React JS

Week	Topics Covered	Practice Examples
1	Introduction to React.js	1. Create a simple "Hello World" React component
	Setting up React Environment	2. Build a basic todo list application
	Components and JSX	3. Create a counter that increments and decrements
	Handling Events	4. Develop a form with input validation

	State and Props	**Build a basic calculator application**
2	Component Lifecycle	1. Create a timer component with start, stop, reset functionality
	Conditional Rendering	2. Build a login form with conditional rendering for error messages
	Lists and Keys	3. Implement a dynamic list of items with delete and edit functionality

	Styling in React	**Create a basic layout using CSS-in-JS libraries like styled-components**
3	React Router and Navigation	1. Create a multi-page website with React Router
	Forms and Form Libraries	2. Build a contact form using Formik and Yup

	State Management with React Context	**Implement a simple theme switcher using React Context**
4	React Hooks	1. Convert class components to functional components using Hooks
	Custom Hooks	2. Build a custom hook for handling form state

	Managing Asynchronous Operations	**Fetch data from an API and display it in a component**
5	Forms in React	1. Create a multi-step form with validation using Formik
	React and Redux	2. Integrate Redux into a React application

	Redux Middleware and Thunk	**Use Redux Thunk middleware to handle asynchronous actions**
6	React and Server-side Communication	1. Connect React app with a backend using Axios or Fetch API
	Error Handling and Debugging in React	2. Implement error boundaries to handle errors gracefully
	Deployment and Optimization	3. Deploy a React app to GitHub Pages or Netlify
	Final Project	4. Build a complete React application of your choice

Node JS

Week	Topics Covered	Practice Examples
1	Introduction to Node.js	1. Create a basic "Hello World" Node.js server
	Setting up Node.js Environment	2. Build a simple web server that serves static files
	NPM (Node Package Manager) and Packages	3. Create a package.json file for a new project
	Synchronous and Asynchronous Programming	4. Implement a file reading and writing application using callbacks
	Introduction to Express.js	**Develop a simple RESTful API with CRUD operations**
2	Middleware in Express.js	1. Create custom middleware to log request details
	Handling Forms and File Uploads in Express.js	2. Build an image upload functionality using Multer middleware
	Error Handling in Express.js	3. Implement error handling for API routes
	Using Template Engines (e.g., EJS) in Express.js	**Create a basic blog application with EJS template**
3	Introduction to MongoDB	1. Set up a local MongoDB database and perform CRUD operations
	Connecting Node.js with MongoDB	2. Create a RESTful API with MongoDB as the data store
	Mongoose ODM (Object Data Modeling)	3. Define and use Mongoose schemas for data validation
	Data Pagination and Sorting	**Implement pagination and sorting for a list of items**
4	Authentication and Authorization in Node.js	1. Add user registration and login functionality using Passport.js
	JSON Web Tokens (JWT)	2. Implement token-based authentication for protected routes
	Password Hashing and Security	3. Enhance user authentication with hashed passwords
	Sending Emails with Node.js	**Create a contact form that sends emails using Nodemailer**
5	RESTful API Design and Best Practices	1. Design and build a scalable API following best practices
	Validation and Error Handling in APIs	2. Implement input validation and error handling for API endpoints
	Rate Limiting and API Security	**Add rate limiting to prevent abuse and secure API endpoints**
6	Deploying Node.js Applications	1. Deploy a Node.js app to a cloud platform (e.g., Heroku)
	Environment Variables and Configuration	2. Utilize environment variables to manage sensitive data
	Performance Optimization	3. Optimize the Node.js application for better performance
	Final Project	4. Develop a complete Node.js application of your choice

JAVA Backend

Week	Topics Covered	Practice Examples
1	Introduction to Spring Framework and Spring Boot	1. Create a simple Spring Boot project with a "Hello World" endpoint
	Dependency Injection in Spring	2. Build a RESTful API with CRUD operations using Spring Boot
	Bean Configuration and Annotations	**Implement bean configuration using annotations**
2	Building RESTful APIs with Spring Boot	1. Develop a user authentication API with JSON Web Tokens (JWT)
	Request and Response Handling in Spring Boot	2. Create endpoints for file uploads and downloads
	Request Validation and Exception Handling	**Implement validation for API inputs and handle exceptions**
3	Introduction to Spring Data JPA	1. Set up a PostgreSQL database and perform CRUD operations
	Defining Data Models with JPA Annotations	2. Define JPA entities and relationships for a simple database
	Querying Data with Spring Data JPA	**Write custom repository methods for specific queries**
4	Advanced JPA Mapping	1. Implement one-to-many and many-to-many relationships in JPA
	Spring Security	2. Secure API endpoints using Spring Security
	Role-based Access Control	**Implement role-based access control for different user roles**
5	Handling Authentication and Authorization	1. Integrate Spring Security with JWT for stateless authentication
	Introduction to Spring Boot Actuator	2. Enable monitoring and management endpoints with Actuator
	Global Exception Handling	**Implement a custom global exception handler**
6	Deploying Spring Boot Applications	1. Deploy a Spring Boot app to a cloud platform (e.g., AWS, Azure)
	Spring Boot Profiles and Configuration	2. Use different profiles for development, testing, and production
	Caching with Spring Boot	3. Enable caching for frequently accessed data
	Final Project	4. Develop a complete Spring Boot application of your choice

