way. For example, you need to set the authorization header Bearer for all requests:

▼ **interact between Parent and Child components**

When passing data from Parent to Child component, you can use the *@Input* decorator in the Child component. When passing data from Child to Parent component, you can use the *@Output* decorator in the Child component.

▼ **lazy loading**

Lazy loading of modules is needed to break the code into pieces. When downloading the app in the browser, it doesn’t load all the application code. During the transition to the route with lazy loading, the module has to load the code into a browser.

Example for using lazy loading modules:

{ path: ‘example’, loadChildren: ‘./example/example.module#ExampleModule’, component: PublicComponent },

▼ **Lifecycle Hooks**

OnChange() – OnInit() – DoCheck() – AfterContentInit() – AfterContentChecked() - AfterViewInit() – AfterViewChecked() – OnDestroy().

▼ **multicasting**

Assume that you want to fetch data by communicating with a backend service by using the HttpClient module. You want to broadcast the data to multiple subscribers. Assume that you want to do all of these tasks in one execution. “Multicasting” refers to responding to multiple subscribers with data.

▼ **ng-content**

”ng-content” in Angular inserts content dynamically inside components. You can reuse components better. “ng-content” passes content inside the component selector.

▼ **ngModel**

Angular NgModel is **an inbuilt directive that creates a FormControl instance from the domain model and binds it to a form control element**. The ngmodel directive binds the value of HTML controls (input, select, textarea) to application data. We can merely achieve it in the component element and the HTML element both.

we should import **FormsModule** and **Reactiveforms** Module to use **ngModel** in Anglar

▼ **optimizing an Angular 6**

AOT compilation, bundling and uglifying the application, tree shaking, lazy loading, separating dependencies and devDependencies, Using OnPush and TrackBy, removing unnecessary 3rd party libraries and import statements, avoid computing values within the template.

▼ **PipeTransform interface**

The PipeTransform interface receives an input value. It then uses the “transform()” method to transform the input value into the desired format.

▼ **promises and observables**

Promises emit a single value. However, observables emit multiple values over time.

Promises aren’t lazy, however, observables are lazy. Observables are called only when there are subscriptions to them.

You can’t cancel a promise. However, the “unsubscribe()” method allows you to cancel observables.

▼ **router events**

NavigationStart;

RouteConfigLoadStart;

RouteConfigLoadEnd;

RoutesRecognized;

GuardsCheckStart;

ChildActivationStart;

ActivationStart;

GuardsCheckEnd;

ResolveStart;

ResolveEnd;

ActivationEnd;

ChildActivationEnd;

NavigationEnd;

NavigationCancel;

NavigationError.

▼ **RouterOutlet vs. RouterLink**

RouterOutlets are directives from the router library. RouterOutlets are placeholders. They are the spots in templates where the router displays the components for that outlet. RouterOutlets are components.

RouterLinks are directives on the anchor tags. They give the router control over the tags. Developers can assign string values to RouterLink directives.

▼ **RouterState**

The RouterState in Angular is a tree of activated routes. Nodes in this structure know the “consumed” URL segment. These nodes also know about the extracted parameter and resolved data. One can use the router service and routerState property to access the current RouterState.

▼ **secure an Angular application**

- Check that all requests come from within your own web app and not external websites
- Sanitize all input data
- Use Angular template instead of DOM APIs
- Content Security Policies
- Validate all data with server-side code
- Compile using an offline template compiler
- Avoid including external URLs in your application
- Make JSON responses non-executable
- Keep all libraries and frameworks up-to-date

▼ **Transpiling in Angular**

Transpiling means converting the source code of one programming language into another. In Angular, that usually means converting TypeScript into JavaScript. You can write the code for your Angular application in TypeScript (or another language such as Dart) that is then transpiled to JavaScript for the application. This happens internally and automatically.

▼ **update the view if your data model is updated outside the 'Zone'**

1. Using the ApplicationRef.prototype.tick method, which will run change detection on the entire component tree.
2. Using NgZone.prototype.run method, which will also run change detection on the entire tree. The run method under the hood itself calls tick, and the parameter takes the function you want to perform before tick.
3. Using the ChangeDetectorRef.prototype.detectChanges method, which will launch change detection on the current component and its children.

▼

▼ Back end

▼ GraphQL

What is GraphQL

GraphQL is a query language for APIs and a runtime for fulfilling those queries with your existing data. GraphQL provides a complete and understandable description of the data in your API, gives clients the power to ask for exactly what they need and nothing more, makes it easier to evolve APIs over time, and enables powerful developer tools.

GraphQL vs. REST

REST has been a popular way to expose data from a server. REST was a good fit for many applications. However, the API landscape has radically changed over the last couple of years. In particular, there are three factors that have been challenging the way APIs are designed.

1. Increased mobile usage creates need for efficient data loading. So Facebook developed GraphQL. GraphQL minimizes the amount of data that needs to be transferred over the network and thus majorly improves applications operating under these conditions.
2. Variety of different frontend frameworks and platforms. It makes difficult to build and maintain one API that would fit the requirements of all. But, with GraphQL, each client can access precisely the data it needs.
3. Fast development & expectation for rapid feature development.

REST offers some great ideas, such as stateless servers and structured access to resources. However, REST APIs have shown to be too inflexible to keep up with the rapidly changing requirements of the clients that access them.

But GraphQL was developed to cope with the need for more flexibility and efficiency. It solves many of the shortcomings and inefficiencies that developers experience when interacting with REST APIs.

For example, in a blogging application, an app needs to display the titles of the posts of a specific user. Also an app needs to display the names of last 5 followers of that user. To do this, with a REST API, we would typically gather the data by accessing multiple endpoints. But, on the other hand, with

GraphQL, we can simply send a single query to the GraphQL server that includes the concrete data requirements.

better things:

Data fetching with Rest vs. GraphQL.

No more Overfetching and Underfetching.

Rapid product iterations on the frontend

insightful analytics on the backend

Benefits of a schema & type system: GraphQL uses a strong type system to define the capabilities of an API. All the types that are exposed in an API are written down in a schema using the GraphQL Schema Definition Language (SDL). This schema serves as the contract between the client and the server to define how a client can access the data. Once the schema is defined, the teams working on frontend and backends can do their work without further communication since they both are aware of the definite structure of the data that’s sent over the network. Frontend teams can easily test their applications by mocking the required data structures. Once the server is ready, the switch can be flipped for the client apps to load the data from the actual API

---

explain

API defines how a client can load data from a server.

GraphQL enables declarative data fetching where a client can specify exactly what data it needs from an API. Instead of multiple endpoints that return fixed data structures, a GraphQL server only exposes a single endpoint and responds with precisely the data a client asked for.

As you know, most applications today have the need to fetch data from a server where the data is stored in a database. So, it’s the responsibility of the API to provide an interface to the stored data that fits an application’s needs.

It has three types like Query for fetching data, Mutation for creating new data, updating existing data, deleting existing data, and Subscription for real-time updates.

▼ MySQL

MySQL is a popular system that manages the database structures that websites use to organize their information. Content management systems like WordPress, for example, often rely on a MySQL database to store and relay information to users, including all of the posts, articles, and user data on the website.

Though there are several database options available to front end developers, MySQL — developed and supported by Oracle — is one of the most popular open-source relational database management systems used across the web. As a relational database, MySQL uses an array of tables to define and organize information while retaining a clear record of its connections and relationships.

MySQL is primarily a back end developer tool because it is often part of the foundation on which back end developers construct the user experience. However, interaction with the back end database is one of the most important responsibilities for any front end developer. When you write an app for use on the front end, it is built on the structure that stores, retrieves, edits, and saves data. Every search and user login on the front end of your website may well involve interacting with the MySQL database, so you need to know and understand how the database works and how best to interact with it to deliver a smooth, seamless user experience.

▼

▼ Services

▼

▼ AWS

What is AWS?

Amazon Web Services (AWS) is the largest cloud computing platform, offering 200+ universally featured resources, from infrastructure to machine learning. These combinable systems provide maximum usability and are designed expressly for the optimization of your

application's performance through content delivery features, data storage, and more.

With AWS, you pay only for the exact amount of assistance you require, resulting in lower capital commitment and enhanced time-to-value without compromising productivity.

Why use AWS services?

Designed around ease-of-use for various skill sets, AWS is tailored for those unaccustomed to software development utilities. Web applications can be deployed in minutes with AWS facilities, without provisioning servers or writing additional code.

Amazon hosts global data centers with a vast network ensuring reduced latency worldwide. AWS' replication capacity allows us to duplicate services regionally, helping us recover quickly and avoid costly downtime.

Authentication and authorization workflows in AWS cognito

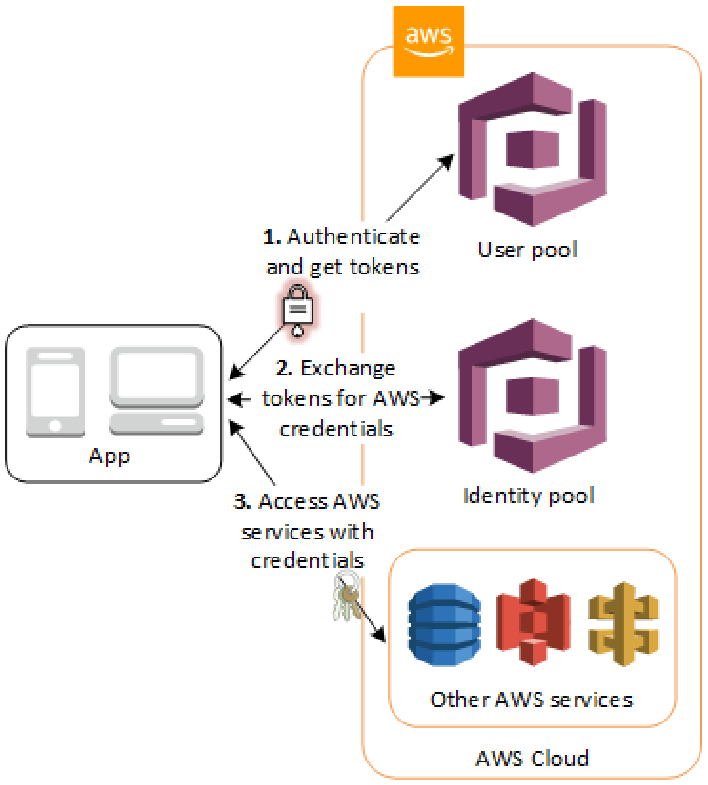
AWS Cognito provides an authentication service for applications. It is serverless. You don't need to manage any database or servers to handle user data and authentication flows. Cognito can be leveraged to handle those tasks.

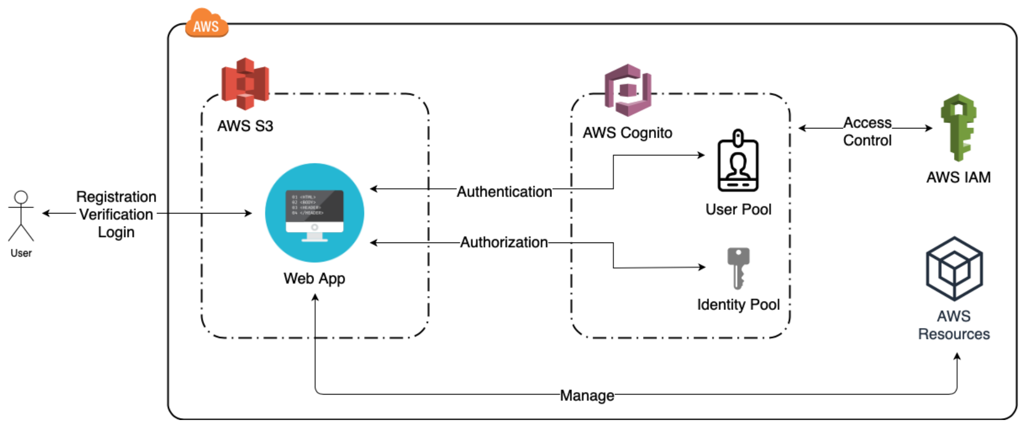
AWS Cognito provides the ability for us to configure our own identity (Authentication) provider. Which means the control of the user sign up, sign in, password management and many more user management features are in our hands.

- **User Registration** User enters email, username and password and registers with the User Pool.
- **User Verification** AWS Cognito User Pool will send verification code by email or sms and the user enters the code to get verified with the User Pool.
- **User Login** User enters username and password and logs in with Cognito User Pool in which case a token will be provided by Cognito upon successful login.
- **Get Temporary Credentials** Cognito Identity Pool will provide temporary credentials to AWS resources using the token that was received on successful login.

- **User Authorization** Cognito will authorize the user with necessary permissions with IAM role.
- **AWS Resource Management** Authorized user will now have the ability to manage AWS resources according to the permissions given by AWS IAM.

  https://docs.aws.amazon.com/cognito/latest/developerguide/amazon-cognito-user-pools-authentication-flow.html


aws
1. Authenticate and get tokens
User pool
2. Exchange tokens for AWS credentials
App
Identity pool
3. Access AWS services with credentials
Other AWS services
AWS Cloud



List of AWS services https://allcode.com/top-aws-services/

1. Amazon EC2 (Elastic Compute Cloud)

   EC2 is a cloud platform provided by Amazon that offers secure, and resizable compute capacity. Its purpose is to enable easy access and usability to developers for web-scale cloud computing, while allowing for total control of your compute resources.

   Deploy applications rapidly without the need for investing in hardware upfront; all the while able to launch virtual servers as-needed and at scale.

2. Amazon RDS (Relational Database Services)

   Amazon Relational Database Service (Amazon RDS) makes database configuration, management, and scaling easy in the cloud. Automate tedious tasks such as hardware provisioning, database arrangement, patching, and backups – cost-effectively and proportionate to your needs.

   RDS is available on various database instances which are optimized for performance and memory, providing six familiar database engines including Amazon Aurora, PostgreSQL, MySQL, MariaDB, Oracle. database, and SQL server. By leveraging the AWS Database Migration Service, you can easily migrate or reproduce your existing databases to Amazon RDS.

   Benefits of Amazon RDS

   Flexibility: Choose from a variety of database engines, such as open source solutions like MySQL and PostgreSQL.

   Cost Savings: Save money and pay for only the resources used, with no initial investment.

   Scalability: Scale your database, compute, and storage resources up and down with a few clicks or an API request.

3. Amazon S3 (Simple Storage Service)

   Amazon S3, at its core, facilitates object storage, providing leading scalability, data availability, security, and performance. Businesses of vast sizes can leverage S3 for storage and protect large sums of data for various use cases, such as websites, applications, backup, and more.

Amazon S3's intuitive management features enable the frictionless organization of data and configurable access controls.

https://allcode.com/amazon-web-services/aws-s3/

4. Amazon Lambda

   Lambda permits you to run code without owning or managing servers. Users only pay for the compute time consumed.

   Operate code for nearly any application or backend utility without administration. Users just upload the code, and Lambda does the rest, which provides precise software scaling and extensive availability.

   https://allcode.com/amazon-web-services/aws-lambda-serverless-compute-guide/

5. Dynamo DB

   DynamoDB is a document database with key-value structuring that delivers single-digit millisecond performance at scale. Dynamo has built-in security with a fully managed, multimaster, multiregion, durable database, backup and restore, and in-memory archiving for web-scale applications.

   DynamoDB can manage upward of 10 trillion requests daily and can support thresholds of more than 20 million requests per second.

   https://aws.amazon.com/dynamodb/

RDS, SimpleDB, DynamoDB

▼ Cloud Databases

1 – Amazon Web Service (AWS)

Amazon has become the market leader in the DBaaS space. It offers supplementary data-management services such as Redshift, a data warehouse and Data Pipeline, which is a data integrating service for easier data management. Amazon's current offerings include:

- Amazon RDS – Amazon's Relational Database Service runs on either Oracle, SQL or MySQL server instances.
- Amazon SimpleDB – This is primarily a schema-less database that is meant to handle smaller workloads.

- Amazon DynamoDB – This falls on the NoSQL databases (SSD), capable of automatically replicating workloads across three availability zones.

Strengths: Lots of Features, Easy to Use, Good Support and Documentation **Weaknesses**: Not Too Customizable, Downtimes as per Amazon's Schedule

2 – Oracle Database

Oracle Database provides companies with enterprise-scale database technology stored in the cloud. Despite its first offering being quite comprehensive, the Generation 2 offering has consistently higher performance with extensive governance and security controls.

Data migration is also covered with a dedicated solution and tight customer support in case any technical issues or questions arise.

Strengths: Intuitive Interface, Easy to Use, Solid Customer Support**Weaknesses:** No Free Version, No Mobile Access, Pricey for Small Companies

3 – Microsoft Azure

In a nutshell, Azure is a cloud computing platform for VM creation, building and running web-based applications, smart client applications, and XML web services. It currently boasts the biggest and strongest global infrastructure, with 55 regions, more than any other cloud provider.

A big point that needs to be considered is that Microsoft arguable offers the biggest range of software that a modern company needs today. This can allow you to create a huge ecosystem that has the same roots, with just one place to go with your questions or issues, if any may arise.

Strengths: Comprehensive Solution, Good Security, Strong Ecosystem**Weaknesses:** Iffy Customer Service, Not User Friendly

4 – Google Cloud Platform

Surprisingly, Google is still playing catch-up with the big players in the market. But its solutions are being adopted by more and more businesses of different sizes, thanks to its no-nonsense approach and comprehensive documentation which reduces stress on developers, IT professionals, and other stakeholders.

The broad open-source compatibility also has its fair share of benefits, allowing you to scale while doing more with analytics and integrations.

Strengths: Comprehensive Documentation, Good for Small and Big Businesses**Weaknesses:** Not Yet at the Level of the Big Three (AWS, Oracle, Azure)

5 – IBM DB2

This is a relational database that delivers advanced data management and analytics capabilities for transactional and warehousing workloads. IBM DB2 is designed to deliver high performance, actionable insights, data availability and reliability, and it is supported across Linux, Unix, and Windows.

