ExpressionLearner
(AI-Assisted Chinese Language Expression Practice Platform for Lower-Grade Students)

Team 08(B)

2253715 Fubin Chen

2252709 Xuanhe Yang

2251653 Tong Li

2251531 Peiyu Duan

Abstract

The project represents a groundbreaking approach to early language education, combining cutting-edge AI technology with pedagogical expertise to transform how lower-grade elementary students learn and practice Chinese language skills. Moving beyond the limitations of traditional teaching methods, this interactive platform creates a dynamic learning environment where students engage in personalized, AI-guided exercises that develop both their expressive abilities and logical thinking skills. The system intelligently adapts to each learner's pace and style, offering immediate feedback and encouragement while tracking progress through natural language processing technology. By integrating storytelling, creative writing, and practical communication scenarios, the platform nurtures students' confidence in self-expression while building essential language fundamentals. This comprehensive approach not only enhances academic performance but also fosters a genuine interest in language learning, supported by detailed progress tracking for educators and parents. The result is a holistic educational experience that prepares students for effective communication in real-world contexts while maintaining their enthusiasm for learning through achievement recognition and personalized guidance.

Team Introduction

This project was collaboratively completed by the following team members:

Fubin Chen
Student ID: 2253715
Phone: +86 15159581396
Email: 1958440015@qq.com

Xuanhe Yang
Student ID: 2252709
Phone: +86 18937082731
Email: 3342675194@qq.com

Tong Li
Student ID: 2251653
Phone: +86 18545556320
Email: 757847466@qq.com

Peiyu Duan
Student ID: 2251531
Phone: +86 19935268512
Email: 2251531@tongji.edu.cn

Table of Contents

Chapter1 Complete List of System Functionalities Implemented

1 JWT Security Authentication

The JWT functionality in the project is implemented using the Spring Security framework. Through the custom `UserDetailsServiceImpl` class for loading user information, working in conjunction with the `UserDetailsImpl` class, we implemented the core logic of user authentication. In the `JwtUtil` class, we used JWT technology to generate tokens containing user information, setting a 14-day validity period. The `SecurityConfig` class configures security policies, including disabling CSRF protection, setting stateless session management, and adding the `JwtAuthenticationTokenFilter` filter. This filter parses the JWT token during each request, verifies user identity, and sets authentication information in the Spring Security context. This series of implementations ensures system security while providing a flexible stateless authentication mechanism, allowing users to operate conveniently after login without frequent identity verification.

2 User Registration

The user registration functionality allows new users to create accounts to access the system. In `UserServiceImpl`, registration logic is implemented through the `register` method. This method accepts username, password, password confirmation, email, and avatar as inputs. The registration process first validates input data completeness and validity, including checking if username and password are non-empty, comply with length requirements, and if the two password entries match. Additionally, it checks if the username already exists to prevent duplicate registrations. If all validations pass, the system encrypts the password using a password encoder and stores the new user information in the database. Upon successful registration, a success message is returned, guiding users to proceed with login. The `register` in `UserController` maps registration requests, handles client-submitted registration data, and calls registration logic through `userService`, ensuring users can smoothly complete the registration process and create accounts.

3 User Login

The user login functionality allows users to access the system by entering their username and password. In `UserServiceImpl`, login logic is implemented through the `getToken` method. This method receives username and password and performs authentication using `AuthenticationManager`. If authentication succeeds, a JWT token is generated as the credential for user identity verification. Error handling mechanisms ensure appropriate error messages are returned when username or password is invalid. The `getToken` in `UserController` maps login requests and returns results to the client. The entire process ensures users can securely and conveniently log into the system.

4 User Profile Center

The user profile center functionality provides user information overview and update capabilities. In `UserServiceImpl`, the `getInfo` method is used to retrieve and return current logged-in user's information, such as username and avatar. This method obtains user details through Spring Security context and then retrieves related information from the database. The `getInfo` in `UserController` maps requests for retrieving user information. Additionally, data transfer objects (DTOs) like `UserUpdateDTO` and `UserAccessibleDTO` are used to encapsulate and transfer user data, enabling the frontend interface to easily display and update user information, while also showing my tasks, displaying tasks the user

has completed. Overall, the user profile center functionality provides users with comprehensive information management and control capabilities.

5 AI Feedback and Prompting

The AI feedback and prompting tool plays a crucial role in our project, with its main function being to assist students in improving their language expression and story conceptualization abilities while completing specific tasks. When students open a task facing three images, they need to coherently narrate the images' content into a story using voice recognition. During this process, the AI tool evaluates students' stories based on specific modes—completeness, logic, and emotion—providing immediate feedback.

In completeness mode, AI analyzes whether the student's story structure and content are complete and cover all image details. If key information is missing or plot jumps are detected, AI poses targeted questions like "What did the character do next?" or "How does the story end?" These questions aim to guide students in supplementing missing plots to make the story more complete.

In logic mode, AI focuses on evaluating story plot coherence and rationality. If contradictions or unreasonable elements exist in student narration, AI helps clarify thinking through prompting questions like "Why did the character suddenly make this decision?" or "What's the connection between these two events?" Such questions help students build more logical storylines.

Emotion mode focuses on emotional expression and atmosphere in students' stories. AI analyzes emotional depth based on narration and provides emotional vocabulary prompts when necessary, such as "How do you think the character feels at this moment?" or "Can you use some words to describe the character's mood?" Such prompts help students better convey emotions in stories, enhancing their impact.

Besides questions, the AI feedback and prompting tool also provides example word prompts, including adjectives, adverbs, or verbs, to enrich students' language expression. For instance, when describing a scene, AI might suggest phrases like "azure sky" or "joyfully jumping" to help students more vividly depict scenes.

Overall, through intelligent analysis and interaction, the AI feedback and prompting tool not only improves students' storytelling abilities but also cultivates their creative thinking and emotional expression capabilities, thus enhancing students' comprehensive qualities through educational entertainment.