MongoDB

Week	Topics Covered	Practice Examples
1	Introduction to NoSQL and MongoDB	1. Install MongoDB and set up a local development environment
	CRUD Operations with the MongoDB Shell	2. Create, read, update, and delete documents using the shell
	MongoDB Data Modeling and Schemas	**3. Design and create simple collections with appropriate schemas**
2	Querying and Basic Aggregation in MongoDB	1. Perform find operations with various filters and projections
	Indexing and Performance Optimization	2. Create and use indexes to improve query performance
	Aggregation Framework and Pipelines	**3. Practice aggregation pipeline operations with sample datasets**
3	Working with MongoDB and Node.js	1. Set up a Node.js project with MongoDB and create a connection
	CRUD Operations with Node.js and MongoDB	2. Build a RESTful API with MongoDB as the data store
	Data Validation and Error Handling in Node.js	**3. Implement validation for API inputs and handle errors**
4	Advanced Querying with Mongoose	1. Use Mongoose to create complex queries with sorting and paging
	Mongoose Middleware and Hooks	2. Implement pre and post hooks for document and query middleware
	Embedded Documents and Referencing	**3. Design and use embedded and referenced documents in MongoDB**
5	Transactions and Data Consistency in MongoDB	1. Perform multi-document transactions with ACID properties
	Geospatial Queries and Indexes	2. Work with geospatial data and create geospatial indexes
	Aggregation with Mongoose	**3. Use Mongoose to perform complex data aggregation**
6	MongoDB Atlas and Cloud Deployment	1. Set up a MongoDB Atlas cluster and migrate data
	Backup and Security Best Practices	2. Implement data backup strategies and secure the database
	Final Project	3. Build a complete MongoDB-backed Node.js application

Python Django

Week	Topics Covered	Practice Examples
1	Introduction to Django and Setting up Environment	1. Install Django and create a basic project with a "Hello World" view
	Django Models and Admin Interface	2. Define models for a simple application and manage them via admin
	Views, URLs, and Templates	3. Create a dynamic web page using views, URLs, and template system
	Working with Forms and User Input	**Build a contact form with Django's built-in form handling**
2	Django ORM (Object-Relational Mapping)	1. Perform CRUD operations with Django's ORM
	Django Querysets and Aggregation	2. Write complex queries using Querysets and aggregate functions
	Model Relationships and Database Design	3. Implement one-to-many and many-to-many relationships in models
	Static Files and Media Handling	**Manage static files and handle user-uploaded media files**
3	User Authentication and Authorization	1. Implement user registration and login functionality
	Custom User Model and Authentication Backends	2. Create a custom user model and implement authentication backends
	Permissions and User Groups	**3. Control user access with permissions and user groups**
4	Django Class-Based Views and Mixins	1. Convert function-based views to class-based views using mixins
	Advanced Template Features and Custom Tags	2. Create custom template tags and filters
	Django Forms and Formsets	**Build a multi-step form using formsets and handle form validation**
5	Django Rest Framework (DRF) Basics	1. Set up a RESTful API with DRF and create endpoints
	Serializers and Model Viewsets	2. Serialize Django models and use Model Viewsets for API views
	Authentication and Permissions in DRF	**Secure API endpoints using authentication and permissions**
6	File Uploads and API Testing	1. Implement file uploads in DRF and write tests for the API
	Pagination and Filtering in DRF	2. Add pagination and filtering to handle large datasets
	Deployment and Production Considerations	3. Deploy a Django app to a production server
	Final Project	4. Build a complete web application using Django and DRF