However, it has fewer regional options, which may impact performance and compliance requirements depending on your development project/s.

Strengths: Well Designed Product, Easy Migration Process**Weaknesses:** Average Customer Service, Pricey, Mediocre Functionality

6 – MongoDB Atlas

MongoDB Atlas is a popular open-source NoSQL database that offers powerful scaling, sharding, and automation capabilities. Another advantage is that most developers using this can speed through continuous delivery models without any database administrator (DBA) hand holding.

On the negative side, some applications require SQL databases to function, which automatically eliminates MongoDB Atlas from consideration.

Strengths: Strong Support Community, Quick Installation, Flexibility**Weaknesses:** NoSQL Only, Can be Challenging for New/Inexperienced Devs

7 – OpenStack

Another interesting open-source rival for Google is OpenStack. These databases come in managed or hosted cloud databases. Rackspace is highly customizable and its architecture is easy to understand and implement. Many reviews have complimented the scaling capabilities of this solution.

The OpenStack community collaborates around a six-month, time-based release cycle with frequent development milestones.

Strengths: Good Value for Money, Easy to Use**Weaknesses:** Cumbersome Interface, Some Stability Issues

▼

▼ MMy work culture

-collaborate with other developers (what means)
-adaptable to any situations
-work well under pressure
-thrive challenge
-team leader experience, problems, main role, events, best behavior
-passionate about delivering high quality, creating quality project
-innovative results
-creative
-detail-oriented -> strong attention to detail
-goal-oriented
-problem solving skills (google what is it)

▼ TTesting

▼ List

Unit Testing

During this first round of testing, the program is submitted to assessments that focus on specific units or components of the software to determine whether each one is fully functional. The main aim of this endeavor is to determine whether the application functions as designed. In this phase, **a unit can refer to a function, individual program or even a procedure**, and a White-box Testing method is usually used to get the job done. One of the biggest benefits of this testing phase is that it can be run every time a piece of code is changed, allowing issues to be resolved as quickly as possible. It’s quite common for software developers to perform unit tests before delivering software to testers for formal testing.

Integration Testing

Integration testing allows individuals the opportunity to combine all of the units within a program and test them as a group. This testing level is designed to **find interface defects between the modules/functions**. This is particularly beneficial because it determines how efficiently the units are running together. Keep in mind that no matter how efficiently each unit is running, if they aren't properly integrated, it will affect the functionality of the software program. In order to run these types of tests, individuals can make use of various testing methods, but the specific method that will be used to get the job done will depend greatly on the way in which the units are defined.

System Testing

System testing is the first level in which **the complete application is tested as a whole**. The goal at this level is to evaluate whether the system has complied with all of the outlined requirements and to see that it meets Quality Standards. System testing is undertaken by independent testers who haven't played a role in developing the program. This testing is performed in an environment that closely mirrors production. System Testing is very important because it verifies that the application meets the technical, functional, and business requirements that were set by the customer.

Acceptance Testing

The final level, Acceptance testing (or User Acceptance Testing), is conducted to **determine whether the system is ready for release**. During the Software development life cycle, requirements changes can sometimes be misinterpreted in a fashion that does not meet the intended needs of the users. During this final phase, the user will test the system to find out whether the application meets their business' needs. Once this process has been completed and the software has passed, the program will then be delivered to production.

▼ Unit testing

What's unit tests?

Unit tests is a technique where we can test a unit - the smallest piece of code that can be logically isolated in an application. It can be a function, a class method, a subroutine, etc. In my case, I have been using this to test functions and React components most of the time.

Why unit testing?

As I said before, unit tests are our best friend. It gives to us confidence to make changes and refactors without worrying if our application it's still working or not. In addition, it ensures to us that our code works properly, avoiding bugs and hours of debugging.

Jest

Jest is a JavaScript/TypeScript framework for testing, it runs the test, check coverage, allow mocks and more. Jest is Awesome.

React Testing Library

React Testing Library is a library that gives you superpowers to test your React Apps. Jest it's framework-agnostic, it doesn't care about which framework you use, so you need to use React Testing Library to handle react for us and made our life easier.

▼ Docker

Why should we use the Docker?

https://www.geeksforgeeks.org/why-should-you-use-docker-7-major-reasons/#:~:text=As Docker reduces the need,deployment process more cost-effective.

1. Consistent & Isolated Environment
2. Rapid Application Deployment
3. Ensures Scalability & Flexibility
4. Better Portability
5. Cost-Effective
6. In-Built Version Control System
7. Security

▼

▼

▼ Coding interview

Note:::::When given a coding interview question, I should start by asking clarifying questions and discussing a few approaches with their interviewers.

1. Visualize the problem by drawing it out.
2. Think about how I would solve the problem by hand.
3. Come up with more examples.
4. Break the question down into smaller independent parts.

▼

▼ Final Round (cultural interview)

▼ Company culture

Types of company culture

While work culture can vary among companies, the following are some of the most common culture types you may see when searching for a job:

Team-first company culture

Organizations with a team-first work culture typically focus on employee engagement. They provide opportunities for meaningful feedback and social gatherings and are flexible enough to accommodate their employees' personal lives. For example, an organization that offers an extended family leave or frequently organizes team outings may have a team-first company culture. A company with a team-first culture typically looks for people with values and goals that align with their own before considering experience and skills. This can help make your work environment more positive and comfortable.

Conventional or hierarchical company culture

Traditional companies often adopt this kind of company culture. These organizations typically have complex structures and decisive management teams. Companies with hierarchical cultures have a clear chain of command that separates employers and employees. Employees perform their duties under supervision, which can increase efficiency. Hierarchical work culture is common in law firms, restaurants, banks, and organizations with strict rules for most teams and departments. In addition, they often have dress codes for employees to follow. This culture has guidelines that often make it risk-averse and stable.

Elite company culture

A company with an elite work culture may hire individuals who are innovative, confident, and talented at developing ideas that can improve their productivity. Companies with this culture often grow quickly in their industry. This corporate culture is result-oriented and competitive. Employees in this work environment aim to reach targets, meet quotas, and achieve results. If your workplace has an elite company culture, you may work for long hours and prioritize tasks. This can help ensure the profitability and success of an organization.

Related: **What Is Corporate Culture?**

Horizontal company culture

This workplace culture is common among new companies and startups because it encourages a collaborative environment that allows employees to pitch ideas that can improve the company. Organizations with this culture typically have a service or product that's flexible or easy to change based on customer feedback and market research. In horizontal company cultures, each employee is typically passionate about the company's goals. This kind of culture can improve teamwork and may attract employees with diverse skillsets who can perform a variety of tasks.

Progressive company culture

Companies that have experienced a merger, market change, new management, buyout, or recent transition to achieve an advanced goal may have a progressive culture. This work environment can allow employees to have open discussions about the company and its competitors. Organizations with this culture often have increased output and may get most of their income from grants, advertisements, and donations. Employees in a progressive culture typically welcome new ideas and can adapt to change quickly.

▼ Do you have any questions for me? (CEO)

::: company’s goals and the role at hand.

- If I were hired for this role, what would you want me to achieve in my first two months?(about the role)

- How many regular meetings do software engineers attend?
- What’s the company’s biggest priority right now?

▼ How will you handle a situation where you are sure your manager is wrong?

---

discussing

I will discuss my point with my manager in a private, one-to-one session. The discussion will help both of us to understand the different views we hold. It will help us to reach a conclusion that is best for the project.

▼ Tell me about a time when you got into a conflict with a coworker. How did you solve it?

In my last job, I had an argument with the tech team representative regarding some system issues that were not resolved even after days. We both handled the situation patiently and after a thorough discussion, we arrived at a solution that worked for both of us.

▼ Do you like to work with a team or individually? Give an example of how you worked on a team.

▼

▼

▼

▼ The most proud accomplishment

The most proud accomplishment is that I've mastered several tech skills in 7 years. I've worked with over 20 projects during this time, and experienced in several tech skills.

▼ What is Your Ambition?

“My long-term goals involve growing with a company where I can continue to learn, take on additional responsibilities, and contribute as much value as possible to the team. I love that your company emphasizes professional development opportunities. I would take full advantage of the educational resources available.”

▼ What are you looking for in your next role

Three things that I'm looking for in my next job are **a collaborative, team-focused culture, opportunities to learn and grow my skills from a technical standpoint, and a chance to learn more leadership skills over time**.

▼ What is the best piece of advice you ever received

There are a lot of good advice, but I think, **It’s just a job.**

The best advice is the hardest to take. I guess that’s advice too. I am just a developer, making a living the best way you know how.

▼ Tell me about a time when you went above and beyond for a customer? What did you do?

A couple of months ago, I noticed that my manager didn't complete most of the staff schedules for the following weeks. My manager normally completes them on time, but he had been working on several other urgent projects. Rather than add to his burden by complaining, I offered to help him with some of his workloads so that he could complete the schedule. We worked overtime hours for a week, and I helped him catch up on everything while learning new skills. I am always ready to go above and beyond to help a colleague or manager with tasks and projects

▼ Describe the work environment in which you flourish?

This is my ideal working environment, as **I thrive working as part of a team**. I prefer working in a group where team members can encourage each other and share their ideas. I also enjoy working for a company where I know I can continue to grow my skills both personally and professionally

▼ What is your preferred work style?

I always keep on top of my projects. Owing to my organizational skills and efficiency, I can successfully juggle multiple projects at once. While I complete most of my work independently, I greatly value input and will consult with team members to ensure we're all on the same track. I also appreciate checking in regularly with my boss to update her on my progress and ask about any issues that have arisen. This open communication helps me complete tasks efficiently and accurately.

▼ What’s your work style?

My work style is **extremely flexible**—working on so many different projects requires me to be adaptive. In general, I try to work on one project at a time, working as quickly and efficiently as possible to achieve the best results.

- ▼ Management style
    - ▼ What’s your management style?

        I have not yet had the opportunity to find out what my particular managerial style would be. At the same time, I’ve found that the best managers are those that have an adaptable supervising style. From my experience, I work best with managers who pay attention to individual's needs. My last manager was a good example. She used slow periods to coach workers who needed more instruction. When things sped up, she gave clear instructions and took on tasks herself if necessary.

        Although I’ve never been a manager, I took on several leadership roles as a student. I was the team captain of our academic team for two years. I made sure that each team member knew what they were responsible for before matches. At the same time, we often switched roles depending on who felt strongest. I took a democratic approach to leadership. I motivated the team by encouraging constructive feedback after matches. We won districts both years, and we’re invited to nationals my senior year.

    - ▼ Example 1: Transformational management

        "I believe a good manager is motivational and encouraging. I'm always working to push myself out of my comfort level and I enjoy doing the same with my employees. They are often capable of achieving many challenging obstacles, so I use my transformational management style to help guide them through this challenging task when needed. I've accomplished this with a content writer I once supervised. I encouraged them to write long-form content pieces on subjects they had little to no knowledge of. This led them to become my strongest research writer on the marketing team."

    - Example 2: Visionary management
    - Example 3: Democratic management
    - Example 4: Mentoring or training management
    - Example 5: Laissez-faire management

    https://www.indeed.com/career-advice/career-development/what-is-my-management-style#:~:text=I'm always working to,this challenging task when needed.

- ▼ What is “Agile” software development and what are your thoughts on it?

Agile software development is a process that focuses on incremental delivery by the team as a whole. The project is broken up into small chunks that are able to be completed within a given timeframe called ‘sprints.’ In my previous role, we were quite successful in adopting the process. We used two-week sprints and kept high contact with many face-to-face discussions to review questions and concerns as they arose

▼ What Is Your Greatest Strength?

I believe my strongest trait is my attention to detail. This trait has helped me tremendously in this field of work. I've always been a great team player. I'm good at keeping a team together and producing quality work in a team environment. After having worked for a couple of years, I realized my strength is accomplishing a large amount of work within a short period of time. I get things done on time and my manager always appreciated it.

▼ What is your Greatest Weakness?

My biggest weakness is that I don't yet have much experience of lead the large groups of people.

However, this is an area I do want to develop, so if there are any opportunities within this role, and within your company to help me develop this skill, then I would certainly be open to that.

▼ Day to day responsibility

- Every day, I have to work closely with the frontend, backend and QA team in completing projects And Troubleshoot any issues relating to frontend development
- Also, create reusable components for my teammates.

▼ What do you consider to be your biggest professional failure, and what did you learn from it?

I was managing a project where a new client wanted a large number of unique product descriptions written to improve the SEO ranking of their site. Because they were a new client and I wanted to impress them with the kind of results we could produce, I assured them we could have it back to them in two weeks. I thought this was doable with multiple writers working on the project, but in the end, it took an extra week, and they were not happy.

We apologized and reassured them that the mistake wouldn’t happen again. I realized that it’s far better to under-promise and over-deliver. The client isn’t going to be upset when you are clear about what the timeline is from the beginning. Problems arise when you can’t meet promised deadlines. I used this experience to be more cautious in managing client expectations. For the next client project I worked on, I made sure to include extra time for unforeseen circumstances and told them we would deliver in four weeks. We delivered in three, and they couldn’t have been more thrilled.

- ▼ What is the most difficult thing you have ever worked on

  The most difficult thing is to estimate time line to finish the project delivered to me. Sometimes I’ve worked with clients non-experienced in development. At that time they asked me to estimate time line to finish project successfully. If I estimate it as I want based on my situation, it would affects the business strategy badly. As a front end leader, I should make milestone blended well with strategy and team members’ ability. So I got inputs from collaborators and client and decided it. Due to urgent launching date we should sometimes work in weekday. After all I made the intended outcomes as they want and they always enjoyed with my ability.

- ▼ Tell us about a time when you had to give someone difficult feedback. How did you handle it?

  I Listen carefully to his feedback, to identify the real problem. And respond quickly, but not in an emotional or reactive manner. If we are in my team, pull him aside, and ask questions and offer proactive solutions to correct the issue and demonstrate my genuine care.

  I always take negative feedback as an opportunity to improve my products and services for the future.

- ▼ How do you handle change?

  I'm good at dealing with change because I'm flexible with my work and abilities. I'm not afraid of learning new and difficult things. Whenever I'm faced with a change, I'll put in extra effort to make the change a smooth transition.

- ▼ How do you handle conflict between team members?
  - ▼ Sample
    - ▼ **Speak to Team Members Individually**

Start by having an informal one-on-one with each team member involved in the conflict. This way I can hear people's concerns in a safe, confidential setting. In these meetings:

- Avoid making assumptions and let people open up in their own time.
- Reassure the employees that the discussion is confidential.
- Ask each party the same questions to remain impartial.

▼ **Bring People Together**

Once I’ve got a better understanding of the conflict and everyone's perspectives, it's time to bring the relevant parties together and act as a moderator.

Set some ground rules before getting the conversation underway. Encourage team members to listen to one another, respect each other's points of view, and not interrupt or make personal comments. During the conversation:

- Moderate to keep the tone of the conversation calm and non-threatening.
- Encourage **active listening** so people really understand where the other person is coming from.
- Encourage individuals to share ideas. What do they want or need? What would they be prepared to commit to? Encourage them to brainstorm some solutions.
- Ask them about situations where they've worked well together in the past. See if they can build on those positive experiences.

If the discussion becomes heated, pause it and reconvene when everyone's had a chance to calm down.

▼ **Ask the Wider Team for Ideas**

When a conflict affects the whole team, provided it's not sensitive or confidential, I can ask for everyone's perspective.

Talking things out helps me and my team to consider different assumptions, beliefs and decision-making approaches. This can also be a part of creating a "**psychologically safe** " environment, where people feel comfortable sharing ideas and concerns, thus preventing future conflicts.

- ▼ **Draw up a Plan**

  Ask the parties to detail agreed-on actions for reconciliation. And get each to commit to this strategy. I can draw up a timetable for actions, ticking them off as and when they are achieved. Hold all relevant parties accountable.

- ▼ **Follow up**

  Ensure that issues have been resolved properly by following up on the situation. For example, people may still feel aggrieved but not want to drag things out. I can use one-on-ones to prevent old disagreements from resurfacing. And try an anonymous **team survey** to get feedback and reveal any lingering frustrations.

▼ d

There are always two sides to every story, which is why it's so important to me to remain as neutral [/ˈnjuːtrəl $ ˈnuː-/] and open-minded as possible whenever I hear of conflict between teammates. I was in a situation a few years ago where two members of my team were clearly unhappy with each other. Rather than let it fester or ignoring it with the hope that they would be able to work it out themselves, I sat down with them individually and asked them to explain what was going on. We discussed reasonable and professional solutions that worked for both parties and the matter was resolved.

▼ How do you handle a disagreement with your colleagues? Give me an example of when you successfully persuaded someone to see things your way at work.

Situation: When I was working as a recruiter at Company X, I noticed that one of the candidates who had sent in their application was perfect for the role. Though he didn't have a university diploma and his resume wasn't too polished, reading his cover letter, it was obvious he knew the industry and had delivered clear results.

Task: I thought it was worth giving him a shot, but my supervisor didn't see it that way. She skimmed through the resume and told me not to waste time, and just discard the candidate.

Action: I was, however, still pretty confident in the candidate, so I talked to the supervisor over lunch. I took a bit of an indirect approach, though. Instead of trying to directly pitch the candidate, I asked her to clarify the job description a bit more. We went a bit in-depth on what, exactly, we were looking for in the candidate, and once we were done discussing it, I told her that we happened to have a candidate that possessed all the relevant experience, but his resume was a bit weak.

Results: Convinced, the supervisor decided to give the candidate's application a more in-depth look and realized that they were, in fact, very qualified. She thanked me for bringing it up and agreed with me that the candidate was worth calling in for an interview.