6 WebSocket-Based Real-time Speech Recognition

Our project integrates an efficient real-time speech recognition functionality implemented through WebSocket technology, allowing students to use voice input directly when describing image content, with the system converting speech to text in real-time to support story narration tasks.

The system uses WebSocket protocol to establish persistent connections, ensuring voice data can be continuously transmitted from client to server in real-time. This connection method reduces traditional HTTP request latency, improving data transmission real-time performance.

When students click the "Start Recording" button on the task interface, voice data captured by the microphone is packaged into frames and sent to the server via WebSocket connection. Upon receiving voice frames, the server processes them through the integrated speech recognition engine, converting student speech into corresponding text and sending it back to the client.

The client displays the recognition results immediately on the interface upon receipt, allowing students to see their speech converted to text and make real-time adjustments to their narration content. Speech recognition functionality combines with AI feedback and prompting tools, providing suggestive questions and recommendations based on students' narration content to help them improve their stories.

Through this functionality, students can more naturally participate in story narration tasks, enhancing task interactivity and engagement.

7 Task Functionality

Task functionality is the core part of our project, organizing the entire learning process and providing students with an interactive, challenging learning platform. Tasks aim to exercise students' language expression, logical thinking, and creativity through a series of carefully designed image description tasks. By completing these tasks, students can improve their narrative techniques while receiving immediate feedback and guidance from AI.

The task service (`TaskService`) is responsible for storing, retrieving, and updating task information, ensuring students can access the latest tasks.

Tasks include three modes:

- **Completeness Assessment Mode**: Evaluates the completeness of students' story narration, ensuring proper coverage of story beginning, development, and ending without missing important plots.
- **Logic Assessment Mode**: Focuses on story structure and logic, ensuring content coherence between images, clear logic, and reasonable plot without contradictions.
- **Emotion Assessment Mode**: Emphasizes students' ability to express story emotions, requiring appropriate use of emotional vocabulary and tone to make stories more impactful.

The system includes:

1. **Mode Selection Interface**: Students can choose a mode before starting tasks, with a clean interface design facilitating mode selection based on learning goals and interests.
2. **Task Execution**: System displays three images, students narrate stories through speech recognition, with real-time speech-to-text conversion.
3. **Mode-Specific AI Feedback**: After task completion, system scores based on selected mode and provides targeted feedback.
4. **Speech Recognition Activation**: Task controller initializes speech recognition service upon "Start Recording" click, establishing WebSocket connection for voice input.
5. **Real-time Speech-to-Text**: Speech recognition service converts student voice to text in real-time during image description.
6. **AI Feedback and Prompts**: AI provides suggestive questions and example words based on narration content and task mode.
7. **Emotion Expression Analysis**: Evaluates students' emotional expression ability by analyzing emotional vocabulary and tone.
8. **Task Completion and Scoring**: System scores based on completeness, logic, and emotion dimensions upon story completion.
9. **History Record and Progress Tracking**: System records all completed tasks, including speech-to-text results, scores, and AI feedback.

As the core of our project, task functionality provides a comprehensive, interactive language learning environment by integrating speech recognition, AI feedback, and scoring systems. It not only helps students improve language expression but also cultivates creative thinking and emotional perception abilities. Through continuous practice and feedback, students can achieve self-improvement and learning goals in a relaxed and enjoyable atmosphere.

Chapter2 User Manual

1 Getting Started with the System

2.1.1 System Overview

This educational platform is designed to help users learn through interactive tasks and AI-assisted guidance. The system consists of several key components including user management, task center, and personal dashboard.

2 User Registration and Login System

2.2.1 Step-by-Step Registration Process

Follow these steps to create your account:

1. Click the ”注册” (Register) button on the login page
2. Complete the following required fields:
    - Username (用户名): Choose a unique identifier
    - Password (密码): Create a strong password using letters and numbers
    - Confirm Password (确认密码): Re-enter your password
    - Email (邮箱): Provide a valid email address for account recovery
    - Avatar: Optional profile picture (default provided)
3. Review your information
4. Submit the registration form
5. Wait for system validation and account creation confirmation

Figure 2.1: User Registration Interface

2.2.2 Login Process

To access your account:

1. Navigate to the login page
2. Enter your username (用户名)
3. Enter your password (密码)
4. Click the ”登录” (Login) button
5. For security, your session will automatically timeout after period of inactivity

2.2.3 Security Features

The system implements several security measures:

- Secure password encryption
- Protection against unauthorized access
- Session management and timeout
- Password recovery system
- Account activity monitoring

Figure 2.2: User Login Interface

3 Using the Task Center

2.3.1 Accessing Learning Tasks

The Task Center provides various educational activities:

1. Navigate to ”任务中心” (Task Center)
2. Browse available tasks in the grid layout
3. Tasks are categorized by:
    - Difficulty level

- Subject matter
- Completion status
- Type of activity

4. Select a task to begin learning

Figure 2.3: Task Center Interface

2.3.2 Interactive Learning Features

Each task includes:

- Clear instructions in both Chinese and Pinyin
- Visual aids and examples
- Progressive difficulty levels
- AI-assisted guidance
- Immediate feedback system

Figure 2.4: Task Presentation Interface

Figure 2.5: AI Guidance

4 Managing Your Personal Center

2.4.1 Personal Dashboard Features

Access your personal information and progress:

1. Click ”个人中心” (Personal Center)
2. View your unique user ID
3. Track completion rates
4. Monitor learning progress
5. Access completed tasks history

Figure 2.6: Personal Center Interface

5 Best Practices for Learning

2.5.1 Recommended Study Approach

Follow these guidelines for optimal learning:

- Complete tasks sequentially
- Review feedback after each task
- Practice regularly
- Track your progress
- Use the AI guidance system

6 Technical Support

2.6.1 Getting Help

If you encounter issues:

1. Check the FAQ section
2. Contact technical support via email
3. Use the help documentation
4. Report any system errors

2.6.2 System Requirements

Ensure your system meets these specifications:

- Modern web browser (Chrome, Firefox, Safari)
- Stable internet connection
- Minimum screen resolution: 1024x768
- JavaScript enabled
- Cookies enabled

