Unit-1: Concepts of NoSQL: MongoDB

Short Questions:

1. What are the advantages of using NoSQL databases like MongoDB over traditional relational databases?

2. Name five common data types supported by MongoDB.

3. How do you create a new database in MongoDB?

4. Explain the purpose of CRUD operations in MongoDB.

5. What is the role of projection operators in MongoDB queries?

6. How can you limit the number of documents returned in a MongoDB query?

7. What is aggregation in MongoDB, and why is it useful?

Long Questions:

1. Discuss the key features of NoSQL databases and how they differ from traditional relational databases. Provide examples.

2. Explain the various data types available in MongoDB, and provide use cases for each.

3. Walk through the steps to create a new database and drop an existing database in MongoDB.

4. Describe the CRUD operations in MongoDB, providing examples of each operation.

5. Explore the different projection operators available in MongoDB, and illustrate their usage with examples.

6. Explain the concept of aggregation in MongoDB and provide examples of aggregation commands.

Unit-2: Fundamentals of React.js

Short Questions:

1. What is React.js, and why is it popular in web development?

2. How do you use React with HTML?

3. What are props in React, and how are they used to pass data between components?

4. What is a class component in React?

5. How do you handle conditional statements and operators in React?

6. Explain React events and how to add event handlers.

7. What is the purpose of the event object in React?

Long Questions:

1. Provide an overview of React.js and its key concepts. Discuss why React is considered a popular choice for building web applications.

2. Explain the process of using React with HTML and the benefits of doing so.

3. Discuss the concept of components within components in React and provide examples.

4. Describe how data is passed through props in React and why it's important.

5. Explore class components in React, covering conditional statements, operators, and lists.

6. Explain React events, including how to add events, pass arguments, and use event objects.

Unit-3: Forms and Hooks in React.JS

Short Questions:

1. How can you add and handle forms in React?

2. What is the purpose of `event.target.name` and `event.target.value` in form handling?

3. Name two form-related components in React.

4. What are React hooks, and what advantages do they offer?

5. Explain the usage of `useState` and `useEffect` hooks in React.

6. How can you create a custom hook in React, and what are its advantages?

Long Questions:

1. Discuss the process of adding and handling forms in React, including form submission.

2. Explain the significance of `event.target.name` and `event.target.value` in form handling, providing practical examples.

3. Describe the components TextArea and Drop-down List (SELECT) in React and how they are used.

4. Explore the concept of React hooks, including `useState`, `useEffect`, and their advantages in managing state and side effects.

5. Provide an in-depth explanation of `useRef`, `useReducer`, `useCallback`, and `useMemo` hooks in React and their use cases.

6. Explain how to build a custom hook in React, discussing its advantages and potential use cases.

Unit-4: Angular JS

Short Questions:

1. What are the key concepts and characteristics of AngularJS?
2. Name some data types that can be used in AngularJS expressions.
3. How do you set up the development environment for AngularJS?
4. Explain the MVC architecture in the context of AngularJS.
5. What is the purpose of AngularJS directives?
6. Name some commonly used AngularJS directives.
7. How do filters work in AngularJS, and what are some examples of filters?

Long Questions:

1. Discuss the fundamental concepts and characteristics of AngularJS, highlighting its key strengths.
2. Explain the role of expressions in AngularJS, covering numbers, strings, objects, and arrays.
3. Walk through the process of setting up the development environment for AngularJS and using AngularJS filters.
4. Explore the MVC (Model-View-Controller) architecture in AngularJS, explaining the responsibilities of each component.
5. Provide a comprehensive overview of AngularJS directives, including `ng-app`, `ng-init`, `ng-controller`, and others. Include use cases.
6. Discuss the concept of filters in AngularJS, covering examples like Uppercase, Lowercase, Currency, and order by.

Unit-5: Angular JS: Single Page Application

Short Questions:

1. How do you create a module in AngularJS?

2. Explain the role of a controller in AngularJS.

3. What is the `$routeProvider` service in AngularJS, and how does it facilitate routing?

4. What are some common HTML DOM directives in AngularJS?

5. How can you handle events and data validation in AngularJS forms?

Long Questions:

1. Walk through the steps to create a module and define a simple controller in AngularJS for building a single-page application.

2. Discuss the embedding of AngularJS scripts in HTML, emphasizing best practices and common patterns.

3. Explain AngularJS's routing capabilities and how the `$routeProvider` service from `ngRoute` enables navigation between different pages in a single-page application.

4. Explore HTML DOM directives in AngularJS, including `ng-disabled`, `ng-show`, `ng-hide`, and `ng-click`. Provide examples for each directive.

5. Describe the use of modules in AngularJS, including application and controller modules. Explain how they help organize code.

6. Discuss forms in AngularJS, covering events, data validation, and the usage of `ng-click` for handling form-related actions.