An Najah National University
Faculty of Engineering & Information Technology
Department of Computer Engineering

Software Graduation Project 1

NexusQuest

Prepared by:

Mohammed Jaddou

Amjad Mousa

Supervised by:

Dr. Samer Arandi

Acknowledgment

We would like to express our profound appreciation to everyone who helped make NexusQuest Platform, a feature-rich interactive coding education platform with real-time collaboration capabilities, a success. We would also like to thank Dr. Samer Arandi, our academic supervisor, for his invaluable advice and support during the project's progress.
As the platform's creators, we are proud to have created a platform that will transform coding education by giving educators and students effective tools for gamified skill development, interactive learning, and real-time collaboration. We would also want to thank our friends and family for their unfailing encouragement and support during this difficult development process. Lastly, we would like to express our gratitude to everyone who helped us construct a platform that really meets the requirements of the educational community by providing input and encouragement.

Abstract

NexusQuest is an all-inclusive interactive coding education platform that aims to revolutionize the way instructors oversee instructional materials and students learn programming. Students and teachers are the two primary user roles on the site, each with unique interfaces and features catered to their particular need.
The first major feature is the Interactive Learning System, where students may take part in timed quizzes, access language-specific lessons, finish coding assignments with automatic grading, and take part in daily coding challenges. Real-time code execution in many languages

(JavaScript, Python, Java, and C++), immediate feedback methods, and thorough progress monitoring are all features of the system.

Real-Time Collaboration, the second key feature, allows students to participate in live pair programming sessions with interactive chat, real-time code synchronization, and video/audio communication. Peer-to-peer communication via WebRTC, cursor tracking, screen sharing, and session management with invite capability are all included in the system.
Gamification & Community, the third major element, allows students to compete on leaderboards, gain experience points, unlock accomplishments, and take part in a Q&A forum. Comprehensive analytics dashboards, accomplishment tracking, streak maintenance, and community-driven learning via Q&A conversations are all features of the system.
Additionally, NexusQuest has an AI-Powered Assistant that uses sophisticated natural language processing to offer intelligent code assistance, recommendations, and explanations. Role-based access control, secure user authentication using JWT tokens, and thorough data management are all features of the system.

The platform is built using MongoDB for data storage, Node.js with Express.js framework for the backend API, React framework for the web application, and React Native for the mobile app, ensuring cross-platform compatibility and optimal performance.

Introduction

Statement of the Problem

In today's digital education landscape, coding education faces significant challenges that hinder effective learning and teaching experiences. Traditional coding education platforms often lack interactivity, real-time collaboration features, and engaging elements. The current coding education ecosystem suffers from:

- Limited Interactivity: Static content without real-time feedback and engagement
- Isolation: Lack of collaborative learning environments and peer interaction
- Fragmented Tools: Separate platforms for learning, practice, and collaboration
- Poor Engagement: Limited gamification and motivation mechanisms
- Teacher Challenges: Inadequate tools for content creation and student management
- Accessibility: Limited mobile support and offline capabilities

These challenges result in reduced student engagement, inefficient teaching workflows, and missed opportunities for creating comprehensive coding education experiences.

Objectives of the Work

The primary objectives of developing NexusQuest platform are:

Primary Objectives:

- To create a comprehensive interactive coding education platform that serves both students and teachers
- To implement real-time collaborative coding features
- To develop a gamified learning system with achievements, leaderboards, and progress tracking
- To provide multi-language code execution
- To create specialized interfaces for different user roles (students and teachers)

Secondary Objectives:

- To support teachers with powerful content creation and student management tools
- To create an engaging community-driven learning environment
- To develop scalable architecture that can accommodate educational institutions
- To provide comprehensive analytics for both learning and teaching optimization

Scope of the Work

The scope of this project encompasses:

Technical Scope:

- Development of a full-stack web and mobile application using React, React Native, Node.js, and MongoDB
- Implementation of WebSocket-based real-time communication using Socket.io
- Multi-language code execution with Docker-based isolated environments
- AI integration for intelligent code assistance and suggestions
- Cross-platform deployment (web and mobile)

Functional Scope:

- Interactive tutorial system with progress tracking
- Real-time collaborative coding sessions

- Gamified learning with XP, achievements, and leaderboards
- Automated code testing and grading system
- Community forum with Q&A capabilities
- Teacher dashboard with analytics and student management
- AI-powered code assistant and explanations

Geographic Scope:

- Global accessibility with multi-language support potential
- Scalable design for educational institutions worldwide
- Cloud-based infrastructure for reliable access

Significance of the Work

This project holds significant importance for several reasons:

Educational Impact:

- **Learning Enhancement**: Transforms passive coding education into interactive, engaging experiences
- **Collaboration Skills**: Develops essential teamwork and communication skills through pair programming
- **Accessibility**: Makes quality coding education accessible to learners worldwide
- **Teacher Empowerment**: Provides educators with powerful tools for content creation and management

Social Impact:

- **Community Building**: Creates a supportive learning community
- **Skill Development**: Addresses the growing demand for coding skills
- **Educational Equity**: Provides quality education regardless of location
- **Innovation in Education**: Sets new standards for interactive learning platforms

Technical Innovation:

- **Real-Time Collaboration**: Demonstrates advanced WebRTC and WebSocket integration

- **Multi-Language Execution:** Shows sophisticated sandboxed code execution
- **AI Integration:** Practical application of AI in educational contexts
- **Cross-Platform:** Unified experience across web and mobile platforms

Constraints, Standards/Codes, and Earlier Coursework

Constraints and Limitations

Technical Constraints:

- **Real-Time Performance:** Network latency affecting collaboration quality
- **Code Execution Security:** Ensuring secure sandboxed execution environments
- **Scalability:** Managing concurrent users in real-time sessions
- **Mobile Limitations**: Device performance constraints for code editing
- **Browser Compatibility**: Ensuring consistent experience across browsers

Regulatory Constraints:

- **Data Privacy:** Compliance with educational data protection regulations
- **Content Security:** Ensuring secure code storage and execution
- **Accessibility:** Adherence to web accessibility standards

Standards and Codes

Backend – Node.js & Express.js REST Framework

Key Standards:

- **RESTful API Design:** Following REST architectural principles with proper HTTP status codes
- **TypeScript Integration**: Type safety and enhanced development experience
- **Security Best Practices:** Following OWASP guidelines for web application security

- **WebSocket Implementation:** Real-time bidirectional communication standards
- **Environment Configuration:** Secure handling of sensitive configuration data

Database – MongoDB

Key Standards:

- **Mongoose ODM:** Using Mongoose for schema validation and data modeling
- **Document Structure:** Following MongoDB document design best practices
- **Indexing Strategy:** Optimized indexes for educational data queries
- **Data Relationships:** Using references and embedded documents appropriately
- **Connection Security:** Database access control and secure connections

Frontend – React (Web) & React Native (Mobile)

Key Standards:

- **Component Architecture:** Functional components with hooks and TypeScript
- **State Management:** React Context API for global state management
- **Responsive Design:** Mobile-first responsive design principles
- **Code Quality:** ESLint/Prettier for consistent code formatting
- **Real-Time Updates:** Socket.io client integration for live features

Real-Time Communication – WebRTC & Socket.io

Key Standards:

- **WebRTC Protocols:** Peer-to-peer communication standards
- **Socket.io Events**: Structured real-time event handling
- **Security:** Secure WebRTC connection establishment
- **Performance:** Optimized data transmission for collaboration

AI Integration – OpenAi

Key Standards:

- **API Security:** Secure communication with AI service providers
- **Data Privacy:** Minimal data sharing with external AI services

- **Response Caching:** Efficient caching of common AI responses
- **Error Handling:** Robust error handling for AI service failures

Methodology

Tools, Methods, and Programming Languages

Tools

Development Environment:

- **Visual Studio Code**: Primary code editor with TypeScript, React, and Node.js extensions
- **Git & GitHub**: Version control system for collaborative development and code management
- **Node.js**: JavaScript runtime environment for backend development
- **Expo CLI**: Development tools for React Native mobile app development
- **Vite**: Fast build tool and development server for React web application

Testing and Quality Assurance:

- **Jest**: JavaScript testing framework for unit testing
- **React Testing Library**: Testing utilities for React components
- **ESLint**: Code linting and style enforcement
- **Prettier**: Code formatting and consistency
- **TypeScript**: Static type checking and error detection

Database and Backend Tools:

- **MongoDB Compass**: Database management and visualization tool
- **Mongoose**: MongoDB object modeling tool for Node.js
- **Postman**: API testing and documentation tool
- **Docker:** Containerization for code execution environments

Deployment and DevOps:

- **Docker Compose**: Multi-container application orchestration

- **Environment Variables**: Secure configuration management
- **MongoDB Atlas:** Cloud database hosting service

Programming Languages & Frameworks

Backend Technologies:

- **Node.js**: JavaScript runtime for server-side development
- **Express.js:** Web application framework for Node.js
- **TypeScript:** Typed superset of JavaScript for better code quality
- **Socket.io**: Real-time bidirectional communication framework

Frontend Technologies:

- **React:** JavaScript library for building user interfaces
- **React Native**: Framework for cross-platform mobile development
- **TypeScript:** Type safety and enhanced development experience
- **React Router:** Declarative routing for React web applications

State Management:

- **React Context API:** Built-in state management for component communication
- **Local Storage:** Client-side data persistence
- **AsyncStorage**: Local storage solution for React Native

Styling and UI:

- **Tailwind CSS:** Utility-first CSS framework for rapid UI development
- **shadcn/ui:** Modern React component library
- **Lucide React:** Icon library for React applications
- **Framer Motion:** Animation library for React components

Authentication and Security:

- **JWT (JSON Web Tokens):** Secure token-based authentication
- **bcryptjs:** Password hashing and security

- **Helmet**: Security middleware for Express.js
- **CORS:** Cross-Origin Resource Sharing configuration

Database

Database System:

- **MongoDB**: NoSQL document-oriented database
- **MongoDB Atlas**: Cloud-hosted MongoDB service
- **Mongoose**: MongoDB object modeling tool for Node.js

Database Design:

- **Document-Based Architecture**: Flexible schema design for educational data
- **Collections**: Organized data storage (Users, Tasks, Tutorials, Quizzes, etc.)
- **Indexing**: Optimized query performance with strategic indexing
- **Data Relationships**: Reference-based relationships between documents

Data Models:

- **User Model**: Student and teacher profiles with gamification data
- **Task Model**: Coding challenges with test cases and automated grading
- **Tutorial Model**: Interactive learning content with progress tracking
- **Quiz Model**: Timed assessments with various question types
- **Collaboration Model**: Real-time session data and communication logs
- **Achievement Model**: Gamification system with badges and rewards

Database Operations:

- **CRUD Operations**: Create, Read, Update, Delete operations
- **Aggregation Pipeline**: Complex data processing for analytics
- **Data Validation**: Schema validation using Mongoose
- **Connection Pooling**: Efficient database connection management
- **Environment Configuration**: Secure database connection strings

Real-Time Communication and AI Integration

Real-Time System Architecture:

- **WebSocket Communication:** Socket.io for real-time bidirectional communication
- **Session Management:** Real-time session creation and management
- **Code Synchronization:** Live code updates with conflict resolution

AI System Integration:

- **OpenAI API**: Integration with GPT models for code assistance
- **Context Management:** Maintaining conversation context and user history
- **Code Analysis:** Intelligent code review and suggestions
- **Response Generation:** AI-powered explanations and help

Real-Time Features:

- **Live Collaboration:** Real-time code editing with multiple users
- **Cursor Tracking:** Visual indicators of user actions
- **Session Invites:** User invitation and session management

Results and Discussion

Web and Mobile Features

This chapter presents the comprehensive results of implementing the NexusQuest platform, showcasing the successful development of a multi-platform application with specialized interfaces for students and teachers.

