MCQS Preparation for NEXT.JS (GIAIC Quarter 2)

Important Topics for Mcq's:

1. Next.js Fundamentals

- SSR (Server-Side Rendering)
- SSG (Static Site Generation)
- ISR (Incremental Static Regeneration)
- File-based Routing
- Dynamic Routes
- Catch-all Routes

2. Data Fetching

- getStaticProps
- getServerSideProps
- getInitialProps
- API Routes
- Static Data vs. Server Data

3. State Management

- useState and useEffect Hooks
- Context API
- Redux or any state management library integration
- Prop drilling and component state

4. Routing and Navigation

- Link Component
- Programmatic Routing (useRouter hook)
- Nested Routes
- Dynamic Routing

5. Styling

- Tailwind CSS integration
- CSS Modules
- Styled-Components
- Global CSS
- Media Queries and Responsiveness

6. Performance Optimization

- Code Splitting
- Lazy Loading and Dynamic Imports
- Image Optimization with next/image
- Prefetching and Preloading
- Performance and SEO best practices

7. Authentication and Authorization

- JWT Authentication in Next.js
- Server-side sessions
- Auth providers (OAuth, Google, etc.)
- Secure routes and private pages

8. Deployment and Hosting

- Vercel Deployment (automatic optimizations)
- Next.js deployment strategies
- Custom Server setup (e.g., Express with Next.js)
- Environment Variables

9. Error Handling and Debugging

- Try-Catch in async functions
- API error handling
- Custom Error Pages (_error.js)
- Logging and monitoring tools

10. Advanced Concepts

- Middleware in Next.js
- Custom Document (_document.js)
- Custom App (_app.js)
- Working with third-party APIs
- SSR vs SSG vs Client-side rendering (CSR)

11. SEO Optimization

- Head Component (next/head)
- Meta tags and Open Graph tags
- Structured Data (JSON-LD)
- Sitemap Generation
- Robots.txt

12. Testing and Quality Assurance

- Unit Testing with Jest and React Testing Library
- Integration Tests with Next.js
- E2E Testing with Cypress
- Performance Testing

13. API Integration

- Fetching data from external APIs
- CRUD operations using API Routes
- Authentication for API calls

14. Modular Component Design

- Reusable components
- Custom Hooks
- Atomic Design Methodology

[illegible]

Practice MCQ's Exam:

MODE: EASY

---

1. Data Fetching

1.1. What is the primary difference between getStaticProps and getServerSideProps?

A) getStaticProps fetches data on every request, while getServerSideProps fetches data only at build time.
B) getStaticProps fetches data at build time, while getServerSideProps fetches data on every request.
C) getStaticProps is used for client-side data fetching, while getServerSideProps is used for API routes.
D) Both methods fetch data at build time but differ in how they cache the results.

1.2. Which data-fetching method in Next.js allows you to build pages that are only generated on request and have the latest data?

A) getStaticProps
B) getServerSideProps
C) getInitialProps
D) useEffect

---

2. Routing and Navigation

2.1. Which of the following Next.js features allows dynamic routes to be generated based on file system structure?

A) Static Site Generation
B) Dynamic Imports
C) File-based Routing
D) API Routes

2.2. What does the useRouter hook in Next.js allow you to do?

A) Fetch data from a server.
B) Manage the state of your app.
C) Access the Next.js router and navigate programmatically.
D) Optimize page rendering.

2.3. Which of the following methods would you use to create a catch-all route in Next.js?

A) pages/[param]/[...catchAll].js
B) pages/[param].js
C) pages/[...catchAll].js
D) pages/[param]/[catchAll].js

---

3. Styling

3.1. Which of the following is NOT a valid way to style a Next.js app?

A) CSS Modules
B) Tailwind CSS
C) Inline styles using JavaScript objects
D) CSS-in-JS with styled-components

3.2. How does Next.js support global styles?

A) Through CSS Modules only.
B) By defining global styles in styles/global.css and importing it in _app.js.
C) By defining styles directly in the pages folder.
D) Through inline styles added in _document.js.