Chapter3 System Deployment and Configurations

1 System Deployment

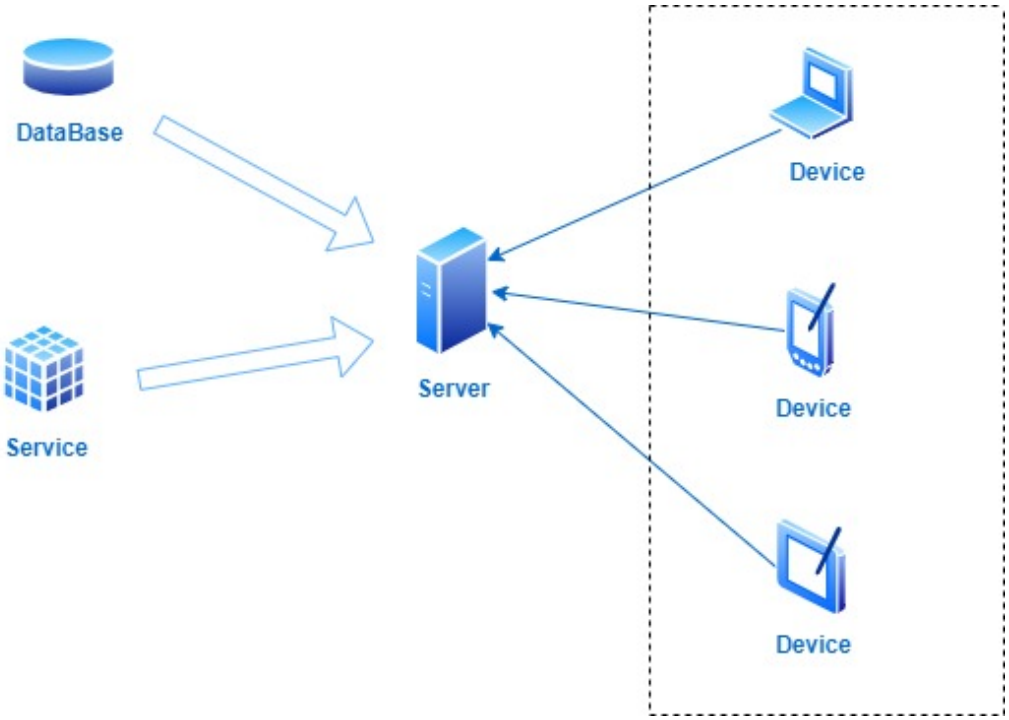

Figure 3.1: Deployment Diagram

-

3.1.1 Database Deployment Instructions

The database is deployed on a remote MySQL server with the following URL configuration:

```
spring.datasource.url=jdbc:mysql://49.235.143.83:3306/ChineseClass?allowPublicKeyRetrieval
```

- **Database IP Address**: `49.235.143.83`
- **Database Port**: `3306`
- **Database Name**: `ChineseClass`
- **MySQL Connection Configuration**:
  - `allowPublicKeyRetrieval=true`: Enables public key retrieval from server
  - `useSSL=false`: Disables SSL connection

Database Deployment Process

1. **MySQL Connection**:
   Connect to the database server using database management tools:
   - Host: `49.235.143.83`

- Port: `3306`

2. **Database Initialization**:
   Create database `ChineseClass` using the following SQL statement:

```
CREATE DATABASE ChineseClass;
```

3. **Table Structure**:
   Implement table structures using SQL scripts.

4. **Network and Firewall Settings**:
   Ensure database port `3306` is open in the firewall and accessible from the application server IP.

3.1.2 Backend Code Deployment Instructions

The backend code is deployed on server `49.235.143.83` at port `3001`.

Backend Deployment Process

1. **Server Environment Preparation**:
   - Install JDK 22 on the server and verify with `java -version`
   - Install and configure MySQL client for database connectivity

2. **Backend Code Deployment**:
   - Upload compiled `.jar` or `.war` file to server via FTP
   - Navigate to upload directory and start backend application:

```
java -jar expressionlearner.jar
```

3. **Server Port Configuration**:
   In `application.properties`, ensure backend listens on port `3001`:

```
server.port=3001
```

4. **Firewall Configuration**:
   Ensure port `3001` is open on the server for client requests.

3.1.3 Deployment Architecture

Architecture Description

- **Client (Browser)**: Users access the application through web browsers
- **Frontend (Web)**: Frontend code deployed locally, communicates with backend server via HTTP
- **Backend Service**: Java application deployed on server `49.235.143.83` at port `3001`, providing API endpoints
- **Database**: MySQL database deployed on the same server `49.235.143.83` using port `3306`

Deployment Diagram

2 Cross-Origin Resource Sharing (CORS)

The CORS configuration is implemented with the following settings:

- `Access-Control-Allow-Origin` is set to the `Origin` from the request, enabling access from specified sources
- `Access-Control-Allow-Headers` and `Access-Control-Expose-Headers` are configured to permit and expose specific request and response header fields, supporting custom headers and cross-origin response reading
- `Access-Control-Allow-Methods` is set to `*`, permitting all HTTP methods
- `Access-Control-Max-Age` is configured to 3600 seconds, caching preflight request results to reduce frequent preflight requests
- `Access-Control-Allow-Credentials` is set to `true`, enabling cross-origin requests to carry credentials (such as Cookies)

3 SSL Configuration

The SSL setup utilizes an `SSLContext` to create a TLS context with a custom `X509TrustManager`. This configuration accepts any certificate without strict validation and does not restrict client certificate authorities.

