e-ISSN: 2278-8875
p-ISSN: 2320-3765

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

Volume 11, Issue 1, January 2022

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Impact Factor: 7.282

| e-ISSN: 2278 – 8875, p-ISSN: 2320 – 3765| www.ijareeie.com | Impact Factor: 7.282|

||Volume 11, Issue 1, January 2022||

|DOI:10.15662/IJAREEIE.2022.1101021|

Design and Implementation of a Pattern-based J2EE Application Development Environment

Sakshi Sharma[1], Natasha Dutta[2]

Senior Technical Project Manager, Kforce Global Solutions Inc. [1]

Information Security Engineer, Online Micro Services, India[2]

ABSTRACT: This paper presents the design and implementation of a pattern-based J2EE application development environment aimed at streamlining the development process, enhancing code quality, and improving maintainability. We begin with an exploration of the fundamental principles of J2EE and the role of design patterns in software engineering. This is followed by a detailed discussion on the selection and application of various design patterns specifically tailored for J2EE, such as Singleton, Factory, Data Access Object (DAO), and Model-View-Controller (MVC).

KEYWORDS: Java 2 Platform, Enterprise Edition (J2EE), Software Design Patterns, Distributed Applications, Multi-tiered Applications, Behavioral Patterns, Data Access Object (DAO) Pattern, Structural Patterns.

I. INTRODUCTION

The evolution of enterprise software development has been significantly influenced by the advent of the Java 2 Platform, Enterprise Edition (J2EE). J2EE offers a robust and scalable framework that supports the development of large-scale, distributed, and multi-tiered applications. However, the complexity inherent in J2EE applications necessitates a methodical approach to ensure maintainability, scalability, and efficiency. One effective methodology that has garnered widespread acceptance is the use of software design patterns. Design patterns provide general reusable solutions to common problems encountered during software design. They encapsulate best practices and can dramatically improve the robustness and maintainability of the code. The application of these patterns in a J2EE environment can significantly streamline the development process and enhance the overall quality of the software [1].
Despite their benefits, the integration of design patterns into J2EE development is not without challenges. Developers often face difficulties in selecting the appropriate patterns for their specific use cases and in implementing these patterns correctly. Additionally, maintaining consistency in the application of design patterns across large teams can be problematic [2]. To address these challenges, this paper explores the design and implementation of a pattern-based J2EE application development environment. This environment aims to provide a structured and systematic approach to incorporating design patterns into the J2EE development process, thereby mitigating common issues and optimizing development workflows.

We begin by examining the fundamental principles of J2EE and the critical role that design patterns play in software engineering. This includes a discussion on various types of design patterns such as creational, structural, and behavioural patterns, with a specific focus on those most relevant to J2EE applications, like Singleton, Factory, Data Access Object (DAO), and Model-View-Controller (MVC) [3]. Following this, we delve into the architectural design of a pattern-based development environment. This involves evaluating existing tools and frameworks that support pattern-based development, such as the Spring Framework and Hibernate, and discussing their integration into our proposed environment.

The implementation phase is illustrated with practical examples and case studies, providing concrete evidence of the benefits and applicability of a pattern-based approach in real-world scenarios. We also address the automation of pattern application to ensure consistency and enhance developer productivity. Testing and maintaining pattern-based J2EE applications are crucial aspects that are covered comprehensively, with strategies to ensure long-term sustainability and reliability. We consider the future directions of pattern-based J2EE development, particularly in the context of emerging trends such as cloud computing and microservices architecture. These trends present both opportunities and challenges, and our discussion aims to provide insights into how pattern-based development can evolve to meet these new demands [4-8]. By implementing a pattern-based J2EE application development environment, developers and organizations can achieve greater efficiency, higher code quality, and improved scalability, thus positioning themselves for success in the dynamic and rapidly evolving field of enterprise software development.