3.3. What is the benefit of using Tailwind CSS in a Next.js project?

A) It allows you to write custom CSS in JavaScript files.
B) It provides a utility-first approach for rapid UI development.
C) It automatically generates optimized CSS files at build time.
D) It supports only responsive design out of the box.

---

4. API Routes

4.1. How do you create an API route in Next.js?

A) By adding a file to the pages/api directory.
B) By creating a custom server with Express.
C) By modifying next.config.js.
D) By using getStaticProps.

4.2. Which of the following is true about middleware in Next.js?

A) Middleware allows you to intercept and modify requests before reaching a page or API route.
B) Middleware is only available for API routes.
C) Middleware runs only on the client-side.
D) Middleware can be used only for error handling.

4.3. How can you handle authentication with API routes in Next.js?

A) By using getServerSideProps to fetch authenticated data.
B) By storing JWT tokens in cookies and verifying them in API routes.
C) By using getStaticProps to fetch data from a secure API.
D) By using the useRouter hook to handle authentication.

---

5. Performance Optimization

5.1. How can you reduce the initial page load time in Next.js?

A) By using getInitialProps on all pages.
B) By using next/image for image optimization.
C) By preloading all images with next/image.
D) By loading CSS files synchronously in the head.

5.2. Which of the following strategies can be used to improve the performance of a Next.js app?

A) Loading images as base64 encoded images.
B) Using static generation for all pages that don't require real-time data.
C) Implementing getServerSideProps for all pages.
D) Always using client-side fetching in useEffect.

5.3. In Next.js, which of the following techniques helps with automatic image optimization?

A) next/image component
B) Manual resizing and format optimization
C) getStaticProps
D) Preloading images with Link

---

6. Deployment

6.1. Which deployment platform is specifically optimized for Next.js applications?

A) Heroku
B) Netlify
C) Vercel
D) AWS Lambda

6.2. What is the main advantage of deploying a Next.js app on Vercel?

A) Automatic optimizations for images, JavaScript, and CSS.
B) Free hosting without any configuration.
C) Vercel automatically adds serverless functions for every page.
D) It requires no GitHub integration.

6.3. When deploying a Next.js app to Vercel, how are static pages handled?

A) They are rebuilt on every request.
B) They are served from the server on every request.
C) They are pre-built and served from a global CDN.
D) Vercel serves them as static files without any optimizations.

---

7. Miscellaneous

7.1. What is the purpose of _app.js in Next.js?

A) To provide global settings for the app like authentication.
B) To define the HTML structure for each page.
C) To handle routing logic.
D) To manage the state of all components in the app.

7.2. What Next.js feature can be used to pre-render dynamic pages for better SEO?

A) getServerSideProps
B) getInitialProps
C) getStaticPaths with getStaticProps
D) useEffect with server-side rendering

7.3. Which of the following is NOT a feature of Next.js?

A) Static Site Generation (SSG)
B) Server-Side Rendering (SSR)
C) Automatic Code Splitting
D) Full-stack backend development with Node.js

