Full Stack Web Development from the Ground Up: Principles, Practices, and Technologies

Book Outline

Last updated: December, 2024

INTRODUCTION

#	Chapter Title	Wd Count*	Chapter Outline (First-level Headings Only)
1	Welcome to Full Stack Web Development	6782	1.1. The Single-Page Web App Box 1.1: What is Speedgolf? 1.2. SpeedScore: A Walkthrough 1.3. Full-Stack Web Development 1.4. Web Development Technologies 1.5. Web Development Practices 1.6. Core Principles for Web Design 1.7. Summary 1.8. References 1.9. Exercises

*Word counts include only the chapter bodies; abstracts, references, exercises, and programming tasks are not included in these counts.

PART I: FRONT END DEVELOPMENT IN HTML, CSS, AND JAVASCRIPT

#	CHAPTER TITLE	WD COUNT*	CHAPTER OUTLINE (FIRST-LEVEL HEADINGS ONLY)
2	The Front-End Code Behind a Single Page Web Application	5269	2.1. Introduction 2.2. Defining Page Content with HTML 2.3. Styling Page Content Using CSS Box 2.1: Search Strategies 2.4. Programming App Behavior Using JavaScript Box 2.2: Quickly Creating a Single-Page Web App in CodePen 2.5. Summary 2.6. References 2.7. Exercises 2.8. Programming Tasks
3	Creating a Single-Page Web App Framework in HTML and CSS: Semantic HTML and ARIA Roles	4713	3.1. Introduction Box 3.1: Exploring the Book's Code using Git, GitHub, and VSC 3.2. Specifying App Metadata in the <head> Section 3.3. Top Navigation Bar Box 3.2: Using FontAwesome Icons 3.4. Summary, Reflections, and Best Practices 3.5. References 3.6. Exercises 3.7. Programming Tasks
4	Creating a Single-Page Web App Framework in HTML and CSS: Part II	5906	4.1. Introduction 4.2. Mode Tabs Box 4.1: Anchors or Buttons? 4.3. Side Menu 4.4. Content Area Box 4.2: Using Browser Tools to Explore App Responsiveness 4.5. Floating Action Button 4.6. Summary, Reflections, and Best Practices 4.7. References 4.8. Exercises 4.9. Programming Tasks

5	Bringing a Single Page Web App to Life with JavaScript: Event Handling and Menus	5987	5.1. Introduction 5.2. Events and Event Handling 5.3. Bringing the SpeedScore App to Life: A Roadmap 5.4. Responding to Click Interaction with the Side Menu 5.5. Supporting Keyboard Interaction with the Menu 5.6. Summary, Reflections, and Best Practices 5.7. References 5.8. Exercises 5.9. Programming Tasks
6	Bringing a Single Page Web App to Life with JavaScript: Tabs and Modal Dialogs	3789	6.1. Introduction 6.2. Responding to Interaction with the Mode Tabs 6.3. Responding to Interaction with the Floating Action Button 6.4. Using the History API to Route App Pages 6.5. Summary, Reflections and Best Practices 6.6. References 6.7. Exercises 6.8. Programming tasks
7	Automated Testing of Client-Side Web Apps	9640	7.1. Introduction Box 7.1: Early Data Gathering and Usability Testing 7.2. Foundations 7.3. Setting Up the Testing Environment 7.4. Writing Tests with Playwright 7.5. Adding Axe DevTools Tests 7.6. Testing Security Vulnerabilities with OWASP ZAP 7.7. Documenting Success (and Failure): Recording a Test 7.8. Summary and Best Practices 7.9. References 7.10. Exercises 7.11. Programming Tasks
8	Obtaining and Validating User Data with HTML Forms	7182	8.1. Introduction 8.2. HTML Form Elements 8.3. Organizing Forms with Fieldsets, Legends, and Labels 8.4. Form Submission and Validation 8.5. Input Sanitization 8.6. Validation Feedback 8.7. Making Forms Accessible 8.8. Building SpeedScore's Login Page 8.9. Summary, Reflections, and Best Practices 8.10. References 8.11. Exercises 8.12. Programming Tasks