1.1 Introduction to Related Technologies

J2EE includes the following three component types: one is the in-app program, which is mostly used for internal training computers; the second is Servlet and JSP components, which are often used in some Internet; the third is the EJB component, which completes its functions on the server-side. J2EE belongs to JAVA, and its programming method is similar to JAVA, but there are some substantial differences. J2EE components can be used in application programs, and at the same time, they can be consistent with the J2EE specification.

Since J2EE has a complete set of specifications, in general, J2EE can be regarded as a constraint of the JAVA2 platform on music education. In addition, J2EE has many advantages; for example, it can meet various requirements of the bottom layer of the system through containers, and the development speed of system personnel has been greatly improved. J2EE uses the middle-tier integration framework for program development, which reduces the cost of system development, and not only improves its performance but also ensures the security of system operation. At the end of the 20th century, to design a cross-platform and distributed software system, Sun Computer Systems researched and launched the JAVA language in the United States, which is an object-oriented design language. For software developers, learning JAVA is simpler to use, and can also perform exception handling and automatic collection of discards [9]. The programming language has functions such as porting and interpretation, so it has been widely used after its launch.

After continuous development, JAVA has been continuously expanded based on the original programming language and has become a mainstream technology in the computer software industry. Chip technology, Internet connection technology, and other fields based on the many advantages of JAVA, its application is also very wide, and its main application directions include game systems developed in large numbers today and mobile Internet-related systems. At the same time, it is also widely used in many colleges and universities, such as educational administration, college teaching, and other management systems. At the same time, due to its various advantages, it is often used in the relevant information systems of government departments.

1.2 Importance of J2EE in building scalable and robust enterprise applications

Java 2 Platform, Enterprise Edition (J2EE) is crucial for developing scalable and robust enterprise applications due to its comprehensive and standardized framework. It offers a set of services, APIs, and protocols that simplify the complexity of building large-scale, distributed, and multi-tiered applications. J2EE's modular component architecture allows developers to build reusable and maintainable components, enhancing the scalability of applications. Its support for transaction management, security, and concurrency ensures that applications are reliable and can handle high volumes of transactions securely and efficiently. Additionally, J2EE's portability across different platforms ensures that applications can run on various systems without modification, further contributing to its robustness and scalability [10]. This makes J2EE a preferred choice for enterprises aiming to build complex applications that can grow and adapt to increasing demands and evolving business needs.

UML Modeling Technology

Unified Modeling Language (UML) is highly effective for visualizing program structures and is widely used in modern program design. Utilizing UML patterns enhances the clarity of program layers, simplifying software testing and maintenance. This approach significantly reduces development and testing time for developers.

UML employs various diagram types, including use case diagrams, sequence diagrams, activity diagrams, class diagrams, and state diagrams. These diagrams represent the entire lifecycle of a subject, from creation to termination, by segmenting the program into distinct sections, thereby streamlining its operation.

The primary function of the UML view is to display program processing outcomes. Programmers use these visual representations to analyze the program. For web-based operations, the model must first be converted into HTML to function within the system's environment, allowing UML's visual graphics to illustrate the program. UML facilitates the creation of view-related content and enables efficient client interaction.

In UML design, the focus is primarily on the data model object. Throughout the process, the control layer in use case diagrams transfers the relevant model objects to the view layer, where operations are displayed. This data model encompasses complex calculation rules, processing flows, and analysis rules, which are used to process and present data. The control layer acts as a bridge between the view and model layers, ensuring seamless interaction and operation [11-13]. This approach enhances the coupling between data and logic interfaces across different program layers. Implementing UML technology makes systematic testing of software, such as music education systems, straightforward and efficient.