▼ Give me an example of when you had to suddenly perform under pressure. What happened and how did you handle it?

Situation: As a seasonal worker, there have been a lot of times where I had to juggle extra responsibilities. My last position as a line cook at Restaurant X comes to mind. During summer, we were pretty much always full, and sometimes, even understaffed to handle all the customers. To make things worse, we didn't have the best shift system at the time either. So, if someone were to unexpectedly not show up for their shift, we'd have to put out the fires as they came up.

Task: Which is exactly what happened when one of our waitresses had to cancel her shift due to an emergency.

Action: So, I stepped up and took her shift as soon as I had clocked out of mine as one of the line cooks. Luckily, I had previous experience working as a waiter.

Results: I was tired and a bit uncoordinated at the beginning, but at the end of the day, everything worked out just fine.

▼ How do you handle irate customers? Give me an example.

Situation: Working in customer support, you really get to talk with many different kinds of people. I remember I had one angry customer that called the helpdesk once to complain. He kept repeating the product he bought was faulty and demanded me to resolve the situation then and there.

Task: Customers calling for refunds happen all the time, but this one was different as he just kept shouting over the phone the whole time. I had to get him to calm down if I wanted the call to go anywhere.

Action: Fortunately, I had experience dealing with loud customers, and knew the first thing I had to do was listen to his story. Halfway through telling his story, he calmed down once he realized I was trying to help. He explained that the product was supposed to be a gift, and that's why he was so frustrated. Then, I offered 2 solutions: a refund or a replacement for his product with express delivery.

Results: The customer opted for the replacement option. I called him back once they received the order just to check-in if he was happy with the product. He turned out to be happy both with the product and our service, and thanked me for the help.

▼ We all make mistakes sometimes we wish we could take back. Is there a time that comes to mind where you wish you had handled a situation with a client or colleague differently?

Situation: This one client we worked with was particularly difficult. They were extremely unpleasant to work with and treated our staff pretty badly. The management, however, insisted on sticking with them, since they made up for a good chunk of our income. At one point, though, the client just barged into our office and started yelling at their account manager for a small mistake on their end.

Task: At this point, I realized that working with the client was really affecting our staff negatively, and we'd be losing some good employees if we kept working with them.

Action: So, I set up a meeting with the management team, and gave them concrete facts and figures about the client. Sure, they were paying us good money, but they were really hurting the workplace morale.

Results: After hearing me out, the management agreed and fired the client. They decided that overall, the impact such clients had on the company wasn't worth it, and started doing stricter vetting during discovery calls.

▼ How do you handle constructive criticism? react

1. Stop my first reaction. Stay calm and try not to react at all.
2. Remember the benefits of getting feedback and try to understand the motivation and perception of my criticizer.
3. Be a good listener.
4. Say thank you.

5. Ask questions to deconstruct the feedback and share my perspective.

▼ What motivates you to do your best work?

I am a very result-oriented person. My primary motivation is to achieve the desired result. While I enjoy working on the project on my own, I am particularly motivated by the buzz of working in a team. It’s very exciting working closely with others, who share the same common goal. I also like to take on the challenge and rise to that challenge as part of a concerted team effort.

▼ What motivates you?.

I am motivated by working on technical challenges. As a frontend developer, I enjoy being given specifications for features to add or problems to solve and figuring out the best and most effective course of action. In my last job, I was able to complete a sprint for an important new feature two days ahead of schedule because I felt motivated to solve the technical challenges associated with implementing the feature

▼ We have an intense work environment at Binance. How do you keep yourself motivated in high-pressure situations?

I find that I work best in an intense work environment surrounded by hard workers who get the job done rather than simply talk about ideas. I know this to be true of Binance. Finding motivation amid stress has never been a problem for me. I firmly believe in the power of cryptocurrency to democratize the global financial system. This belief forms the backbone of my work and keeps me motivated even when facing high-pressure situations.

▼ Who inspires you?

The person who has most greatly inspired me is my Project manager. He introduced me to the world of decentralized finance. We've worked together on various crypto-related projects in our free time. He introduced me to the concept of putting 80% of your effort into the 20% that will get you the biggest return. It's completely changed the way I work and think about projects.

▼ How do you keep up-to-date on new technologies and trends in the cryptocurrency space?

"I utilize resources like CoinDesk and Blockchain News to keep up with cryptocurrency news and technologies. I also follow three blogs from industry leaders and read multiple articles online per week. When possible, I take online training courses to keep my industry knowledge sharp and up-to-date. While researching **Binance**, I came across the company's 'Learn and Earn' initiative, which I am eager to explore further.”

- ▼ What is your greatest accomplishment?

  “In my most recent job I was responsible for managing the orientation and training programs for our new hires. Unfortunately, the content was not engaging. While it was necessary information for our new hires to have, we found that only 35% 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, 93% of participants complete the training and provide positive feedback about their experience. My manager was so pleased with the improvements that she asked me to lead a training seminar in our New York office."

- ▼ why do you want to leave (or have left) your job?

  I want to leave my job because I felt it is time for a change. I've reached my full potential there and I want to work for your company, where I felt I will be pushed the challenged and my diverse set of skills , qualities and experiences will be put to good use. I have enjoyed working for my employer and they have been very supported and now I'm ready for fresh challenge.

- ▼ why do you want this job

  - ▼ Information Technology & Services

    I'm interested in a career in technology **because I want to use my skills and knowledge to make other people's jobs easier and more productive**. I believe IT support technicians are important team members for any organization because they combine technical knowledge with interpersonal skills.

  I want this job because it is the type of role that I genuinely love doing. We spend a lot of time at work, and I want that time to be put to good use.

  I feel I already have the skills and qualities to match the job description / which means I can come into the role and start helping the team to make an immediate and positive impact.

  I also want this job because I believe it will give me lots of job satisfaction, and I will get to develop essential workplace skills that we all need to live a productive and fulfilling life.

I have applied for this vacancy because it is an excellent match for my skills and experience. This role is exactly the sort of role I am currently targeting, and I am confident I will be able to make a major contribution.

▼ Why do I want this job? why should you hire me?

You should hire me because my experience is almost perfectly aligned with the requirements you asked for in your job listing. I have seven years' progressive experience in building startups (or implementing IMS), advancing from my initial role as a front-end developer to my current position there as a senior Full-stack developer. I'm well-versed in providing world-class customer service to an upscale clientele, and I pride myself on my ability to quickly (build your project) resolve problems so that our guests enjoy their time with us (you can put your idea into reality in the shortest time).

▼ What are your long term goals?

I've always loved to teach. I like to grow newer employees and help co-workers where ever I can. So in the future, I would love to be an instructor. I would like to become a director or higher. This might be a little ambitious, but I know I'm smart, and I'm willing to work hard.

▼ What are your short term goals?

My short term goal is to find a position where I can use the knowledge and strengths that I have. I want to partake in the growth and success of the company I work for.

As a program manager, it's important to understand all areas of the project. Although I have the technical abilities to be successful in my job, I want to learn different software applications that might help in work efficiency.

▼ Do you manage your time well?

I manage my time well by planning out what I have to do for the whole week. It keeps me on track and evens helps me to be more efficient.

▼ How do you make important decisions?

I believe all decisions should be made by having all the information. If you are missing an important detail, it's easy to make a bad decision. So I make important decisions by having all of the information.

▼ How Do You Deal With Pressure or Stressful Situations?

I stay motivated by thinking about the end result. I've found that even in the midst of a challenging situation, reminding myself of my goals helps me take a step back and stay positive

▼ How to resolve the conflict between designers and developers?

Point 01: Communication

A good communication is the need of the Design and Development Teams. Even before you design mockups or any solution to the problem, the discussion is very important between both to avoid conflict.

Point 02: Share the Components Library

A User Interface needs lots of components while developing text styles, blocks, forms, containers, and these major components needed to prepare the template of the website or etc. So it should be very clean for both designer and developer what are these components and needed to share between both.

Point 03: Having a good understanding of User Interface or understanding review

The process of reviewing the build UI or etc. really needs to include both designer and developer to discuss the making of the right set of changes that are needed. Usually, it does not take too much time to discuss but it adds value to the solution.

▼ How do you accomplish tasks when under a tight deadline? Give me an example.

Situation: Well, typically, I try to never commit to a deadline I don't think I can make. But sometimes, unexpected things happen and you're forced to think on your feet. For example, at my last job, my coworker had to take some time off work because of an emergency, and his project was left without a manager.

Task: My supervisor then instructed me to take over his project and complete what work was left. Suddenly, I had a new project on my hands, and I wasn't really sure how to handle it, as the deadline was in 1 week.

Action: First, I requested a reduction on my own daily sales goals - which I was granted. This way, I could pay more attention to the project, and only a few hours per day to my original tasks. Once I had a consistent schedule and hours set for each of my tasks, it was mostly easy from there.

Results: Thanks to my teammates and my good time management skills, I managed to finish up 2 days early before the deadline. And once my coworker came back to work, I was able to review the whole thing with him before submitting it. For what it’s worth, he was thoroughly impressed. And a few months later, I even got promoted based on my performance.

▼ How to improve your website's SEO in 5 steps:

- Choose the right URL.
- Create titles and descriptions for each page.
- Utilize anchor text.
- Add alt text to all your images.
- Give your site structure with the right headers.

▼ Top Reasons to Work with Us

- VERY cutting edge company!
- Founding team has made successful exits...this could be another!
- Fun, fast-paced environment
- Touch many technologies
- Work with some extremely successful entrepreneurs
- Remote work

▼ Describe a long-term project you managed. How did you make sure everything was running smoothly?

Situation: When I was at Company X, I was managing the web development team in charge of setting up a new website for one of our biggest clients at the time. With most projects, we had a process set up and we would get most sites done in up to 2 months. This project, however, was a bit different, as the website was supposed to be more detailed, with a lot of unique pages. So, we had to be a lot more careful with our time-management.

Task: We had a strict deadline of 15 weeks, and I had to make sure that we used up our time as efficiently as possible.

Action: Before getting to actual work, I decided that we should plan everything out by the week. After some research and consulting with our team of developers, we decided to split the workload between different stages. We would devote around 1 week to the discovery phase, 5 weeks to design, 3 weeks to initial development and the rest to any modifications and updates.

Results: In the end, we actually finished the website with all the promised functionalities in just under 3 months. The client was very satisfied with the result and eventually ended up recommending partners to our firm.

▼ Sometimes, it's almost impossible to get everything done on your to-do list. What do you do when your list of responsibilities becomes overwhelming?

Situation: As a senior at University X, there were times when I just couldn't physically get everything done on time. For example, towards the end of my final semester, I was the Student Council President and I was also writing my University thesis. I had to submit my thesis the next day, and I was also working with my fellow student council members to organize the end-of-the-year ceremony for the University.

Task: If I had tried to multitask both, I would just have done a poor job. Now, for me, the University thesis was clearly higher up in my list of priorities. After all, this was what my studies were building up to for so long. But I couldn't just abandon my council members either. With 24 hours until my thesis deadline, I had to think fast.

Action: I decided that the best approach was to send all of my notes and outlines for the event to the Student Council VP, who was also a close friend of mine. Luckily, he understood my situation and took over my event-management responsibilities. Then, I had just enough time to edit and finalize my paper.

Results: Thanks to the VP, I was able to fix and finalize my Thesis. And fortunately, the event went without a hitch too. In the end, I learned a valuable lesson on time-management, and the importance of having the right team around you who you can rely on.

▼ Tell me about a time you set a personal goal for yourself. How did you ensure you would meet your objectives and what steps did you take?

Situation: I think the most recent, and important, personal goal that comes to mind is that I managed to teach myself web development from scratch. You see, I wasn't very satisfied as a sales rep at Vival company. My coworkers were nice, and the pay was decent too, but I just didn't see myself growing there.

Task: So, I decided that I wanted a career change in a field I've always been interested in - web development. Now, because I was working full-time, I had to be very efficient with my time-management skills.

Action: I did some research, and all that was left to do was just follow my routine and stay committed. I set up a personal calendar and made sure to study HTML, CSS, and JavaScript for at least 2 hours every day. I gathered a list of beginner-friendly books to start with, and once I was done with those, I paid for some advanced online courses to improve my React and Vue.js skills. When I felt comfortable, I started working on some personal projects for my portfolio and did some freelance work part-time while I was still working at Company X.

Results: In the end, I'm glad I stuck to my plan and continued with my set curriculum. If I did not have my calendar planned out with specific objectives, I surely would have been overwhelmed. Sure, at times, it felt like I was basically working 2 jobs and that a lot of the material wasn't making sense. But I just kept moving forward, and then, I got my first real break as a junior web dev at Loop Markets company.

▼ Can you describe an instance where your supervisor or manager just gave you too much work with not enough time? What did you do?

Situation: I had a pretty rocky start with my manager at Agency X, as we had different expectations for my workload. Normally, I don't have a problem with a fast-paced working environment, and I tend to thrive when I'm thinking on my feet. But at the agency, I had just finished onboarding, and I was already bombarded with tasks and weekly reports. For the most part, I was managing to get everything done on time, but I realized the quality would suffer if my list of tasks kept getting longer.

Task: So, I had to take up my work schedule issue with my manager and let him know about my concern. I decided that being direct, and also respectful was the best approach, and booked the meeting.

Action: During the meeting, I remained calm, and just went straight to the point. I explained how I liked my work, but the heavy workload was really impacting the quality of the work.

Results: Luckily, he was understanding. I was the first in-house designer they'd hired, and they weren't 100% sure what was a lot of work, and what wasn't. We ended up working together to better define my responsibilities. From then on, I was, for the most part, only getting the workload I could handle without diminishing the quality of my work.

▼ What would you do if you misunderstood an important task on the job? Give me an example.

Situation: At my previous internship at Company X, I underestimated the amount of time it would take me to finish a presentation for a team meeting. The deadline my boss gave me was around a week, which was completely fair and I didn't think it would be a problem. However, apparently, we had some miscommunication with what he'd meant with the deadline. I thought it was the date where we would go through the presentation, edit it together, and submit it like that. Apparently what he'd meant, though, was to have the presentation 100% ready on that date.

Task: So, I had to submit a draft presentation first, edit it based on my manager's comments, and then present the report, all within 2 days.

Action: I booked a meeting with the manager for the following day, and spent 4 extra hours at the office to make sure that the first draft of the presentation was spotless. We held the meeting the next day, and went through the presentation together to make sure it's spotless.

Results: The manager loved the work, and it only took us around 30 minutes to finalize the whole thing.

▼ Have you ever had to work under someone who wasn't very good at communicating? What happened?

Situation: Yes, at my last job as a tech recruiter the hiring manager I was working directly with was somewhat more difficult to communicate with. He had very strict and precise requirements on the type of candidates he wanted to invite for interviews. He wasn't open to much communication on the matter or trying new things even when the company desperately needed new hires. This one time, I got a candidate that was a pretty good fit for the job, but was lacking in some aspects.

Task: I wanted to make sure that we got the person in for an interview, but I was 100% sure that my hiring manager would shut me down.

Action: So, before running the candidate through him, I called them and collected his biggest strengths to present to the hiring manager.

Results: The hiring manager did, indeed, end up liking the candidate and calling them in for an interview.

▼ Tell me about a time when you successfully explained a technical problem to a colleague or a customer who didn't have a tech background?

Situation: I've worked as a tech support specialist before, so I really excel at this. I've had to explain complex concepts to customers on a regular basis, but to give you one single example, I've had to explain to clients with next to no understanding of computers how to delete a virus on their computer.

Task: After trying to give basic instructions to the client, they still didn't really understand much, so I had to come up with a smarter solution.

Action: So what I did was, I walked them through the entire thing step by step while explaining it simply but in no condescending terms. Instead of making them do most of the work, I walked them through the process of getting me to connect with their computer, and then I explained to them what, exactly, I did.

Results: The customer was very happy with my work, and we managed to fix the issue with their computer.

▼ Can you tell me about a time you gave a presentation that was particularly successful? Why do you think it went well?

Situation: Sure thing. As the business development manager at Firm X, there were quite a few opportunities when I had to speak in front of a crowd. The most recent, and successful, one was for the new project we were launching.

Task: I was called on to speak for a 2 department-wide meeting, of up to 50 people. Now, I had never delivered a presentation to this many people, but luckily, I knew most of them quite well after years of working with them.

Action: Working with 2 other members of my team, I decided to take a more creative approach, and create a short video (a skit) to hook the audience. That was the intro, and then we used PowerPoint and hands-on examples to show what to expect from the new project launch. And finally, we dedicated the last 5 minutes to a Q&A session.

Results: It felt longer, but the whole speech took about 15 minutes in total. We got great feedback from the audience, and I was later asked to present at the all-hands meeting the next month. I knew my colleagues well enough and I tried to make the speech as if I was having a one-on-one conversation with a friend - with a few jokes in-between.

▼ Tell me about a time when you had to work with someone completely different from you. How did you adapt to collaborate better?

Situation: Sure, I always enjoy working with new and different people. Usually, because they bring something new to the table. At Company X, there was a particularly young developer who was assigned to work with me on a new software development project, and I was to run him through what our typical coding process was like.

Task: It was also my job to get to know him, and find common ground so that we could effectively work together. The fact that he was younger wasn't an issue for me, but because he was completely self-taught, he didn't know a lot about the industry methodologies we used.

Action: Teaching him everything from scratch would take too much time. So, instead, I briefly explained the development process (waterfall model) we were using for that specific project, and taught him how to write tests for our code-base. Writing tests is the number 1 way to learn what code does. After all, that's how I got started with development.

Results: I also sat down and helped him go through the material at times, but in the end, he surprised me by how much of a fast-learner he was. He just needed a bit of encouragement and guidance. Through this approach, he learned our whole routine in less than a week, while most of our new hires needed at least up to 2 weeks. In return, I learned a lot about multitasking and time-management from him. The whole thing was a win-win situation, and it was all smooth sailing the next time we worked together (which was quite often).