Answer Key to EASY MODE

---

1. Data Fetching

1.1.
Answer: B) getStaticProps fetches data at build time, while getServerSideProps fetches data on every request.

1.2.
Answer: B) getServerSideProps

---

2. Routing and Navigation

2.1.
Answer: C) File-based Routing

2.2.
Answer: C) Access the Next.js router and navigate programmatically.

2.3.
Answer: A) pages/[param]/[...catchAll].js

---

3. Styling

3.1.
Answer: D) CSS-in-JS with styled-components

3.2.
Answer: B) By defining global styles in styles/global.css and importing it in _app.js.

3.3.
Answer: B) It provides a utility-first approach for rapid UI development.

---

4. API Routes

4.1.
Answer: A) By adding a file to the pages/api directory.

4.2.
Answer: A) Middleware allows you to intercept and modify requests before reaching a page or API route.

4.3.
Answer: B) By storing JWT tokens in cookies and verifying them in API routes.

---

5. Performance Optimization

5.1.
Answer: B) By using next/image for image optimization.

5.2.
Answer: B) Using static generation for all pages that don't require real-time data.

5.3.
Answer: A) next/image component

---

6. Deployment

6.1.
Answer: C) Vercel

6.2.
Answer: A) Automatic optimizations for images, JavaScript, and CSS.

6.3.
Answer: C) They are pre-built and served from a global CDN.

---

7. Miscellaneous

7.1.
Answer: A) To provide global settings for the app like authentication.

7.2.
Answer: C) getStaticPaths with getStaticProps

7.3.
Answer: D) Full-stack backend development with Node.js

MODE: NORMAL

---

1. Data Fetching

1.1. Which of the following statements about getStaticProps is correct?

A) It is executed at runtime on every request.
B) It is executed only at build time.
C) It cannot be used with dynamic routes.
D) It always fetches the latest data from the server.

1.2. Which function does Next.js use to pre-render pages at build time?

A) getServerSideProps
B) getStaticProps
C) getInitialProps
D) useEffect

1.3. When using getServerSideProps, what is the purpose of the context parameter?

A) It provides data fetched during the build process.
B) It gives access to the request object and other information about the page.
C) It is used for static generation of the page.
D) It allows you to access session information for the current user.

---

2. Routing and Navigation

2.1. What is the correct file structure for a dynamic route with a parameter in Next.js?

A) pages/[param].js
B) pages/{param}.js
C) pages/[param]/index.js
D) pages/param.js

2.2. Which of the following is the correct way to access query parameters in Next.js?

A) useRouter().query
B) useQueryParams()
C) getInitialProps().query
D) next.query()

2.3. How do you handle route transitions in Next.js using the useRouter hook?

A) router.push('/new-route')
B) router.transitionTo('/new-route')
C) navigate('/new-route')
D) next.js.push('/new-route')

---

3. Performance Optimization

3.1. Which of the following techniques improves Next.js app performance by reducing the size of JavaScript bundles?

A) Code splitting
B) Lazy loading of images
C) Using getServerSideProps
D) Using global CSS files

3.2. What does the next/image component do in Next.js?

A) It automatically adjusts the size and format of images for optimization.
B) It allows for manual resizing of images.
C) It is used for lazy loading images.
D) It fetches images from a content delivery network (CDN).

3.3. Which of the following features is used to preload fonts in Next.js for better performance?

A) <link rel="preload" href="font-url" />
B) next/font package
C) getStaticProps with fonts parameter
D) Preloading images using next/image

---

4. Styling

4.1. Which of the following Next.js styling methods enables scoped styles within components?

A) Tailwind CSS
B) CSS-in-JS with styled-components
C) Global CSS
D) CSS Modules

4.2. What is the benefit of using CSS Modules in Next.js?

A) It makes global styles easy to manage.
B) It automatically loads styles based on the component's name.
C) It scopes styles to the current component, avoiding conflicts.
D) It only works with class-based components.

4.3. How can you add Tailwind CSS to a Next.js project?

A) Manually import Tailwind CSS files into _app.js.
B) Use the Tailwind CLI to build CSS files.
C) Install the Tailwind package, configure it in tailwind.config.js, and import tailwind.css into _app.js.
D) Add Tailwind via a CDN link.

---

5. API Routes and Middleware

5.1. Which of the following Next.js functions is used to handle API routes?

A) next.js/api()
B) getStaticProps()
C) pages/api/
D) serverSideProps()

5.2. How can you handle authentication in API routes in Next.js?

A) Use getStaticProps to authenticate users.
B) Store authentication tokens in cookies and check them in API route handlers.
C) Use the useRouter hook to check the session.
D) Use middleware to authenticate API requests.

5.3. How can you handle errors in API routes?

A) By using the next callback function in the API handler.
B) By catching errors within try-catch blocks in the API route handler.
C) By setting a statusCode header in the response.
D) By using custom error components in pages directory.