II. ADVANTAGES OF UML AND ITS APPLICATION

Unified Modeling Language (UML) offers numerous advantages that significantly enhance software development processes. As a standardized modeling language, UML provides a comprehensive way to visualize, specify, construct, and document the artifacts of a software system. One of its primary benefits is the ability to create clear and detailed visual representations of a system's architecture and design, making complex systems easier to understand and manage. These visual models help bridge the communication gap between stakeholders, including developers, designers, and non-technical team members, ensuring everyone has a shared understanding of the system's functionality and structure.
UML's versatility is evident through its wide range of diagrams, such as use case diagrams, sequence diagrams, activity diagrams, class diagrams, and state diagrams, each serving a specific purpose in depicting different aspects of a system. For instance, use case diagrams help identify and organize system requirements, while class diagrams illustrate the static structure of the system, showing the system's classes, attributes, operations, and the relationships among objects. Sequence diagrams, on the other hand, capture the interaction between objects over time, providing insights into the dynamic behavior of the system. This multi-faceted approach allows developers to tackle both the structural and behavioral aspects of a system comprehensively. UML enhances the maintainability and scalability of software systems. By clearly defining system components and their interactions, UML facilitates easier updates and modifications. It also supports better planning and design, reducing the risk of costly errors and rework. Additionally, UML aids in the seamless integration of new features and technologies, ensuring the system can evolve with changing requirements.

In practice, UML's application extends across various stages of the software development lifecycle. During the analysis phase, UML diagrams help in gathering and analyzing requirements. In the design phase, they assist in creating detailed system architectures. During implementation, UML models guide developers in writing coherent and consistent code. Finally, in testing and maintenance, these models provide a reference for verifying system functionality and making necessary adjustments.

The adoption of UML also promotes better documentation practices. Well-documented UML models serve as valuable references for future development and maintenance activities, facilitating knowledge transfer and reducing dependency on individual developers. This is particularly beneficial in large, distributed teams or in scenarios where the original development team may not be available for ongoing support.

UML's ability to improve communication, enhance system understanding, and provide a structured approach to software design and development makes it an indispensable tool in modern software engineering. Its application leads to more efficient, reliable, and maintainable software systems, ultimately contributing to the success of software projects.

III. ANALYSIS OF FUNCTIONAL REQUIREMENTS

Information Release Module
The information release management module comprises two primary functions: publishing news and announcements. News is intended for all users of the music teaching auxiliary system, involving tasks such as editing, publishing, and maintaining news content. Announcements, on the other hand, target registered users—teachers, students, and administrators—mirroring the news management process.

Document Management Module
The document management module in the music teaching auxiliary system oversees the handling of official documents within the school's teaching and administrative processes. This module manages the entire lifecycle of documents, from drafting and approval to issuance, leveraging network and computer technologies to facilitate a paperless office environment, enhance efficiency, and monitor the document circulation process.

Teaching Resource Management Module
This module focuses on managing self-study resources for music education, including audio and video materials, homework exercises, and general music knowledge dissemination.

Auxiliary Teaching Management Module
Aimed primarily at system administrators, this module manages courses, oversees student online examinations and exercises, facilitates interactive Q&A and discussions, and conducts performance analysis.

System Management Module
The system management module supports the activities of students and teachers, ensuring the smooth operation of the system and fostering collaboration and communication across different modules. This management area can be divided into the following sections:

Music-assisted Teaching: Through data statistics and analysis, this function aids school leaders and teachers in making informed decisions to improve teaching quality and management.

Log Management: This is split into operation logs and system logs. Operation logs track all functional activities, while system logs record the system's running processes.

User Management: This function manages user identities (teachers, students, leaders, and managers) and their basic information.

3.1 Model Analysis
Object Model Analysis
The needs of music teaching auxiliary management are mainly reflected through the use case model, which can build a bridge between ordinary customers and system implementers, allowing users to describe their own needs and the functions to be achieved by the required system in the most detailed way. There is a Controller