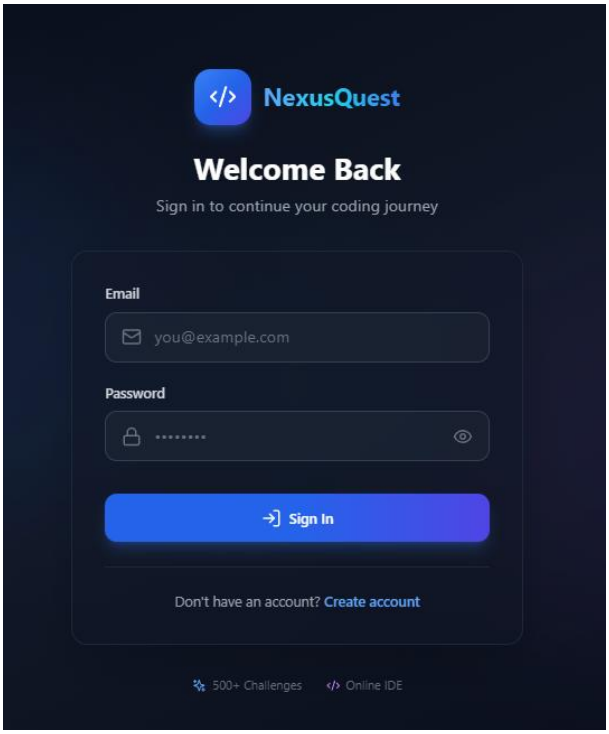
Common Features Among Users

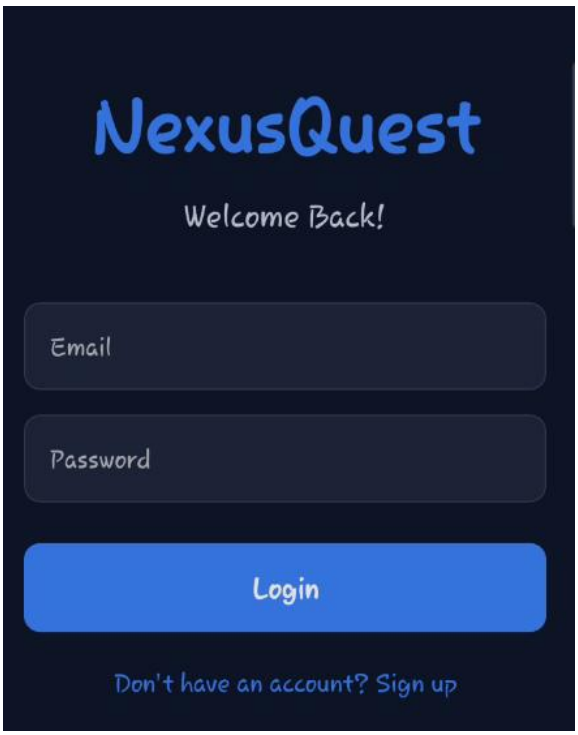
Authentication System

Login Screen

- **Implementation**: Professional login interface with NexusQuest branding
- **Features:** Email and password authentication, "Remember Me" functionality
- **Security**: JWT token-based authentication with refresh token mechanism
- **UI/UX:** Responsive design with animated logo and form validation

1: login screen

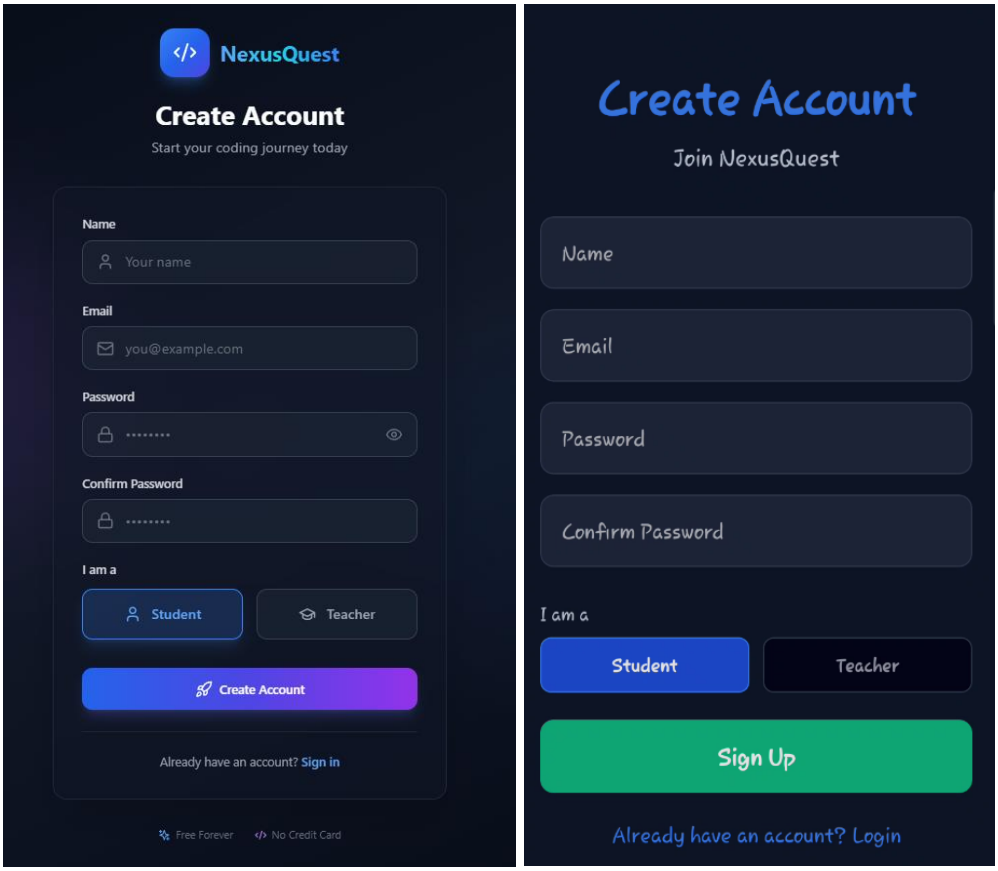




Registration

- **Implementation:** Multi-step registration process for different user types
- **Features:** Student and teacher registration with role-specific information
- **Validation**: Real-time form validation and error handling
- **UI/UX:** Step-by-step wizard with progress indicators

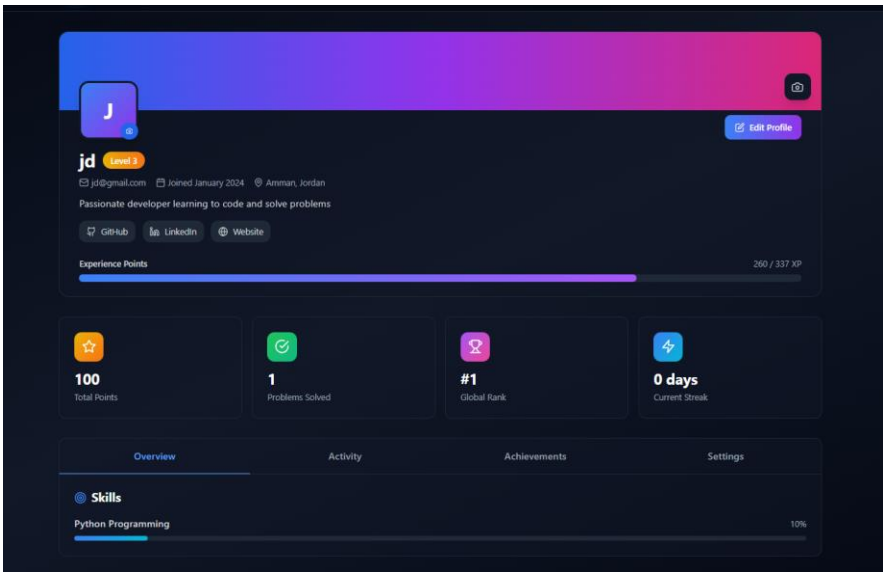
2: SignUp screen

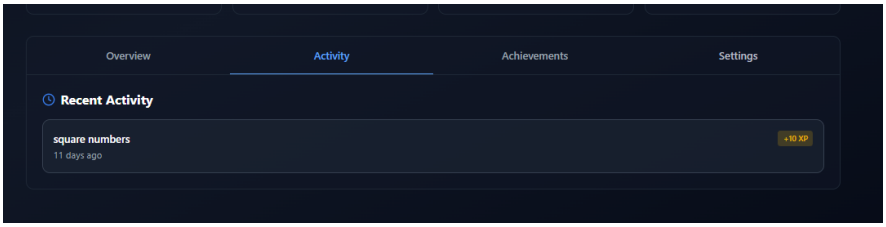


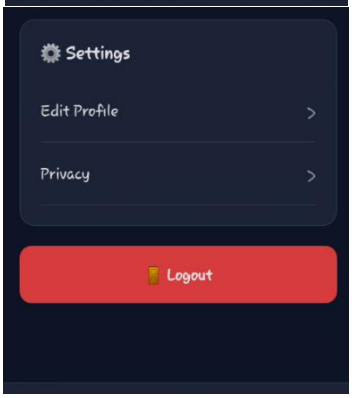
Profile Management

- **Implementation:** Comprehensive user profile display with gamification stats
- **Features:** Personal information, achievements, XP points, and level progression
- **UI/UX:** Clean profile layout with edit functionality and achievement showcase
- **Data**: Real-time profile data from MongoDB with gamification integration

3: Profile screen







Multi-Language Project Builder

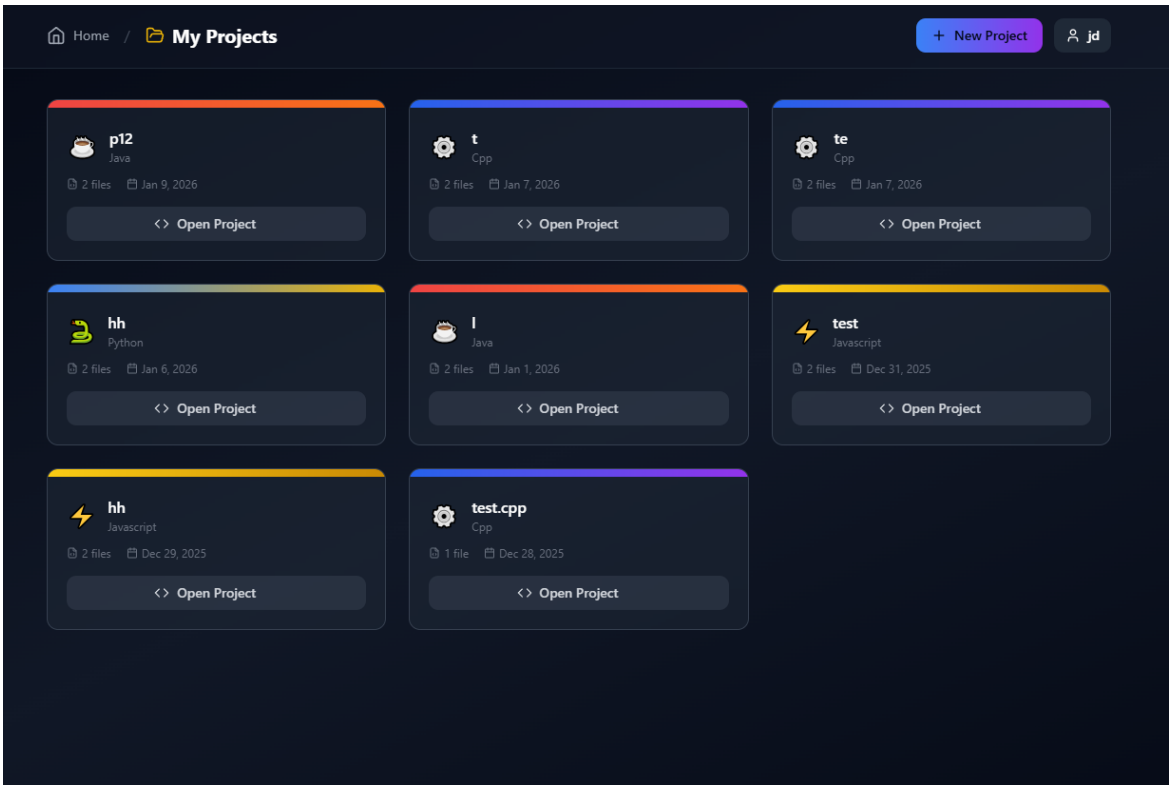
• **Implementation:** Unified development environment supporting Python, C++, Java, and JavaScript projects