4 WebSocket Configuration

The WebSocket configuration in the Spring Boot application includes:

- Enabling WebSocket functionality
- Registering WebSocket endpoint at `/ws`
- Configuring cross-origin access with `setAllowedOrigins("*")`
- Adding WebSocket handshake interceptor (`HttpSessionHandshakeInterceptor`)
- Implementing a custom message handler `AudioWebSocketHandler`
- Enabling Spring WebSocket support through `ServerEndpointExporter`

5 Zhipu AI Service Configuration

The Zhipu AI service client configuration includes:

- Loading configuration properties with `@ConfigurationProperties` annotation for `ai` prefixed settings (e.g., `apiKey`)
- Network configuration parameters:
  - Connection timeout: 30 seconds
  - Read timeout: 60 seconds

Chapter4 System Architecture and Component Design

1 System Architecture

The system adopts a classic **three-tier architecture** (frontend, backend, database) design, including the following parts:

- **Presentation Layer**: Responsible for user interaction, displaying task selection, task details, user login, personal center, and other pages.
- **Business Logic Layer**: Processes user requests, handles business logic and data interactions, returns response results. The backend implements interface logic related to tasks, students, and users.
- **Data Layer**: Stores system-related data, including user information, task data, student responses, and image data.

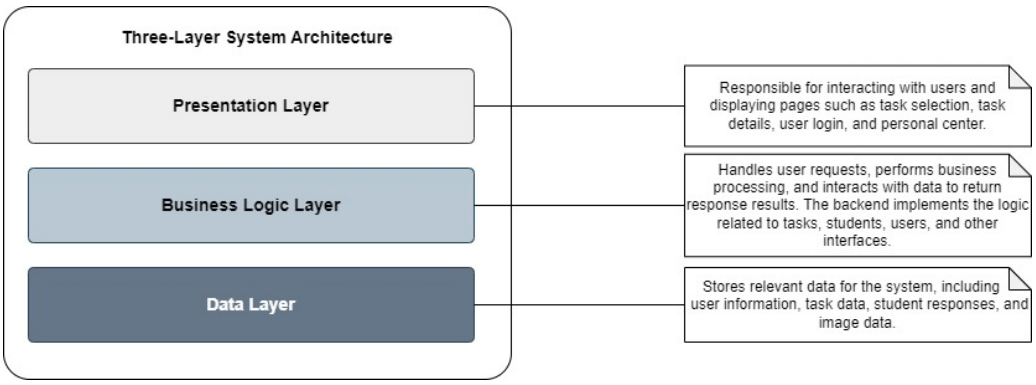

Figure 4.1: Three-tier Architecture Diagram

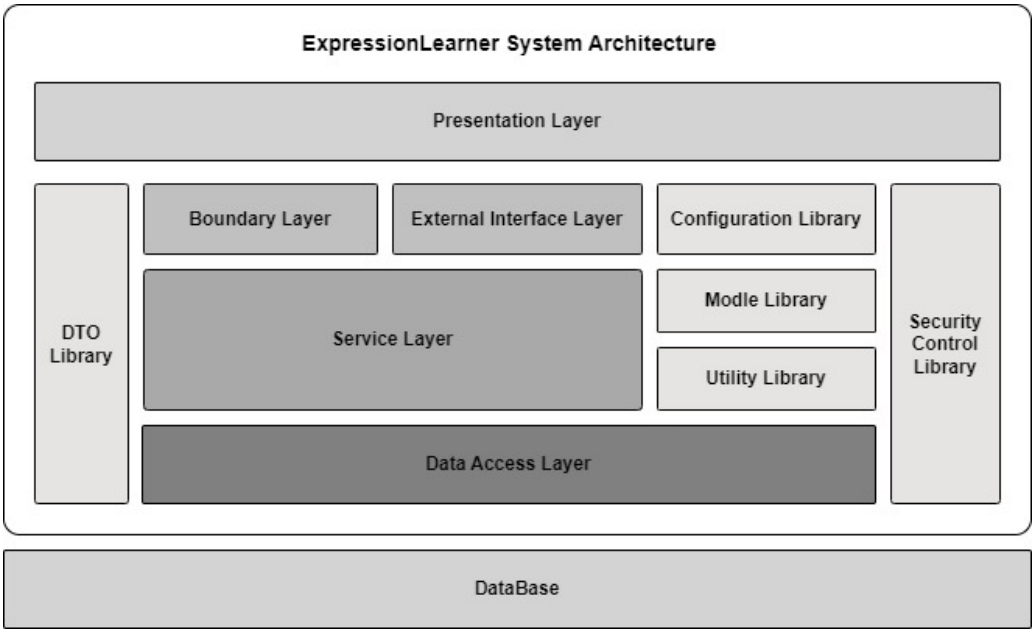

Figure 4.2: Overall Architecture

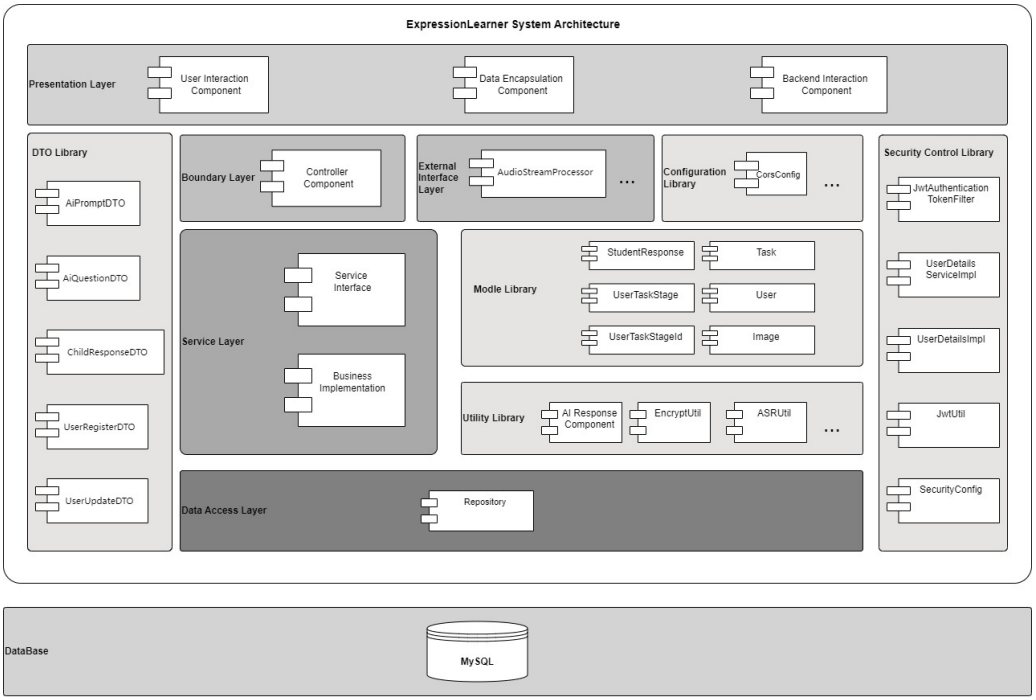

Figure 4.3: Detailed Architecture

2 Presentation Layer

4.2.1 Interface Components

- **Task Selection Page**: Displays available tasks for selection, users can browse and choose tasks to start. Clicking a task directs users to the task details page.

- **Task Details Page**: Shows detailed task information and images, students can perform task-related operations and submissions here.

- **Login Page**: Users enter username and password to log in, entering the website upon successful verification. Error messages prompt for re-entry upon login failure.