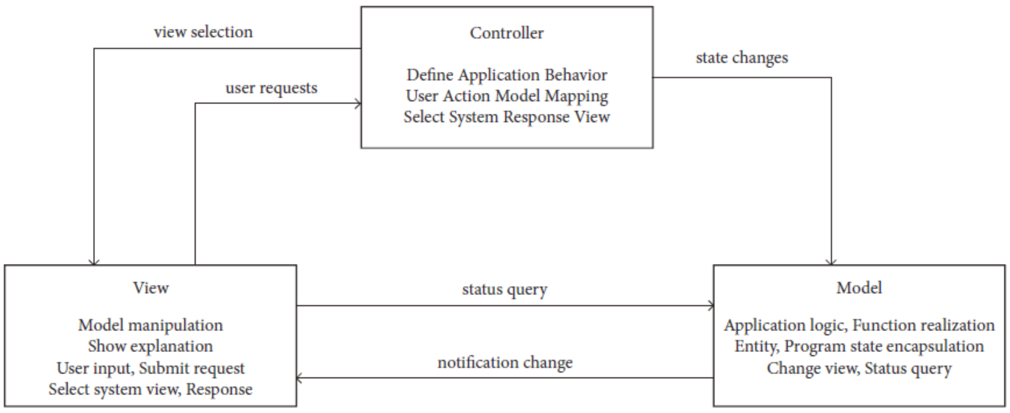
view selection
Controller
Define Application Behavior
User Action Model Mapping
Select System Response View
state changes
user requests
View
Model manipulation
Show explanation
User input, Submit request
Select system view, Response
status query
notification change
Model
Application logic, Function realization
Entity, Program state encapsulation
Change view, Status query

Figure 1: Schematic diagram of MVC mode function implementation

Scientific Programming and the Relationship Between Use Case Diagrams and Modularity
In scientific programming, a crucial aspect is the refinement of use case diagrams and their modularity. Using UML (Unified Modeling Language) to model a system allows for a clear visualization of system functions and specific, feasible methods for implementation. The primary goal is to address the needs of all stakeholders, starting with making the system easy for users to understand and ensuring developers have a clear understanding to initiate development.

The establishment of use cases must prioritize system participants. This involves a detailed analysis of actual system users and their interactions, followed by a combination of their actions and behaviors to realize the required functions. This process leads to a specific and practical use case analysis. UML can describe use cases from multiple perspectives, detailing participants and their intricate relationships. Addressing these aspects helps define the construction of the use case.

Designing use cases follows a top-down approach, progressing from basic to detailed requirements. Initially, the most fundamental use cases are created based on the project's specific needs. These use cases are then refined step by step to meet deeper customer requirements and target analyses. For example, Figure 2 illustrates the use case diagram for school leaders.

School leaders primarily perform tasks such as handling official documents, browsing news and announcements, and managing auxiliary decision-making for music teaching. Music teachers, on the other hand, manage operations like examination question bank management, courseware management, student communication, and homework and examination administration, as shown in Figure 3.

Students engage in activities like consulting on music-related questions, online learning, and online testing. Consultations cover music knowledge, exam-related queries, and general discussions, illustrated in Figure 4.
System administrators handle user management (including managing teachers, students, and authorities, with leaders classified under teacher management with higher authority), log management, system settings, and information release (encompassing news and announcement management), among other operations.

By following this structured approach, the system can effectively meet the needs of all participants, ensuring a comprehensive and user-friendly design.

3.2 Data Model Analysis

The analysis of the data model is mainly carried out using the data flow chart, which can systematically and comprehensively describe the data logic of the music teaching auxiliary management system. In the process of information storage, processing, and flow reflection, the data flow diagram is mainly realized employing centralized and general symbols. The characteristics of the data flow graph are mainly composed of the following two aspects:
(1) Abstraction: abstract data information into specific information storage, processing and flow, and reduce unnecessary object processes
(2) Generality: express all requirements as a whole, and associate all related information or business processing
A data flow diagram consists of the following four basic elements, namely, external entities, data flow, processing (function), and data storage. Each module uses a data flow diagram to represent the source of data and the relationship between data. The top-level data flow of the system is shown in Figure 5.