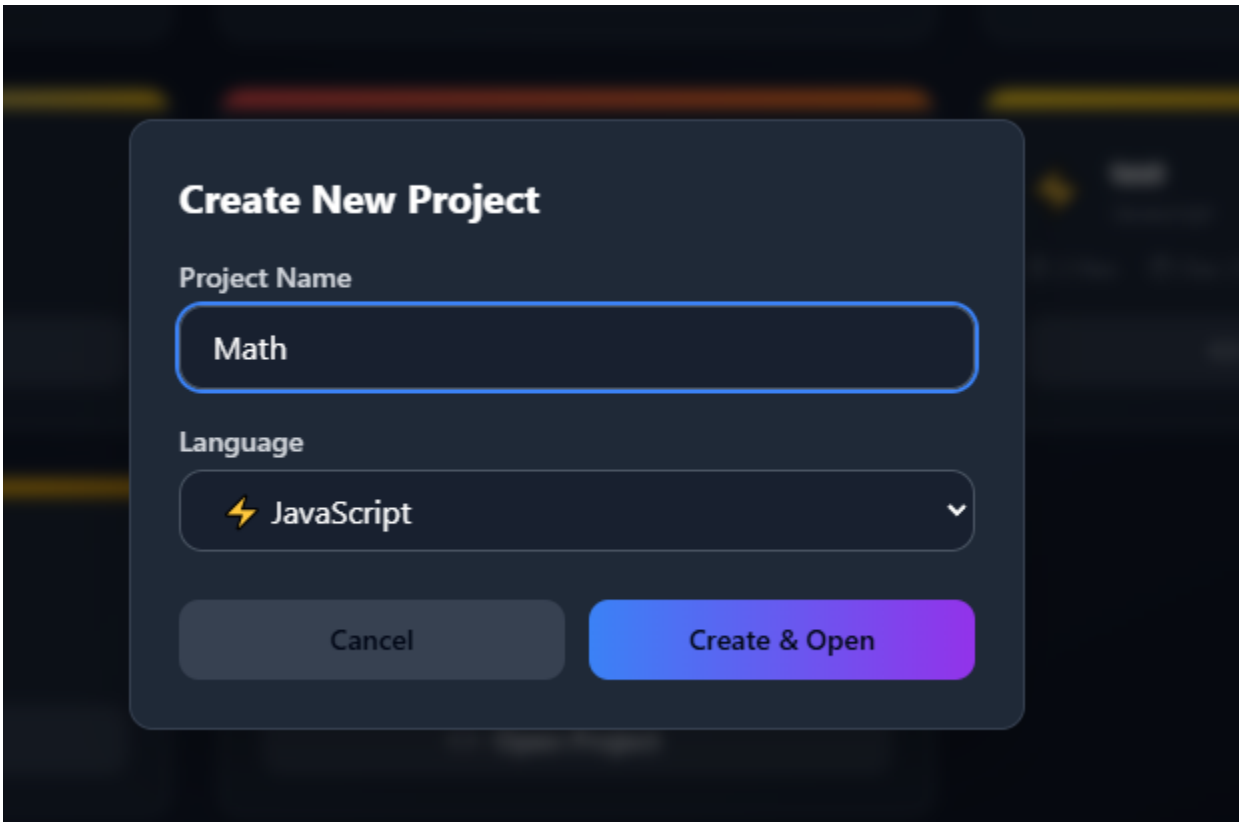
• **Features:** Dedicated package manager per language (pip, conan, Maven, npm), dependency management, and custom library integration

• **UI/UX:** Structured project workspace with language separation, dependency viewer, and intuitive project configuration

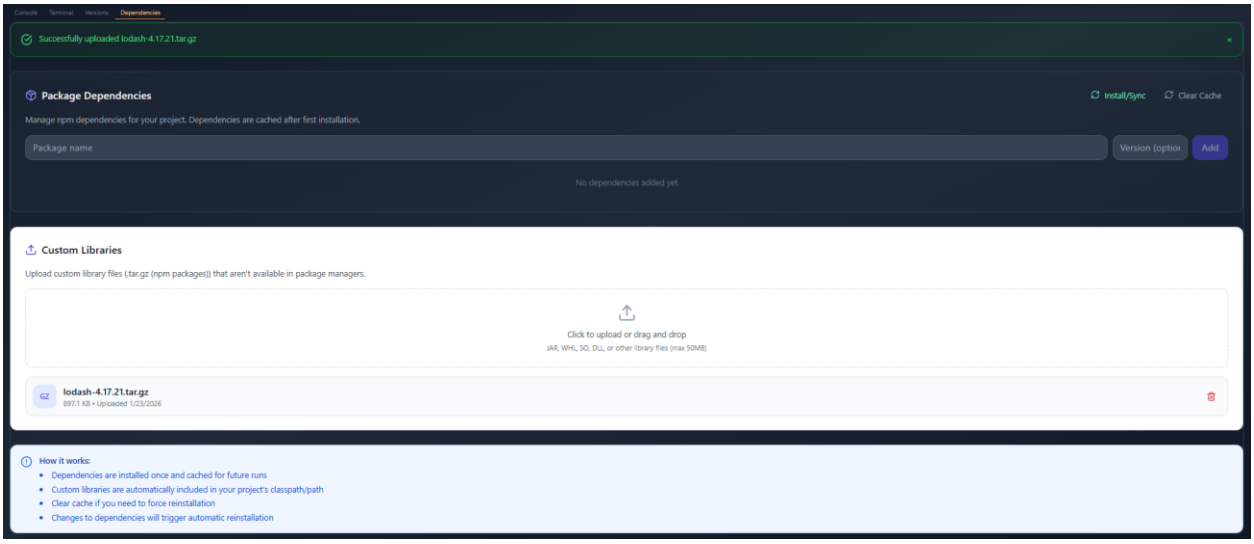
• **Data:** Built-in version control using snapshot-based tracking for project states, dependency versions, and custom libraries

