

Branch Name:	MCA
Program Code:	CS201
Course Title:	Full Stack Development-II
Course Code:	4CS2010307T
Pre-requisite Course:	Basic knowledge of HTML, CSS, JavaScript, React fundamentals, component-based architecture, basic Node.js concepts, REST API basics, database CRUD operations, and version control using Git.

Course Objective:

The objectives of the course are to:

1. Enhance students' knowledge of advanced React concepts including hooks, state management, and performance optimization.
2. Enable students to develop dynamic and scalable front-end applications using modern React techniques.
3. Provide understanding of backend integration using Node.js, Express, and RESTful APIs.
4. Familiarize students with database integration using SQL and NoSQL technologies.
5. Develop the ability to build complete full-stack applications with proper authentication, validation, and security practices.
6. Introduce best practices in application development, testing, deployment, and version control.

Teaching and Examination Scheme:

Teaching Scheme (Hours per week)				Evaluation Scheme (Marks)				
Lecture (L)	Tutorial (T)	Practical (P)	Credit	Theory (Marks)		Practical (Marks)		Total (Marks)
				University Assessment	Continuous Assessment	University Assessment	Continuous Assessment	
3	-	-	3	60	40	-	-	100

Course Contents:

Unit No	Topic	Total Hours	Weightage (%)
1	Advanced React.js Development: • Advanced React Overview: React.js for building scalable Single Page Applications (SPAs) using Virtual DOM and component-based architecture. • Component Architecture: Functional components, reusable design, and proper project structure for maintainable applications. • React Hooks: useState, useEffect, useReducer, and useContext for efficient state and lifecycle management, along with custom hooks for reusable logic. • State Management & Optimization: Redux for centralized state management and performance techniques like memoization and lazy loading. • Forms and Validation: Handling forms using controlled components with validation and error handling techniques.	9	25
2	Advanced React Features & Application Development	10	25

	• Component Interaction & Rendering: Communication between components, conditional rendering, and rendering lists using map() with proper key usage. • Routing in React: Client-side routing using React Router including BrowserRouter, Route, Link, and navigation between multiple pages. • API Integration: Fetching data using Fetch API / Axios, handling loading states, and managing error responses. • Global State Management: Context API for sharing data across components and reducing prop drilling in applications. • Best Practices & Project Development: Code organization, performance best practices, and development of a small React-based application.		
3	Backend Integration with Node.js & Database • Introduction to Backend Integration: Role of backend in full-stack applications, client-server architecture, and interaction between React frontend and server. • Node.js & Express Basics: Setting up Node.js environment, creating server using Express, routing, and handling HTTP requests and responses. • RESTful API Development: Designing APIs using GET, POST, PUT, DELETE methods and structuring request-response data using JSON. • Database Integration: Introduction to SQL (MySQL) and NoSQL (MongoDB), connecting backend with database using drivers and Mongoose ORM. • CRUD Operations: Performing Create, Read, Update, and Delete operations in both MySQL and MongoDB. • Data Validation & Security: Basic validation techniques, handling errors, and securing data using best practices.	10	25
4	Full Stack Integration & Application Development • Frontend-Backend Integration: Connecting React applications with backend APIs, handling data exchange using HTTP requests, and managing responses effectively. • Authentication & Authorization: Implementing user authentication using JWT, managing sessions, and role-based access control for secure applications. • Project Structure & Deployment: Organizing full-stack project structure, environment configuration, and basic deployment concepts using platforms like Netlify or Vercel. • Testing & Debugging: Identifying and fixing errors in frontend and backend, debugging APIs, and improving application reliability.	10	25

	• Best Practices: Code optimization, security practices, version control using Git, and maintaining scalable and maintainable applications.		
--	--	--	--

Text Books:

1. Vasan Subramanian, *Pro MERN Stack: Full Stack Web App Development with Mongo, Express, React, and Node*, 2nd Edition, Apress.
2. Alex Banks, Eve Porcello, *Learning React: Modern Patterns for Developing React Apps*, 3rd Edition, O'Reilly.
3. Ethan Brown, *Web Development with Node and Express*, 2nd Edition, O'Reilly Media.
4. Frank Zammetti, *Modern Full-Stack Development*, Apress.

Reference Books:

1. Marijn Haverbeke, "Eloquent JavaScript: A Modern Introduction to Programming", 3rd Edition, No Starch Press.
2. David Herron, "Node.js Web Development", 5th Edition, Packt Publishing.
3. Shama Hoque, "Full-Stack React Projects", 1st Edition, Packt Publishing.
4. Kyle Banker et al., "MongoDB in Action", 2nd Edition, Manning Publications.
5. Kyle Simpson, "You Don't Know JS Yet", 2nd Edition, O'Reilly Media.

List of Open Source Software/learning website:

- <https://react.dev>
- <https://nodejs.org>
- <https://expressjs.com>
- <https://developer.mozilla.org>
- <https://www.mongodb.com/docs>
- <https://www.w3schools.com>
- <https://www.tutorialspoint.com>
- <https://github.com/explore>

Course Learning Outcomes (CLO): On completion of this course, the students will be able to:

CLO	Description	Bloom's Taxonomy Level
CLO1	Understand advanced React concepts including hooks, state management, and performance techniques.	1.Remembering, 2. Understanding
CLO2	Develop scalable front-end applications using React with routing, API integration, and component interaction.	3.Applying 6.Creating
CLO3	Design and implement backend services using Node.js and Express with RESTful API principles.	3.Applying 4.Analyzing
CLO4	Perform database operations using SQL and MongoDB with proper data handling and validation.	3.Applying 4.Analyzing
CLO5	Integrate frontend, backend, and database to build full-stack applications.	6.Creating
CLO6	Apply authentication, security, and best practices in full-stack application development.	5.Evaluating, 6. Creating

Mapping of CLOs with POs & PSOs

Course Learning Outcomes	Program Outcomes (POs)												Program Specific Outcomes (PSOs)	
	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO 10	PO11	PO12	PSO1	PSO2
CLO1	H	M	M	–	H	–	–	–	–	–	–	L	H	M
CLO2	H	H	H	M	H	–	–	M	L	L	–	L	H	H
CLO3	H	H	H	M	H	–	–	M	–	–	–	L	H	H
CLO4	H	H	H	H	H	–	–	–	–	–	–	L	H	M
CLO5	H	H	H	M	H	–	–	M	M	L	M	L	H	H
CLO6	H	H	H	M	H	M	–	H	M	L	M	L	H	H

H:High, M:Medium, L:Low