Official Documentations:

- ReactJS Documentation:https://reactjs.org/docs/getting-started.html
- NodeJS Documentation: https://nodejs.org/docs/latest-v17.x/api/
- ExpressJS Documentation: https://expressjs.com/en/guide/routing.html
- MongoDB Documentation: https://www.mongodb.com/docs/manual/tutorial/getting-started

Prerequisites:

Install NodeJs: https://www.youtube.com/watch?v=JINE4D0Syqw
Install VSCode or any code editor of your choice.
Install Git: https://www.youtube.com/watch?v=2j7fD92g-gE

ReactJS:

React is a popular JS frontend library open-sources by Facebook. It allows you to create and reuse UI elements. React was born out of the need to solve the problem of the browser DOM being slow.
React is component based meaning that components are the building blocks of a React UI. A component describes how a smaller section of the UI looks like and is reusable. Complex applications can be built by nesting components within other components.
Another important thing about React is that it is used to create Single Page Applications. This means React doesn't fetch a completely new web page when the user has to be shown a different view, it just rewrites the current web page with any new data fetched from the backend server.

Creating your first React Application:

npx create-react-app <app_name>

Example:
npx create-react-app online-judge

This will create a folder named as **online-judge** in which you can see the React starter code.

Steps to start the react app.

1. **cd <app_name>**
2. **npm start**
3. **Open Web Browser and go to http://localhost:3000**

Resources:

Firms that uses React (Just for some motivation)

Creating React App
Understanding JSX
Introduction to JSX
React: supported HTML attributes

React Components
Component and Props
State and Lifecycle
Prop and State Explained
Props vs State

ReactJS Lifecycle

CodeSandbox:
Hello, World
Counter (Don't worry about syntax, focus on the code reuse)

NodeJS:

"Node.js is an open source, cross platform, backend, Javascript runtime environment that executes Javascript code outside a web browser". - Wikipedia.

- **Open source** - its source code is available on github for use and modification. You can also contribute to this project on Github.
- **Cross Platform** - Works across multiple platforms like Linux, OSX and Windows.
- **Backend** - Receives requests from clients and contains the logic to respond to it.
- **JS runtime environment** - Where the JS code gets parsed and executed

It's easier to learn if you are familiar with JS. This also means that both the frontend and backend can now be written with just JS knowledge.

For initialisation of the project type the following command.
npm init

Resources:
How to install NodeJs
Getting Started with NodeJs
Routing in NodeJS
Curl Command for HTTP Requests
NodeJS Child Processes

Express JS

Express.js is a Node.js framework,it makes it easier to build APIs.

Command to install ExpressJS in your node project.

- **npm install express**

Resources:
Getting started with express
Express basic routing
Express: req.params, req.query and req.body
Middleware Functions
How nodejs middleware work

MongoDB

Relational databases were popular due to the reduced data duplication they brought with them. However, the reduction in storage costs and massive amounts of unstructured data have paved the way for NoSQL based databases. In a lot of cases where it is not required to deal with complex relational data models that occur due to the large number of tables and the relationships between them, NoSQL comes to the rescue. While the relational database store data in the form of rows and columns, the NoSQL databases store data as documents (e.g. BSON).

MongoDB, CouchDB, Cassandra, HBase and many more are examples of NoSQL databases.

Resources:
MongoDB vs MySQL database
When to go for MongoDB
JSON vs BSON

Introduction to MongoDB
Database and Collections
MongoDB Documents

MongoDB Update Operators
MongoDB Logical Operators
MongoDB Object Id
MongoDB Query Selectors

Introduction to Mongoose

ODM vs ORM

Topics To Be Covered in Class:

- What is MERN ?
- How do these technologies communicate with each other?
- Practical Implementation of basic Web App in MERN.
- How to create an API in NodeJS, With and Without Express.
- How to connect MongoDB with NodeJS.
- How to create Schemas in Mongoose.
- How to make an API call from the React App.
- How to render the data fetched from an API on the frontend.