IV. ANALYSIS OF NONFUNCTIONAL REQUIREMENTS

4.1 Performance Requirements

(1) Performance - The safe and reliable operation time of the music teaching auxiliary platform is not less than ten years; the number of simultaneous online teachers and students is not less than 400; the response time of business processing shall not exceed five seconds; to adapt to the continuous increase of functions, automatic expansion support is required; all data transmissions must be stable and secure;

(2) Security - Security requirements mainly include data transmission encryption, data backup, and virus prevention. Data transmission encryption: different levels of users set different access rights, and MD5 encryption is performed for user authentication; Data backup: data backup is performed regularly to ensure safe operation; Virus prevention: because it is in the form of a network, virus prevention and awareness enhancement are required.

(3) Scalability - In the process of music teaching, various teaching methods, teaching modes, and functions are changing with each passing day, and need to be maintained and upgraded from time to time. Therefore, the auxiliary teaching system must be scalable and configurable.

The auxiliary management of music teaching is realized by modularization, which improves the reusability and maintainability of the code by reducing the association between different modules.

The upgrade of the system must not affect the normal operation and only needs to be upgraded online on the server side.

Design Goals - The design goals of the music teaching assistant management system based on J2EE mainly include the following aspects:

| e-ISSN: 2278 – 8875, p-ISSN: 2320 – 3765| www.ijareeie.com | Impact Factor: 7.282|

||Volume 11, Issue 1, January 2022||

|DOI:10.15662/IJAREEIE.2022.1101021|

Practicality: simultaneously design information functions based on traditional operation methods to provide a concise and clear operation interface for teachers and students when teaching or learning music; Fault tolerance: when data errors are caused by a program running or Scientific Programming 5

