Classroom

Overview

Classroom is a comprehensive learning management system (LMS) designed to bridge the gap between instructors and students through a unified digital platform. Built with modern web technologies, it provides a seamless experience for course creation, enrollment management, assignment distribution, and progress tracking. The platform addresses the core challenges of online education by consolidating scattered workflows into a cohesive, intuitive system that prioritizes clarity and structure.

Features

For Instructors

- **Course Creation & Management**: Design and publish courses with detailed metadata including title, description, category, difficulty level, and thumbnail images
- **Dashboard Analytics**: Monitor total courses created, student enrollment statistics, and pending assignment reviews through an intuitive dashboard
- **Course Portfolio Management**: View and manage all published courses with real-time enrollment data and completion rates
- **Assignment Distribution**: Create and distribute assignments with deadlines and track submission status
- **Material Upload**: Organize and share learning materials (PDFs, videos, links) within course modules

For Students

- **Course Discovery**: Browse comprehensive course catalog with advanced search and filtering by category, level, and rating
- **Enrollment System**: One-click enrollment in courses with instant access to materials and assignments
- **Progress Tracking**: Visual dashboard displaying enrolled courses, completion percentages, and recent activity
- **Personalized Learning**: Access enrolled courses through a dedicated "My Courses" section with progress indicators
- **Assignment Management**: View assignments with due dates and submission status tracking

Platform Features

- **Role-Based Access Control**: Separate authentication flows and dashboards for instructors and students
- **Responsive Design**: Fully responsive UI optimized for desktop, tablet, and mobile devices
- **Real-Time Updates**: Dynamic content updates without page reloads using React client components
- **Course Rating System**: 5-star rating display with review counts and student enrollment metrics
- **Smooth Navigation**: Intersection Observer-based active section highlighting in navigation

Tech Stack

Frontend Framework

- **Next.js 16.1.6**: React-based framework with App Router for server-side rendering and routing
- **React 19.2.3**: Component-based UI library with hooks for state management
- **TypeScript 5.x**: Static typing for enhanced code quality and developer experience

Styling & UI

- **Tailwind CSS 4.x**: Utility-first CSS framework for rapid UI development
- **Custom Design System**: Consistent color palette (#4F46E5 primary, #F9FAFB background) and typography
- **Geist Font Family**: Optimized font loading using next/font for improved performance

Development Tools

- **ESLint 9.x**: Code linting with Next.js configuration for code quality enforcement
- **PostCSS**: CSS processing with Tailwind CSS integration
- **TypeScript Compiler**: Strict type checking with ES2017 target and JSX support

Architecture Patterns

- **App Router Architecture**: File-based routing with nested layouts and route groups
- **Client-Side State Management**: React Context API for enrollment state management
- **Component Composition**: Reusable components with props-based customization
- **Image Optimization**: Next.js Image component for automatic image optimization

Contributors

Harsh

Role: Frontend Developer & UI/UX Designer
Responsibilities:

- Designed and implemented the landing page with hero section and feature highlights
- Built responsive navigation with active section tracking using Intersection Observer
- Developed consistent UI using Tailwind CSS and the defined design system
- Ensured layout responsiveness across desktop, tablet, and mobile screens

Sunil

Role: Data Modeling & Application Logic Developer
Responsibilities:

- Defined TypeScript interfaces for courses, instructors, and assignments
- Structured the centralized course data module with sample course entries
- Implemented logic for course metadata including ratings and enrollment counts
- Designed assignment and material data structures for instructor workflows

Kishan

Role: Full-Stack Developer & Authentication Lead
Responsibilities:

- Implemented role-based authentication flow for instructor and student access
- Developed instructor dashboard with course creation and management features
- Built validated course creation form with category selection and media handling
- Configured Next.js App Router structure and project-level TypeScript setup

Payaswini

Role: Student Experience Developer
Responsibilities:

- Developed student dashboard with enrolled course tracking and progress indicators
- Implemented course browsing with search, filtering, and sorting functionality
- Built enrollment handling using React Context API for state management
- Created "My Courses" section with completion tracking and activity updates

Chandhini

Role: UI Component Developer & Testing Support
Responsibilities:

- Developed reusable UI components such as course cards and stat panels
- Implemented detailed instructor course view interfaces
- Tested UI responsiveness across devices for consistent user experience
- Assisted in refining component structure and improving accessibility

Vaishnavi

Role: Application Quality & Optimization Support
Responsibilities:

- Assisted in improving UI performance and reducing unnecessary re-renders
- Verified build configurations and ensured smooth production build generation
- Supported code quality checks using ESLint and TypeScript strict settings
- Helped coordinate testing and final prototype refinement before submission

---

Getting Started

Prerequisites

- Node.js 20.x or higher
- npm, yarn, pnpm, or bun package manager

Installation

1. Clone the repository:

```
git clone <repository-url>
cd classroom
```

2. Install dependencies:

```
npm install
```

3. Run the development server:

```
npm run dev
```

4. Open http://localhost:3000 in your browser

Available Scripts

- `npm run dev` - Start development server with hot reload
- `npm run build` - Create optimized production build
- `npm run start` - Start production server
- `npm run lint` - Run ESLint for code quality checks

Project Structure

```
classroom/
├── app/
│   ├── (auth)/             # Authentication routes (login, signup)
│   ├── data/               # Course data and TypeScript interfaces
│   ├── instructor/         # Instructor dashboard and course management
│   ├── student/            # Student dashboard and course browsing
│   ├── globals.css         # Global styles and Tailwind imports
│   ├── layout.tsx          # Root layout with navigation
│   └── page.tsx            # Landing page
├── components/             # Reusable React components
├── public/                 # Static assets (images, icons)
└── package.json            # Project dependencies and scripts
```

License

This project is developed for educational purposes.