4: My Project screen

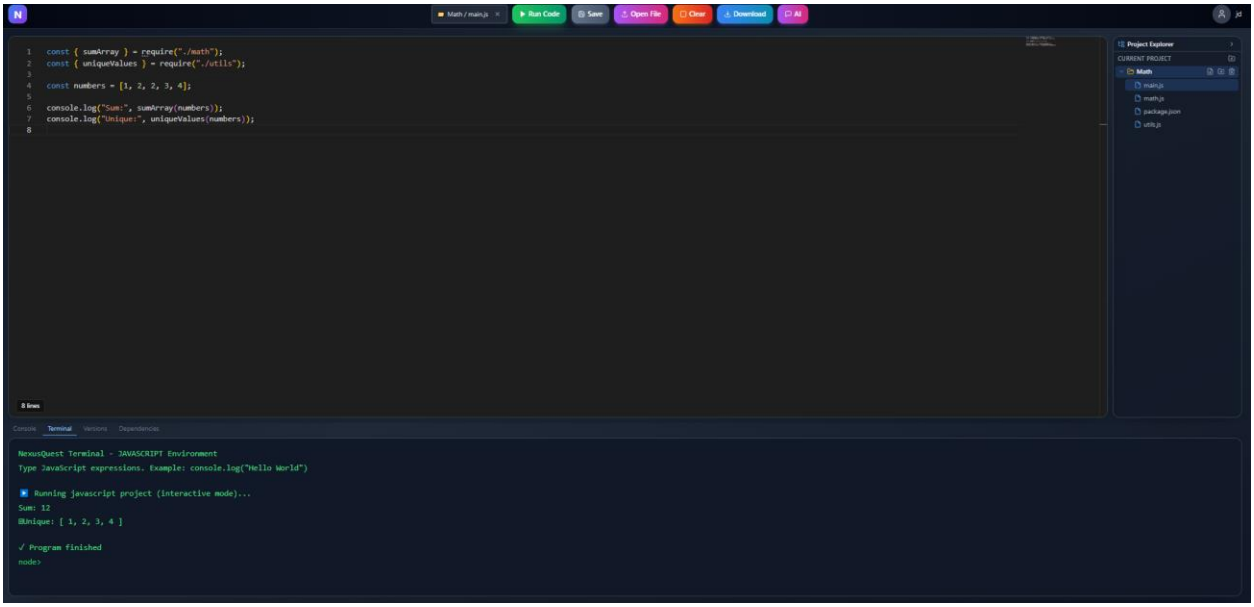




5: Dependencies tap



6: Project screen



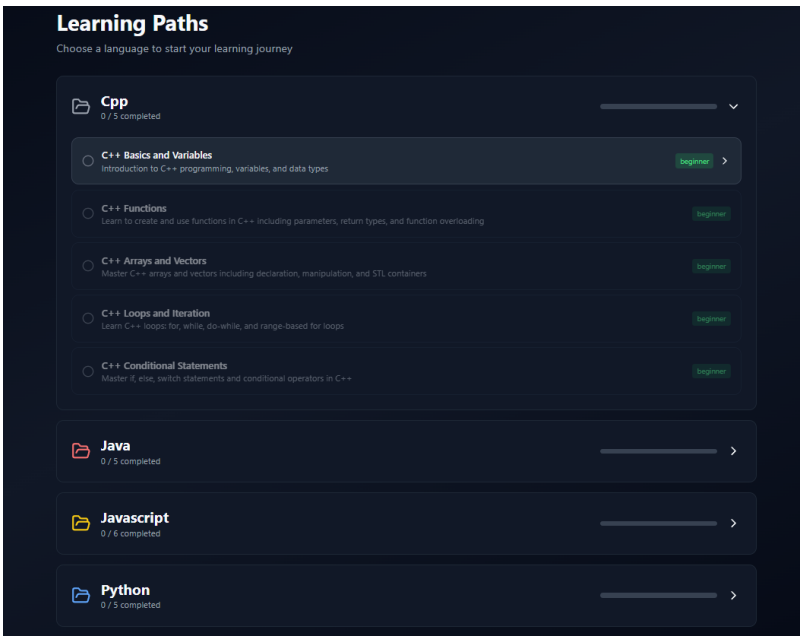
Student Features

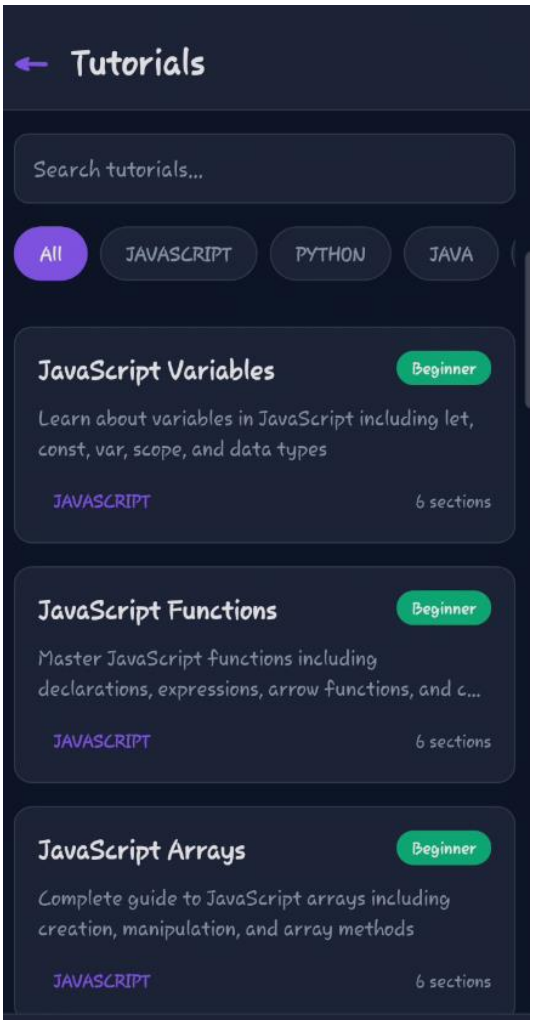
Interactive Learning System

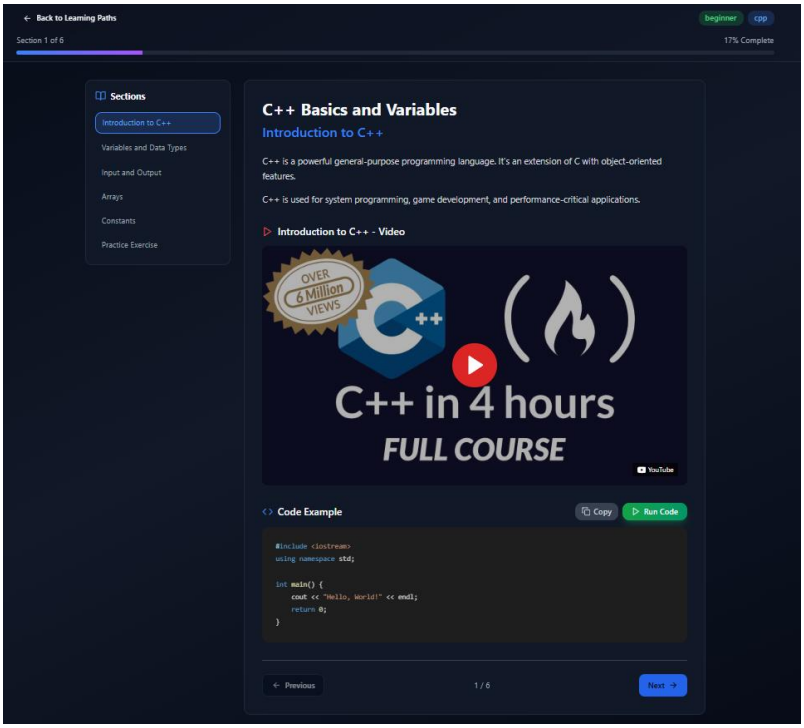
Tutorial Navigation

- **Implementation**: Language-specific tutorial browsing with progress tracking
- **Features:** Interactive tutorials with code examples, video integration, and quizzes
- **UI/UX:** Card-based layout with progress indicators and difficulty levels
- **Performance:** Optimized loading with lazy loading for large content

7: Tutorials screen





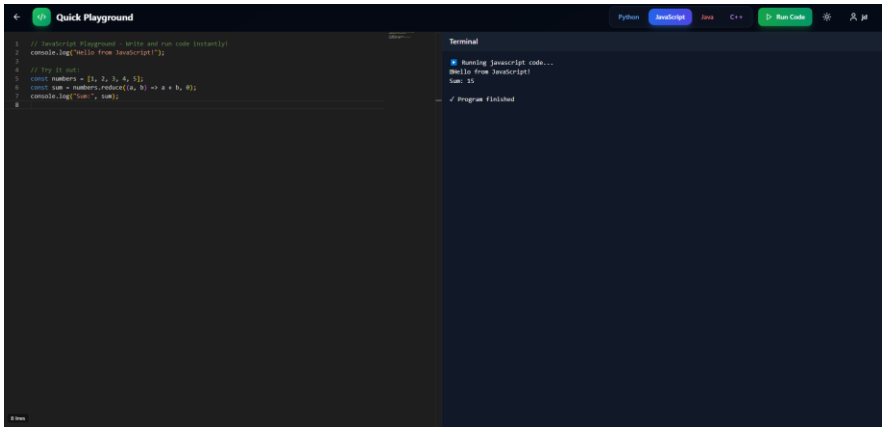


Code Playground

- **Implementation:** Multi-language code execution environment
- **Features:** Support for JavaScript, Python, Java, and C++ with real-time output
- **UI/UX:** Monaco Editor integration with syntax highlighting and IntelliSense

- **Security:** Sandboxed execution with resource limits and timeout protection

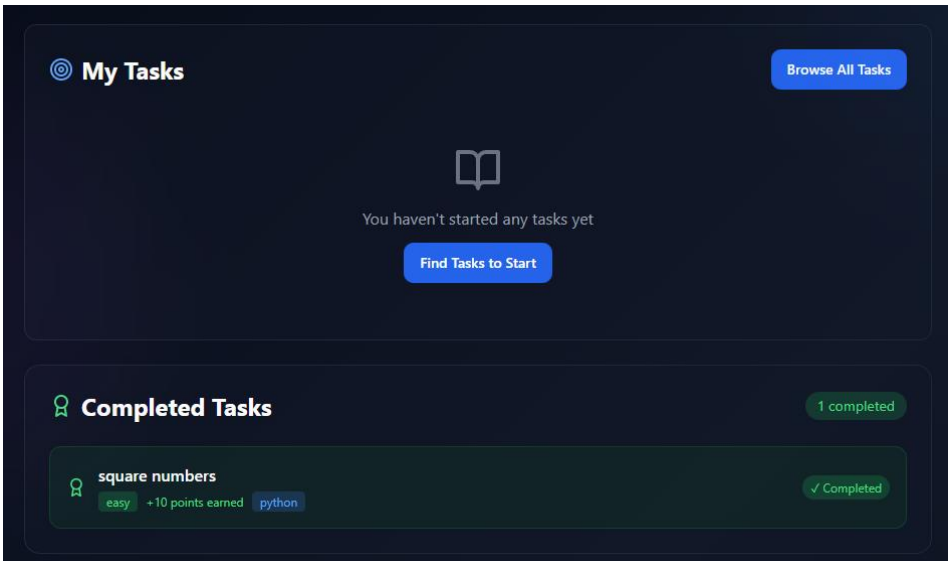
8: Playground screen

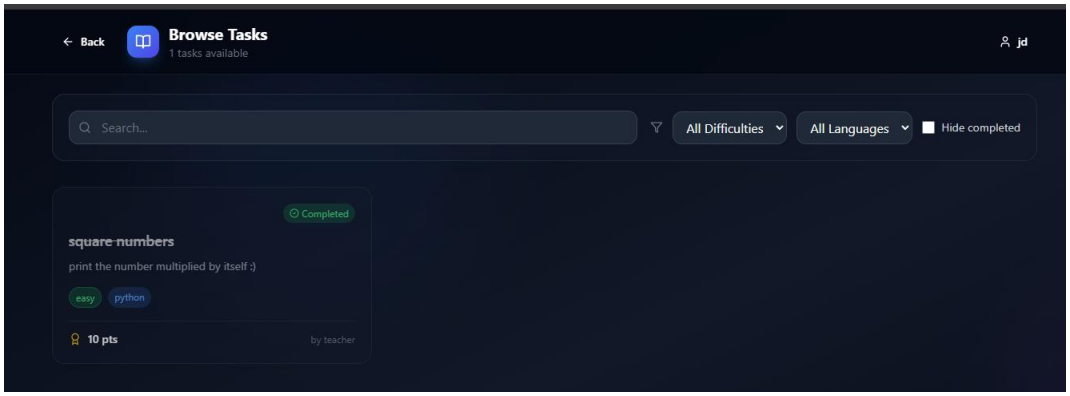


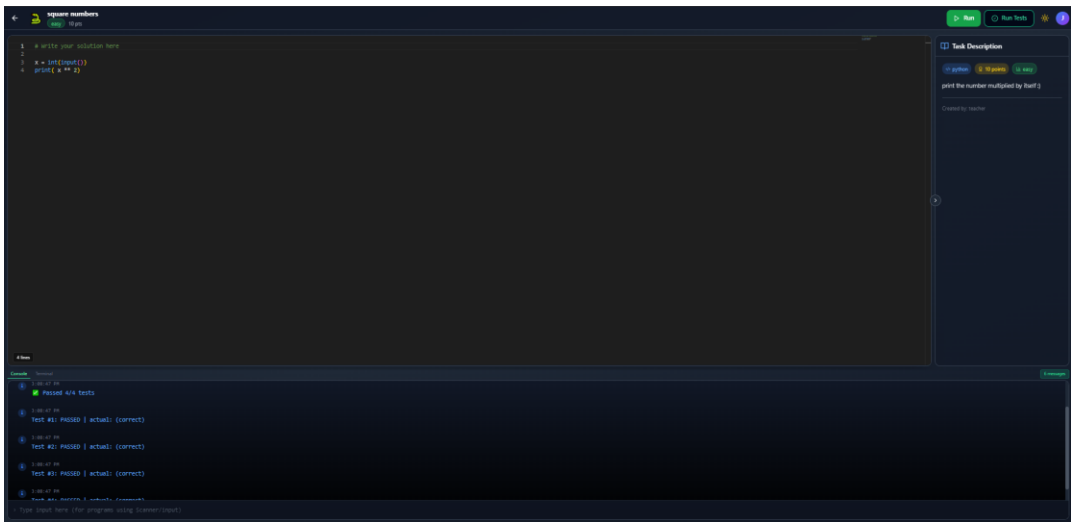
Task Management

- **Implementation:** Coding challenge system with automated grading
- **Features:** Difficulty-based tasks, instant feedback, and solution validation
- **UI/UX:** Task cards with points, difficulty indicators, and completion status
- **Integration:** Real-time test case execution and automated scoring