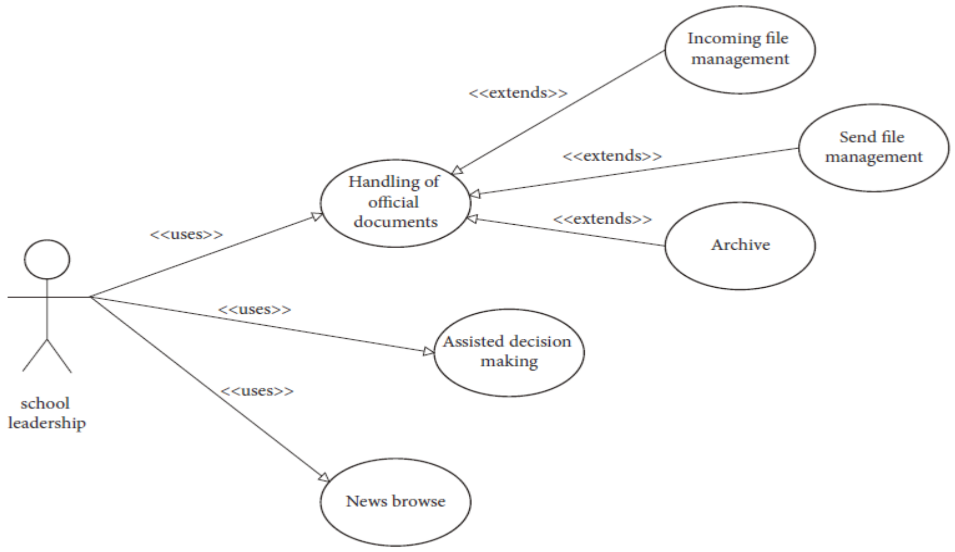

Figure 2: School leadership use case diagram

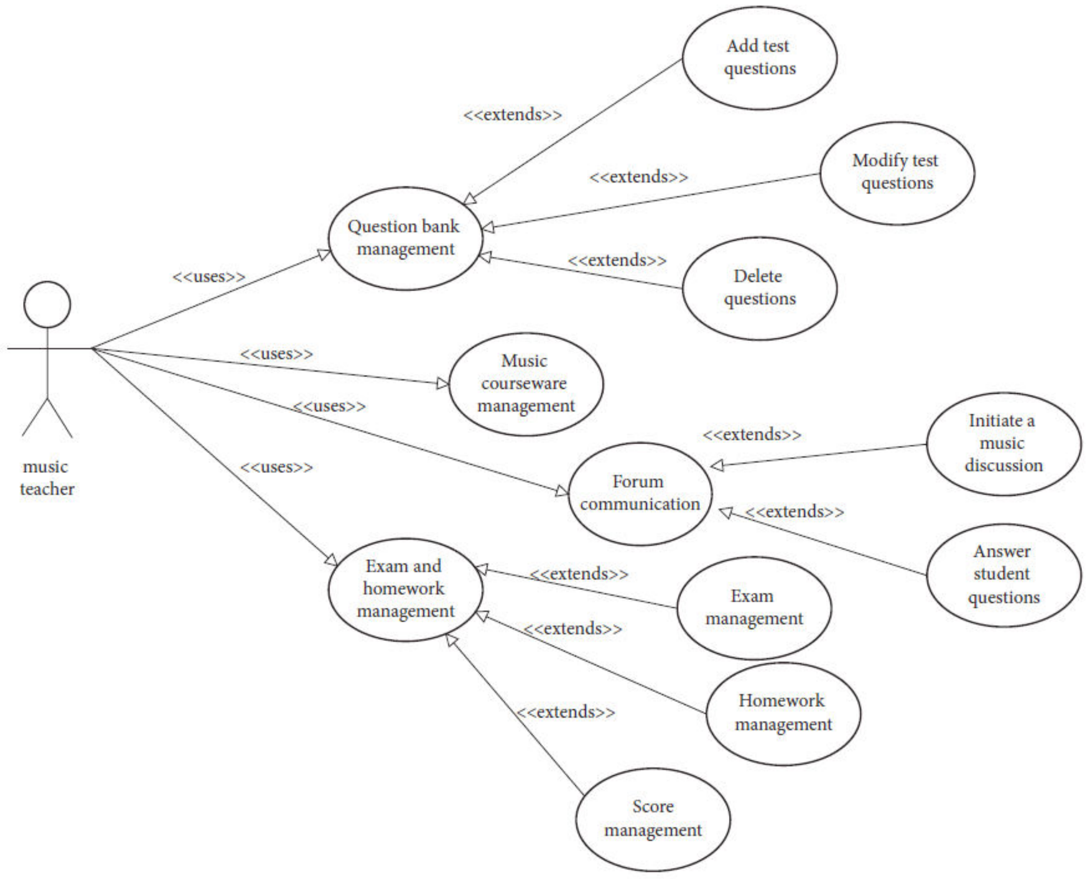

Figure 3: Music teacher use case diagram

Scientific Programming
In MIS operation, the system must have good fault tolerance capabilities, such as error prompts and automatic rollbacks, and both security and reliability must be guaranteed during the fault tolerance process.

Versatility: the system cannot be limited to a specific browser access form to meet the needs of learners and operators' operating habits;

Openness: the system adopts a standardized processing form and provides a good extension interface in the research of music teaching curriculum setting to ensure that the system can be used in multiple platforms and environments.

Timeliness: as long as there is a network, the learning and management of music-related courses and knowledge can be carried out at any time.

Stability: we use a data management system to achieve data management, use J2EE technology and MVC design pattern to achieve system development, and ensure the stability of system operation.
The music teaching assistant management platform has the characteristics of real-time communication, functional versatility, and object openness.