9	Using HTML Forms: An Advanced Example	4285	9.1. Introduction 9.2. Creating a User Account Box 9.1: Should We Explicitly Set the Focus When a Web Page Loads? 9.3. Testing the "Create Account" Dialog 9.4. Summary, Reflections, and Best Practices 9.5. References 9.6. Exercises 9.7. Programming Tasks
10	Saving User Data in Local Storage	6260	10.1. Introduction 10.2. Foundations: Web Storage 10.3. Saving Accounts to Web Storage 10.4. Implementing "Account & Profile" Dialog Box 10.1: Using Busy Indicators 10.5. Summary, Reflections and Best Practices 10.6.
11	Working with Data in Tables	7202	11.1. Introduction 11.2. Foundation: HTML Tables 11.3. Adding a Rounds Table to "Rounds" Mode 11.4. Logging a New Round 11.5. Adding a Round to the Table 11.6. Populating the Table with Rounds Data 11.7. Viewing and Editing a Round ~~Box 11.1: Using the History API to Route Pages~~ 11.8. Sorting Rounds by Table Column 11.9. Searching Rounds for a Target Text String 11.10. Summary, Reflections, and Best Practices 11.11. References 11.12. Exercises 11.13. Programming Tasks

*Word counts include only the chapter bodies; abstracts, references, exercises, and programming tasks are not included in these counts.

PART II: FRONT-END DEVELOPMENT IN REACT

#	Chapter	Wd Count*	Chapter Outline (First-Level Headings Only)
12	Welcome to React	6805	12.1. Introduction 12.2. Diving Right In: React with No Installs 12.3. JSX 12.4. Customizing Components with Props 12.5. Composing Components 12.6. Rendering Components Conditionally 12.7. Adding State to Components 12.8. Summary, Reflections, and Best Practices Box 12.1: Class Components 12.9. References 12.10. Exercises 12.11. Programming Tasks
13	Transitioning to the Create React App Framework	5884	13.1. Introduction 13.2. Getting Started with Create React App 13.3. Creating a Starter React App 13.4. Migrating SpeedScore to the Create React App Framework 13.5. Implementing "Courses" Mode as React Component 13.6. Summary, Reflections, and Best Practices 13.7. References 13.8. Exercises 13.9. Programming Tasks
14	Using Web APIs to Obtain Data and Enhance Functionality	7080	14.1. Introduction 14.2. Introduction to Web APIs 14.3. Using a Web API to Implement Golf Course Search 14.4. Initial Implementation: Google Autocomplete Widget 14.5. Improved Implementation: Custom Autocomplete Widget Box 14.1: Supporting Page Routing with React Router 14.6. Summary, Reflections, and Best Practices 14.7. References 14.8. Exercises 14.9. Programming Tasks

15	Architecting React Component Hierarchies	6042	15.1. Introduction 15.2. Rearchitecting `CoursesMode` as a Component Hierarchy 15.3. Parent Component: `Courses` Mode 15.4. Adding Golf Courses: `CoursesModeAdd` Box 15.1: We Can Get Pictures through the Google Places API. Why Not Use Them? 15.5. Viewing a Table of Golf Courses: `CoursesModeTable` 15.6. Searching and Filtering Golf Courses: `CoursesModeSearchFilter` 15.7. Viewing/Editing Golf Courses: `CoursesModeDetails` (Placeholder) 15.8. Summary, Reflection, and Best Practices 15.9. References 15.10. Exercises 15.11. Programming Tasks
16	Managing State and Complexity in React Component Hierarchies	9589	16.1. Introduction 16.2. Inventorying Golf Course Data 16.3. Overview of the Updated Course Details Dialog 16.4. Implementing the Parent Component: `CoursesModeDetails` 16.5. Implementing the Course Info Tab 16.6. Implementing the Speedgolf Info Tab 16.7. Implementing the Scorecard Tab 16.8. Advanced State Management: Using Context and Reducer 16.9. An Alternative Implementation: Using the *Provider* Pattern Box 16.1: Using Redux for Complex State Management 16.10. Summary, Reflections, and Best Practices 16.11. References 16.12. Exercises 16.13. Programming Tasks

*Word counts include only the chapter bodies; the abstract, references, exercises, and programming tasks are not included in these counts.
**Should add material on React router to Chapter 16