- **Personal Center Page**: Displays user basic information, including name, avatar, and completed tasks.

4.2.2 Data Encapsulation Components

- Formats and encapsulates user input or system return data to meet backend interaction requirements. Encapsulates data into VOs for transmission and processing.

4.2.3 Backend Interaction Components

- Sends formatted data to the backend via API, where business logic processing occurs based on requirements and returns results.

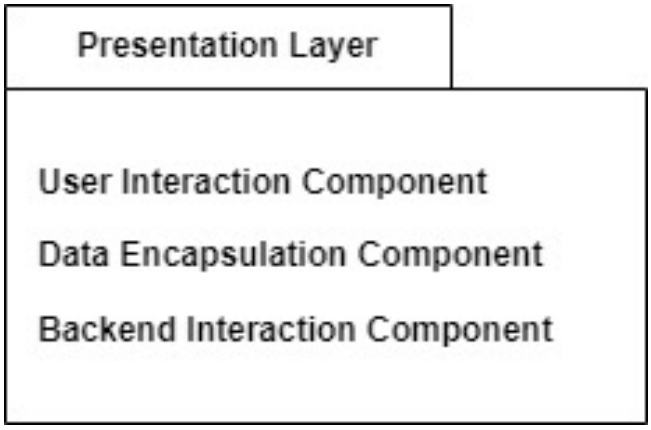

Figure 4.4: Presentation Layer Components

3 Business Logic Layer

4.3.1 Boundary Layer Controllers

- **AIController**: Handles AI question requests and task session information retrieval, calls `AiService`.
- **ImageController**: Processes image uploads and task image queries, calls `ImageService`.
- **StudentTaskController**: Manages student task operations like homework submission, feedback retrieval, and grading, calls `StudentTaskService`.
- **TaskController**: Handles task retrieval, detail viewing, and creation, calls `TaskService`.
- **UserController**: Manages user operations like login, registration, and user information retrieval, calls `UserService`.

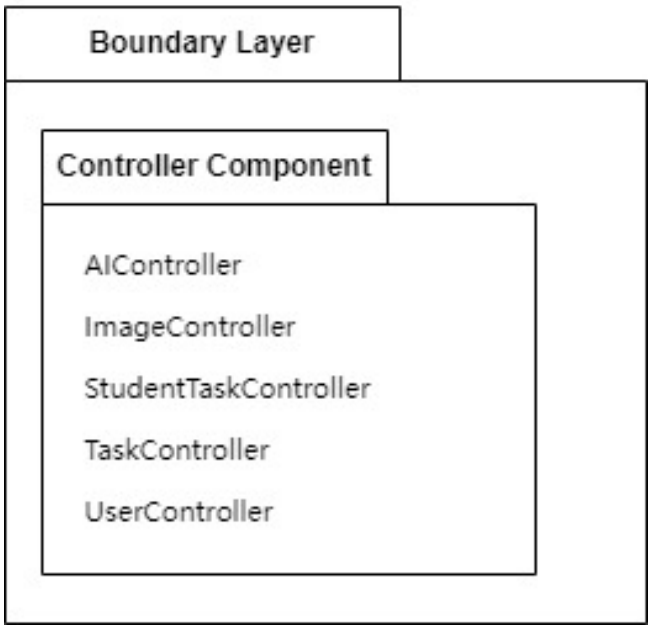

Figure 4.5: Boundary Layer Components

4.3.2 External Interface Connection Layer

- **AudioStreamProcessor**: Processes audio stream data, sends audio data blocks to WebSocket clients.
- **AudioWebSocketHandler**: Handles WebSocket connections, establishes connections with voice recognition service.

- **CustomWebSocketClient**: Custom WebSocket client for voice recognition service connection.
- **DraftWithOrigin**: Custom WebSocket handshake protocol supporting cross-origin access.

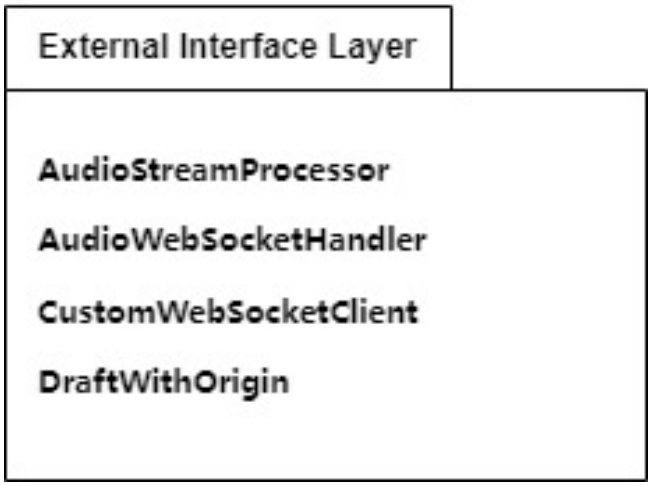

Figure 4.6: External Interface Components

4.3.3 Service Layer

Service Interfaces

- **AiService**: Interacts with AI services, processes generated AI questions and student feedback.
- **ImageService**: Handles image upload, query, and deletion operations.
- **StudentTaskService**: Processes student homework submission, grading, and feedback.
- **TaskService**: Handles task creation and queries.
- **UserService**: Manages user login, registration, and information queries.