▼ What do you do when your team member refuses to, or just can't complete their part of the work? Give me an example.

Situation: There was one co-worker at my Company who was notorious for being bad at deadlines. But she would always end up delivering exceptional work, just a few hours (or worse - days) late. For some reason, the company was ok with this as her work was just too good. So, this one time, the management put us together to work on a time-sensitive project.

Task: Our task was to turn in a sales presentation together and have our manager go over it before sending the client the final version. Because of how important the project was, I didn't want to risk going over the deadline - as this would also directly impact other people. Either way, for everyone's sake, I had to somehow get her to hurry up with the project. So, I decided to try and push her a little and see what would happen.

Action: I started regularly checking in on her to see where she was with work. I would bring it up at times over lunch, send a quick Slack message, and so on.

She wasn't taking this quite well, but it DID get her to work faster and more efficiently.

Results: At the end, the constant check-ins and pushing did have a positive effect, even though the co-worker didn't particularly like me too much once we were finished with the work. We even managed to submit the final version of the presentation 2 days before the deadline.

▼ Clients can be difficult to work with sometimes. Can you describe a situation when a client was wrong and you had to correct them?

Situation: Absolutely. One of our past clients at Agency X came to us because his Facebook advertising strategy wasn't working. He was driving traffic but wasn't getting any conversions, so they thought that it was because they weren't reaching the right audience. We realized, though, that it was actually because their product homepage wasn't really that convincing. The client, however, was adamant about "not fixing what wasn't broken."

Task: I had to somehow communicate with the client that the service he wanted wasn't what he wanted - there was no way for us to fix his Facebook ads if his homepage wasn't selling the product.

Action: We had to give the client an ultimatum - they either go with our approach, or we wouldn't be able to get the results (and hence, work with them).

Results: After some back and forth, the client grudgingly agreed to do an A/B test between the existing landing page, and one that we'd propose. So, we tested the two landing pages with the same ads he'd been running, and ended up getting 2.5x better results. From then on, the client was a lot more willing to allow us to experiment with whatever we proposed.

▼ Tell me about your first job in the industry. What did you do to learn the ropes?

Situation: Well, my first job in the field was as a junior dev ops engineer. While I did have extensive knowledge of the field, I didn't have too much experience doing it.

Task: This made it very hard for me to get started with the job. While I was working almost all the time, I wasn't getting too much done.

Action: So, what I did was, taking a lot of my personal time to really work and learn the ins and outs of dev ops. I also made sure to talk to my team members and get their input on daily tasks.

Results: A few months into the job, I managed to learn the ropes and ended up being a lot more productive.

▼ Can you give me an example of when you had to adapt to a new and sudden change in the workplace? What happened?

Situation: Sure thing. In my previous position as an account manager at Company X, we had to suddenly change all of our CRM software and move all the data to a new tool. The CRM tool we'd been using till now wasn't fit for a growing team, and on top of that, they were upping their pricing, so it wasn't really worthwhile for us.

Task: I was put in charge of finding the replacement CRM, as I was the one who knew the previous one inside-out. And this was also an opportunity for me to clean up our outdated info and start fresh. All the while, I still had to handle my daily responsibilities and as usual.

Action: So, the first thing I did was ask our sales associates and lead generation teams what they thought of the old CRM, and if there were any new features they were lacking. After doing a bit of research and asking around, I found the perfect tool that had it all - sales analytics, email integration, and more. And because I typically have no problem with learning new tools, I stayed in one evening, transferred our data to the new tool, and wiped the old account. Finally, I sent a new announcement to the entire team about the new software, as well as a video on how to use it.

Results: We completed the transfer with 4 days to spare, the team was satisfied with the new CRM, and my daily responsibilities as an account manager didn't suffer.

▼ Tell me about a time when you successfully delegated tasks to your team.

Situation: Well, at my first job as a team lead, I had to really get to know most of my team in order to delegate tasks appropriately.

Task: Most team members were new to the company, so I didn't have much to go with.

Action: So, I sat down with each team member individually, and really got to know them and their strengths and weaknesses, and distributed tasks based on their personality.

Results: Team members were pretty happy with the tasks they got, and started off their relationship with our company on a positive note.

▼ Can you tell me about a time when you had to perform a task or work on a project you had no previous experience before? How did you approach this situation and what did you learn?

Situation: In my previous position at Company X, my manager had to leave unexpectedly for about a month due to a medical condition. Fortunately, she was able to give us a week's notice.

Task: Because of that, our director asked me to fill in as the interim manager. I was familiar with the basics of management on a theoretical level, and I had worked with my manager closely before, but I certainly wasn't trained to be a manager yet. Though, I wasn't going to say no, and I, more or less, felt confident about my ability to take on the new challenge.

Action: So, I accepted the position. The first thing I did was gather the team and let them know about the situation. I was very open about my lack of experience, and asked them to be open about giving feedback when possible. I also asked a manager for an hour of their time to pick their brain and make sure I'm doing everything right.

Results: In the end, we managed to get through the month without any problems, and delivered all the projects on time. When my manager returned, she was very pleased with the work, and I even got compliments from our director. Because of my success with the role, I was then promoted to team manager at the end of that year.

▼ Have you ever faced conflict with a coworker? How did you resolve the situation?

S - "I'm usually a very easy-going employee and get along with most of my coworkers. There was this one time though, when I was working for Company Y. At the time they had just purchased new software for the company."

T - "I was in charge of introducing it to my coworkers since I had previous experience with it. The CEO however, also brought in someone from the software company to assist with the transition as well. We had to cooperate on this task and well.. the guy wasn't too thrilled about the idea. He often interrupted me whenever I was explaining something, and sometimes didn't even invite me to some of the training we were supposed to organize together. "

A - "I decided to approach him and suggested we split the responsibilities. He would handle the theoretical part of the training (introduction to how it works), and I'd explain the practical stuff (as in, how we can implement the software for our business case)."

R - “He agreed, and the transition went smoothly since then. Took us around a week and a half to get the entire team on board and productive with the new tool.”

▼ Tell me about a time when you made a mistake at work

S - “The store I was working for would be opening a new location soon and they offered me the position of manager there.”

T - “That meant I was also in charge of monitoring the work until the store opened. I ordered the clothing shipments, trained the new employees, and organized the inauguration event. One hour before the event, the last shipment of boxes hadn’t arrived yet. The clothes the mannequins had to wear were in those boxes.”

A - “I called the delivery company and they said the boxes had already been shipped...to the other location of the store...across the city. I’d given them the wrong address. There was absolutely no time for those boxes to arrive in time at the store.”

R - “The mannequins were dressed in other clothes, none of them belonging to the new collection. I explained the situation to my superiors. They weren’t very happy about it of course, but they acknowledged that it was a human mistake and it could happen to anyone.”

▼ Do you have any question for me?

Do you have any concerns about my ability to do this job/fill this role

▼ day to day

· Develop high quality software which meets requirements, promotes re-use of software components, is secure, and facilitates ease of support

· Diagnose, isolate, and implement remedies for system failures caused by errors in software code

· Build and manage the DevOps processes including test automation, CI/CD pipeline, product installation, upgrades, and infrastructure provisioning

· Conduct unit tests, track problems, and implement changes to ensure adherence to test plan and functional/nonfunctional requirements

· Provide input and technical content for technical documentation, user help materials and customer training

· Analyze, design and implement software mechanisms to improve code stability, performance, and re-usability

· Participate and lead code review sessions

· Participate with industry groups, stay current with technology and industry trends, disseminate knowledge to team members, form best practices. (Retail and Self-service Retail Industry groups)

· Communicate with Solution Management and other internal teams. Participate in cross-functional collaboration within the organization

▼ What are the main features of an ecommerce website?

Most customers look for a few key features when evaluating an ecommerce website. These are elements that improve the overall online shopping experience by making it highly functional and user-friendly.

- Easy to use features: Simple navigation tools, easy checkout flows, etc.
- Mobile compatibility: Compatible and functional on all mobile devices.
- Discount code and promotional capabilities: Allows shoppers to use discounts on-site.
- Security features: Payment processing is secure and reliable.
- Social proof: Validation from past customers and trusted sources.
- User-generated content: Reviews, ratings and photos that add to the ethos of offerings.

▼ How do you respond to new project assignments?

I look forward to the first project planning meeting with a new team. Listening to everyone's input and ideas helps everyone get to know each other and figure out what roles we will take on

▼ What have you learned from your mentor

One of the traits I most respect is her ability to show empathy to everyone she meets. I think it helps people trust her quickly. She can mediate conflict, and people are willing to try her suggestions.

▼ Tell me about a challenge or conflict you've faced at work, and how you dealt with it.

In my last big project for my previous employer, the client needed to make frequent changes to the scope, which affected many of our task deadlines. As the project manager, it fell to me to help my team learn to be flexible and avoid frustration. We held frequent meetings, I gave people a voice to express their concerns and we all found ways to adapt and complete that project to our client's satisfaction.

▼ Code quality

Code quality can have a major impact on software quality, on the productivity of software teams, and their ability to collaborate

https://www.sealights.io/code-quality/code-quality-metrics-is-your-code-any-good/

▼ What is your favorite ES (es6) functionality?

Arrow functions, Modules, Destructuring, Generators, Promises, Template literals, Spread operator, New object literal features (including method definitions), let and const, Rest parameters

▼ Tell us about your interests in current development trends.

JavaScript Frameworks, Automation Testing, Accelerated Mobile Pages

▼ Do you consider unit testing essential, or a waste of time?

Unit testing is an essential instrument in the toolbox of any serious software developer. However, it can sometimes be quite difficult to write a good unit test for a particular piece of code

▼ What do you think are the most important aspects to pay attention to when reviewing another team member's code?

Elements of the code, including styling, formatting, design patterns, and naming conventions, should be kept consistent. Having someone who works with the same standards review your code will help keep it formatted correctly and easy to use for other team members.

▼ What are you most passionate about in programming?

I think that great software developers are passionate about **coding and problem solving in general**. We like to solve puzzles. We like to sink our teeth into a problem, break it down, and work out the best solution.

▼ How do your qualifications support your success as a <level/> developer?

1. Work independently with little supervision.

2. Have excellent organizational and problem-solving skills.
3. Have an analytical mind with an aptitude for problem-solving.
4. Take the lead on operational and technical projects.
5. Demonstrate the ability to create and maintain relevant processes.

▼ What development processes have you had to compromise on and why?

My first semester in college, I was a political science major. My introductory government class professor had a differing political view than I. We disagreed on everything, and many classes were filled with criticizing each others' view. However, on one test I answered a question with the view I believe in, and she marked it wrong. So I asked her how an opinion can be wrong, and she said because her opinion is the way she taught it in class. I pointed out that my answer showed I understood the concepts of the question. She agreed, and I also agreed, not to be so combative in answers on tests. Compromise is the key to problem resolution.

▼ How would you describe your ideal work environment?

From the research I've conducted, it appears your company operates in a collaborative environment that fosters professional advancement. This is my ideal working environment, as I thrive working as part of a team. I prefer working in a group where team members can encourage each other and share their ideas. I also enjoy working for a company where I know I can continue to grow my skills both personally and professionally.

▼ Which package manager do you prefer and why?

▼ what is the package manager

A package manager is a programming language’s tool to create project environments and easily import external dependencies. You don’t have to reinvent the wheel and are able to make the most of the tools at your disposal. When working on a project or library, you may “package” your project and publish it for others.

npm is the package manager for JavaScript. It is the world’s largest software repository. npm hosts extremely popular packages like jQuery, Bootstrap, React, Angular etc. Linking your GitHub repository with npm also allows you to create and share your own projects. As the npm online repository is so large and diverse, JavaScript front-end and Node.js backend developers make use of npm as the packages can be used in either environment.

▼ How do you explain technical challenges to stakeholders who do not have technical knowledge or backgrounds?

1. Use humor and humility to better explain technical information. ...
2. Be attentive to your audience throughout your presentation. ...
3. Incorporate storytelling when sharing technical information. ...
4. Use visual content to explain technical information and processes.

▼ What achievements are you most proud of?

I'm most proud of the latest social media marketing campaign I developed and launched with my current team. Although I have experience creating a variety of social media content and have created marketing campaigns in the past, the most recent project was the first time I effectively led and managed an entire team of employees. This experience taught me a lot about the value of teamwork and highlighted the importance of nurturing everyone's unique gifts and talents. While it was satisfying to develop digital content and outreach initiatives on my own, I found it more rewarding to lead others through the process. Effective communication was crucial, and once all members were clear about their role in the project, we were able to achieve engagement levels that were 30% higher than our initial goals.

▼ How would you explain APIs to non-technical stakeholders?

▼ Know my Stakeholder

Learn about my stakeholders! Find out where their expertise lies and try to gauge their level of knowledge on the subject beforehand. Pick up on clues during my first engagements to better determine their level of technical expertise and communicate at a level slightly below that – being cautious that you don't undermine their intelligence, as that could be offensive. If you're unsure, start with a simple "I don't mean to undermine my intelligence, if this is something you already know, please let me know." Starting at a lower level and covering ,y basis is better than rambling about something they know nothing about, as they are less likely to speak up and ask questions. Knowing my stakeholders' background and being observant of gestures, tones, and body language while you communicate is an essential step towards ensuring everyone is on the same page.

▼ Cut Out Tech Jargon

To increase my stakeholders' understanding, start by eliminating intimidating technical phrases and acronyms. Although it can be a challenge to communicate my ideas and the details without using the IT jargon I am comfortable with, there are ways to overcome this issue to make sure everyone is on the same page. Consider the use of analogies, metaphors, and similes that my audience would be familiar with when explaining more technical details. Reserve the high-tech jargon for internal conversations inside the IT department where it's appreciated and understood.

▼ Translate and Educate

Never assume what someone does or doesn't know – only that they are intelligent and capable of understanding provided you appropriately explain. Translate key terms and acronyms that are critical to the discussion in a manner that makes it simple for the stakeholder to understand. If it is not vital, cut out the jargon and instead provide context and analogies that are relatable. Metaphors are also an excellent way to improve a stakeholder's understanding by putting a foreign concept in a familiar context.

▼ Speak in Terms of Results

Communicating effectively with my stakeholders is about using terms they'll understand, in a manner that matters to them – end results. I need to have an understanding of their overall business perspective and be able to relay what my technology solution will deliver to impact their business needs or goals positively. Tell them how my technology solution is the answer to their problems, makes their life easier, helps them achieve their goals or any combination thereof, and describe what the user will see in the finished product – not how you're going to build the technology or product. Talking about the technology benefits and what it does for them and the business rather than the technical design, development, or implementation details will allow my stakeholders see that you know exactly what they're looking to achieve and you have the best interest for their business in mind.

▼ Use Visuals

Some people are visual learners and learn immensely better with the use of creative communication tools. There are a variety of tools available to illustrate even the most technical concepts, which makes this much easier of a task. Utilizing an array of resources such as whiteboards to draw what's happening step-by-step, visual charts, screen sharing, or video tutorials can make a world of difference in effectively communicating with someone who is not as familiar with the technical concepts I am presenting as I am.

Think about how I best learn new concepts and would want something like this communicated to me if I was in their position.

▼ Encourage Question

As mentioned previously, people are often scared or embarrassed to ask questions on subjects I am expecting them to know, so actively encourage my stakeholders to ask questions. Throughout the conversation, pause and ask things such as: "Does that make sense?" or be straightforward, "Do you have any questions so far?" This will make the stakeholder feel at ease while clearing up any questions during the conversation and encourage them to be more willing to ask about ideas or concepts they don't understand before moving forward.

▼ How do you keep your skills sharp and up to date?

▼ 1. Consider Professional development courses.

Professional development courses can help us expand my professional skill set, learn something new, or even earn academic credit to put towards a degree. Online courses are particularly convenient because they are affordable and flexible. Just be careful to do my homework—evaluate instructor bios, read reviews, and check the syllabus carefully before putting down my credit card. I can also find professional development courses through vendor-taught classes, traditional universities, and training institutions.

▼ 2. Make use of online resources

The Internet is a limitless source of free information and educational resources. Attend educational webinars, follow the blogs or social media accounts of industry experts, or bookmark and regularly check industry news sites and online forums to stay current on the latest trends.

▼ 3. Sign Up for Professional Events

Professional events are valuable ways to learn about growth and development in my industry. Local companies, business associations, and professional groups often host seminars, forums, or workshops that can give you direct access and insight to experts in my profession. Treat these events as constructive networking opportunities to brainstorm and share ideas with colleagues who can provide fresh insight and perspective.

▼ 4. Take Networking Online

As an independent consultant, I know the importance of building and maintaining a list of contacts to ensure a steady flow of work. Use LinkedIn to connect with high-ranking people at companies I am interested in working with. Employ social media platforms to promote my own service or brand, network with industry experts, and keep in touch with former and current clients.

Start by finding which social site works best for you—connect with fans and followers on Facebook, creatively network and share news on Twitter, or utilize blogging to boost my online credibility, and connect with potential clients.

▼ 5. Continue My Education or Get a Certification

While continuing education and certification programs typically require a more intensive time and financial obligation, they can help boost reliability, and demonstrate a commitment to my profession. Becoming proficient in a new software platform before it becomes mainstream, committing to upholding industry standards through a certification program, or staying on top of market trends by taking a class can increase my income and position you competitively within my line of business.

▼ 6. Learn from Others

Social media platforms have given thought leaders a new outlet for sharing valuable information, insights, and practical advice. Seek out and follow both industry leaders in verticals I specialize in and those who specialize in skill areas I have or want to build. By reading regular posts, you can not only gain additional knowledge, but I can also build relationships by commenting and reacting to posts and increase visibility of my business and skills by sharing their content.

