Lab	Type	Practical
I.	**Javascript Revision**	
LAB-1		**Basic JavaScript program**
	A	1. Write a program to check that the given number is prime or not. (A)
	B	2. Write a program to print all the prime numbers between given range. (B)
	A	3. Write a program to print factors of a given number. (A)
	A	4. Write a program to implement basic calculator features, take input using prompt function. (A)
	A	5. Write a program to convert celsius to fahrenheit. (A)
	A	6. Write a program to print given patterns, take number of rows as user input. (A)
	C	7. Design Calculator with HTML/CSS and implement basic calculator features using javascript. (C)
	B	8. Write a program to create a static login page using HTML and JavaScript. (B)
	C	9. WAP to create a scientific graphical calculator using HTML and JavaScript. (C)
	A	10. Write a JavaScript program to validate user data given from the HTML form (A) a. username must be of minimum 8 characters b. password must contain at least 1 digit and 1 special character and should be between 8- 12 characters c. password and repeat password must be same d. age must be greater than 18 (calculate from date of birth) e. enrollment must be of 11 digits f. email validation
LAB-2		**Document Object Model Programs**
	A	1. WAP to change background color on click of button. (A)
	A	2. WAP to recognize which mouse event is fired. (A)
	B	3. WAP to recognize which keyboard event is fired. (B)
	B	4. WAP to recognize which form event is fired. (B)
	A	5. WAP to demonstrate change in properties of HTML element using JavaScript. (A)
	A	6. WAP to prepare student registration form and validate it using JavaScript. (A)
	C	7. WAP to perform CRUD operation on Array using HTML and JavaScript. (C)
	B	8. Write a JavaScript that handles following mouse events. Add necessary elements. (B) a. JavaScript gives the key code for the key pressed. b. If the key pressed is “a”,”e”,”i”,”o”,”u”, the script should announce that vowel is pressed. c. When the key is released background should change to blue

LAB-3		**ES6 Programs**
	A	1. WAP to demonstrate callbacks in JavaScript. (A)
	A	2. Demonstrate the difference between let and var. (A)
	A	3. Demonstrate the default parameter while using a function. (A)
	A	4. Demonstrate the spread operator. (A)
	A	5. Demonstrate the 'for of' loop. (A)
	B	6. Demonstrate the Array and Object Destructuring. (B)
	B	7. Demonstrate the Arrow functions. (B)
	B	8. Demonstrate how to create a class in Java Script. (B)
	C	9. Create a Snake game using Java script. (C)
	B	10. Write JavaScript that handles following mouse event. (B) • If mouse left button pressed on browser, it displayed message "Left Clicked". • If mouse right button pressed on browser, it displayed message "Right Clicked".
	C	11. Write a JavaScript having a list of checkbox and by clicking on checkbox, it should show list of selected value in comma separated format. (e.g. list of roll number as checkbox value, and display selected roll number in comma separated format) (C)
II.	**Next.js basics**	
LAB-4		**NextJS application setup and project sturecture**
	A	1. Setup a NextJS project using "npx create-next-app@latest" command. (A)
	A	2. Explore Default Project Structure for NextJS created with create-next-app script. (A)
	A	3. Update first page.tsx file to display "Hello World from Next Application". (A)
	B	4. Write a detailed note on project structure of a NextJS application. (B)
LAB-5		**Basic Routing and Navigation in NextJS application**
	A	1. Create a NextJS application with home, contact and about page. (A)
	A	2. Create a NextJS application with basic Layout. (A)
	B	3. Create a NextJS application with layout using bootstrap. (B)
	B	4. Create a NextJS application with navigation between different pages. (B)
LAB-6		**Apply CSS on NextJS application**
	A	1. Apply CSS on the static website created in the previous lab. (A)
	B	2. Implement HTML/CSS template into NextJS application. (B)
	C	3. Implement minimum 3 different template into NextJS application. (C)