Business Implementation

- **ImageServiceImpl**: Implements image-related business logic.
- **PureChatAiImpl**: Implements AI question generation and feedback logic.
- **StudentTaskServiceImpl**: Implements student homework submission and grading logic.
- **TaskServiceImpl**: Implements task creation and query logic.
- **UserServiceImpl**: Implements user management functionality.

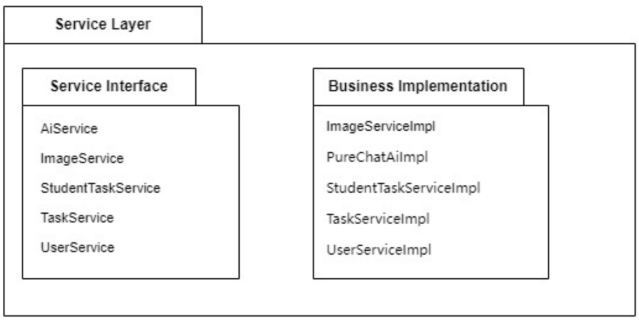

Figure 4.7: Service Layer Components

4.3.4 Data Access Layer

- **ImageRepository**: Stores and manages image data.
- **StudentResponseRepository**: Stores and manages student response information.
- **TaskRepository**: Stores and manages task information.
- **UserRepository**: Stores and manages user information.
- **UserTaskStageRepository**: Stores and manages user task stage information.

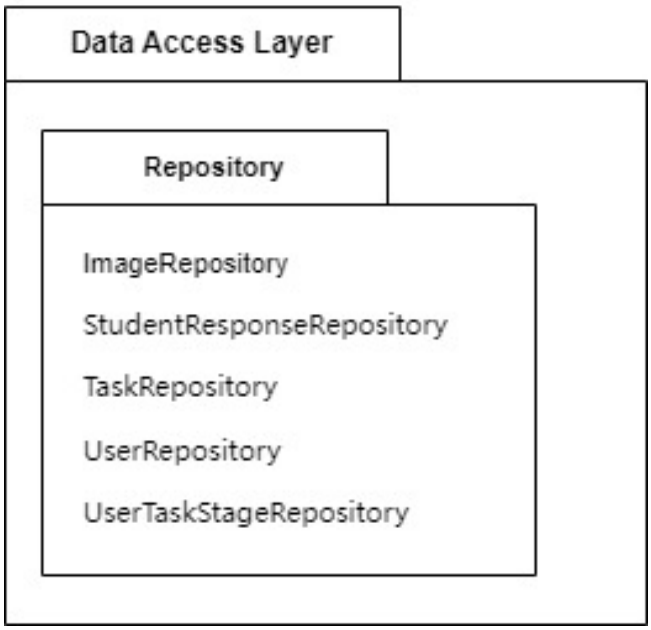

Figure 4.8: Data Access Layer Components

4.3.5 Configuration Library

- **CorsConfig**: Configures cross-origin resource sharing.
- **SslConfig**: Configures SSL context for WebSocket clients.
- **SwaggerConfig**: Configures Swagger for API documentation.
- **WebSocketConfig**: Configures WebSocket endpoints.
- **ZhipuAiConfig**: Configures Zhipu AI client connection.

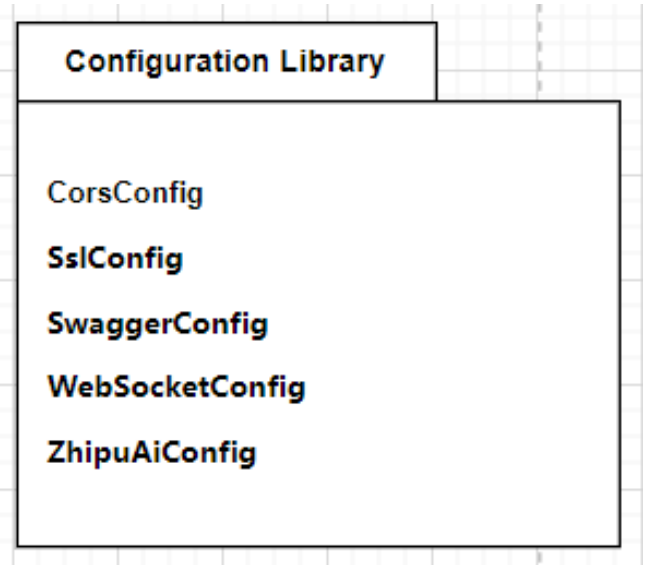

Figure 4.9: Configuration Library Components

4.3.6 DTO Library

- **AiPromptDTO**: AI prompts and image links.
- **AiQuestionDTO**: User questions and prompts for AI service.
- **ChildResponseDTO**: Student responses and task stage data.
- **UserRegisterDTO**, **UserUpdateDTO**: User registration and update information.