---

6. Deployment

6.1. What is the main advantage of deploying a Next.js app to Vercel?

A) Automatic serverless function generation.
B) Automatic image optimization for all pages.
C) Pre-rendering static pages for SEO.
D) Automatic database integration.

6.2. Which of the following statements about static page generation in Next.js is true?

A) Static pages are generated at runtime for each request.
B) Static pages can only be generated with getStaticProps.
C) Static pages are pre-built and served from a CDN.
D) Static pages require a database connection for rendering.

6.3. How does Next.js handle server-side rendering (SSR) when deployed on Vercel?

A) SSR happens at build time and is served from a CDN.
B) SSR happens at runtime on every request.
C) SSR happens in the browser using JavaScript.
D) SSR is not supported in Vercel.

---

7. Miscellaneous

7.1. What does the _app.js file in Next.js do?

A) It wraps all pages and can be used to add global CSS, state management, and layout components.
B) It is used to add custom server logic.
C) It defines the layout for a specific page.
D) It contains API route handlers.

7.2. What is the purpose of the getStaticPaths method in Next.js?

A) It pre-fetches data for static generation.
B) It defines a set of paths that should be pre-rendered at build time.
C) It is used to fetch data on each request.
D) It allows for dynamic routing.

7.3. How can you fetch data client-side in Next.js?

A) Using getStaticProps
B) Using getServerSideProps
C) Using useEffect and fetch or axios
D) Using getInitialProps

Answer Key to NORMAL MODE

---

1. Data Fetching

1.1. Which of the following statements about getStaticProps is correct?
Answer: B) It is executed only at build time.

1.2. Which function does Next.js use to pre-render pages at build time?
Answer: B) getStaticProps

1.3. When using getServerSideProps, what is the purpose of the context parameter?
Answer: B) It gives access to the request object and other information about the page.

2. Routing and Navigation

2.1. What is the correct file structure for a dynamic route with a parameter in Next.js?
Answer: A) pages/[param].js

2.2. Which of the following is the correct way to access query parameters in Next.js?
Answer: A) useRouter().query

2.3. How do you handle route transitions in Next.js using the useRouter hook?
Answer: A) router.push('/new-route')

3. Performance Optimization

3.1. Which of the following techniques improves Next.js app performance by reducing the size of JavaScript bundles?
Answer: A) Code splitting

3.2. What does the next/image component do in Next.js?
Answer: A) It automatically adjusts the size and format of images for optimization.

3.3. Which of the following features is used to preload fonts in Next.js for better performance?
Answer: B) next/font package

4. Styling

4.1. Which of the following Next.js styling methods enables scoped styles within components?
Answer: D) CSS Modules

4.2. What is the benefit of using CSS Modules in Next.js?
Answer: C) It scopes styles to the current component, avoiding conflicts.

4.3. How can you add Tailwind CSS to a Next.js project?
Answer: C) Install the Tailwind package, configure it in tailwind.config.js, and import tailwind.css into _app.js.

---

5. API Routes and Middleware

5.1. Which of the following Next.js functions is used to handle API routes?
Answer: C) pages/api/

5.2. How can you handle authentication in API routes in Next.js?
Answer: B) Store authentication tokens in cookies and check them in API route handlers.

5.3. How can you handle errors in API routes?
Answer: B) By catching errors within try-catch blocks in the API route handler.

---

6. Deployment

6.1. What is the main advantage of deploying a Next.js app to Vercel?
Answer: A) Automatic serverless function generation.

6.2. Which of the following statements about static page generation in Next.js is true?
Answer: C) Static pages are pre-built and served from a CDN.

6.3. How does Next.js handle server-side rendering (SSR) when deployed on Vercel?
Answer: B) SSR happens at runtime on every request.

---

7. Miscellaneous

7.1. What does the _app.js file in Next.js do?
Answer: A) It wraps all pages and can be used to add global CSS, state management, and layout components.