▼ 7. Read White Papers and Case Studies

Top companies, consulting organizations, agencies, and think tanks regularly publish white papers and case studies on industry trends and often offer them for download at no cost. Stay up to date on industry and business trends by taking advantage of these resources.

▼ 8. Target the Hard and Soft Skills You Need to Develop

Conducting a self-assessment to determine my hard and soft skills and target those you want to develop should be a core activity in my professional development. A self-assessment test such as CliftonStrengths Assessment will measure my natural ways of thinking, feeling, and behaving, and you'll get access to personalized reports that help you better understand what makes you unique and how to use my strengths to reach my full potential.

Maintaining enhanced knowledge and skills in my field shows clients you are well informed and dedicated. Set myself up for success by investing in my job skills and knowledge today.

▼ Describe your strengths relating to software architecture

- I have design skills and knowledge.
- I have knowledge of building and construction.
- I am thorough and pay attention to detail.
- I have thinking and reasoning skills.
- I have customer service skills.
- I have excellent verbal communication skills.
- I have analytical thinking skills.
- I have the ability to use your initiative.

▼ What do you do when you're having difficulty solving a problem?

I always do my best solving difficult problems. If I have trouble, I'll use resources around me such as looking through the Internet, asking co-workers, asking my manager, or looking through some books.

▼ We encourage innovation at Binance. When have you created something from scratch in the workplace?

In my previous role, I coded a new software program that became a key component of our customer management system. Doing so required me to put in additional work outside of my typical hours, but it was a project that I was passionate about, so I had no problem doing so. Further, I took the lead in deploying the new software by conducting training and serving as the point person. This project was a huge success at my company, and I look forward to contributing to building something from the ground up at Binance.

▼ Walk us through your post-secondary education and training. How has your education prepared you for this role with Binance?

I completed my Bachelor of Applied Technology from University of North Carolina in 2014. During that time, I specialized in data management and analysis, which taught me how to grasp abstract data models to drive better decision-making for my team. Additionally, my capstone project helped me sharpen my teamwork skills and ability to work with various perspectives.

▼ Culture fit is important to us at Binance. How would you describe your personality?

"I would describe my personality as driven and focused. I believe that, if asked, my colleagues and supervisor would say the same about me. I am confident that I would thrive in Binance's results-driven culture. Binance describes itself as 'Hardcore,' which is a trait that resonates with me. I, too, am results-driven. I work hard to get things done, and I am passionate about my work."

▼ At Binance, we take pride in our ability to stay one step ahead of our competitors. How will you contribute to this trend

"Binance is one of the leading crypto platforms. The company has achieved this by offering unmatched customer support and building an easy-to-use user experience platform. I believe the best way for Binance to stay ahead of its competitors is to pay close attention to the needs of its users. I will contribute to the success of Coinbase by providing my most attentive work every day, ensuring each decision I make speaks to the promise Binance has made to its users."

▼ How comfortable are you with failure?

"I spent the first four years of my career in venture capital, where I became quite comfortable with risk-taking and learning from failure. I learned that while it was crucial to prepare adequately to make an important decision, it was even more important to trust my gut and take a chance on a company sometimes. The trust I built with myself was incredibly beneficial for my team and clients in multiple instances. I think it's truly inspiring that Binance has fostered such an innovative community that is not afraid to take risks and learn from perceived /pəˈsiːv $ pər-/ failure /ˈfeɪljə $ -ər/."

▼ Tell us about a time you took independent ownership of a project.

"In my previous position, I took over as the client lead for one of our most profitable but demanding clients. While various people on my team contributed to projects for this client, I adopted the mindset that any failure to reach full potential ultimately landed on me, as I was the group leader. This approach helped me maintain diligence and an eye for detail when working on client tasks. Client profitability increased by 22% in that first quarter as client lead."

▼ How do you approach explaining a complex idea in simple terms?

"I strive for a straightforward and concise communication style with clients and colleagues. In my previous role as a financial advisor, I focused on listening to my clients and their most pressing financial needs and goals. I made an effort to echo their language and avoid esoteric ideas that I knew wouldn't be relevant to what they needed to hear. Above all, I ensure that my tone is never condescending /ˌkɒndɪˈsendɪŋ◂ $ ˌkɑːn-/ towards clients or colleagues. I often learn something new from those I work with and value their input."

▼ How do you further your education as a professional in this space? (blockchain, web3)

"I take my personal and professional development very seriously. I believe it's important to avoid stagnation in my career, especially in an industry that changes daily. I've recently opted to participate in a 50-hour blockchain course and am about halfway through it. So far, it's been beneficial and has given me a wealth of knowledge and new resources. I have also found the Binance Academy to be a golden resource, including the lessons on the Metaverse. I'm just getting started on my journey in blockchain, crypto, and Web3 topics, and I look for every opportunity to develop a solid foundation in this space."

▼

▼ Leader

leadership skills

ex: decisiveness(problem-solving, initiative, research, project evaluation, expectation setting), integrity(diplomacy, ethics, reliability, professionalism, responsibility, confidentiality, honesty), creativity(critical thinking, curiosity, diversity, innovation, collaboration, open-mindedness), flexibility(negotiation, adaptability, feedback, work-life balance), positive attitude(conflict management, social skills, rapport, empathy, positive reinforcement, respect), communication(active listening, verbal communication, written communication, nonverbal communication, public speaking, presentation skills), relationship-building(collaboration, management, interpersonal, social teamwork), problem-solving(critical thinking, analytical skills, research, decisiveness, team-building), dependability(realistic goal-setting, timeliness, initiative, detail-oriented, loyalty), ability to teach and mentor(motivation, clarity, able to recognize and reward, understanding employee differences, feedback, help, helpfulness)

Interpersonal skills examples

Emotional intelligence

Emotional intelligence is your ability to be aware of and manage your emotions — as well as the ability to perceive and deal with the emotions of others, both on an individual level and in groups. Generally, when you have higher levels of emotional intelligence, you'll be able to manage your emotions and deal with stress easier. And what software developer doesn't have stress, right?

But emotional intelligence doesn't stop with managing stress and emotions better, though. It also includes a subset of skills that include empathy, self-awareness, self-regulation, and motivation. And some of these are especially important for software developers.

For example, empathy is all about understanding other people's problems and putting yourself in their shoes. Conversely, software development is about solving people's problems. So, with empathy and the ability to understand people's problems better, you'll be able to find better solutions to them. Ultimately, this allows you to build better software.

Moreover, as you know, in software development, you'll continuously strive to improve your skills both on a personal and professional level. With higher emotional intelligence, you'll have more self-awareness which means you'll be able to identify your shortcomings better. And if you're able to do this, you'll be able to implement the necessary strategies to improve.

Read More: How to Build a Software Engineer Portfolio

Communication

Communication skills can be broadly divided into three distinct categories:

- **Verbal communication**. Verbal communication relates to *what* you say and *how* you say it. This is often the foundation of effective communication and getting your message across to other people.
- **Non-verbal communication**. Effective communication doesn't stop with verbal communication, though, and non-verbal communication is just as important. Here, your gestures, tone of voice, and body language can reinforce the message you're trying to communicate or, in some cases, even undermine it.
- **Listening**. Communication doesn't come only from one side of a conversation, right? That's why listening skills are also crucial when it comes to communication skills. In fact, you'll spend up to 45% of your time listening.

Now, it might be that, in the past, you wouldn't have needed stellar communication skills as a software developer. It was simple: you got the tasks you needed to complete from your project manager, and off you went.

But modern software development is much more of a team affair that requires — yes, you guessed it — effective communication. Actually, some say that excellent communication skills are vital for a successful career in software development. And, let's face it, when you have the communication skill chops, you'll also be more approachable.

Apart from discussions with your team, you'll often also need to communicate directly with clients. And when you do, you'll need the right communication skills. These will let you not only listen to their issues and requirements, but also be able to distill complicated technical aspects into language that's relatable and easy to understand. Projects progress more smoothly when everyone's on the same page!

Teamwork

Teamwork is probably one of the most important skills you can have as a software developer. It involves the way you work with others in formal settings like meetings as well as more informal project chats.

In software development, it can relate to how you work as a team towards a common goal and how you collaborate. Thanks to project management methodologies like Agile, effective and efficient teamwork is a must-have.

Effective teamwork can increase:

- **Efficiency**. It’s simple, when team members work together effectively to reach a common goal, the entire team will be more efficient. As a result, they’ll be able to deliver quality software faster. This, however, also depends on effective communication and collaboration between team members.
- **Creativity**. When team members communicate and collaborate well, they’ll be more willing to share their ideas with the rest of the team. This, in turn, leads to more creativity and can improve the product they’ll deliver.
- **Innovation**. Flowing from increased creativity, is the ability to innovate more. Simply put, when teams are more creative, they’ll be able to find more innovative solutions to the problems their customers experience.

Ultimately, effective and efficient teamwork allows teams to develop better software and keep their clients satisfied.

Read More: Common Interview Questions for Software Engineering Jobs

Dependability and Responsibility

As a software developer, you’ll need to take responsibility for any new task you get. In other words, you’ll need to know what to do and by when. In turn, your colleagues, teammates, and other stakeholders should be able to depend on you to get the work done on time.

This means, in other words, you need to be reliable. Reliability means that when you take on new, often complex tasks, you see them through to completion, no matter how challenging they are.

Not only does reliability ensure that you, as a software developer, will deliver high-quality work within any agreed time limits, but it also makes you a valuable member of the team. And, let’s be honest, employers value dependable and reliable workers and trust them with their most important tasks and duties. This means, when you’re reliable, you’re more likely to be promoted.

Leadership

Leadership skills is a vital set of abilities to have, especially if you’re a more senior software developer. Why? You’re more likely to end up leading a team. Leadership typically consists of a subset of other skills that you use to effectively lead a team. These include skills like organizing, planning, risk management, and also decision-making and problem-solving skills.

Apart from these, to be an effective leader, you also need excellent emotional intelligence, communication, empathy, negotiation, and conflict resolution skills. With these skills in your arsenal, you'll be able to effectively lead any team to deliver high-quality software based on a specific client's needs and requirements.

But it goes further than this. Leadership is about more than just leading a team and taking charge. It also allows you to inspire and motivate your teammates to deliver their best work consistently.

Read More: Do You Need a Degree to Become a Software Engineer?

Negotiation and persuasion skills

Negotiation and persuasion skills are often thought of as a subset of communication skills but they're so important that it warrants seeing them as separate interpersonal skills. This is especially true for software developers.

Let's face it, if you put 10 software developers in a room and give them a problem to solve, you'll likely find a lot of differing opinions and disagreements on the best way to solve the problem. And this is where effective negotiation and persuasion skills come in.

They'll allow you to not only convince other team members on the best way forward but also allow you to find common ground where there are differences. This, ultimately, leads to finding the best solutions for a problem that takes into account every team member's thoughts and proposals.

Apart from this, you'll need effective negotiation and persuasion skills in many other aspects of life as a software developer. For instance, apart from convincing your team members about the best way to solve a problem, you'll often have to do the same with a client.

These skills are also invaluable when you need to:

- Convince clients to make use of your services,
- Determine workloads and schedules within a software development team,
- When you're applying for a software development position, or
- When you need to convince management that you should get a raise or promotion.

Conflict resolution skills

It’s a fact that negotiations won’t always work out well — and then you’ll have to deal with disagreements and conflict. This is not necessarily a bad thing, as differing points of view can lead to innovative solutions to problems and, therefore, better software. Unfortunately, it doesn’t always turn out this way.

And this is where conflict resolution skills become relevant. These skills help you positively resolve conflict while leaving valuable relationships intact. Conflict resolution skills generally include a subset of skills like mediation skills, the ability to communicate in difficult circumstances, and the ability to deal with feedback and criticism.

The last mentioned is especially crucial in software development as you’ll often get feedback or criticism on the code you write. Here, you’ll have to deal with this feedback effectively and often incorporate suggestions into your workflow.

If you don’t, you’ll more than likely appear unapproachable and not open to any suggestions to improve. Even worse, you could appear arrogant, and let’s face it, no one wants an arrogant software developer on their team. Conversely, if you’re able to accept constructive criticism and deal with feedback effectively, you’ll be more successful in your role as a software developer.

Read More: Best Resume Builders & CV Makers for Software Developers

Decision making

Now, you might immediately think that decision-making is not an interpersonal skill. After all, you make your decisions on your own and you have to deal with the consequences of your decisions.

The thing is, the decisions you make directly impact those around you. This is especially true when working in a team. Just think about it: you need to make decisions about the code you write, the features you include in a product, and how the team will work towards the goals among others.

To be a good decision-maker, you need to be able to effectively consider the advantages and disadvantages of every decision you need to make or the course of action you take. And you need to also take others’ viewpoints into account. Moreover, you need to be able to make tough decisions when you need to, despite potential criticism from your team.

Ultimately, in software development, a wrong decision can not only impact the success of the project but also how satisfied the client is with the outcome. And here, another vital aspect of decision-making comes into play. When making decisions, you also need to be accountable for them and face the consequences of any wrong decisions.

▼

▼ Tell me about your leadership skills and experience.

leadership skills are the skills you use to organize a software development team, inspire and motivate them, and build a sense of common purpose. In other words, with leadership skills, you guide and help a team to reach a shared goal.

- Technical knowledge
  - Our team's top programming languages and frameworks
  - Software design and architecture
  - Implementation
  - Database design, manipulation, and management
  - Debugging and software testing
  - Application deployment
  - Project management and project management methodologies
- Emotional intelligence
  - Empathy
  - Self-confidence
  - Commitment
  - Understanding
  - Optimism
  - Advanced interpersonal skills
- Communication

  It's a fact that when you want to lead a software development team, you need to have excellent communication skills. These skills allow you to communicate effectively with your team, and clearly spell out every team member's specific responsibilities and what you expect of them.

  As such, you'll communicate your expectations, their roles and responsibilities, and — more importantly — the project's specifications to them in a way that's easy to understand, consistent with your processes, and straightforward to implement.

Excellent communication skills are also invaluable when it comes to dealing with customers and other stakeholders. This is simply because, as the leader of your team, you'll often be the bridge between them and your team.

For example, you'll typically need to translate the customer's requirements into actionable information your team can use. You'll also need to convey technical feedback *back* to the customer in language that is simpler to understand.

And this is where effective communication comes in. Without it, your team will struggle to know what to do and your customer will — quite probably — get a product with different specifications than they asked for.

- **Customer-centric (The customer is always right)**
- **Self-development**
- **Team building**
- **Problem-solving**
  - Analytical skills
  - Critical thinking
  - Strategic planning

That's good question. Leadership skills is a vital set of abilities to have as a senior software engineer. I have organizing, planning, risk management, decision-making and problem-solving skills.

Also, I have excellent emotional intelligence, communication, empathy, negotiation, and conflict resolution skills.

As the result: I could deliver high-quality software based on a specific client's needs and requirements.

Now, I am trying to improve interpersonal skills.

▼ As a mentor yourself, how do you inspire your student mentees?

As a mentor to some young or non-experience developers, I usually share my own experience with them, hoping to bring them inspiration and help them realize their interests.

I always lead them by examples and mistakes and place great emphasis on thorough testing. And then what is more important is to encourage them to grow and have a strong vision by themselves. I usually demanded that they usually keep developing themselves. And second, the important thing has to have a good appearance as a mentor. So I tried to keep my positive attitude and engage in honest, open communication. and care about everyone on my team.

▼ What motivates you to do your best work?

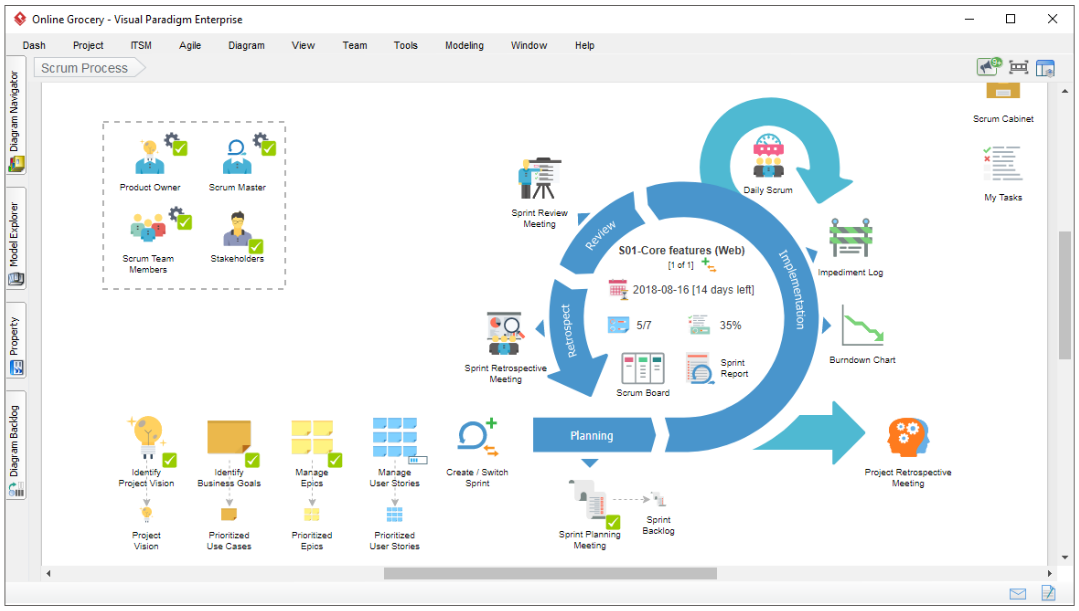
Working in a team and leading one, is my motivation to do a good job. United we stand, divided we fall has been my mantra for success right along. Working in a team has taught me so much more. It enlightened me and opened up my mind to a whole new world. The confluence of ideas, thoughts, and opinions encouraged me to know and learn from my peers. There was a time when I believed that I could do everything on my own. With time I realized the magnitude of achievement possible with teamwork. It took me some time getting used to leading a team of varied individuals. Gradually I realised that it was more about dealing with their personalities and less about their skills sets. It was a challenge which I took up and gained valuable insights related to team building and motivation. As a leader, it is imperative that you trust your team members and support them. It is also important to create synergy from their individual capabilities.