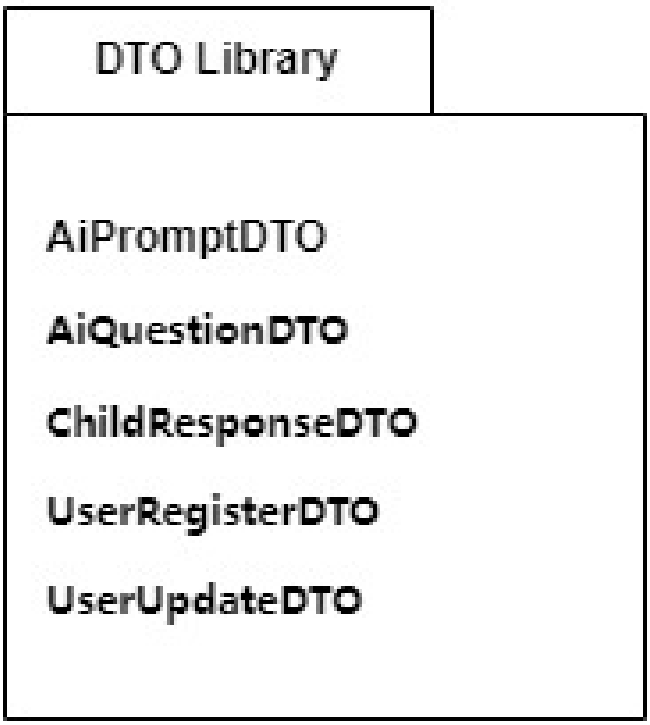

Figure 4.10: DTO Library Components

4.3.7 Security Control Library

- **JwtAuthenticationTokenFilter**: JWT verification filter.

- **JwtUtil**: JWT generation and parsing utility.
- **SecurityConfig**: Spring Security configuration with JWT.
- **UserDetailsImpl**: Custom user information implementation.
- **UserDetailsServiceImpl**: Custom user details service.

Figure 4.11: Security Control Library Components

4.3.8 Utility Library

AI Response Components

- **GlmGenerator**: Interacts with GLM model.
- **AiResponseGenerator**: AI response generation interface.
- **ASRUtil**: Processes voice recognition service parameters.
- **EncryptUtil**: Provides HMAC-SHA1 and MD5 encryption.
- **ImageProcessUtil**: Generates pinyin-annotated images.
- **PinyinGenerator**: Converts Chinese characters to pinyin.

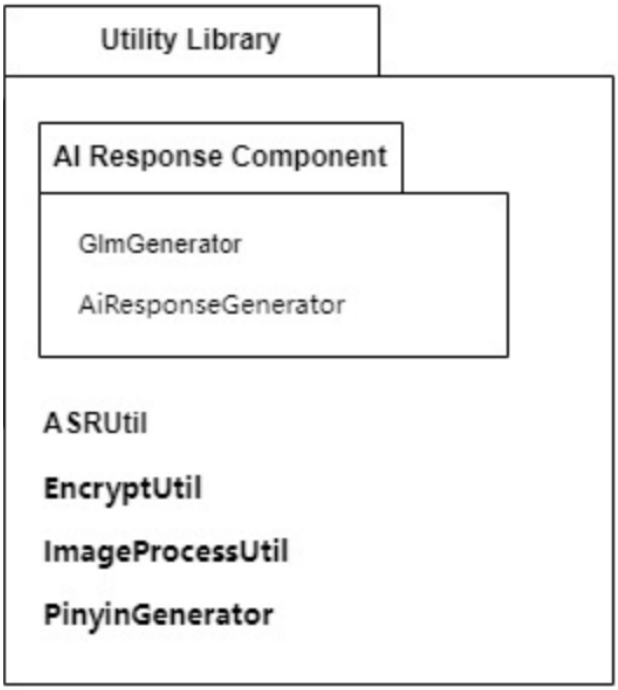

Figure 4.12: Utility Library Components

Model Library

- **Image**: Stores images required for tasks
- **StudentResponse**: Stores student response information
- **Task**: Stores task-related information
- **User**: Stores user-related information
- **UserTaskStage**: Records user's current task and stage information
- **UserTaskStageId**: Composite primary key including task_id and user_id

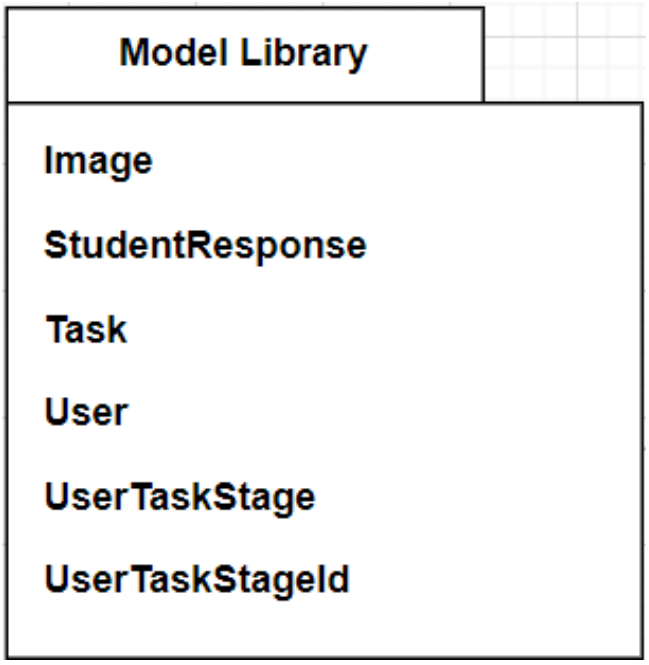

Figure 4.13: Model Library Components

4 Database

The database stores data related to users, tasks, images, and student responses. Main tables include:

- **Image**: Stores task-related images.
- **StudentResponse**: Stores student answers and feedback.
- **Task**: Stores detailed task information.
- **User**: Stores user account information.
- **UserTaskStage**: Records user task stage information.

Chapter5 Database Design

1 Database Overview

The Expression Master database design encompasses task management, student responses, and user management functionalities. The system supports creation and management of image-based tasks, allowing students to submit responses and receive scores with AI feedback.