7.2. What is the purpose of the getStaticPaths method in Next.js?
Answer: B) It defines a set of paths that should be pre-rendered at build time.

7.3. How can you fetch data client-side in Next.js?
Answer: C) Using useEffect and fetch or axios

MODE: EXPERT

---

1. Data Fetching

1.1. In Next.js, what happens when both getStaticProps and getServerSideProps are defined for the same page?

A) getStaticProps will be used exclusively.
B) getServerSideProps will be used exclusively.
C) The page will fail to build.
D) Both methods will be executed together.

1.2. How does Next.js handle caching for pages generated with getServerSideProps?

A) By default, the cache expires immediately after the page is generated.
B) Next.js caches the page indefinitely until you clear the cache.
C) The cache is automatically managed and expires based on the Cache-Control header.
D) It does not cache pages generated by getServerSideProps.

1.3. How do you generate static pages with dynamic parameters using getStaticProps in combination with getStaticPaths?

A) You define dynamic routes directly in getStaticPaths.
B) You use getStaticProps without needing to define getStaticPaths.
C) getStaticPaths is used to specify which paths to pre-render at build time.
D) getStaticPaths is optional and is only used in server-side rendered pages.

---

2. Routing and Navigation

2.1. What is the correct way to handle nested dynamic routes in Next.js?

A) Use pages/[...param] in the parent folder and define the children routes in separate files.
B) Use pages/[param] and add a parameter for each nested route.

C) Define the parent route in a pages/[param]/index.js file, and nested routes in pages/[param]/[child].js.
D) Define the parent route in pages/[param].js and use useRouter for navigation.

2.2. Which Next.js hook allows you to programmatically navigate between pages while retaining state and scroll position?

A) useRouter
B) useState
C) useEffect
D) useNavigate

2.3. How would you handle route transitions with animated pages in Next.js?

A) Use next/dynamic to lazy load each page with an animation.
B) Use useEffect and CSS animations for handling transitions between pages.
C) Use next/router and CSS-in-JS libraries to manage transitions.
D) Use framer-motion and getServerSideProps for page transitions.

---

3. Performance Optimization

3.1. What is the purpose of next/dynamic in Next.js?

A) It enables client-side routing between dynamically generated pages.
B) It allows for dynamic imports of JavaScript and React components to improve initial page load.
C) It optimizes the images automatically for faster load times.
D) It reduces the server-side rendering time.

3.2. How can you optimize images in Next.js beyond using next/image?

A) Preload images with the preload tag.
B) Use the useEffect hook to delay image loading until the user scrolls to the image.
C) Leverage next/font to optimize fonts before images.
D) Compress images using server-side logic before delivery.

3.3. Which Next.js feature can help you analyze and optimize the size of your JavaScript bundles?

A) next/bundle-analyzer
B) next/performance
C) next/split-chunks
D) next/analyze

---

4. Advanced Styling

4.1. How can you implement a global state with styled-components in Next.js?

A) By passing props down from the _app.js component.
B) By using React Context to manage global styles across components.
C) By using a ThemeProvider to apply global styles.
D) By defining the global styles inside _document.js and applying them to the body tag.

4.2. What is the role of getInitialProps in styled-components in a Next.js app?

A) It helps to fetch external stylesheets for the app.
B) It allows for server-side rendering of styled-components.
C) It is used to dynamically adjust the theme based on user preferences.
D) It provides a way to inject CSS-in-JS into the head.

4.3. Which CSS-in-JS solution is fully compatible with Next.js's static rendering methods like getStaticProps?

A) styled-components
B) emotion
C) tailwindcss
D) CSS modules

---

5. API Routes and Middleware

5.1. What is the purpose of middleware in Next.js?

A) It allows you to intercept requests and modify them before reaching the page or API route.
B) It can be used only for handling authentication in API routes.
C) Middleware is responsible for routing between different pages.
D) Middleware is used to perform static file management in the build process.