▼

▼ Concept

▼

▼ AAgile, Scrum and Kanban



What is Agile methodology in project management?

The Agile methodology is a way to manage a project by breaking it up into several phases. It involves constant collaboration with stakeholders and continuous improvement at every stage. Once the work begins, teams cycle through a process of planning, executing, and evaluating. Continuous collaboration is vital, both with team members and project stakeholders.

Agile's four main values are:

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

Agile project management

So what is Agile methodology in project management? It’s a process for managing a project that involves constant collaboration and working in iterations. Agile project management works off the basis that a project can be continuously improved upon throughout its life cycle, with changes being made quickly and responsively.

Agile is one of the most popular approaches to project management due to its flexibility, adaptability to change, and high level of customer input.

Agile methodologies frameworks

Agile project management is not a singular framework — rather, it can be used as an umbrella term to include many different frameworks. Agile project management can refer to terms including Scrum, Kanban, Extreme Programming (XP), and Adaptive Project Framework (APF).

Agile pros and cons.

There are a range of advantages and disadvantages to following an Agile methodology in your business. Consider these Agile pros and cons to help decide if it's the right direction for you.

Benefits of Agile project management.

There are various advantages of an Agile project methodology, which include:

- Freedom for employees to work on models that leverage their strengths.
- More efficient use of resources and rapid deployment.
- Greater flexibility and adaptability to changing needs.
- Quicker detection of and remedies to problems.
- Improved collaboration with co-workers and users, leading to better functionality in products that better meet user needs.
- Clearly defined goals and processes do not need to be firmed up before work can start.

Disadvantages of Agile project management.

A few drawbacks to consider before you implement an Agile project methodology are that it:

- Is easy to slide off-road without predetermined paths of action.
- Provides less predictable outcomes.
- Works less well for businesses that require plenty of time to analyze problems or undertake market research.
- Can fall flat without good collaborative skills and good personal relations.

What is Scrum?

https://www.scrum.org/resources/what-is-scrum

Scrum is a framework within which people can address complex adaptive problems, while productively and creatively delivering products of the highest possible value.

Scrum is a lightweight framework that helps people, teams and organizations generate value through adaptive solutions for complex problems.

Agile v Scrum: what’s the difference?

The main difference between Agile and Scrum meetings is that Agile is a general approach and philosophy to project management – Scrum is a specific method within it.

Agile vs. Kanban

Kanban project management is a type of Agile methodology that seeks to improve the project management process through workflow visualization using a tool called a Kanban board. A Kanban board is composed of columns that depict a specific stage in the project management process, with cards or sticky notes representing tasks placed in the appropriate stage. As the project progresses, the cards will move from column to column on the board until they are completed.

A key difference between Kanban and other Agile methodologies, such as Scrum, is that there are typically limitations regarding how many tasks can be in progress at one time. Project management teams will typically assign a specific number of tasks to each column on the board, which means that new tasks cannot begin until others have been completed.

What is Kanban

https://kanbanize.com/kanban-resources/getting-started/what-is-kanban

Kanban is a popular Lean workflow management method for defining, managing, and improving services that deliver knowledge work. It helps you visualize work, maximize efficiency, and improve continuously. Work is represented on Kanban boards, allowing you to optimize work delivery across multiple teams and handle even the most complex projects in a single environment.

Benefits of Kanban

- Increased visibility of the flow
- Improved delivery speed
- Alignment between goals and execution
- Improved predictability

- Improved dependencies management
- Increased customer satisfaction

Scrum vs. Kanban

The most important difference between Kanban and Scrum is that the former is a method, while the latter is a framework. Kanban builds a continuous delivery model where teams release value as soon as they are ready, while Scrum organizes work in Sprints. Applying either one depends on the nature of your process, however, it can be said that Kanban offers a more tailor-made approach while Scrum relies on predetermined rules. Another key distinguishing characteristic between the two is the mindset and founding belief systems of Scrum and Kanban.

	Kanban	**Scrum**
Nature	Kanban is an adaptive method	Scrum is a prescriptive framework
Principles	1. Start with what you do now 2. Agree to pursue evolutionary change 3. Encourage acts of leadership at all levels 4. Focus on customer’s needs 5. Manage the work 6. Regularly review the network of services	1. Empiricism 2. Transparency 3. Inspection 4. Adaptation
Cadences	- Team-level cadences - Service-oriented cadences	- Sprint with a fixed length - Sprint planning - Daily Scrum - Sprint Review - Sprint Retrospective
Roles	- Service Delivery Manager* - Service Request Manager* (*no pre-defined roles are required)	- Product Owner - Scrum Master - Development Team
Metrics	- Cycle Time - Throughput - Work In Progress	- Velocity - Planned Capacity

6 rules of Kanban

1) Never pass on defective products;

2) Take only what is needed;

3) Produce the exact quantity required;

4) Level the production;

5) Fine-tune production;

6) Stabilise and rationalise the process.

I have good knowledge of Scrum and Agile concepts and vast experience in Agile methodologies especially scrum, and scrum events such as sprint planning, daily scrum, sprint, sprint review and sprint retrospective.

Scrum is a popular framework that enables teams to work together. Based on Agile principles, Scrum enables the development, delivery, and sustenance of complex projects. It enables teams to hypothesize how they think something works, try it out, learn and reflect from their experiences, and make appropriate changes.

Scrum is generally recommended for teams or industries that expect to face change. Because they are designed to adapt quickly and operate in short cycles, Scrum teams are generally small, making it hard for pure Scrum to work for projects with large teams without some modifications.

Agile, Scrum (from other origin)

▼ What did you do?

1. I participated in daily stand-up meetings, story grooming, reviews, demos, and other project-related meetings.
2. Supported team members in their tasks.
3. Coaching the team on Scrum principles and best practices.
4. Facilitating open discussion and conflict resolution.
5. Proactively identify and resolve issues.

▼ Agile

Agile is a collection of methods and practices that focuses on iterative development. It is time-boxed and iterative, focusing on delivering products incrementally throughout the project, rather than all at once, in the end.

The shorter projects are completed in short two-to-four week cycles called iterations. The requirements and solutions are obtained with the collaboration of self-organizing cross-functional teams. Some of the popular Agile methodologies are Scrum, Kanban, XP, Lean, Crystal, etc.

▼ Artifacts of the Scrum Process

- Product Backlog: It is a list that consists of new features, changes to features, bug fixes, changes to the infrastructure, and other activities to ensure a particular output can be obtained.
- Sprint Backlog: It is a subset of the product backlog that contains tasks focused on by the team to satisfy the sprint goal. Teams first identify the tasks to be completed from the product backlog. These are then added to the sprint backlog.
- Product Increment: It is a combination of all product backlog items completed in a sprint and the value of previous sprints' increments. The output must be in usable condition, even if the product owner doesn’t release it.

▼ Burn-up and Burn-down charts

Burn-up Chart:

It is a type of chart that is used to display or represent the amount of work that has been completed and the total amount of work for a sprint or iteration.

Burn-down Chart:

It is a type of chart that is used to display or represent the amount of work that is remaining to be completed in the project. These charts are very simple and easy to understand.

▼ Different types of Burn-Down charts are listed below:

- **Product Burndown Chart:** It is a type of chart that is used to show story points of each completed sprint so that it depicts the completion of requirements over time. It mainly shows how many of the product goals are being achieved by the team and how much work is remaining.
- **Sprint Burndown Chart:** It is a type of chart that is used to show the remaining works for the scrum team of a particular sprint. It makes the work of the team visible and shows the rate at which work is completed and how much is remaining to be completed.
- **Release Burndown Chart:** It is a type of chart that is used to show how a team is progressing against the work for a release. This chart is updated by the scrum team at the end of each sprint. It is very essential to see what process is being made during each sprint.
- **Defect Burndown Chart:** It is a type of chart that is used to show the total number of defects that are being identified and fixed or removed.

▼ Complexity and Effort

Complexity and effort are measured through “Story Points”. In Scrum, it’s recommended to use the Fibonacci series to represent it.

▼ Daily Standup Session

Stand-up sessions are daily discussions that take place and are usually 15 minutes long. Daily Stand-up sessions help understand:

- What tasks went well
- What tasks were completed
- What tasks are pending, and
- The obstacles the team is facing

The meeting helps in understanding the overall scope and status of the project. Further discussions can take place after the stand-up sessions.

▼ DoD

- Definition of Done (DoD) refers to the collection of deliverables, which includes written codes, comments on coding, unit tests, integration testing, design documents, release notes, etc. This adds verifiable and demonstrable values to project development. DoD is very helpful to scrum while identifying the deliverables to achieve the objective of the projects
- It helps with:
    - Defining the steps required to deliver the iteration
    - The usage of appropriate tools like burndown to make the process more effective
    - Ensuring on-time feedback throughout the project life cycle
    - Ensuring the walkthrough of the product backlog items are done and understood correctly
    - The creation of a checklist for the product backlog items
    - Ensuring the DoD is defined to become task-oriented
    - Involving the product owner for reviewing during the sprint and sprint retrospective

▼ **Events**

There are some events in Scrum and all these events are timeboxed which means all these events are allotted with a maximum and fixed unit of time for the task. The events that are time-boxed are listed below:

1. **Sprint Planning:** This event happens at the start of the sprint, where the scrum team discusses the items in the product backlog and decides which ones will form part of the current sprint. The deliverable of the sprint planning meeting is the sprint backlog.
2. **Daily Scrum:** Daily scrum is timeboxed to 15 minutes and is held usually at the start of the day. In the daily scrum, the scrum team discusses the sprint backlog items to understand their progress or any issues/blockers that need to be resolved.
3. **Sprint:** The sprint itself is the duration in which all the sprint backlog items are delivered. This is usually 2 weeks but can be a month.
4. **Sprint Review:** Sprint review is held at the end of the sprint where the team reviews the product with the stakeholders. The sprint review is a maximum of 4 hours.
5. **Sprint Retrospective:** This meeting reviews the last sprint to answer questions: What went well and what could be improved and then develops a plan to improve. The sprint retrospective is generally 1 hour to 3 hours.

▼ Key skills of scrum master

- A strong understanding of Scrum and Agile concepts
- Fine-tuned organizational skills
- Familiarity with the technology used by the team
- To be able to coach and teach the team to follow Scrum practices
- Having the ability to handle conflicts and resolve them quickly
- To be a servant leader

▼ Handle **Conflict**

For a Scrum Master, paying attention to the source of the problem and listening and acting accordingly would go a long way. Any disagreements should be shared with other team members in a manner that they would be open to suggestions for resolving the issue. When a conflict arises, the Scrum Master must intervene so that the process runs smoothly and without hiccups.

The following steps help in handling conflicts within the team:

Step 1 - Scene setting

First, we must determine the source of the team's quarrel. Before taking any action, it is necessary to understand the discrepancy between two groups or two persons. In times of dispute, Scrum Masters typically react aggressively against team members in the hopes of resolving the conflict on their own. However, while this may temporarily cure the problem, it does not address any underlying concerns. The Scrum Master must lead the team and teach them that disagreement is a regular occurrence in the workplace and it can be resolved with assertiveness. It is the leader's responsibility to guarantee that team members' concerns are acknowledged and addressed.

Step 2 - Gathering Information

Gathering facts about the conflict is usually crucial before coming to a conclusion about a certain individual or suppressing the topic. This could be accomplished by listening to each party separately and comprehending the situation from their point of view. The Scrum Master should also consider other team members' perspectives and also respect every team member’s decisions. As a result, the Scrum Master must elicit everyone's assistance in order to gain a picture of the workplace conflict.

Step 3 - Brainstorming to find a solution

It is often impossible for the leader to resolve problems on his or her own. Furthermore, several members of the Scrum Team would have better answers that would quickly remedy the problem. Organizing spontaneous group talks and sharing opinions on various activities would stimulate good discourse between the two people or groups in these situations. This would urge both sides to see things from the other's perspective. This also provides opportunities for superior ideas to be pushed and for the disparity to be bridged.

Step 4 - Solution conferring

Listing all of the possible answers to an issue would only be useful if those solutions were put into action. Scrum Master removes the team's roadblocks by implementing the solution in this step. Throughout the conflict resolution process, remembering to stay calm and respectful will aid in a speedier and more efficient resolution.

▼ MVP vs MMP

- Minimum viable product (MVP) is a Lean Startup concept that stresses the impact of learning while performing product development. This allows one to test and understand the idea by getting exposed to the initial version for target customers & users. To accomplish this, one has to collect all the relevant data and learn from that collected data. The thought behind MVP is to produce the product, to provide access to the users, and to observe how the product is used, perceived, and understood. This will also provide more insight towards what the customers’ or users’ needs are.
- The MMP (Minimal Marketable Product) refers to the description of the product, which will have a minimal number of features that address the requirement of the users. The MMP would help also help the organization reduce the time to market

▼ Pillars in scrum

The three pillars of Scrum are summarized below -

Adaption: The method being processed must be changed if an inspector determines that one or more aspects of a process are outside of permitted limits. A correction must be made as quickly as possible to avoid future deviation.

Transparency: Transparency mandates that those elements be specified by a consistent standard in order for viewers to understand what they are viewing. For example, while referring to the process, all participants must use the same terminology. Those reviewing as well as those executing the job and the resulting addition must have the same definition of "done."

Inspection: Scrum users must check Scrum artifacts and progress toward a Sprint Goal on a regular basis to discover unwanted deviations. Inspections should not be carried out so frequently that they constitute a burden to their work. Inspections are most successful when skilled inspectors do them attentively at the point of work.

▼ Product roadmap

A product roadmap, as the name suggests, is a powerful tool that describes how a product is likely to grow over time. It is a holistic view of product features that create the product vision. It also indicates what development is building, business goals that the new product will achieve, problems that the product will solve, etc. A product roadmap is owned by the product manager. It also encourages the development team to work together to achieve the desired goal for the successful delivery of the product.

▼ **Pros & Cons**

▼ Pros:

- Scrum can help teams complete project deliverables quickly and efficiently.
- Scrum ensures effective use of time and money.
- Large projects are divided into easily manageable sprints.
- Developments are coded and tested during the sprint review.

▼ Cons

Scrum has several benefits, but it isn't for every organization or team. A Scrum master should know the limits of Scrum, and when best to leave a project to other types of project management.

Scrum is generally recommended for teams or industries that expect to face change. Because they are designed to adapt quickly and operate in short cycles, Scrum teams are generally small, making it hard for pure Scrum to work for projects with large teams without some modifications.

Scrum also might not be the best choice if you're working with very strict constraints, like a budget or timeline. If you're familiar with other project management methods, talk about how those might replace or supplement Scrum in a project.

- Scrum requires individuals with experience
- Teams need to be collaborative and committed to ensuring results
- A scrum master with lesser experience can cause the collapse of the project
- Tasks need to be well defined, lest the project has many inaccuracies
- It works better for smaller projects and is difficult to scale to larger, more complex projects

▼ Pair programming

Pair programming, as the name suggests, is a type of programming where two people write code together and work side-by-side on one machine or computer. It is basically a technique mostly used in agile software development. In this type of programming, one person writes code and another person checks and reviews each line of code. Both of them also switch their roles while doing work.

Advantages of Pair Programming

- Develop higher-quality code
- Reduce the risk of errors
- An effective way to share knowledge
- Enhanced productivity
- Improved team collaboration

▼ Responsibilities of Scrum team

The Scrum Team is one that's self-organizing and involves five to seven members. The following are their responsibilities:

- Working products must be developed and delivered during each sprint.
- Ownership and transparency must be ensured for the work assigned to the team members.
- Correct and crisp information must be provided to ensure a successful daily scrum meeting.
- They must collaborate with the team and themselves.

▼ Risks in Scrum

- Budget: The risk of exceeding budgets
- People (team): Team members need to be of appropriate skill and capability
- Sprint (duration and deliverables): Exceeding the duration, addition of the scope of work
- Product (user stories, epics): Having ill-defined user stories and epics
- Knowledge and capability: Having the appropriate resources

▼ Risk Management Steps

The five steps of Risk Management are given below -

Risk Identification: To identify the risks that your company is exposed to in its current operating environment. There are several types of risks, such as market risks, legal risks, regulatory risks, environmental risks, etc. It's crucial to be aware of as many risk factors as possible.

Risk Analysis: Once a risk has been identified, it must be investigated. The scope of the danger must be determined. It's also important to understand the connection between other internal factors and risk. It's critical to determine the risk's severity and importance by examining how it affects the business operations.

Ranking the risk: Risks must be ranked and prioritized. Most risk management solutions include numerous risk categories based on the severity of the danger. Risks that may cause minor discomfort are prioritized the least, but risks that can result in significant loss are prioritized the highest.

Treating the risk: As much as possible, all risks should be avoided or reduced by contacting experts in the field in question. In a manual environment, this would include contacting each and every stakeholder and setting up meetings for everyone to discuss the issues.

Risk review: To ensure that it has been entirely eradicated, the risk evaluation is done.

▼ Roles in scrum

- Product Owner: The product owner is an individual who is responsible for increasing the ROI by determining product features, prioritizing these features into a list, what needs to be focused on the upcoming sprint, and much more. These are constantly re-prioritized and refined.
- Scrum Master: This individual helps the team in learning to apply Scrum to ensure optimum business value. The scrum master removes impediments, shields the team from distractions, and enables them to adopt agile practices.
- Scrum Team: They are a collection of individuals who work together to ensure that the requirements of the stakeholders are delivered.