9: Tasks screen





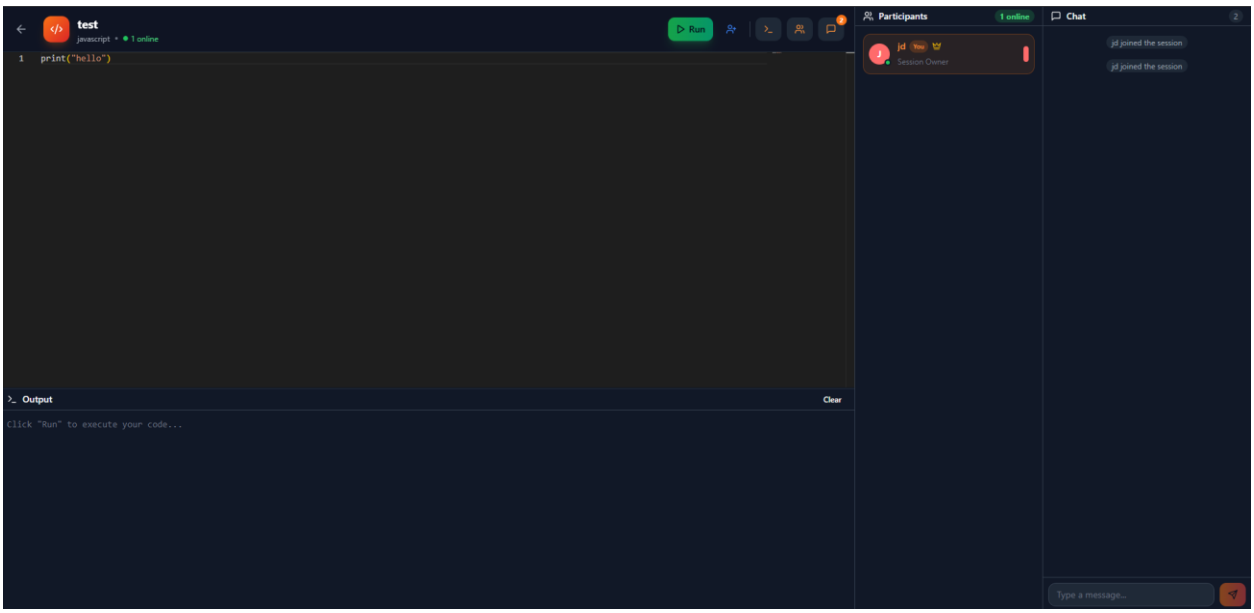


Real-Time Collaboration

Live Coding Sessions

- **Implementation:** WebRTC-based peer-to-peer collaboration
- **Features:** Real-time code synchronization, video/audio chat, and cursor tracking
- **UI/UX:** Split-screen interface with participant indicators and chat panel
- **Performance:** Optimized for low-latency collaboration with conflict resolution

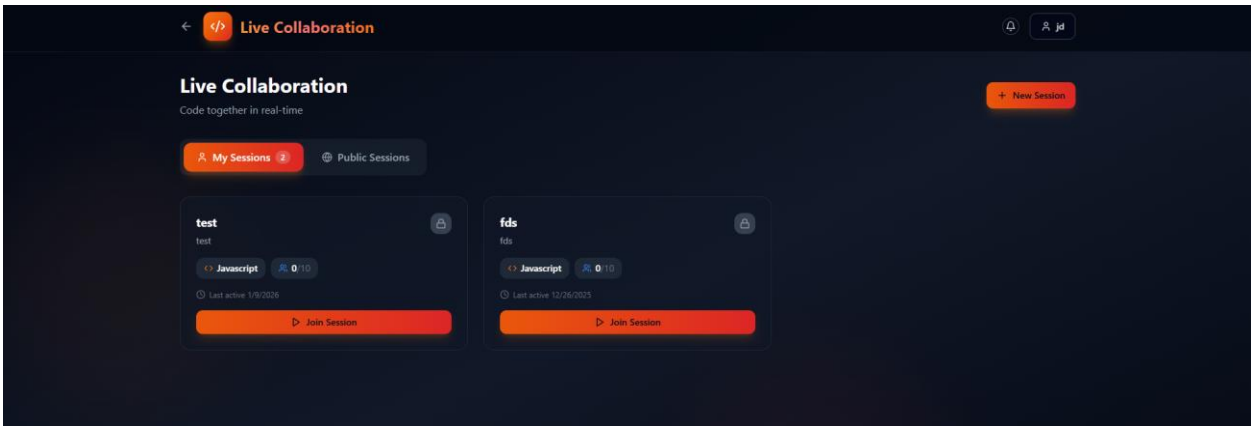
10: Collaboration screen



Session Management

- **Implementation: Session creation, invitation system, and participant management**
- **Features: Session scheduling, invite links, and participant permissions**
- **UI/UX: Clean session dashboard with active sessions and history**
- **Security: Secure session establishment with authentication and authorization**

11: Collaboration screen

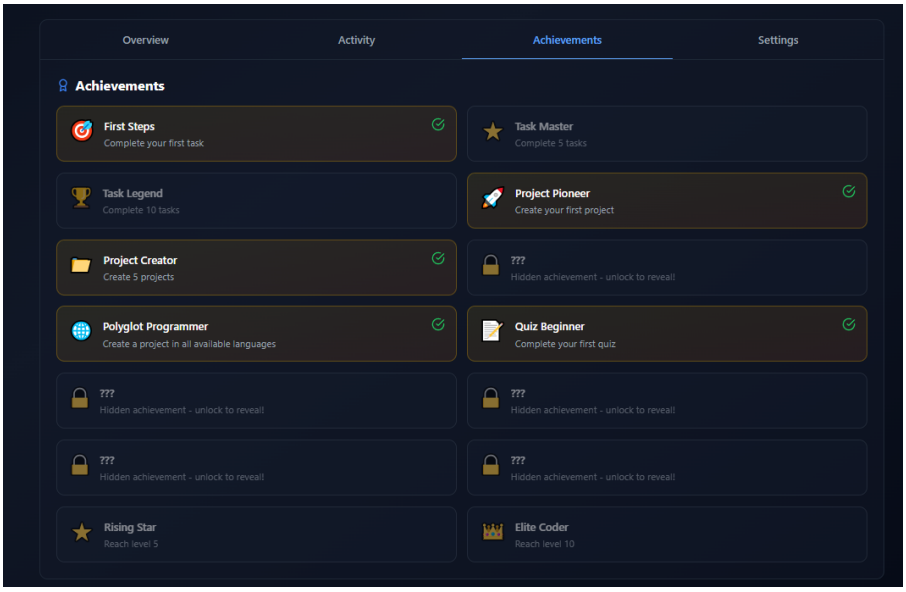


Gamification System

Achievement System

- **Implementation: Comprehensive achievement tracking with badges and rewards**
- **Features: Milestone achievements, skill-based badges, and special rewards**
- **UI/UX: Achievement showcase with progress tracking and unlock animations**
- **Data: Real-time achievement calculation and notification system**

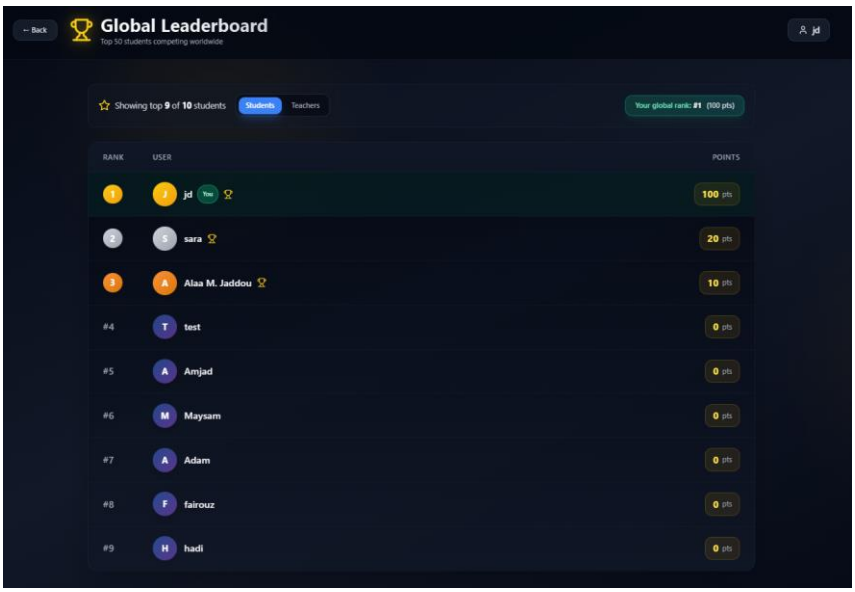
12: Achievements Tap

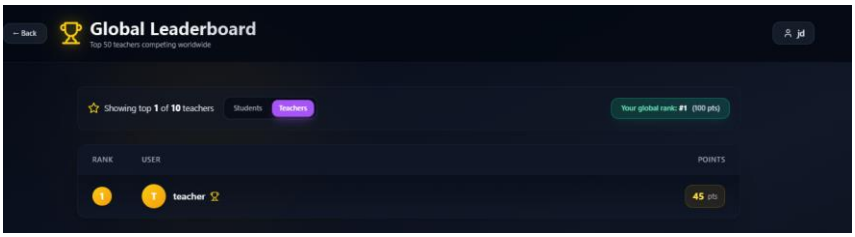


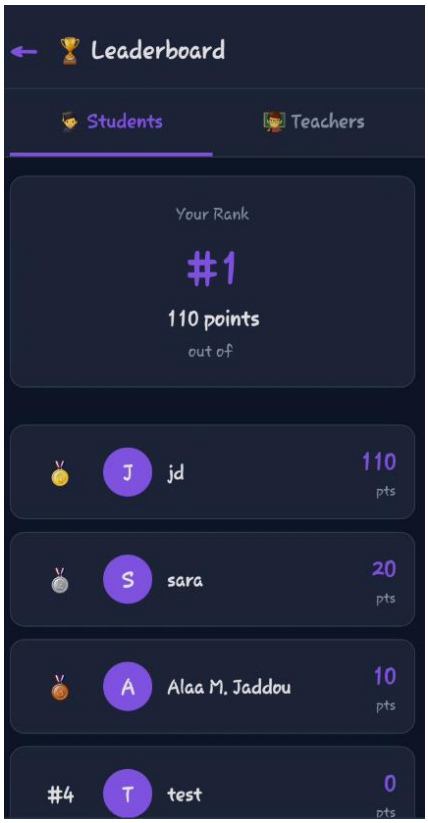
Leaderboard

- **Implementation: Global and friends-based leaderboards with multiple categories**
- **Features: XP ranking, level progression, and skill-specific leaderboards**
- **UI/UX: Interactive leaderboard with filters and search functionality**
- **Updates: Real-time leaderboard updates with ranking changes**

13: Leaderboard screen



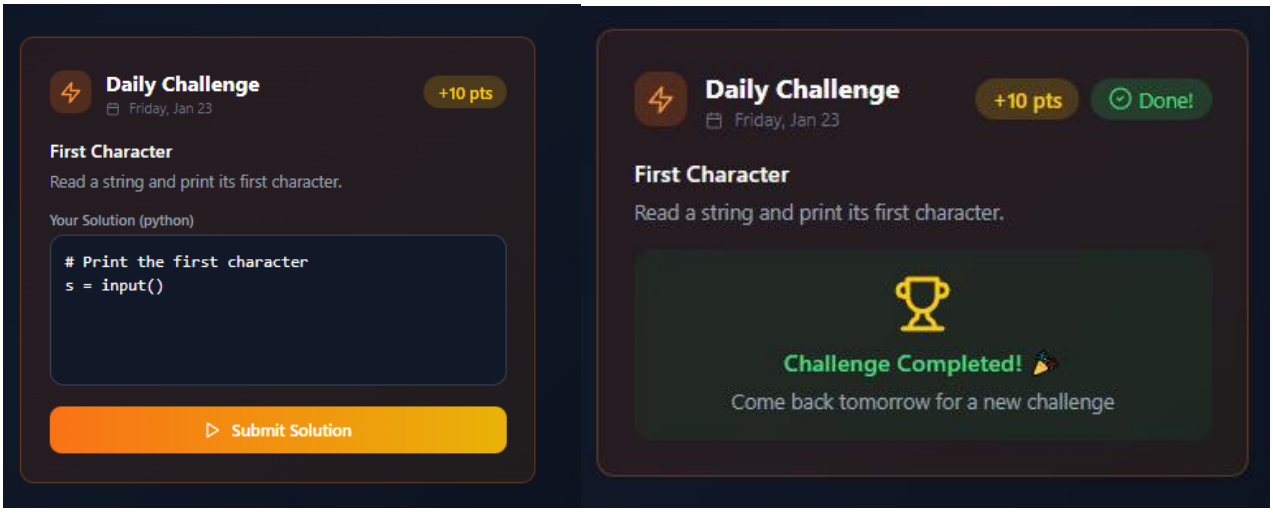




Daily Challenges

- **Implementation: Daily coding challenges with varying difficulty**
- **Features: 24-hour challenge cycle, bonus XP rewards, and streak tracking**
- **UI/UX: Prominent challenge display with countdown timer and participation stats**
- **Engagement: Push notifications and social sharing capabilities**