PHP Laravel

Week	Topics Covered	Practice Examples
1	Introduction to Laravel and Setting up Environment	1. Install Laravel and create a new project with a basic "Hello World"
	Laravel Routing and Views	2. Define routes and create views for a simple web page
	Blade Templating Engine	3. Use Blade to create reusable templates and layouts
	Working with Databases and Eloquent ORM	**Set up a database and perform CRUD operations using Eloquent ORM**
2	Form Handling and Validation	1. Create a contact form and implement server-side validation
	Middleware and Request Lifecycle	2. Implement custom middleware to handle request manipulation
	User Authentication and Authorization	**Set up user authentication and manage user roles and permissions**
3	Eloquent Relationships	1. Define and use one-to-many and many-to-many relationships
	Working with File Uploads and Storage	2. Implement file uploads and manage files using Laravel Storage
	Sending Emails with Laravel	**Create and send emails using Laravel's built-in email capabilities**
4	RESTful APIs with Laravel	1. Build a RESTful API with Laravel and create endpoints
	API Authentication with Laravel Passport	2. Secure API endpoints using Laravel Passport for token-based auth
	API Versioning and Rate Limiting	**Implement versioning and rate limiting to control API access**
5	Error Handling and Logging	1. Implement custom error handling and log application events
	Caching with Laravel	2. Use caching to optimize application performance
	Localization and Internationalization	**Add multi-language support to your application using localization**
6	Testing and Test-Driven Development (TDD)	1. Write unit tests and practice Test-Driven Development (TDD)
	Laravel Mix and Front-end Asset Management	2. Use Laravel Mix for front-end asset compilation and optimization
	Deployment and Production Considerations	3. Prepare and deploy a Laravel application to a production server
	Final Project	4. Build a complete web application using Laravel

Flutter

Week	Topics Covered	Practice Examples
1	Introduction to Flutter and Setting up Environment	1. Install Flutter and create a new Flutter project with a basic UI
	Flutter Widgets and Layouts	2. Build a UI with various Flutter widgets and layout components
	Handling User Input and Gestures	3. Implement interactive elements and handle user input using gestures
	Navigation and Routing in Flutter	**Create multiple screens and navigate between them**
2	State Management in Flutter	1. Manage state using StatefulWidget and StatefulWidget
	Flutter UI Styling and Themes	2. Apply custom styles and themes to Flutter widgets and components
	Working with APIs and HTTP Requests	**Fetch data from an API and display it in your Flutter app**
3	Flutter Packages and Plugins	1. Integrate third-party packages and plugins to add functionalities
	Flutter Animations and Transitions	2. Create animated UI elements and transitions
	Local Data Storage in Flutter	**Implement data persistence using local storage (SharedPreferences)**
4	Flutter and Firebase	1. Integrate Firebase Authentication for user login and registration
	Firebase Realtime Database	2. Implement real-time data synchronization with Firebase
	Firebase Cloud Firestore	**3. Store and retrieve data using Firestore in your Flutter app**
5	Flutter and State Management Libraries	1. Use popular state management libraries like Provider or Bloc
	Internationalization and Localization in Flutter	2. Add multi-language support to your Flutter app using localization
	Flutter Performance Optimization	**Optimize your app for better performance and faster rendering**
6	Flutter Web and Desktop Support	1. Build and test your Flutter app for the web and desktop platforms
	Deployment of Flutter Applications	2. Deploy your Flutter app to Google Play Store and Apple App Store
	Final Project	3. Develop a complete Flutter application of your choice

XRPL Blockchain

Week	Topics Covered	Practice Examples
1	Introduction to XRPL and Blockchain Concepts	1. Understand the fundamentals of XRPL and its consensus algorithm
	XRP Ledger Architecture and Components	2. Explore the ledger, transactions, and accounts on the XRP Ledger
	XRPL APIs and Data Retrieval	**Use XRPL APIs to fetch ledger data and transaction history**
2	XRPL Account Creation and Management	1. Create an XRPL wallet and manage keys and account information
	Sending and Receiving XRP	2. Perform XRP transactions between different accounts
	XRPL Transactions and Transaction Types	**Send various types of transactions, such as payment and escrow**
3	XRPL Smart Contracts	1. Understand the concept of escrows and create an escrow transaction
	Conditional Payments and Payment Channels	2. Implement conditional payments using XRPL
	XRPL Multisigning and Security Considerations	**Create multisigning transactions and explore security best practices**
4	XRPL Hooks and Plugins	1. Explore hooks and plugins to extend XRPL's functionality
	XRPL Validators and Consensus	2. Set up a validator on the XRPL network
	Exploring XRPL Decentralization	**Study XRPL's decentralized nature and consensus mechanism**
5	XRPL Interoperability and Cross-Border Payments	1. Investigate XRPL's potential for cross-border payments
	Interacting with XRPL through APIs	2. Build a simple application that interacts with XRPL using APIs
	XRPL Integration with Existing Systems	**Integrate XRPL with other applications or systems**
6	XRPL Scaling and Future Developments	1. Learn about XRPL's scaling solutions and future development plans
	XRPL Testnet and Testing Strategies	2. Utilize XRPL Testnet for testing your applications
	Final Project	3. Develop a complete XRPL-based application or use case

Please note that for the "Practice Examples" section, the examples mentioned are just suggestions to give you an idea of what you can build there are multiple assignments related to these topics are uploaded in LMS. As you progress through the weeks, try to work on more complex and challenging projects to reinforce your learning.