15-Day Advanced Java Full-Stack Mini-Project Roadmap: Smart E-Learning Platform with Real-Time Features

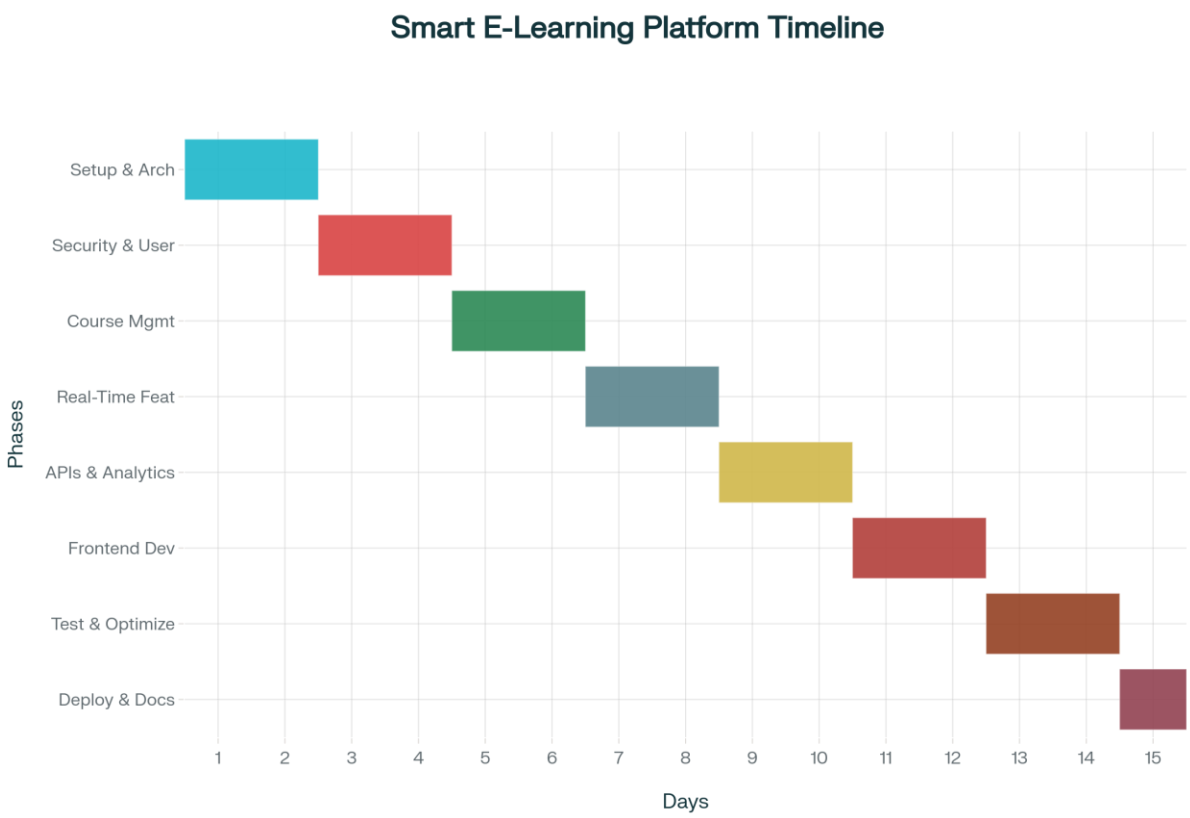
Building upon foundational concepts while introducing **moderately advanced technologies**, this roadmap guides you through creating a **Smart E-Learning Platform** that incorporates **real-time notifications**, **advanced security**, **performance optimization**, and **interactive learning features**. This project strikes the perfect balance between complexity and achievability for college students.[1][2][3]

Recommended Project: "EduSmart - Interactive Learning Management System"