14: Daily Challenge

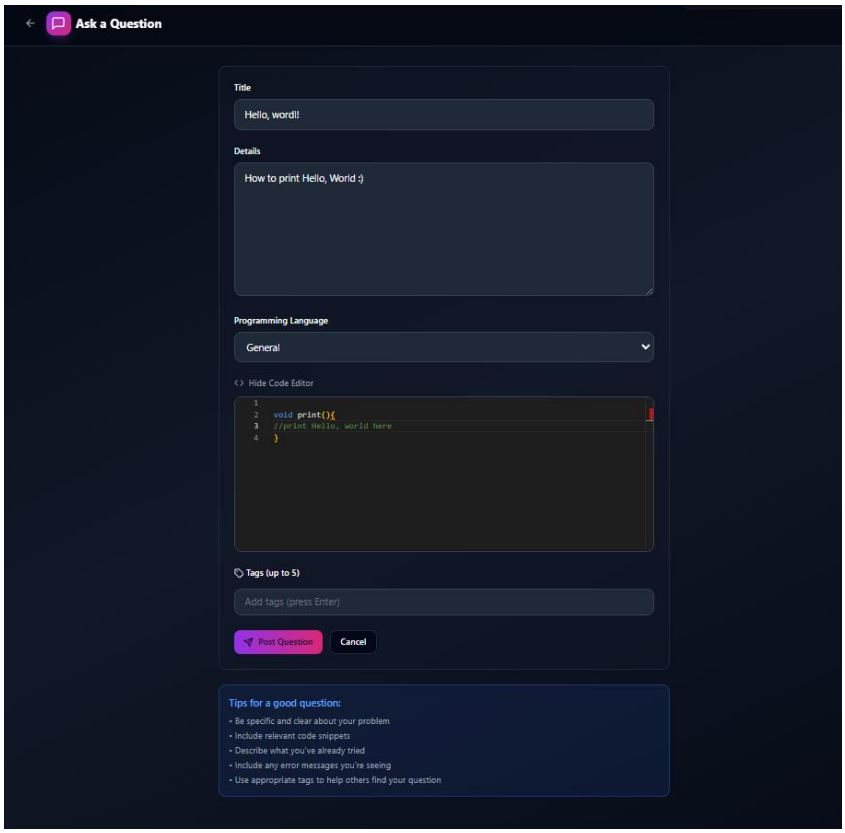


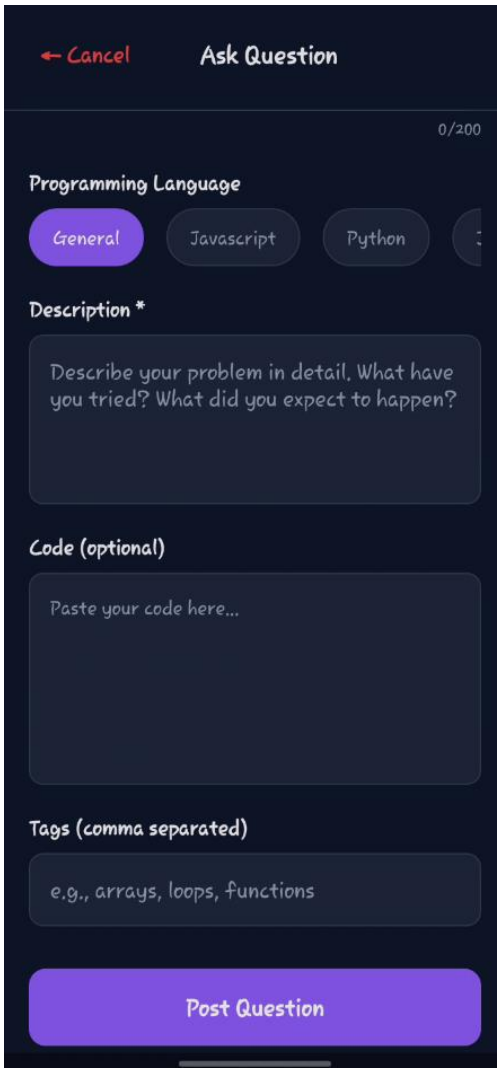
Community Features

Q&A Forum

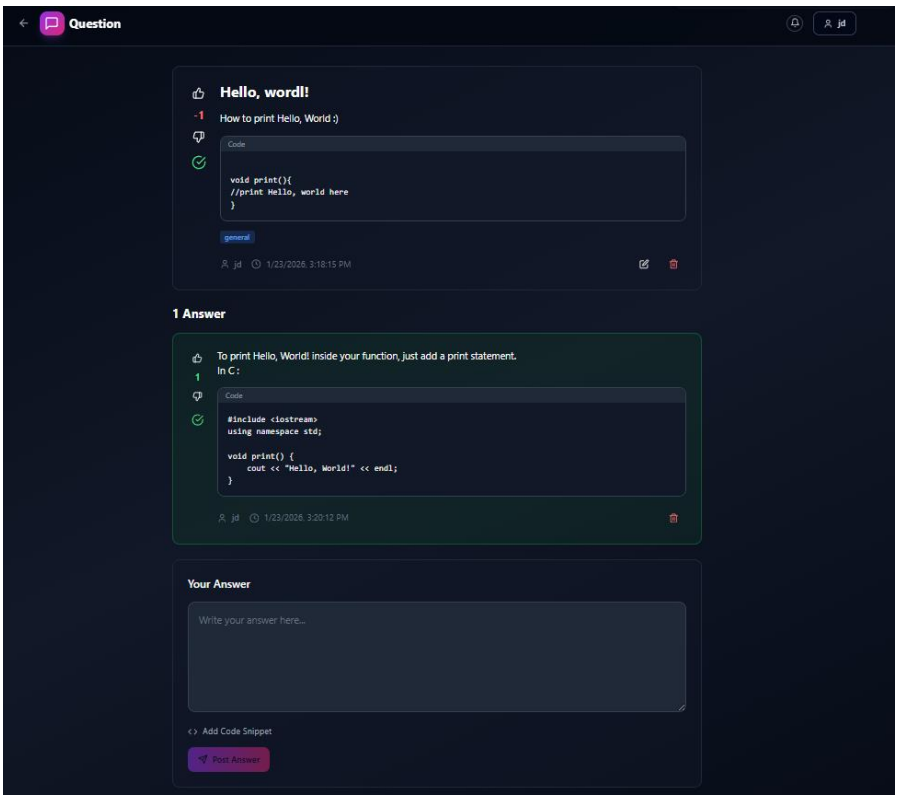
- **Implementation:** Stack Overflow-style forum with voting and reputation
- **Features:** Question tagging, answer acceptance, and user reputation system
- **UI/UX:** Clean forum interface with rich text editor and code highlighting
- **Moderation:** Teacher moderation tools and content management

