1. Project Setup and Initial Planning

Define Requirements: Finalize functional requirements, including CRUD operations for book and user management.

Design Architecture: Create an architectural diagram detailing the interaction between AWS services (EC2, ALB, RDS, S3).

Set Up Git Repository: Create a GitHub repository for version control, set up branching strategy, and initialize README with project details.

2. Frontend Development (React)

Set Up Project Structure: Initialize a React project and organize folders for components, services, and assets.

Develop UI Components: Implement essential components like book listings, product details, cart, and user management pages.

Implement API Calls: Set up services for API communication with the backend for CRUD operations.

Initial Testing: Conduct local testing for UI functionality and responsiveness.

3. Backend Development (Spring Boot)

Set Up Project Structure: Create a Spring Boot project, defining layers for controllers, services, and repositories.

Database Models and API Endpoints: Define models for books, users, orders, etc., and create corresponding CRUD endpoints.

Configure AWS RDS: Set up an RDS instance for database management, then connect Spring Boot to RDS using configuration properties.

Implement Business Logic: Add logic for data processing, validation, and error handling.

Test API Endpoints: Test endpoints locally using Postman or equivalent tools to ensure proper functionality.

4. Cloud Infrastructure Setup on AWS

EC2 Instances for Backend: Launch EC2 instances to host the Spring Boot backend, configure security groups, and set up SSH access.

S3 Bucket for Frontend: Create an S3 bucket, configure static website hosting, and upload React build files for serving the frontend.

RDS Database Instance: Launch RDS with appropriate configurations for storage, scalability, and backup settings.

Application Load Balancer (ALB): Configure ALB to route traffic across EC2 instances, enabling high availability.

5. Deployment and Integration

Connect Frontend to Backend: Ensure the React frontend communicates correctly with the backend using deployed API endpoints.

CI/CD Pipeline Setup: Implement a pipeline for continuous integration and deployment, automating updates to the backend and frontend.

Load Testing and Optimization: Test the application under different traffic conditions to ensure ALB is distributing load effectively.

Security Measures: Apply AWS IAM roles and policies to restrict access, secure data in transit and at rest, and handle sensitive user data.

6. Monitoring and Optimization

Implement CloudWatch: Set up CloudWatch for monitoring the health of EC2 instances, load balancer performance, and RDS metrics.

Optimize Cost: Review AWS resource usage and optimize settings for cost efficiency (e.g., reserved instances, storage configurations).

Error Logging and Alerts: Enable error logging and set up alerts for unusual activity, downtime, or critical failures.

7. Testing and Final Adjustments

Comprehensive Testing: Conduct end-to-end tests covering functional, integration, and user experience aspects.

User Acceptance Testing: Gather feedback and make adjustments based on usability and performance.

Finalize Documentation: Complete documentation for codebase, architecture, and operational procedures.

8. Project Deployment and Maintenance

Production Deployment: Deploy the final version to production and conduct a final round of testing to ensure stability.

Maintenance Plan: Establish a plan for regular maintenance, updates, and scalability adjustments based on usage metrics.