2 Table Structure Design

5.2.1 User Table

Field Name	Type	Nullable	Key	Description
user_id	int	NOT NULL	PK	User ID, auto-increment
username	varchar(255)	NOT NULL	-	Username
password_hash	varchar(255)	NOT NULL	-	Password hash value
email	varchar(255)	NULL	-	Email address
registration_date	timestamp	NULL	-	Registration time, default current
avatar	text	NULL	-	User avatar URL

Table 5.1: User Table Structure

5.2.2 Task Table

Field Name	Type	Nullable	Key	Description
task_id	int	NOT NULL	PK	Task ID, auto-increment
task_prompt	json	NOT NULL	-	Task prompts/requirements
creation_date	timestamp	NULL	-	Creation time, default current
image_description	varchar(255)	NOT NULL	-	Image description
task_name	varchar(50)	NOT NULL	-	Task name, default ”Picture Description”

Table 5.2: Task Table Structure

5.2.3 Image Table

Field Name	Type	Nullable	Key	Description
image_id	int	NOT NULL	PK	Image ID, auto-increment
task_id	int	NOT NULL	FK	Associated task ID
image_url	varchar(255)	NOT NULL	-	Image URL
image_order	int	NOT NULL	-	Image sequence

Table 5.3: Image Table Structure

5.2.4 StudentResponse Table

Field Name	Type	Nullable	Key	Description
response_id	int	NOT NULL	PK	Response ID, auto-increment
student_id	int	NULL	FK	Student ID
task_id	int	NULL	FK	Task ID
response_text	text	NULL	-	Response content
score	decimal(5,2)	NULL	-	Score
submission_date	timestamp	NULL	-	Submission time, default current
ai_feedback	json	NOT NULL	-	AI feedback content

Table 5.4: Student Response Table Structure

5.2.5 UserTaskStages Table

Field Name	Type	Nullable	Key	Description
user_id	int	NOT NULL	FK	User ID
task_id	int	NOT NULL	FK	Task ID
stage	int	NOT NULL	-	Current stage
update_time	timestamp	NULL	-	Update time, auto-update

Table 5.5: User Task Stages Table Structure

3 Relationship Constraints

5.3.1 Foreign Key Constraints

- Image table:
  - task_id references Task(task_id) ON DELETE CASCADE
- StudentResponse table:
  - student_id references User(user_id) ON DELETE CASCADE
  - task_id references Task(task_id) ON DELETE CASCADE
- UserTaskStages table:
  - user_id references User(user_id) ON DELETE CASCADE
  - task_id references Task(task_id) ON DELETE CASCADE

5.3.2 Unique Constraints

UserTaskStages table:

- Combination of (user_id, task_id, stage) must be unique

5.3.3 Index Design

- Image table:
  - Index on task_id
- StudentResponse table:
  - Index on student_id
  - Index on task_id
- UserTaskStages table:
  - Index on task_id
  - Unique composite index on (user_id, task_id, stage)

4 Storage Engine

All tables use the InnoDB storage engine, supporting transaction processing and foreign key constraints.

5 Character Set

The database uses utf8mb4 character set with utf8mb4_0900_ai_ci collation, supporting the complete Unicode character set.

Chapter6 Other Technical Details and Information

1 Development Environment and Configuration Requirements

To ensure proper development and operation of the Expression Master system, the following configurations are required:

- **Backend Development Environment:**
  - Java Development Kit (JDK) 17 or higher
  - Spring Boot 2.7.0 or later
  - Maven 3.8+ for dependency management
  - MySQL 8.0 or higher for database
- **Frontend Development Environment:**
  - Node.js 16.0 or higher
  - React 18.0+
  - npm or yarn package manager
  - WebSocket client support
- **AI Service Requirements:**
  - ZhipuAI API access credentials
  - Minimum 4GB RAM for AI model operations
  - Stable internet connection for API calls
- **System Requirements:**
  - 8GB RAM minimum (16GB recommended)
  - 4-core CPU or better
  - 20GB available storage
  - Broadband internet connection

2 Security Implementation Details

6.2.1 JWT Token Implementation

The system implements JWT-based authentication with the following specifications:

- **Token Structure:**
  - Header: Algorithm and token type
  - Payload: User ID, username, and roles
  - Signature: HMAC-SHA256 encryption
- **Security Measures:**

  - 14-day token expiration
  - Secure token transmission over HTTPS
  - Token blacklisting for logged-out users
  - Regular security audits and updates

3 Performance Optimization

6.3.1 Frontend Optimization

- **React Component Optimization:**
  - Implemented React.memo for performance-critical components
  - Used useCallback and useMemo hooks for callback and value memoization
  - Lazy loading for route-based code splitting
- **Asset Optimization:**
  - Image compression and lazy loading
  - CSS minification
  - JavaScript bundle optimization

6.3.2 Backend Optimization

- **Database Optimization:**
  - Implemented database indexing for frequently queried fields
  - Connection pooling for efficient database connections
  - Query optimization and caching
- **API Performance:**
  - Response caching for static content
  - Pagination for large data sets
  - Compression for response payload

4 Testing Strategy

6.4.1 Unit Testing

- **Backend Testing:**
  - JUnit for service layer testing
  - Mockito for dependency mocking
  - Coverage target of 80% for critical components
- **Frontend Testing:**
  - Jest for component testing
  - React Testing Library for UI testing
  - Snapshot testing for UI components

6.4.2 Integration Testing

- **API Testing:**
  - Postman collections for API endpoint testing
  - WebSocket connection testing
  - End-to-end flow validation
- **System Integration:**
  - AI service integration testing
  - Database integration testing
  - Authentication flow testing

5 Maintenance and Monitoring

6.5.1 System Monitoring

- **Performance Monitoring:**
  - API response time tracking
  - Database query performance monitoring
  - Memory usage and CPU utilization tracking
- **Error Tracking:**
  - Centralized error logging
  - Real-time error notifications
  - Error pattern analysis

6 Future Improvements

6.6.1 Planned Enhancements

- **Feature Additions:**
  - Advanced voice recognition capabilities
  - More sophisticated AI feedback algorithms
  - Enhanced student progress analytics
  - Mobile application development
- **Technical Improvements:**
  - Microservices architecture migration
  - Real-time collaboration features
  - Enhanced security measures
  - Performance optimization for larger scale

6.6.2 Documentation

- **Documentation Updates:**
  - Comprehensive API documentation
  - Detailed deployment guides
  - User manual updates
  - Developer contribution guidelines
- **Knowledge Base:**
  - Troubleshooting guides
  - Best practices documentation
  - Common issues and solutions
  - Performance optimization guidelines