Full Stack Website Development I

Chapter 1: Introduction to Full-Stack Development
Lecture Notes

Instructor: YAB Datuk Seri Utama Prof. Dr. Lim Zhi Wei
Organization: Smol Tako

July 10, 2025

Contents

1 Course Overview

1.1 Course Information

This course introduces students to full-stack web development using the MERN stack (MongoDB, Express.js, React.js/Next.js, Node.js). Students will learn to build complete web applications from database design to user interface implementation.

Course Details

Course Code:	FSWD101
Credit Hours:	3
Contact Hours:	4 hours per week (2 hours lecture + 2 hours tutorial)
Duration:	14 weeks
Assessment:	Continuous Assessment (50%) + Final Examination (50%)

1.2 Prerequisites

Before taking this course, students should have:

- Basic knowledge of HTML, CSS, and JavaScript
- Understanding of programming fundamentals
- Familiarity with version control systems (Git)

2 What is Full-Stack Development?

2.1 Definition

Full-stack development refers to the practice of working on both the **frontend** (client-side) and **backend** (server-side) portions of a web application. A full-stack developer has the skills to handle all aspects of web development.

2.2 Components of Full-Stack Development

Layer	Description	Technologies
Frontend	User interface and user experience	HTML, CSS, JavaScript, React.js, Next.js
Backend	Server-side logic, APIs, business logic	Node.js, Express.js, RESTful APIs
Database	Data storage and management	MongoDB, NoSQL databases
DevOps	Deployment and infrastructure	Cloud services, CI/CD pipelines

3 Overview of Web Development

3.1 Traditional vs Modern Web Development

Aspect	Traditional	Modern
Architecture	Monolithic applications	Microservices, API-first
Frontend	Server-rendered pages	Single Page Applications (SPAs)
Data Exchange	Form submissions, page reloads	RESTful APIs, JSON
User Experience	Static, page-based	Dynamic, interactive
Development	Separate frontend/backend teams	Full-stack developers

3.2 Benefits of Full-Stack Development

1. **Comprehensive Understanding:** Complete control over the entire application
2. **Efficient Communication:** Better coordination between frontend and backend
3. **Rapid Prototyping:** Faster development and iteration cycles
4. **Cost Effectiveness:** One developer can handle multiple aspects
5. **Problem Solving:** Ability to debug issues across the entire stack

4 Introduction to MERN Stack

4.1 What is MERN?

MERN is an acronym for four key technologies that work together to create dynamic web applications:

Technology	Purpose	Description
MongoDB	Database	NoSQL document database for storing application data
Express.js	Backend Framework	Web application framework for Node.js
React.js	Frontend Library	JavaScript library for building user interfaces
Node.js	Runtime Environment	JavaScript runtime for server-side development

4.2 MERN Stack Architecture

MERN Stack Flow

Client (React.js) → **HTTP Requests** → **Server** (Express.js/Node.js) → **Database Operations** → **MongoDB**

4.3 Why Choose MERN Stack?

Advantage	Description
JavaScript Everywhere	Single language for frontend, backend, and database queries
JSON Data Flow	Seamless data exchange between all layers
Rapid Development	Pre-built modules and extensive community support
Scalability	Easy to scale both horizontally and vertically
Performance	Fast rendering and efficient data handling
Community Support	Large developer community and extensive documentation

5 Development Environment Setup

5.1 Required Software

Software	Purpose	Installation
Node.js	JavaScript runtime	Download from nodejs.org
npm/yarn	Package manager	Comes with Node.js
MongoDB	Database	MongoDB Atlas (cloud) or local installation
VS Code	Code editor	Download from code.visualstudio.com
Git	Version control	Download from git-scm.com
Postman	API testing	Download from postman.com

5.2 VS Code Extensions

Essential extensions for MERN development:

- ES7+ React/Redux/React-Native snippets
- Bracket Pair Colorizer
- Auto Rename Tag
- GitLens
- Prettier - Code formatter
- MongoDB for VS Code
- Thunder Client (Postman alternative)

6 Basic Project Structure

6.1 Typical MERN Project Organization

Listing 1: Project Directory Structure

```
my-mern-app/
          client/                  # React frontend
                public/
                src/
                      components/
                      pages/
                      utils/
                package.json
          server/                  # Node.js backend
                controllers/
                models/
                routes/
                middleware/
                package.json
          README.md
          .gitignore
```

6.2 Package.json Structure

Key package.json Scripts

```
{
  "name": "mern-app",
  "version": "1.0.0",
  "scripts": {
    "dev": "concurrently \"npm run server\" \"npm run client\"",
    "server": "cd server && npm run dev",
    "client": "cd client && npm start",
    "build": "cd client && npm run build"
  }
}
```

7 Key Concepts and Terminology

7.1 Frontend Terminology

Term	Definition
Component	Reusable UI building blocks in React
State	Data that changes over time in a component
Props	Data passed from parent to child components
JSX	JavaScript XML syntax for writing React components
Virtual DOM	In-memory representation of the real DOM

7.2 Backend Terminology

Term	Definition
API	Application Programming Interface
REST	Representational State Transfer architecture
Middleware	Functions that execute during request-response cycle
Route	URL endpoint that handles specific HTTP requests
Controller	Functions that handle business logic for routes

8 Learning Outcomes

Upon completion of this chapter, students should be able to:

1. Explain the concept of full-stack development
2. Identify the components of the MERN stack
3. Understand the advantages of using MERN for web development
4. Set up a proper development environment
5. Recognize the basic project structure of a MERN application
6. Use fundamental terminology related to full-stack development

9 Next Steps

In the next chapter, we will dive deeper into:

- JavaScript ES6+ features essential for modern development
- Asynchronous programming concepts
- Module systems and their importance
- Setting up your first Node.js project

10 Tutorial 1: Environment Setup and Basic Project Structure

10.1 Objectives

- Install and configure development tools
- Create a basic MERN project structure
- Verify installation and setup
- Initialize version control

10.2 Tasks

1. Install Node.js and verify installation
2. Set up MongoDB Atlas account
3. Install and configure VS Code with recommended extensions
4. Create a new MERN project directory structure
5. Initialize Git repository and create .gitignore
6. Test basic setup with simple "Hello World" examples

Important Notes

- Always keep your development environment updated
- Use LTS versions of Node.js for stability
- Practice proper version control from the beginning
- Document your setup process for team collaboration