▼ Scrum

Scrum is a popular framework that enables teams to work together. Based on Agile principles, Scrum enables the development, delivery, and sustenance of complex projects. It enables teams to hypothesize how they think something works, try it out, learn and reflect from their experiences, and make appropriate changes.

Step 1: Create a Scrum project

Step 2: Create user stories or tasks in the backlog

Step 3: Create a Sprint

Step 4: Hold the Sprint planning meeting

Step 5: Start the Sprint in Jira

Step 6: Hold the daily standup meetings

▼ Step 7: View the Burndown Chart

It is a type of chart that is used to display or represent the amount of work that is remaining to be completed in the project. These charts are very simple and easy to understand.

Step 8: View the Sprint Report

Step 9: Hold the Sprint review meeting

Step 10: Hold the Sprint retrospective meeting

Step 11: Complete the Sprint in Jira

Step 12: Repeat the whole process from step 2

▼ Scrum ban

- Scrum-ban is a methodology that’s a combination of Scrum and Kanban. Scrum-ban can be used to meet the needs of the team, and to minimize the batching of work, and to adopt a pull-based system.
- It ingeniously includes the structure of Scrum and the flexibility and visualization of Kanban.

▼ Scrum Master

A Scrum Master is someone who promotes and supports the usage of Scrum within the team.

- He/She understands the theory, practices, rules and, values of Scrum
- He/She ensures that the team follows the values, principles and, practices of Scrum
- They remove any distractions and impediments that hamper the progress of the project

- The Scrum Master ensures that the team delivers value during the sprint

▼ Scrum of Scrums

- It is a terminology used for scaled agile technologies, which is required to control and collaborate with multiple scrum teams. It is best used in situations where teams are collaborating on complex assignments.
- It is also used to ensure that the required transparency, collaboration, adaption, and adoption are established and to ensure that the products are deployed and delivered.

▼ Scrum vs waterfall

Answer: Scrum and Waterfall are quite different. They are two different methodologies that can be used to deliver any particular project.

Following are the major differences between them:

1. Waterfall is a sequential method where one phase is followed by the other in a sequence. Scrum is more value-driven and is an agile process, which is iterative.
2. In Waterfall approach, the end-user will see the final product near the end. In Scrum, the end-user is involved at each stage of the process right from the design phase.
3. Change management is easy in the scrum, where a change can be incorporated even later in the stage without much cost. In Waterfall, making a change later in the process is very costlier and is generally not feasible.
4. Waterfall is broken into phases, usually referred to as the requirements phase, development phase, testing phase, deployment phase, etc. Scrum is broken down into sprints (usually 2 weeks) in which the planning, development, testing, and deployment happens for a set of features.

▼ Scope Creep

Scope creep is used to describe how a project's requirements tend to grow over time, like - a single deliverable product becomes five when a product with three essential features becomes ten, or when the customer's needs change midway through a project, requiring a reassessment of the project requirements. Changes in project needs from internal miscommunication and disagreements, and key stakeholders are some of the common causes of scope creep.

To manage scope creep, we need to use the change control mechanism to keep it under control. This includes the following -

- Maintaining a baseline scope and keeping track of the project's progress.
- To evaluate actual work performance metrics to the baseline scope, i.e., "How different is the current project from the original plan?", we need to perform Variance analysis.
- Identifying the severity and source of the observed alterations.
- Selecting whether to take preventive or corrective action in response to requests regarding changes.
- To recommend actions and manage all change requests by using the Perform Integrated Change Control method (whether preventive or corrective).

▼ Sprint

- Sprint is a terminology used in Scrum, used to describe a time-boxed iteration.
- During a sprint, a specific module or feature of the product is created.
- The duration of a sprint can vary between a week or two.

▼ Sprint 0 and spike

- Sprint 0 refers to the small amount of effort put in to create a rough skeleton of the product backlog. It also includes insights towards estimating the release of products. Sprint 0 is required for:
    - Creating the project skeleton, along with research spikes
    - Keeping minimal design
    - Developing some stories completely
    - Having low velocity and being lightweight
- The spike is a set of activities that involve Extreme Programming (XP) for research, design, investigation, creating POCs, etc.
- The spike aims to reduce risks of the technical approach, helping gain knowledge to better understand requirements and improve reliability

▼ Sprint meetings

- **Sprint Planning Meeting:** In this meeting, the discussion takes place about features and product backlog items (user stories) that are important to the team. This meeting is usually attended by the product owner, Scrum Master and Scrum Team. It is a weekly meeting and usually lasts for about an hour.
- **Sprint Review Meeting:** In this meeting, the Scrum team gives a demonstration of the product. After this, the product owner determines which items completed and which are not completed. He also adds some additional items to the product backlog on the basis of feedback from customers or stakeholders. Its main aim is to inspect the product being created in the sprint and modify it if required.
- **Sprint Retrospective Meeting:** This meeting takes place after the Sprint planning meeting. In this meeting, the Scrum team meets again to inspect itself and discuss the past mistakes, potential issues and methods to resolve them. Main aim of this meeting is to improve the development process. This meeting lasts for about 2-3 hours.

▼ Sprint retrospective

The sprint retrospective takes place after the sprint review. During this meeting, past mistakes, potential issues, and new methods to handle them are discussed. This data is incorporated into the planning of a new sprint.

▼ Story point

A story point is calculated by taking into consideration the development effort+ testing effort + resolving dependencies and other factors that would require to complete a story.

▼ Time boxing

Timeboxing is the practice of devoting a set amount of time to a single activity. A timebox is a unit of time measurement. A timebox should not exceed 15 minutes in length. A Sprint can be canceled before the Sprint timebox limit ends. Only a Product Owner can cancel the sprint.

▼ User story

- A user story is an agile software development/ project management tool that provides teams with simple, natural language explanations of one or more features of the project that’s written from the perspective of the end-user.

- The user story doesn’t go into detail but only mentions how certain types of work will bring value to the end-user. The end-user, in this case, could be an external component or an internal customer/colleague within the organization.
- They also form the building block of agile frameworks like epics and other initiatives.
- They ensure that the teams work towards the goals of the organization, with the help of epics and initiatives.
- The requirements to make a user story a reality are added later, after discussions with the team.
- They are recorded on post-it notes, index cards, or project management software.

▼ User story mapping

- User story mapping represents and arranges user stories that help with understanding system functionalities, system backlog, planning releases, and providing value to customers.
- They arrange user stories based on their priority on the horizontal axis. On the vertical axis, they are represented based on the increasing levels of sophistication.

▼ User stories vs epics vs tasks

- User Stories: They provide the team with simple explanations of the business’ requirements created from the end user's perspective.
- Epics: An epic is a collection of related user stories. They are usually large and complex.
- Tasks: Tasks are used to break down user stories further. They’re the smallest unit in Scrum that is used to track work. A person or a team of two people usually work on a task.

▼ BBest practices

Best practices are an industry—accepted set of guidelines. They represent the most efficient or sensible course of action.

- Start by planning.

Some developers jump right into coding without a planned approach. This can lead them to rewrite the code again and again.. Unfortunately, rewriting code wastes valuable time. The first step of a project should be to identify the overall goal. Simplifying the user-interface of an e-commerce app or customizing a retail website are both examples of projects goals.

Once the overall goal has been defined, the next step is to plan out how you are going to reach that goal. Planning helps save time and effort in the long run. Development should be significant and precise, and a plan will highlight the relevant aspects and features of a project to focus on.

It may feel like you’re wasting time that could be used to build projects on the planning stages. But, it’s quite the contrary. Planning will make a project organized, minimize the space for errors, and allow you to stay focused without questioning the next step. So, it actually saves you a lot of time.

- Stick to the standards.

  You need to know these steps.

  - The navigation map should be the same for all pages of your website. It makes your website more uniform and coherent.
  - There is an ‘ideal’ position for every feature, particularly on the homepage. While the company logo should be at the top-left corner of the homepage, contact information, organizational details, and registration options should be towards the top-right.
  - The size of the hyperlinks should be appropriate and easily visible to the user.

- Code smarter, not longer.

  Every line of code that you write should fulfill a necessary and particular purpose. Avoid writing anything too complex that could make your code hard to understand and edit in the future. This is particularly important while working in a team, since your team members will often proof read or change each other’s work.

  Writing code without a defined purpose invites bugs into your program. Each line of code should provide an important feature to a webpage or solve a specific problem. Also, you should realize the value of code commenting: a practice of inserting short, single-line notes throughout your code. These notes allow you to easily understand why a code snippet works in a particular way or what is it intended for.

- Sprint, don’t leap.

  Website development is a rigorous process of improvement, and it thrives on incremental updates rather than implementing huge features at once. Whether it’s a minor change in fonts or introducing a new feature, incremental updates create fewer errors and bugs than full updates do.

  The trick is to maintain a balance between today’s demand and future innovation. The user-interface created today will naturally become outdated a few months or years down the line. Therefore, it’s imperative to design and develop the backend accordingly. This makes sure that the User Interface (UI) can be modified in the future without doing a complete overhaul.

- Establish multi-device compatibility.

  Most users visit websites from more than one device. In fact, the majority of your site visitors will likely come from a mobile device. Devices such as smartphones, laptops, and tablets have different screen resolutions as well as form factors. To ensure a quality user experience, it’s important to make sure your data is displayed correctly across all devices.

  This is particularly important for the primary website functions as well as its control elements. The adaptive design of your website is crucial if you want to expand your user base.

- Link your social media channels.

  Social media is the most powerful branding tool in the modern world. It’s indispensable for any online business. Whether a retail company or a food delivery service, a company’s existence on social media helps with its own personalized branding.

  Historically, a corporate image would take years to build through conventional media platforms such as television and print. Social media helps your business build your own unique corporate image quickly. Social media can expedite your services, increase your customer base, and most importantly, highlight your presence in the market.

  As you can see, it’s important to integrate your social media links seamlessly across all platforms. Make the links and witness your company’s image rise to a whole new level!

- Write only by using the coding languages you master.

- Take UX/UI into account - user experience and user interface.
- High performance in web development is also about the download speed.
- Implement Search Engine Optimization (SEO) strategies.

  Search engine optimization is an important part of business success online.
- Explain which div you're closing. (make comments after div tag)
- Use a CSS reset.
- Use the best case style → camel, pascal, snake, kebab case.
- Don't use @import. (don't waste your visitor's time using @import.)
- Don't mix CSS with HTML.
- Don't mix JavaScript with HTML.
- "Smush" your images.
- Use conditional comments.

▼ SSEO

Everyone is aware of the fact that SEO is the most vital digital marketing tool.

importance

SEO is important as it keeps the search results of search engines like Google, Bing, and Yahoo fair. It eliminates or reduces the possibility of manipulating search results. In the absence of SEO, it would be extremely easy to manipulate the search results.

In simple words, SEO is Google's way to determine the rank of sites for the query entered into the search engine. To gain higher SEO ranks, websites must appeal to its visitors along with meeting all other criteria.

Even the users trust search engine due to SEO. Whenever they find a website ranking on top, then they believe that the site is a credible source for their entered query. The ranking is very crucial as it will fetch more clicks and traffic for your site.

The search engine utilizes web crawlers to determine any website's ranking on the search results.

A web crawler is nothing but a bot whose job is to regularly visit the web pages and analysing it as per the specific criteria established by the respective search engine.

Every search engine has its own crawler. For example, Google's crawler name is Googlebot.

Challenges

Optimization of single-page applications is a tough job as it involves several challenges. As we already discussed above that in a SPA, the page first loads on the client-side which acts as the empty container. This empty container is then filled by the content integrated by JavaScript.

Moreover, you will also require a browser for running the script into the SPA. Only after this, it will be able to load the web pages dynamically.

Now, when the search engine bots visit any SPA website, then they won't be able to crawl the page. They can only crawl it if the entire content is already updated in the user's browsers.

In case, if bots don't find any relevant content then they will regard your website as blank and poorly constructed. If this happens, then the search engine won't index your website or web application.

These are not the only reasons that make React development so difficult with respect to SEO. Let's have a look at some other reasons one by one.

solve challenges

Prerendering

Prerendering is one of the common approaches that is used for making single as well as multi-page web apps SEO-friendly. One of the most prominent way of doing it by using the prerendering services such as prerender.io.

https://prerender.io/blog/how-prerender-crawls-and-renders-javascript-websites/

Server-side rendering

If you're looking to build a React web application, then you must know the difference between server-side and client-side rendering.

Client-side rendering means that the Google bot and browser get HTML files or files that have very little content. After that, JavaScript code downloads the content from the server which enables the users to view it on their screens.

If we see it from the SEO perspective, then client-rendering poses few problems. It's because the Google bots get very little to no content and thus they are not able to index it properly.

However, with server-side rendering, the Google bots and browsers can get HTML files along with all the content. This helps Google bots to index the page well.

Server-side rendering is one of the easiest ways to create React web applications that are SEO-friendly. Although, if you need to build a single page application that can render on the server, then you'll require to add Next.js.

Isomorphic React apps

Isomorphic React application is something that can run both client-side as well as the server-side. With the help of isomorphic JavaScript, you can run the React app along with capturing the rendered HTML which is usually rendered by the browser. This rendered HTML file can be then sent to anyone who requests the site.

The HTML file is used as a base by the app on the client-side and then continues operating in the browser as if it was rendered on the browser only.

An isomorphic app determines if the client can run the scripts or not. The code is rendered on the server when a JavaScript is turned off. This enables bots and browser to get all the required meta content and tags in CSS and HTML.

The moment JavaScript is switched on, the first page is rendered on the server which enables the browser to get CSS, HTML, and JavaScript files. After that JavaScript begins running which loads rest of the content dynamically.

This is the reason why the first screen is showed faster. Not only this, but it also makes the app more compatible with older browsers. Even the user interactions are smoother as compared to the client-side rendering of websites.

Developing real-time isomorphic apps can be a pain as it consumes a massive amount of time. However, there are few frameworks that can make real-time isomorphic app development simpler and faster. The two of the most popular frameworks are Gatsby and Next.js.

Gatsby is a free open-source compiler which enables developers to make scalable, fast, and powerful web applications. It's important to notice that Gatsby doesn't offer server-side rendering, rather than that, it generates static website and then stores the generated HTML files on the hosting service or cloud.

This was Gatsby, now let's have a look at Next.js in detail.

▼ BBEM

BEM

It stands for Block(.site-search {}, .site-search__field {}, .site-search—full {}), Element, Modifier and is a smart way of naming your CSS classes to give them more transparency and meaning to other developers.

Why use BEM?

Well, we use BEM because it tries to get the best out of today's CSS concerning modularity.

In a nutshell, first, it avoids inheritance and provides some sort of scope by using unique CSS classes per element like ".my-component__list-item".

Second, it reduces style conflicts by keeping CSS specificity to a minimum level. They were my opinion.

▼ HHard coding

Hard coding (also **hard-coding** or **hardcoding**) is the software development practice of embedding data directly into the source code of a program or other executable object, as opposed to obtaining the data from external sources or generating it at runtime. Hard-coded data typically can only be modified by editing the source code and recompiling the executable, although it can be changed in memory or on disk using a debugger or hex editor. Data that are hard-coded is best for unchanging pieces of information, such as physical constants, version numbers and static text elements.

▼

▼ Back Knowledge

▼

▼ AAnalytical skills

Analytical skills help you assess information, problem-solve, and implement solutions at work.

Analytical skills are problem-solving skills that help you parse data and information to develop creative, rational solutions. An analytical person in the workplace focuses on making sense of the facts and figures and using logical thinking practices to identify a fix.

Analytical skills are the ability to invest

Software development relies heavily on strong analytical skills because of the amount of problem-solving involved in the filed.

- Detect patterns and make connections
- Brainstorm and theorize
- Observe and interpret data
- Integrate new information
- Pick between many solutions
- Make decisions based on multiple factors

Types of analytical skills

- Problem-solving skills
- Critical thinking skills
- Research skills
- Creativity skills
- Communication skills

Analytical thinking helps you develop a better understanding of a complex technical problem so you can solve it. The process is derived from the scientific method and has the following steps:

1. Encounter a problem.
2. Observe the problem and conduct research.
3. Hypothesize possible solutions based on an understanding of the problem and your past knowledge.

4. Evaluate and compare the effectiveness of each solution and the feasibility of implementation.
5. Communicate results and your chosen solution.

---

experience

When I was developing product management application, there were some problems. I needed to solve these problems fast. There were critical bugs to fix and new features to develop for customers. So, I needed to have a reliable process for analyzing problems that allows me to find solutions in an efficient way.

First, I observed it, collected information about it, and tried to understand it. I analyzed the surrounding code to find clues or patterns. Walking through the functions and variables in the code helped me find the root of the problem.

Once I had found the root of the problem, I started brainstorming possible solutions. Past experience with similar problems was also helpful for coming up with reliable solutions. I could also go on Stack Overflow to find out whether anyone else had encountered a similar problem because there were some creative solutions worth exploring.

Once I had identified multiple solutions, I begun to narrow down my options based on factors like technical complexity, impact on performance, cost, availability of talent, etc. I couldn't test all possible solutions so I need to pick the best solution based on the information I had.

Also, I asked for feedback or suggestions from other software developers or tech leads to find the best solution to the problem. As you know, collaborating with other team members is key to finding creative solutions.

When I had decided on my final solution, I communicated my findings in a clear and concise way. This included articulating why my solution was the optimal solution for the problem and what impact it would have on the greater organization.

So, I could solve all problems on time.

▼ AAPI

https://aws.amazon.com/what-is/api/

What is an API?

APIs are mechanisms that enable two software components to communicate with each other using a set of definitions and protocols. For example, the weather bureau’s software system contains daily weather data. The weather app on your phone “talks” to this system via APIs and shows you daily weather updates on your phone.

What does API stand for?