LAB-7	A C A A C	**Route group and dynamic routes in NextJS** 4. Demonstrate the Route Group feature of NextJS. • Create auth and admin route group. (A) • Create auth, admin, client route group and apply different layout on each route group. (C) 5. Demonstrate the Dynamic Route feature of NextJS. • Create a route with id as a parameter and print the id. (A) • Create a route with start and end as a parameter and print the prime number between. (A) • Create a route with pageNo as a parameter and print the start and end index of the record, assume we are getting 10 record per page. (C)
LAB-8	A A B	**intercepting routes, route handler** 1. Demonstrate the use of intercepting routes. (A) 2. Demonstrate the use of route handler. (A) 3. Create Rest API to perform CRUD operation on array using route handler. (B)
LAB-9	A B C A	**middleware, configuring NextJS application** 1. Create a middleware in NextJS to add pageNo to be zero if it does not have any parameter in the request. (A) 2. Create a middleware to check for the token in the request, if it does not have token redirect them to login page. (B) 3. Create a middleware to get new token if the received token is expired. (C) 4. Explore NextJS configurations. (A)
III.	**Fetching, Testing and Deploying NextJS application**	
LAB-10	A A B B C	**Fetching Data from mockapi** 1. Write a NextJS application to consume getALL api from mockapi and display the data on the page. (A) 2. Write a NextJS application to consume getByID api from mockapi and display the data on the page. (A) 3. Write a NextJS application to consume getALL and getByID api from mockapi and display on different pages, also provide navigation between them. (B) 4. Write a NextJS application to consume getALL and getByID operation for minimum 3 different APIs. (B) 5. Write a NextJS application to consume getALL and getByID operation for minimum 10 different APIs. (C)

LAB-11	A A B B C	**Connect with database from NextJS** 1. Write an application in NextJS to fetch the data from mysql. (A) 2. Write an application in NextJS to fetch the data from postgres. (A) 3. Write an application in NextJS to fetch the data from MongoDB. (B) 4. Write an application to perform getAll, getByID and search operation on a table stored in MySQL database. (B) 5. Write an application to perform getAll, getByID, search and getTasksByUserID operation on a database stored in MySQL with following tables (C) • **User** table with UserID, UserName, Password • **Task** table with TaskID, TaskTitle, TaskDescription, IsCompleted, UserID
LAB-12	A A A	**Connect with database using Prisma ORM** 1. Download and initialize Prisma ORM. (A) 2. Explore Prisma methods. (A) 3. Using Prisma ORM write an application to perform getAll, getByID, search and getTasksByUserID operation on a database stored in MySQL with following tables (A) • **User** table with UserID, UserName, Password • **Task** table with TaskID, TaskTitle, TaskDescription, IsCompleted, UserID
LAB-13	A A B C	**Server Actions** 1. Write an application in NextJS to demonstrate Server action. (A) 2. Create server action to print the form data. (A) 3. Create server action on a separate file and use it in your component. (B) 4. Create a server action and validate the form data. (C)
LAB-14	A A B A	**Client Component and delete operation** 1. Write an application in NextJS to create a client component (A) 2. Write an application in NextJS to create a basic calculator. (A) 3. Create a basic snake game using NextJS's client component (B) 4. Write an application to perform delete operation using Prisma ORM on a table from previous lab. (A)
LAB-15	A B A B	**Insert and Update the records using Prisma ORM.** 1. Create a registration form to insert data into database using prisma ORM. (A) 2. Create a form to add task to database table using Prisma ORM with following fields TaskID, TaskTitle, TaskDescription, IsCompleted, UserID. (B) 3. Create a form to change password into database using Prisma ORM. (A) 4. Create a form to edit Task stored in database. (B)