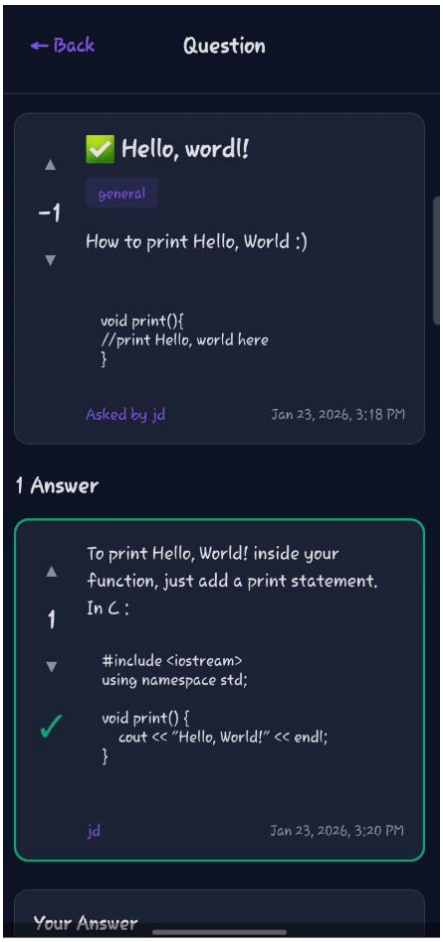
15: Ask Question screen

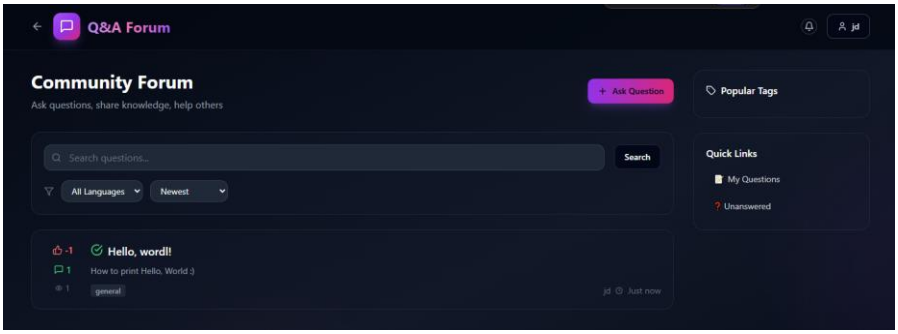




16: Q&A screen



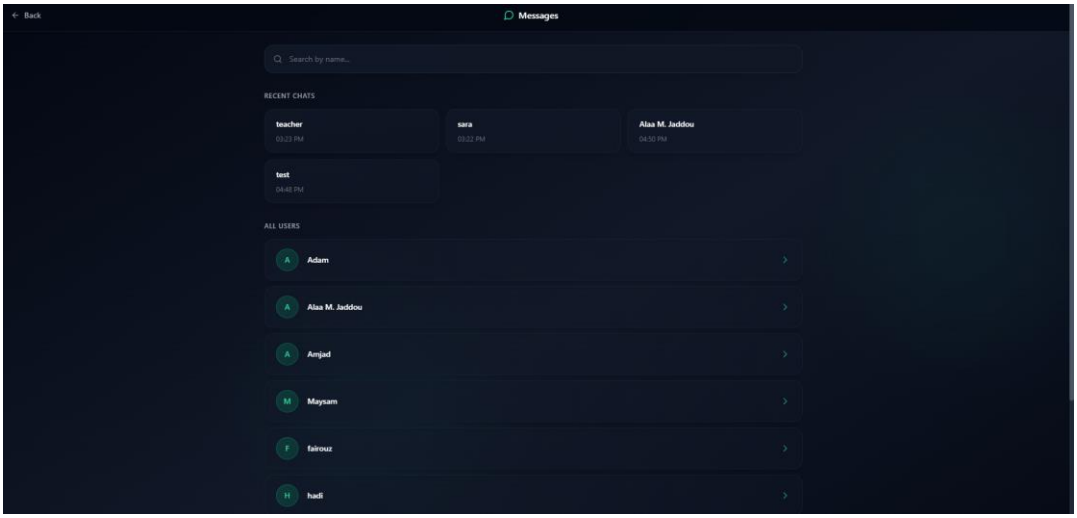


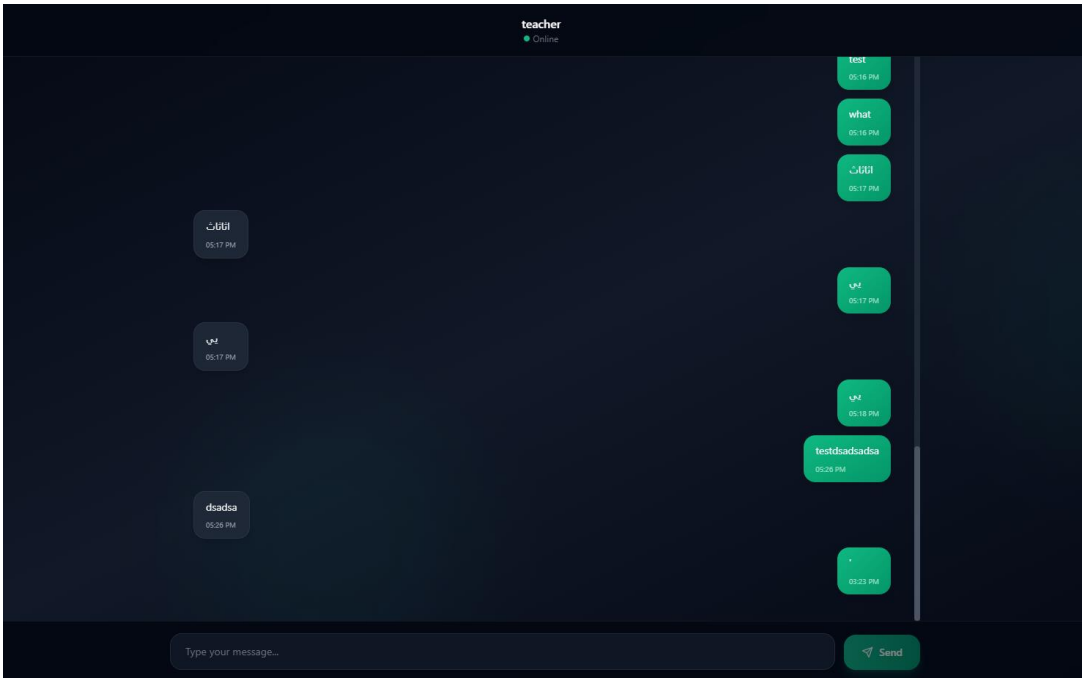


Direct Messaging

- **Implementation:** Real-time messaging system between users
- **Features:** Online status indicators, read receipts, and file sharing
- **UI/UX:** Modern chat interface with emoji support and search functionality

17: Chat screen





Teacher Features

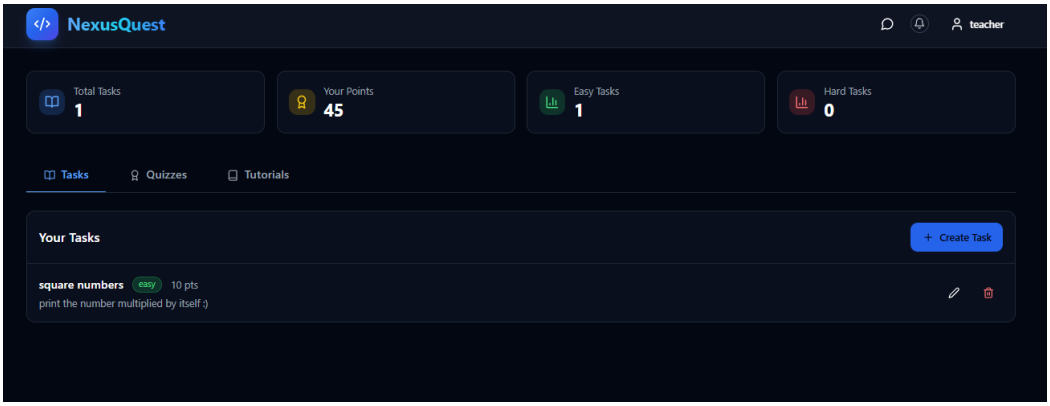
Content Management

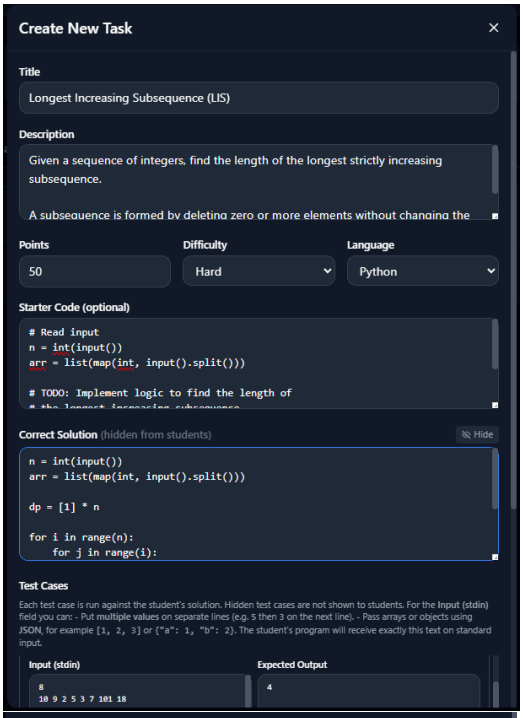
Task Creation

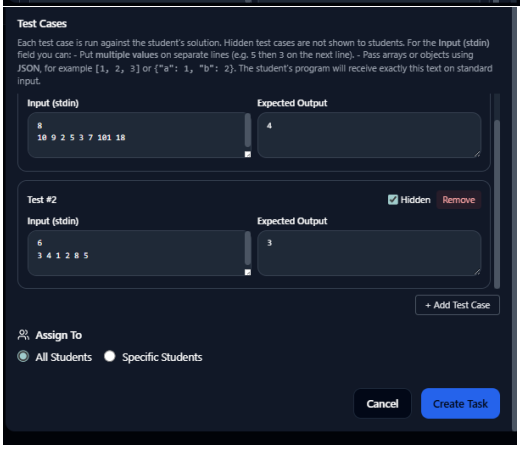
- **Implementation:** Comprehensive task builder with test case management
- **Features:** Multi-language support, automated testing, and difficulty assignment
- **UI/UX:** Visual task builder with code editor and test case interface

- **Validation:** Real-time validation and preview functionality

18: Tasks screen (teacher)



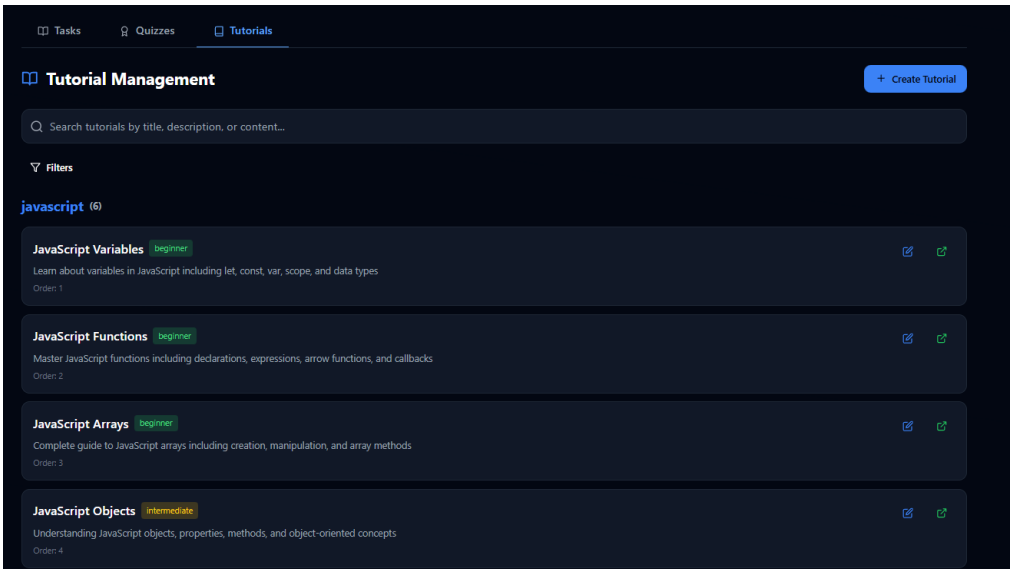


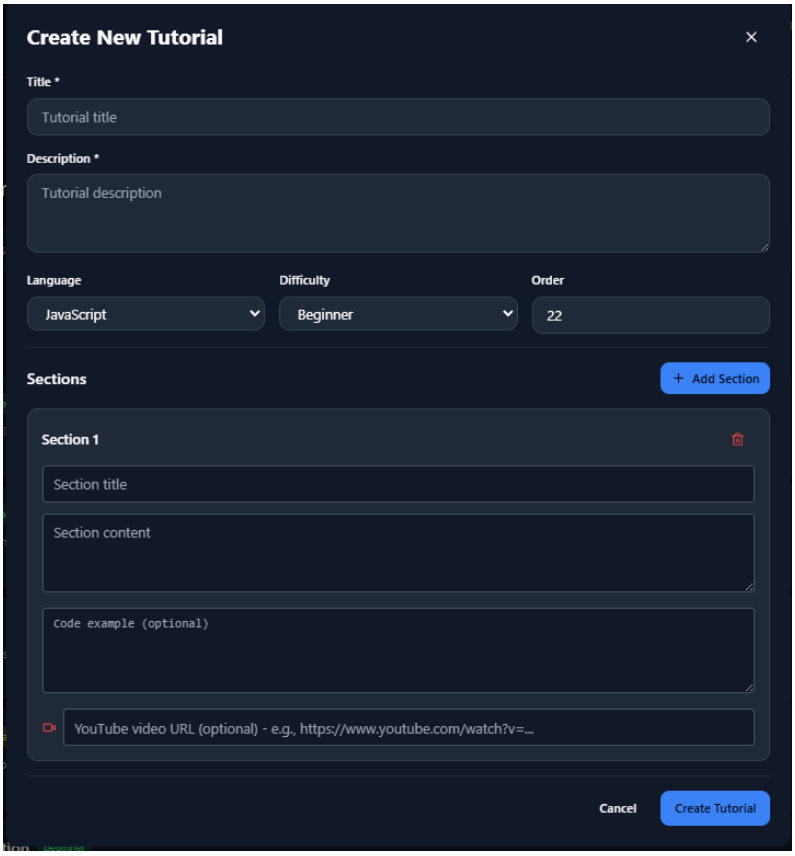


Tutorial Builder

- **Implementation: Interactive tutorial creation with rich content support**
- **Features: Markdown support, video integration, and interactive code examples**
- **UI/UX: Visual tutorial editor with drag-and-drop content blocks**
- **Publishing: Draft management and publication workflow**

19: Tutorials screen (teacher)