API stands for Application Programming Interface. In the context of APIs, the word Application refers to any software with a distinct function. Interface can be thought of as a contract of service between two applications. This contract defines how the two communicate with each other using requests and responses. Their API documentation contains information on how developers are to structure those requests and responses.

How do APIs work?

API architecture is usually explained in terms of client and server. The application sending the request is called the client, and the application sending the response is called the server. So in the weather example, the bureau’s weather database is the server, and the mobile app is the client.

There are four different ways that APIs can work depending on when and why they were created.

REST APIs

These are the most popular and flexible APIs found on the web today. The client sends requests to the server as data. The server uses this client input to start internal functions and returns output data back to the client. Let’s look at REST APIs in more detail below.

What are REST APIs?

REST stands for Representational State Transfer. REST defines a set of functions like GET, PUT, DELETE, etc. that clients can use to access server data. Clients and servers exchange data using HTTP.

The main feature of REST API is statelessness. Statelessness means that servers do not save client data between requests. Client requests to the server are similar to

URLs you type in your browser to visit a website. The response from the server is plain data, without the typical graphical rendering of a web page.

What are API integrations?

API integrations are software components that automatically update data between clients and servers. Some examples of API integrations are when automatic data sync to the cloud from your phone image gallery, or the time and date automatically sync on your laptop when you travel to another time zone. Enterprises can also use them to efficiently automate many system functions.

What are the benefits of REST APIs?

REST APIs offer four main benefits:

1. Integration

APIs are used to integrate new applications with existing software systems. This increases development speed because each functionality doesn’t have to be written from scratch. You can use APIs to leverage existing code.

2. Innovation

Entire industries can change with the arrival of a new app. Businesses need to respond quickly and support the rapid deployment of innovative services. They can do this by making changes at the API level without having to re-write the whole code.

3. Expansion

APIs present a unique opportunity for businesses to meet their clients’ needs across different platforms. For example, maps API allows map information integration via websites, Android,iOS, etc. Any business can give similar access to their internal databases by using free or paid APIs.

4. Ease of maintenance

The API acts as a gateway between two systems. Each system is obliged to make internal changes so that the API is not impacted. This way, any future code changes by one party do not impact the other party.

▼ FFront end development?

Front end developers build with the user in mind. Front end development is a style of computer programming that focuses on the coding and creation of elements and features of a website that will then be seen by the user. It's about making sure the visual aspects of a website are functional. We can also think of front end as the "client side" of an application. So, let's say we're a front end developer. This means our job is to code and bring to life the visual elements of a website. We'd be more focused on what the user sees when they visit a website or app. And you'd want to make sure the site is easy to interact with while also running smoothly.

Front end developers take the visual designs from UX and UI designers and bring the website to life, making sure it functions well for the user. One of the many ways they could use front end skills is in creating a static website, which is a website with fixed content that's delivered to a user's browser exactly as it's stored. We might run into a static website if we happen upon a simple landing page or a small business website that doesn't allow users to perform any interactive tasks.

Front end developers build elements like: Buttons, Layouts, Navigation, Images, Graphics, Animations, Content organization, etc.

▼ BBack end development?

Back end development focuses on the side of the website users can't see. It's what makes a site interactive. You can also refer to the back end as the "server side" of a website. For instance, let's you're running a social media website. You need an accessible place to store all of your users' information. This storage center is called a database and a few widely used examples include Oracle, SQL Server, and MySQL. Databases are run from a server, which is essentially a remote computer. A back end developer will help manage this database, as well as the site contents stored on it.

This ensures that front end elements on your social media website can continue to function properly as users browse uploaded content and other user profiles.
While users do not directly interact with the back end of a website, they'll indirectly interact with elements these developers work on through a front-end application.
Back end development deals with storing and arranging data while also ensuring the front end is functioning well.

Back end web developers work on tasks like: Building code, Troubleshooting and debugging web applications, Database management, Framework utilization, etc.

▼ Vs. FB front end vs. back end?

That’s very interesting question. I think there are 4 main distinctions that set front and back end development apart.

First of all, front and back end developers have different strengths.

Second, they work in different languages. Let me elaborate further. When you’re coding, you’ll use a programming language. Much like human languages, these languages allow programmers to communicate with their computers through a series of symbols. Very, simply, it’s like giving your computer instructions. Front end developers work in languages like HTML, CSS and JavaScript. Front end also works in its own set of frameworks and libraries such as React, Vue, Sass, etc. But back end developers work in languages like JavaScript and Node.js, Python, Ruby and C++, etc. And back end frameworks include Express, Django, Rails and Laravel.

Third, they have different strengths. Finally, they work together to create awesome applications.

▼ FFront End Skills

- Break apart interesting problems, as well as design engaging solutions.
- Design, create, and modify static web pages that conform to HTML5 specifications.
- Analyze the client-side performance of a webpage to better understand the consumer experience.
- Imagine, create, and deploy interactive and mobile-friendly applications for the web using the latest web technologies, including HTML5, CSS3, JavaScript (ES6+), and React.
- Pair those skills with back-end technologies like databases and Node.js, as well as developer tools like Bash, Git, and automated tests.
- Understand how to effectively work and collaborate on a software project, also how to interview confidently.
- excellent problem-solving skills
- strong knowledge of HTML, CSS, JavaScript

▼ BBack End Skills

- Level up with a second, popular programming language (Python 2 & 3), as well as its own most common web framework, Django. Also use language features like lists, sets, and dictionaries appropriately for simple algorithmic tasks.
- Become adept at interacting with behind-the-scenes technologies, like databases and servers, also at solving more complex sets of problems.
- Identify and fix performance bottlenecks in a web application. Additionally, propose a viable fix to a specific bottleneck in a provided sample application.
- Learn to make applications faster, more secure, more stable, and more capable.

▼ FFull Stack Skills

- Understanding how to best structure code and data
- Communication skills
- Problem solving skills
- Knowing how to connect a database using programming language
- Having the ability to manage data storage using a database
- Attention to detail
- Creativity
- Patience

▼ DDeveloper Skills

- Analytical thinking skills
- Coding expertise
- Knowledge of HTML, CSS, and JavaScript
- Communication skills
- Collaboration skills
- Experience with testing and debugging
- Ability to problem-solve
- Break down problems and design engaging solutions
- Design, create, and modify static web pages that conform to language specifications

- Analyze the client-side performance of a webpage to better understand the consumer experience
- Imagine, create, and deploy interactive and mobile-friendly applications for the web using the latest web technologies
- Understand how to effectively work and collaborate on a software project
- Learn how to interview confidently
- Become adept at interacting with behind-the-scenes technologies, like databases and servers
- Gain the ability to solve more complex sets of problems
- Correctly identify and fix performance bottlenecks in a web application and propose viable solutions
- Functionally improve the speed, security, stability, and capability of applications
- Develop an engineering mindset toward problem solving and receive support in your job search

▼ DDesigner Skills

- Communication skills
- Empathy
- Creativity
- Detail-oriented
- Experience with design software
- Knowledge of SEO and content management systems

▼ FFull-stack developer?

Full-stack developers are web developers who work in both front and back-end development. This means they can complete all tasks involved in web or software development and will often operate as team leader of project management. But before we get into the role of a full-stack developer, it’s important to understand their counterparts: frontend developers and backend developers.

- **Frontend developers** focus on using programming languages to create what the user can see. This side of a website is called the “client-side” or “frontend.”

Frontend developers make web pages visually appealing and functional. Whether it's clickable buttons, scrolling hero images, or rolling videos, frontend developers use coding languages and frameworks to create areas for user interactions within the website. They're also typically proficient in HTML, CSS, and JavaScript.

- **Backend developers** work on the "backend" or "server-side" of a website. This is the side of a website that users can't see or interact with directly. Additionally, this is where databases are stored. This type of development ensures a website continues to run efficiently. Backend developers will eliminate bugs as they come into play.

Frontend developers work in sync with backend developers. Each has their own responsibilities in the project development process. Together, they utilize specific coding languages exclusive to either front or backend development to create one website or application. Without either position, a finished product does not exist.

Now, full-stack developers operate using both frontend and backend skills. Though knowledgeable of both web development skills, full-stack developers aren't required to write code for an entire website or application on their own. They work as a part of a team to design, code, and launch a website or application.

▼ FFrontend engineer?

What is frontend engineer?

When designing web applications, front-end engineers work to perfect a website's performance to ensure processes such as loading speed, security, proper coding, and overall efficiency. While duties vary by employer, they often work within a team that develops, improves, or maintains various websites.

What does a frontend engineer do?

Front end engineers plan, design, build, and implement the user interface systems of websites, software programs, and web-based applications. Their primary goal is to provide a satisfactory user experience with no issues, errors, or downtime. They create and optimize systems, play an active role in testing and troubleshooting, and resolve issues such as those related to accessibility and browser compatibility. They often serve as an intermediary bridging the gap between the user and the backend developers, and tweak tools and platforms to adapt to real-world use situations. Front end engineers typically have a bachelor's degree in computer science or a related field, and a background in software development. They must be extremely proficient with the use of programming languages such as JavaScript and CSS. Experience troubleshooting compatibility and accessibility issues is helpful.

What responsibilities are common for front end engineer jobs?

Work closely with the product, design, and engineering teams.
Research new technologies and propose solutions to user needs.
Identify UI problems and bugs and devise elegant solutions.
Participate in code reviews and ship code on a daily basis.
Develop fully functional web applications that align with business objectives.
Mentor and onboard junior members of the team.
Work directly with the client to define and clarify requirements.
Accept payments, ensuring all prices and quantities are accurate.
Follow the best practices in developments and improve them.
Create high resolution mockups to test interfaces with users.
Complete all required paperwork and documentation according to guidelines and deadlines as assigned.
Exemplify both technical expertise and empathy.
Overhaul user interfaces to optimize for speed and ease of use.
Expedite all cashiers and service desk transactions.
Ensure customer service is provided in a timely fashion.
Assist in the test strategy and execution of master test plan.

Analyze business requirements
Architect the structure of the system
Production build and deployment

Understand the differences between developers and front end engineers.
A front-end engineer is different from a front-end developer, and some developers expand their knowledge to become front-end engineers. Many of the classes and certifications used in both jobs are the same, but front-end engineers spend more time working on programs and apps than they do developing websites.

▼ UUI (UI, design best practice)

https://medium.com/@FlowMapp/bad-bad-ui-10-common-mistakes-in-user-interfaces-ac89767ac43d

Please, don’t do this

- **Grey text on colored backgrounds**

- **Filling the whole screen**
- **User-uploaded content**
- **Overloaded style**
- **Bad font**
- **Uninviting CTAs**
- **No social proof**
- **Too much text**
- **Too complex navigation**
- **Poor padding and spacing**

▼ OOptimize React?

How to optimize React?

- Use React & TypeScript
- React DevTools -> React tools are used for performance analysis with the help of profiler
- Content Delivery Network(CDN) -> Using a content delivery network improves the speed and performance of the site.
- Using CSS Animation
  It is recommended to use CSS animations instead of JavaScript animations as they contain UI and separate elements.
  For eg: Change the state of user interface elements and display the exact layout you want without letting your web browser crash or locking the user interface. You also have more control over the duration and repetition of the animation and change multiple functions at the same time.
- Using web workers
  With Web Workers, the browser can run scripts in the background and the web application can perform multiple tasks at the same time to animate, manipulate DOM elements, process response data, direct UI interactions and more. It provides additional workloads to remove workload from the main "thread" and performs operations separately. It does this without disturbing the main "string", which reduces the execution load on the main "string" and increases rendering and processing speed.

- Error boundaries
  Component rendering produces an unexpected error that causes the components to become empty. But you can avoid such situations with margins of error. This will help you identify and keep track of JavaScript errors that exist throughout the substructure. It also shows a feedback interface that makes it easy to deal with "bugs". Note that only class components have an error limit. It only records errors in the components below the tree and cannot catch the error within itself.
- Use Jest and Enzyme for Testing
  These are testing frameworks. If used wisely, it will be useful in the React application development process. Fast, contains an extensive simulation library, controls dependencies, controls the main time, controls timer functions, supports ES6 classes and takes snapshots. In addition, Enzyme is a JavaScript test tool that exposes multiple components, scrolls and interacts while using virtual DOM. ReactJS developers often use them to improve React application development.
- Dependency Optimization
- Spliting the Code
  Used to break up the base code into component packages. This reduces the amount of "code" required to compile, resulting in faster page loading. It also helps with slow loading as it only loads components that are currently in use. Single-page React applications use tools like Webpack, Browserify and Rollup to group files. Sharing "code" improves performance and provides a better development experience.

▼ WWebpack

Webpack is a static module bundler for modern JavaScript applications.

- Entry
- Output
- Loaders
- Plugins
- Mode
- Browser Compatibility

Entry

An **entry point** indicates which module webpack should use to begin building out its internal dependency graph. Webpack will figure out which other modules and libraries that entry point depends on (directly and indirectly).

▼ RRollup.js

Overview

Rollup is a module bundler for JavaScript which compiles small pieces of code into something larger and more complex, such as a library or application. It uses the new standardized format for code modules included in the ES6 revision of JavaScript, instead of previous idiosyncratic solutions such as CommonJS and AMD. ES modules let us freely and seamlessly combine the most useful individual functions from your favorite libraries. This will eventually be possible natively everywhere, but Rollup lets you do it today.

▼ BBundle

Bundling

Most React apps will have their files “bundled” using tools like Webpack, Rollup or Browserify. Bundling is the process of following imported files and merging them into a single file: a “bundle”. This bundle can then be included on a webpage to load an entire app at once.

▼ How does the internet work, what?

What?

The internet is a vast, sprawling collection of networks that connect to each other. (interconnected networks)

The internet is a worldwide computer network that transmits a variety of data and media across interconnected devices. It works by using a packet routing network that follows Internet Protocol (IP) and Transport Control Protocol (TCP).

Why?

Developing software is usually easier if you break your project into smaller separate pieces, since that often removes unexpected interactions and dramatically reduces the complexity of the problems you'll need to solve, and simply writing smaller projects in the first place isn't necessarily the answer. Unfortunately, JavaScript has not historically included this capability as a core feature in the language.

This finally changed with the ES6 revision of JavaScript, which includes a syntax for importing and exporting functions and data so they can be shared between separate scripts. The specification is now fixed, but it is only implemented in modern browsers and not finalized in Node.js. Rollup allows you to write your code using the new module system, and will then compile it back down to existing supported formats such as CommonJS modules, AMD modules, and IIFE-style scripts.

▼ QQA (Quality Assurance) and QQC (Quality Control)

https://www.bam.tech/article/a-developers-guide-to-quality-assurance

---

QC focuses mainly on fulfilling quality requirements. It makes sure that the end product is working as expected and is free of bugs.

Quality Assurance is much more than that. By focusing on the entire process, it builds engineers' confidence over quality, improves testing and development efficiency, and by that it makes us deliver faster in smaller batches. The end goals of QA are better SDLC processes, team scalability, and agility. Quality and bugs-free products are (only) derivatives of that.

---

steps

1. Draw the tree of paths.
2. Write a test plan. (It is a roadmap of our tests. I then break down every test case in a series of actions to be performed and their expected results. In our project, there are 9 ways to create an account by accepting the Terms & Conditions aside from Google, Weibo & Facebook options.)
3. Include boundary tests to your test plan.
4. Execute the test plan.
5. Isolate component tests.
6. Monkey test our app

   - Go offline

   - Press buttons several times in a row quickly

   - Open & Close the keyboard several time

   - Open background applications and reopen your application (Multi tasking on your phone)

- Go from portrait to landscape

- Enter special characters in search inputs

- Play with sliders

- Scroll rapidly and repeatedly

- Focus on the back end API calls. Go offline, kill the app, go back and forth around those. This is where it helps to have a developer's vision and understand the ins and outs of the application.

▼

▼ Hunting developers

▼

▼

▼ Adding profile info

▼ Job post for hiring

This is Colin. Self-motivated, energetic collaborator who thrives on working in a fast-paced environment with tight deadlines and have a desire to stay updated on current and new technologies.

As a passionate full-stack developer with can-do attitude in Java script Framework, I'm thrilled to leverage my technical experience to implement client's ideas into reality.

Recently, leaving a contract job from comcate, looking for an opportunity to work in remote position.

DM is the best way to reach out.

Familiarity with the non-profit environment is a plus.

I'm in vacation right now, **this why so long response**. Sorry for that.

We're **thrilled** that you'd like to join us here at Hims & Hers.

Looking for a small team (greater than 5 engineers) that resolves hard/interesting problems. Prefer to work on user-facing stuff. My strengths are mostly in backend, but I can do HTML/CSS, HAML./SASS ,JS(React, jQuery, etc) . I'd like to work in a mission-driven company that enables people, especially if it is online education.

Passionate about creating something completely new in the world that people need. Driven by creating technology that helps people be more human. Do you believe, with the right guidance and support, everyone has the power to improve their life and deserves that opportunity?

We’re looking for an expert full stack, Backend engineer, to help grow our product and team and work closely with the founding team to help make Coral the indispensable sexual wellness resource.

It was crazy, I have been left from busy stuff, taking some part-time jobs meanwhile. Now I am looking for an another challenge with full time availability and super excited with JAMStack - Strapi/Nuxt/Gatsby/DatoCMS.

Hello, Nice to meet you. Hope you're doing well. As a passionate full-stack developer with can-do attitude in Java script Framework, I'm excited to leverage my technical expertise to implement your ideas into reality. Nobody disappointed me. You will be good luck with me. I ensure that
Looking for a good opportunity from you. thanks. Colin

▼ LinkedIn note

Hello, Nice to meet you. Hope you're doing well. As a passionate full-stack developer with can-do attitude in Java script Framework, I'm excited to leverage my technical and hands-on experience to implement your ideas into reality.
Looking for a good opportunity from you. thanks. Colin

▼

▼