EduSmart differentiates itself from typical college projects by incorporating **industry-relevant features** similar to modern platforms like Coursera, Udemy, and Khan Academy. The system enables **interactive learning experiences** with real-time progress tracking, smart notifications, and advanced analytics.[1][2][4][3][5]

Key Advanced Features (Moderately Complex):

- **Real-time notification system** using WebSocket and Server-Sent Events[6][7]
- **Smart progress tracking** with visual analytics[3][5]
- **Advanced security** with JWT and role-based access control[2][8]
- **Performance optimization** with Redis caching[9][10]
- **Interactive assessment system** with automated grading[2][3]
- **File management** for course materials and assignments[11][12]
- **Discussion forums** with real-time updates[8][3]



15-Day Smart E-Learning Platform Development Timeline

Advanced Technology Stack (Moderately Complex)

Backend Framework: Spring Boot 3.x with Strategic Advanced Features

- **Spring Security 6** with JWT authentication[2][13]
- **Spring Data JPA** with custom repository methods[3][8]
- **Redis** for session management and caching[10][14]
- **WebSocket + SSE** for real-time notifications[6][7]
- **Spring Boot Actuator** for monitoring and health checks[15][16]

Database & Storage:

- **PostgreSQL/MySQL** with optimized queries[2][3]
- **Redis** for caching frequently accessed data[10][17]
- **File storage** with proper upload/download management[11][12]

Frontend Technologies:

- **Thymeleaf** with modern HTML5/CSS3[2][8]
- **Bootstrap 5** for responsive design[3][2]
- **JavaScript ES6+** for interactive features[18][19]
- **WebSocket client** for real-time updates[6][18]

Detailed 15-Day Development Schedule

Phase 1: Advanced Setup & Security Foundation (Days 1-2)

Day 1: Project Architecture & Environment Setup

Morning (4-5 hours):

- Initialize **Spring Boot 3.2+** project with dependencies:[2][8]

```
spring-boot-starter-web
spring-boot-starter-security
spring-boot-starter-data-jpa
spring-boot-starter-data-redis
spring-boot-starter-websocket
spring-boot-starter-thymeleaf
spring-boot-starter-actuator
```

- Set up **PostgreSQL** database with proper configuration[3]
- Configure **Redis** using Docker for development environment[10]
- Design comprehensive **database schema** with relationships[2][3]

Afternoon (4-5 hours):

- Create **multi-layered project structure** (controller, service, repository, dto)[8]
- Set up **Git repository** with proper branching strategy
- Configure **application properties** for multiple environments
- Create **entity classes** with JPA annotations[3][8]

Day 2: Advanced Security Implementation

Morning:

- Implement **Spring Security configuration** with custom authentication[8][2]
- Create **JWT token service** with access token generation[13][20]
- Set up **user registration and login** endpoints[2][3]
- Implement **role-based access control** (Student, Instructor, Admin)[8]

Afternoon:

- Configure **password encryption** with BCrypt[13]
- Create **security filters** and exception handling[2]
- Set up **CORS configuration** for frontend integration
- Implement **user profile management** functionality[3][8]

Phase 2: Core Learning Management Features (Days 3-4)

Day 3: Course Management System

Focus: Comprehensive course and content management[2][3]

Tasks:

- Create **Course entity** with category relationships[8][2]
- Implement **CRUD operations** for courses and categories[3]
- Design **content management system** for lessons and materials[5][11]
- Set up **file upload service** for course materials[11][12]

Day 4: User Management & Enrollment System

Tasks:

- Create **user management service** for all roles[2][8]
- Implement **course enrollment** functionality[3][8]
- Set up **instructor dashboard** for course management[2][3]
- Create **student dashboard** for enrolled courses[8][3]

Phase 3: Advanced Features & Caching (Days 5-6)

Day 5: Redis Caching Implementation

Focus: Performance optimization with strategic caching[10][17]

Tasks:

- Configure **Redis cache manager** with TTL settings[16][10]
- Implement **@Cacheable** for frequently accessed courses[17][16]
- Set up **session management** with Redis[14][21]
- Create **cache warming** strategies for popular content[10]

Day 6: Assessment & Quiz System

Tasks:

- Design **quiz and assessment entities** with relationships[3][8]
- Implement **question bank management** system[2][3]
- Create **automatic grading** functionality[8][3]
- Set up **progress tracking** with completion percentages[5][3]

Phase 4: Real-Time Features Implementation (Days 7-8)

Day 7: WebSocket & Real-Time Notifications

Focus: Real-time communication and notifications[6][7]

Tasks:

- Configure **WebSocket with STOMP** for real-time messaging[22][6]
- Implement **Server-Sent Events (SSE)** for notifications[7][23]
- Create **notification service** with different types[6][7]
- Set up **real-time progress updates** for course completion[5]

Day 8: Interactive Discussion System

Tasks:

- Implement **discussion forum** with real-time updates[3][8]
- Create **chat functionality** for course discussions[11][12]
- Set up **announcement system** with push notifications[5][6]
- Add **real-time user presence** indicators[18][24]

Phase 5: Advanced API Development & Analytics (Days 9-10)

Day 9: Comprehensive REST API Development

Focus: Advanced API design with proper architecture[2][8]

Tasks:

- Create **RESTful endpoints** with proper HTTP methods[25][8]
- Implement **API versioning** and documentation[2]
- Set up **request/response DTOs** with validation[8]
- Create **search and filtering** capabilities[3][5]

Day 10: Analytics & Reporting System

Tasks:

- Implement **learning analytics** dashboard[5][3]
- Create **progress visualization** with charts[1][11]
- Set up **performance metrics** collection[15][11]
- Implement **certificate generation** system[3][8]

Phase 6: Interactive Frontend Development (Days 11-12)

Day 11: Responsive UI Design

Focus: Modern, interactive user interface[2][11]

Tasks:

- Create **responsive Thymeleaf templates** with Bootstrap 5[3][2]
- Implement **dynamic course catalog** with search functionality[5][3]
- Design **interactive dashboards** for different user roles[2][3]
- Set up **form validation** with real-time feedback[8]

Day 12: Advanced JavaScript Integration

Tasks:

- Implement **WebSocket client** for real-time features[6][18]
- Create **interactive quiz interface** with timer functionality[3]
- Set up **progress bars** and completion tracking[5][3]
- Add **file upload** with progress indicators[11][12]

Phase 7: Testing & Performance Optimization (Days 13-14)

Day 13: Comprehensive Testing Strategy

Focus: Quality assurance and reliability[15][26]

Tasks:

- Write **unit tests** for service layer methods[26][27]
- Create **integration tests** for REST endpoints[15][26]
- Implement **security testing** for authentication flows[13][15]
- Set up **WebSocket connection testing**[6][24]

Day 14: Performance Tuning & Monitoring

Tasks:

- **Profile application performance** and identify bottlenecks[28][15]
- Optimize **database queries** and connection pooling[3][8]
- Configure **Spring Boot Actuator** for monitoring[16][15]
- Implement **caching optimization** for improved response times[10][17]

Phase 8: Deployment & Documentation (Day 15)

Day 15: Production Deployment

Tasks:

- Create **Docker containerization** for easy deployment[5]
- Set up **environment-specific configurations**[15][28]
- Deploy to **cloud platform** (Heroku, AWS, or similar)[29]
- Create **comprehensive documentation** and user guides[3][15]
- Set up **basic monitoring** and logging[28][15]

Smart E-Learning Platform Feature Set

Core Learning Features:

1. **Course Management:** Create, organize, and manage educational content[2][3]
2. **Interactive Assessments:** Quizzes, assignments with automatic grading[3][8]
3. **Progress Tracking:** Visual progress indicators and completion analytics[5][3]
4. **Discussion Forums:** Real-time communication between students and instructors[8][3]