5.2. How can you implement role-based authentication in Next.js API routes?

A) By using custom headers in the request to authenticate roles.
B) By adding middleware that checks for role permissions before sending the response.
C) By storing roles in cookies and verifying them with each API request.
D) By modifying the getServerSideProps function to handle user roles.

5.3. How do you access the req and res objects in Next.js API routes?

A) By using the useRouter hook inside the API handler function.
B) By defining the API route handler as an asynchronous function.
C) By passing them as parameters to the handler function in the pages/api folder.
D) They are automatically available in the global scope of API routes.

---

6. Deployment

6.1. What happens when you deploy a Next.js app with Incremental Static Regeneration (ISR) enabled?

A) The page is regenerated on every request.
B) The page is pre-rendered at build time and updated only when necessary.
C) The page is never cached and always served freshly.
D) ISR is not supported when deployed to Vercel.

6.2. Which of the following Next.js configuration files is used to control image optimization settings?

A) next.config.js
B) next/image.js
C) next/image-config.js
D) next/optimize.js

6.3. When deploying a Next.js app to a serverless environment, which method is used to ensure optimal page rendering performance?

A) Enable the ssr option for pages requiring server-side rendering.
B) Use getServerSideProps exclusively for dynamic routes.
C) Pre-render pages at build time using getStaticProps and getStaticPaths.
D) Always use API routes for every page to ensure dynamic data fetching.

---

7. Advanced Concepts

7.1. What is the role of next/plugin in a Next.js project?

A) It allows you to customize the Next.js server for additional functionality.
B) It provides support for integrating third-party services.
C) It optimizes the build and deployment process.
D) It enables plugins to extend Next.js capabilities for dynamic routing.

7.2. What is the advantage of using getServerSideProps over getStaticProps in a Next.js application?

A) getServerSideProps allows the page to be built once and cached, while getStaticProps regenerates on each request.
B) getServerSideProps ensures that the data fetched is always up to date on every request.
C) getStaticProps only works for static pages, while getServerSideProps is only for dynamic pages.
D) getServerSideProps does not run on the server, but getStaticProps does.

7.3. How can you implement client-side rendering with code splitting in Next.js?

A) By wrapping components in dynamic imports using next/dynamic.
B) By manually dividing the JavaScript code into separate files.
C) By using the useEffect hook to import components only when needed.
D) By configuring the splitChunks option in next.config.js.

Answer Key to Expert Mode:

---

1. Data Fetching

1.1. C) The page will fail to build.
1.2. C) The cache is automatically managed and expires based on the Cache-Control header.
1.3. C) getStaticPaths is used to specify which paths to pre-render at build time.

---

2. Routing and Navigation

2.1. C) Define the parent route in a pages/[param]/index.js file, and nested routes in pages/[param]/[child].js.

2.2. A) useRouter
2.3. B) Use useEffect and CSS animations for handling transitions between pages.

---

3. Performance Optimization

3.1. B) It allows for dynamic imports of JavaScript and React components to improve initial page load.
3.2. B) Use the useEffect hook to delay image loading until the user scrolls to the image.
3.3. A) next/bundle-analyzer

---

4. Advanced Styling

4.1. C) Leverage next/font to optimize fonts before images.
4.2. B) It allows for server-side rendering of styled-components.
4.3. A) styled-components

---

5. API Routes and Middleware

5.1. A) It allows you to intercept requests and modify them before reaching the page or API route.
5.2. B) By adding middleware that checks for role permissions before sending the response.
5.3. C) By passing them as parameters to the handler function in the pages/api folder.

---

6. Deployment

6.1. B) The page is pre-rendered at build time and updated only when necessary.
6.2. A) next.config.js
6.3. C) Pre-render pages at build time using getStaticProps and getStaticPaths.

---

7. Advanced Concepts

7.1. A) It allows you to customize the Next.js server for additional functionality.
7.2. B) getServerSideProps ensures that the data fetched is always up to date on every request.
7.3. A) By wrapping components in dynamic imports using next/dynamic.