PART III: BACK-END DEVELOPMENT WITH NODE, EXPRESS, AND MONGODB

#	Chapter Name	Wd Count*	Chapter Outline (First-Level Headings only)
17	Making Client Web Apps Accessible to the World	6,252	17.1. Introduction 17.2. How Web Apps are Delivered to Users through a Network 17.3. How Web Browsers Communicate with Web Servers 17.4. Deploying a Web App Locally 17.5. Deploying a Web App Remotely 17.6. Summary, Reflections, and Best Practices 17.7. References 17.8. Exercises 17.9. Programming Tasks
18	Using Node and Express to Build a Web App's Back End	10,255	18.1. Introduction 18.2. Using Node.js to Execute Server-Side Code Box 18.1: Server-Side Web Page Generation with Template Systems 18.3. Using Express.js as a Framework for Server-Side Middleware 18.4. Example: An Express App to Maintain SpeedScore Users and Rounds 18.5. Summary, Reflections, and Best Practices 18.6. References 18.7. Exercises 18.8. Programming Tasks
19	Storing App Data Persistently with MongoDB	10,072	19.1. Introduction 19.2. From Relational (SQL) to Document (NoSQL) Databases Box 19.1: Should I Use Email Addresses as a Primary Key? 19.3. Getting Started with MongoDB 19.4. Creating a MongoDB Database for SpeedScore 19.5. Reimplementing SpeedScore Server App with MongoDB Database 19.6. Manual Route Testing with Postman 19.7. Summary, Reflections and Best Practices 19.8. References 19.9. Exercises 19.10. Programming Tasks
20	Architecting Web APIs with Express and MongoDB	9,803	20.1. Introduction 20.2. API Architectural Frameworks 20.3. Making the SpeedScore API RESTful 20.4. Updating the SpeedScore API with a Six Layer Architecture Box 20.1: Default Exports or Named Exports? 20.5. Summary, Reflections, and Best Practices 20.6. References 20.7. Exericises 20.8. Programming Tasks

21	Authenticating Users	10,987	21.1. Introduction 21.2. Security Weaknesses in SpeedScore's Current User Authentication 21.3. Password Salting 21.4. Email Account Verification Box 21.1: Setting Up SendGrid to Work with a Custom Domain 21.5. Email Password Reset 21.6. Multi-Factor Authentication 21.7. Third-Party Authentication: OAuth 21.8. Summary, Reflections and Best Practices 21.9. References 21.10. Exercises 21.11. Programming Tasks
22	Securing API Routes	9,965	22.1. Introduction 22.2. Restricting API Access Based on Authentication Status Box 22.1: CORS Preflight Requests 22.3. Restricting API Access Based on User Roles and Permissions Box 22.2: Security Auditing 22.4. Limiting Route Requests 22.5. Summary, Reflections and Best Practices 22.6. References 22.7. Exercises 22.8. Programming Tasks
23	Testing and Documenting Web APIs	10,267	23.1. Introduction 23.2. API Testing Frameworks 23.3. Developing a Test Suite Collection for the SpeedScore API Box 23.1: API Versioning 23.4. Documenting APIs 23.5. Summary, Reflections, and Best Practices 23.6. References 23.7. Exercises 23.8. Programming Tasks
24	Implementing and Deploying SpeedScore with a Web API	10,819	24.1. Introduction 24.2. Wiring the Client and Server Together 24.3. Supporting Offline Use of Client Applications 24.4. Deploying the Web API 24.5. Interacting with the Deployed SpeedScore Application 24.6. Summary, Reflections and Best Practices 24.7. References 24.8. Exercises 24.9. Programming Tasks

*Word counts include only the chapter bodies; the abstract, references, exercises, and programming tasks are not included in these counts.

APPENDIX A: TEAM WEB DEVELOPMENT LIFECYCLE: AGILE BEST PRACTICES

APPENDIX B: TEAM WEB DEVELOPMENT PROJECT

APPENDIX C: DEPLOYING A CLIENT-SIDE WEB APP TO AMAZON WEB SERVICES, GOOGLE CLOUD PLATFORM, AND AZURE