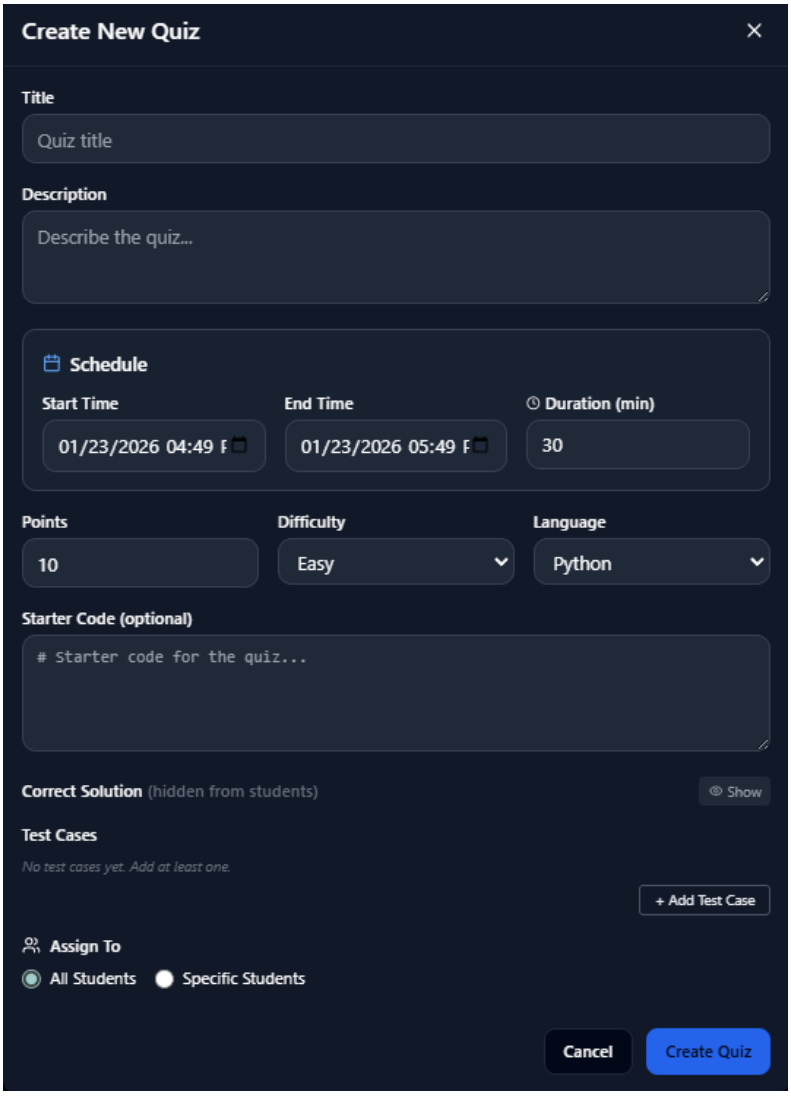




Quiz Designer

- **Implementation: Flexible quiz creation with multiple question types**
- **Features: Multiple choice, code-based questions, and timed assessments**
- **UI/UX: Intuitive quiz builder with question bank and randomization**
- **Analytics: Detailed quiz performance analytics and student insights**

20: Quiz screen (teacher)



Student Management

Assignment Management

- **Implementation: Assignment creation, distribution, and grading system**

- **Features: Due date management, submission tracking, and bulk grading**
- **UI/UX: Assignment dashboard with status indicators and quick actions**
- **Automation: Automated grading for code-based assignments**

Database

In the NexusQuest project, we use MongoDB as the primary database to store and manage all application data. MongoDB is a NoSQL document-oriented database designed for flexibility, scalability, and high performance, making it an excellent fit for modern educational platforms.

Why MongoDB?

- **Schema Flexibility**: MongoDB stores data in JSON-like documents (BSON), allowing dynamic and flexible schemas. This is ideal for an educational platform where different entities (users, tasks, tutorials, quizzes) may have diverse structures and evolve over time.
- **High Scalability:** MongoDB supports horizontal scaling and can handle large volumes of data and concurrent users efficiently. This is crucial for an educational platform that needs to manage thousands of students, teachers, and real-time collaboration sessions.
- **Real-Time Performance**: With features like indexing and fast queries, MongoDB ensures responsive performance, crucial for real-time features like collaborative coding, live chat, and instant feedback.
- **Native Support for Relationships**: While being a NoSQL database, MongoDB supports references and embedded documents, which allows modeling both relational and hierarchical data efficiently.

Database Schema Design

User Collection:

- Comprehensive user profiles with role-based fields (student, teacher)
- Extended gamification information including XP, level, achievements, and statistics
- Learning progress tracking and skill assessment data
- Social features including friends, followers, and activity history

Task Collection:

- Complete coding challenge management with multiple language support
- Automated test case system with hidden and visible test cases
- Difficulty grading and point allocation system
- Student assignment and submission tracking

Tutorial Collection:

- Interactive learning content with step-by-step instructions
- Progress tracking and completion status for each student
- Rich media support including videos, images, and code examples
- Teacher analytics and engagement metrics

Quiz Collection:

- Flexible quiz system supporting multiple question types
- Timed assessments with automated grading
- Student performance tracking and detailed analytics
- Question bank management with categorization

Collaboration Collection:

- Real-time session data and participant information
- Code snapshots and change history for collaboration sessions
- WebRTC connection details and session metadata
- Chat logs and communication history

Achievement Collection:

- Comprehensive gamification system with badges and rewards
- Achievement criteria and automatic unlocking system
- User achievement tracking and progress monitoring
- Leaderboard data and competitive metrics

Database Optimizations

- **Indexing Strategy:** Optimized indexes for frequently queried fields including user roles, task difficulty, tutorial categories, and collaboration sessions.
- **Connection Pooling:** Efficient MongoDB connection management for high concurrent usage during peak learning hours.
- **Query Optimization:** Aggregation pipelines for complex analytics and reporting on student performance and engagement metrics.
- **Data Validation:** Mongoose schema validation ensuring data integrity and consistency across all educational content.
- **Security Implementation:** Role-based access control, data encryption, and secure authentication mechanisms for protecting student and teacher data.

Discussion

The development of NexusQuest aimed to provide a comprehensive interactive coding education platform that supports real-time collaboration, gamified learning, and effective teaching tools. This section discusses how the system met its objectives, the effectiveness of the implemented features, challenges faced during development, and the rationale behind key design and technology choices.

Achievement of Objectives:

Through the integration of features such as real-time collaborative coding, gamified learning systems, AI-powered assistance, and comprehensive teacher tools, NexusQuest successfully delivers a revolutionary coding education experience. The system enables students to learn programming through interactive tutorials, collaborate with peers in real-time, and engage with gamified elements that enhance motivation. Teachers benefit from powerful content creation tools and comprehensive student management systems.

Feature Effectiveness:

One of the most significant features is the real-time collaboration system, which enables students to engage in pair programming with video/audio communication and live code synchronization. The decision to use WebRTC and Socket.io enabled smooth, low-latency collaboration experiences that closely mimic in-person pair programming sessions.

Another major feature is the comprehensive gamification system, which includes XP points, achievements, leaderboards, and daily challenges. This system significantly increases student engagement and motivation through game-like elements that make learning programming more enjoyable and rewarding.

The AI-powered code assistant provides intelligent help and suggestions, offering students personalized guidance when they encounter difficulties. This feature reduces frustration and helps maintain learning momentum by providing immediate assistance.

Project Impact:

NexusQuest stands out by offering a holistic solution tailored specifically for coding education. Unlike traditional learning platforms that focus on static content delivery, NexusQuest brings together interactive learning, real-time collaboration, gamification, and AI assistance in one unified, engaging environment.

The multi-platform approach (React web app, React Native mobile app) ensures accessibility across all devices and use cases, while the comprehensive backend API supports all frontend implementations with consistent data and functionality.

Conclusions and Recommendations

In this section, we present a summary of the project outcomes along with suggestions for future enhancements.

Summary

In summary, the NexusQuest platform successfully delivers an interactive coding education environment tailored for modern learning needs. By combining real-time collaboration, gamified learning, AI-powered assistance, and comprehensive teacher tools, the system addresses multiple challenges in coding education within a unified solution. The thoughtful use of modern technologies such as MongoDB, React, React Native, Node.js, WebRTC, and Socket.io ensured scalability, responsiveness, and real-time interactivity across web and mobile platforms.

The implementation of JWT authentication, role-based access control, real-time communication, and comprehensive error handling further enhanced user trust and system reliability. Through overcoming technical and educational challenges, NexusQuest demonstrates its value as a scalable and future-ready coding education platform.

Future Works

To further improve the platform and expand its capabilities, the following future enhancements are proposed:

- **Advanced AI Integration**: Implement more sophisticated AI models for personalized learning paths, adaptive difficulty adjustment, and intelligent code review with detailed explanations.
- **Virtual Classroom Features**: Expand collaboration capabilities to include virtual classrooms with screen sharing, whiteboard functionality, and teacher-led coding sessions.
- **Advanced Analytics:** Implement comprehensive learning analytics with predictive modeling for student success, engagement patterns, and learning optimization recommendations.
- **Integration with Learning Management Systems**: Develop LMS integration capabilities for seamless adoption by educational institutions with existing infrastructure.
- **Certification System**: Implement a comprehensive certification program with industry-recognized credentials and skill assessment validation.
- **Multi-language Support**: Complete internationalization system with dynamic language switching and culturally adaptive content for global markets.
- **Advanced Mobile Features**: Enhance mobile capabilities with offline-first architecture, push notifications, and native performance optimizations.
- **Enterprise Features**: Develop enterprise-grade features including single sign-on (SSO), advanced security controls, and administrative tools for institutional deployment.

References

1. Node.js: Node.js Foundation. (n.d.). Node.js Documentation. Retrieved from: https://nodejs.org/docs/latest/api/
2. Express.js: Express.js. (n.d.). Express – Node.js web application framework. Retrieved from: https://expressjs.com

3. MongoDB: MongoDB Inc. (n.d.). MongoDB Manual – Official Documentation. Retrieved from: https://www.mongodb.com/docs/
4. ReactJS: Meta Platforms, Inc. (2024). React – A JavaScript library for building user interfaces. Retrieved from https://reactjs.org
5. React Native: Meta Platforms, Inc. (2024). React Native – Build native apps using React. Retrieved from https://reactnative.dev
6. Socket.io: Socket.io. (n.d.). Socket.io Documentation. Retrieved from: https://socket.io/docs/
7. WebRTC: Google. (n.d.). WebRTC Documentation. Retrieved from: https://webrtc.org/
8. TypeScript: Microsoft. (2024). TypeScript Documentation. Retrieved from https://www.typescriptlang.org/
9. Monaco Editor: Microsoft. (n.d.). Monaco Editor Documentation. Retrieved from: https://microsoft.github.io/monaco-editor/
10. Tailwind CSS: Tailwind Labs. (n.d.). Tailwind CSS Documentation. Retrieved from: https://tailwindcss.com/
11. OpenAI: OpenAI. (n.d.). OpenAI API Documentation. Retrieved from: https://platform.openai.com/docs/
12. JWT: Auth0. (n.d.). JSON Web Tokens (JWT) Documentation. Retrieved from: https://jwt.io/
13. bcrypt: bcrypt. (n.d.). bcrypt Documentation. Retrieved from: https://www.npmjs.com/package/bcrypt
14. Docker: Docker Inc. (n.d.). Docker Documentation. Retrieved from: https://docs.docker.com/
15. Mongoose: MongoDB Inc. (n.d.). Mongoose Documentation. Retrieved from: https://mongoosejs.com/docs/