Advanced Technical Features:

- **Smart Notifications:** Real-time updates for course activities[6][7]
- **Performance Analytics:** Detailed learning analytics and reporting[5][11]
- **Secure Authentication:** JWT-based security with role management[2][13]

- **Optimized Performance:** Redis caching for improved response times[10][17]
- **File Management:** Robust upload/download system for course materials[11][12]
- **Real-time Collaboration:** Live discussion and instant messaging[18][6]

Technical Implementation Highlights

Smart Caching Strategy:

```
@Service
@EnableCaching
public class CourseService {

    @Cacheable(value = "courses", key = "#categoryId")
    public List<Course> getCoursesByCategory(Long categoryId) {
        return courseRepository.findByCategoryId(categoryId);
    }

    @CacheEvict(value = "courses", allEntries = true)
    public Course createCourse(Course course) {
        return courseRepository.save(course);
    }
}
```

Real-Time Notification System:

```
@Controller
public class NotificationController {

    @Autowired
    private SimpMessagingTemplate messagingTemplate;

    public void sendProgressUpdate(Long userId, ProgressUpdate update) {
        messagingTemplate.convertAndSendToUser(
            userId.toString(),
            "/queue/progress",
            update
        );
    }
}
```

Advanced Security Configuration:

```
@Configuration
@EnableWebSecurity
public class SecurityConfig {

   @Bean
   public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {
      http
         .authorizeHttpRequests(authz -> authz
            .requestMatchers("/api/admin/**").hasRole("ADMIN")
            .requestMatchers("/api/instructor/**").hasRole("INSTRUCTOR")
            .requestMatchers("/api/student/**").hasRole("STUDENT")
            .anyRequest().authenticated()
         )
         .oauth2ResourceServer(oauth2 -> oauth2.jwt());
      return http.build();
   }
}
```

Advanced Learning Outcomes & Skills

Technical Skills Demonstrated:

- **Spring Boot ecosystem** mastery with advanced features[15][28]
- **Real-time application** development with WebSocket/SSE[6][7]
- **Security best practices** with JWT and Spring Security[2][13]
- **Performance optimization** with caching strategies[10][17]
- **Modern web development** with responsive design[11][2]

Software Architecture Concepts:

- **Layered architecture** with proper separation of concerns[8][12]
- **RESTful API design** with proper HTTP methods[25][2]
- **Database optimization** with efficient query patterns[3][8]
- **Real-time system design** for notification delivery[6][7]

Project Scalability & Extension Options

If Ahead of Schedule:

- Add **email notification** integration for course updates[11]
- Implement **mobile-responsive PWA** features[4][11]
- Create **advanced analytics** with data visualization[5][30]
- Add **AI-powered recommendation** system for courses[1][30]

If Behind Schedule:

- Focus on **core CRUD operations** first[2][8]
- Use **Bootstrap templates** for faster UI development[3]
- Implement **basic caching** without Redis complexity[16]
- Simplify **real-time features** to basic polling[7]

Portfolio Impact & Industry Relevance

Why This Project Stands Out:

- **Addresses real-world needs** in the growing e-learning market[1][31]
- **Demonstrates modern tech stack** used in enterprise applications[15][28]
- **Shows understanding of user experience** and interface design[4][11]
- **Includes performance considerations** essential for scalable applications[5][10]

This **Smart E-Learning Platform** represents a **sweet spot** between complexity and achievability, incorporating **enterprise-level concepts** without overwhelming complexity. The project demonstrates your ability to work with **modern technologies**, **real-time systems**, and **performance optimization** - skills highly valued in today's software development landscape.[11][12][1]

The platform's **educational focus**, **real-time features**, and **comprehensive user management** showcase your understanding of **full-stack development**, **system design**, and **user-centered application development** - making it an excellent addition to your portfolio for **mid-level developer positions**.