LAB-16	A A A B C	**Automated Testing in NextJS application** 1. Write a test case to perform unit testing on NextJS application using Vitest. (A) 2. Write a test case to perform testing on NextJS application using Jest. (A) 3. Write a basic test cases to perform e2e testing on NextJS application using Playwrite. (A) 4. Write a test case to fill a form to add user and generate the report of the same using Playwrite. (B) 5. Write a test case to perform full CRUD operation using Playwrite and generate the report. (C)
IV.	**NestJS**	
LAB-17	A A B	**Setting up NestJS project** 1. Installing NestJS (A) 2. Setup NestJS project and explore the project structure (A) 3. Practice NestJS CLI commands. (B)
LAB-18	A A C	**Controllers in NestJS** 1. Create a student controller with helloWorld method which returns “hello world” (A) 2. Create a student controller with findAll, findOne, insert, update and delete method and return appropriate string, make sure you use relevant HTTP methods. (A) 3. Create 3 different controllers with all above specified methods. (C)
LAB-19	A A A A A A B C	**Routing** 1. Demonstrate @All decorator. (A) 2. Demonstrate @HttpCode decorator. (A) 3. Demonstrate @Redirect decorator. (A) 4. Demonstrate @Header decorator. (A) 5. Demonstrate Route Wildcards. (A) 6. Demonstrate Route Parameters, • Write a code to get id from the parameters and print it. (A) • Write a code to get start and end from parameters and return all the prime numbers between that range. (B) • Write a code to get pageNo as parameter and return start and end record number, consider 5 records per page. (C)
LAB-20	A A A	**Providers** 1. Create a student service with findAll, findOne, insert, update and delete method to return “Hello world”. (A) 2. Write a NestJS application to combine student controller and service. (A) 3. Demonstrate module in NestJS. (A)

LAB-21	 A A C	**Resoruces** 1. Createa a student resource. (A) 2. Implement static array CRUD API for students (A) 3. Implement static array CRUD API for 3 different array. (C)
LAB-22	 A B C	**Setting up NestJS project** 1. Create complete Rest API for students table stored in MySQL with following fields, (A) • StudentID • StudentName • StudentAge • StudentGender • StudentRollNo • StudentSemester. • Etc… 2. Create a complete authentication API with login, register features (B) 3. Implement forgot password feature API. (C)
V.	**React Hooks**	
LAB-21	 A A A A	**React Hooks (useState, useEffect, useActionState, useCallback)** 1. Write a react application to implement a counter feature, each time use click on a button it should add a counter and display current counter on the page.(A) 2. Write a React Application which print "Hello world" on a console after every second. (A) 3. Write a React Component which display a form validation error using useActionState hook. (A) 4. Demonstrate useCallback hook in React. (A)
LAB-22	 A A A A	**React Hooks (useContext, useDebugValue, useDeferredValue)** 1. Write a react component named "A" which has a child component called "B", which again has a child component called "C", now pass a property from the "A" to "C" without using any hooks (prop drilling). (A) 2. Write a react component named "A" which has a child component called "B", which again has a child component called "C", now pass a property from the "A" to "C" using useContext hook. (A) 3. Demonstrate the use of useDebugValue hook in React. (A) 4. Demonstrate the use of useDeferredValue hook in React. (A)

LAB-23	A A A A A	**React Hooks (useImperativeHandle, useLayoutEffect, useMemo, useOptimistic)** 1. Demonstrate the use of useImperativeHandle hook in React. (A) 2. Demonstrate the use of useLayoutEffect hook in React. (A) 3. Write a react component which returns addition of first 5000 prime number. (A) 4. Write a react component which returns addition of first 5000 prime number, use useMemo to demonstrate the performance diffrence. (A) 5. Demonstrate the use of useOptimistic hook in React. (A)
LAB-24	A A A A A	**React Hooks (useReducer, useRef, useSyncExternalStore, useTransition, useFormStatus)** 1. Demonstrate the use of useReducer hook in React. (A) 2. Demonstrate the use of useRef hook in React. (A) 3. Write a code in React to check weather the user is online or offline using useSyncExternalStore hook. (A) 4. Demonstraste the use of useTransition hook in React. (A) 5. Display the status information of the last form submission using useFormStatus hook in React. (A)
VI.	**State management**	
LAB-25	A B	**ContextAPI** 1. Implement user authentication with Context API. (A) 2. Implement “add to cart” feature using Context API. (B)
LAB-26	A B	**Redux** 1. Implement user authentication with Redux toolkit. (A) 2. Implement “add to cart” feature using Redux toolkit. (B)
LAB-27	A	**socket.io** Implement chat application using socket.io. (A)
VII.	**Mini Project**	
LAB-28	A	**Create a mini project on library management system with NestJS as backend and NextJS as a frontend, also implement state management and chat features (1/3)**
LAB-29	A	**Create a mini project on library management system with NestJS as backend and NextJS as a frontend, also implement state management and chat features (2/3)**
Lab-30	A	**Create a mini project on library management system with NestJS as backend and NextJS as a frontend, also implement state management and chat features (3/3)**