1. https://elearningindustry.com/elearning-in-2025-key-trends-shaping-the-future-of-education
2. https://www.codewithmurad.com/2024/07/learning-management-system-project-using-springboot-reactjs.html

3. https://github.com/PATMESH/Learning-Management-System
4. https://www.vis10dwarka.com/blog/virtual-classrooms-unpacked-choosing-the-right-e-learning-platform-for-your-school
5. https://www.temjournal.com/content/142/TEMJournalMay2025_1657_1669.pdf
6. https://blog.cubed.run/real-time-notifications-in-spring-boot-e69b5966183d
7. https://roytuts.com/server-sent-events-spring-push-notifications/
8. https://codebun.com/learning-management-system-in-spring-boot-and-hibernate-with-source-code/
9. https://dev.to/olymahmud/building-high-performance-apps-with-redis-postgresql-spring-boot-3m36
10. https://www.geeksforgeeks.org/advance-java/spring-boot-caching-with-redis/
11. https://www.techavidus.com/case-studies/e-learning-app-development
12. https://moldstud.com/articles/p-developing-e-learning-platforms-with-java
13. https://www.bezkoder.com/spring-boot-jwt-authentication/
14. https://developer.okta.com/blog/2020/12/14/spring-session-redis
15. https://www.coursera.org/learn/java-spring-boot-advanced
16. https://docs.spring.io/spring-boot/reference/io/caching.html
17. https://www.squash.io/tutorial-on-integrating-redis-with-spring-boot/
18. https://www.djamware.com/post/687c4fd6753e267cbbd0376e/realtime-chat-app-with-java-websocket-and-spring-boot
19. https://www.geeksforgeeks.org/advance-java/building-real-time-applications-with-java-spring-boot-and-websocket/
20. https://blog.devops.dev/comprehensive-guide-to-spring-security-with-jwt-refresh-token-database-and-redis-671acaf0f2ed
21. https://stackoverflow.com/questions/55967830/how-to-save-jwt-in-redis
22. https://www.toptal.com/java/stomp-spring-boot-websocket
23. https://vaadin.com/blog/send-web-push-notifications-java
24. https://videosdk.live/developer-hub/websocket/spring-boot-websocket
25. https://spring.io/guides/tutorials/rest
26. https://www.coursera.org/learn/advanced-spring-boot-development

27. https://www.baeldung.com/spring-boot

28. https://www.geeksforgeeks.org/advance-java/spring-boot/

29. https://www.bacancytechnology.com/case-study/java/elearning

30. https://360learning.com/blog/ai-learning-platforms/

31. https://savvycomsoftware.com/blog/top-10-e-learning-platforms-in-the-2025/

32. https://github.com/abdomohsen228/Learning_Management_System

33. https://www.upgrad.com/blog/java-project-ideas-topics-for-beginners/

34. https://ijarsct.co.in/Paper9226.pdf

35. https://www.guvi.in/blog/java-project-ideas-of-all-levels/

36. https://www.youtube.com/watch?v=Pulk8JrPPoA

37. https://www.interviewbit.com/blog/java-projects/

38. https://www.geeksforgeeks.org/blogs/java-projects/

39. https://www.ccbp.in/blog/articles/java-projects

40. https://spring.io/projects/spring-boot

41. https://unstop.com/blog/java-project-ideas

42. https://github.com/SuwaidAslam/E-Learning-System-Java-GUI-Application

43. https://www.youtube.com/watch?v=MyiwqspNcLg

44. https://www.youtube.com/watch?v=oRfx3UDiYoY

45. https://www.appvizer.com/magazine/hr/learning-management-system/e-learning-platform

46. https://www.youtube.com/watch?v=gJrjgg1KVL4

47. https://www.kashipara.com/project/idea/java/online-learning-platform_4483.html

48. https://thirst.io/blog/top-12-learning-experience-platforms/

49. https://spring.io/guides/gs/spring-boot