4.2 Design of the Music Teaching System
Overall Design
The overall design of the system includes functional structure design and architecture design, which can generally be simplified as a modular design. When n carrying out modular design, not only should the entire system be divided into components, but also the Consulting

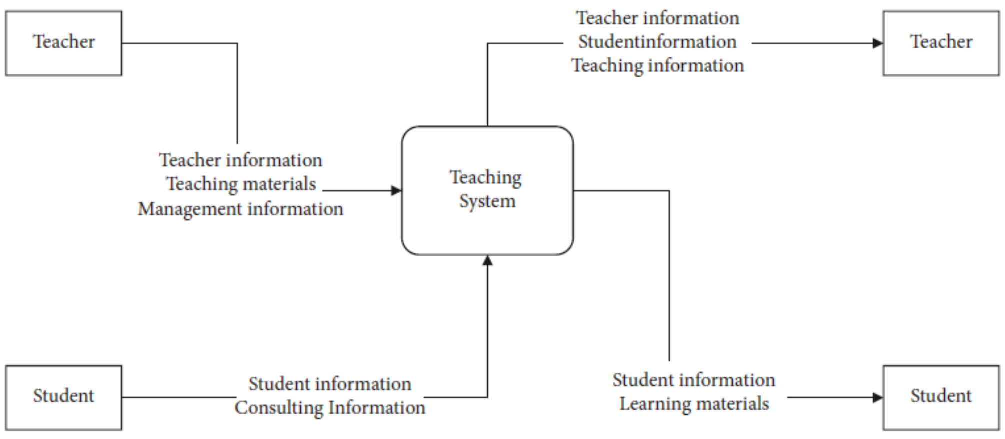

Figure 4: System top-level data flow diagram

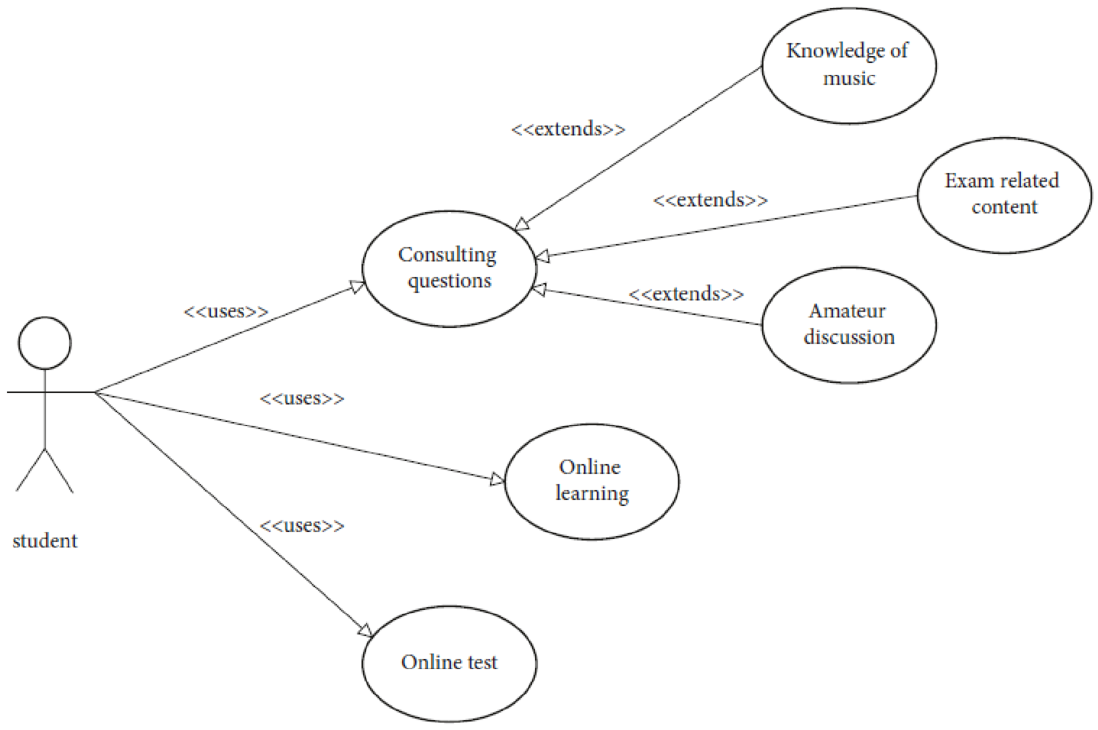

Figure 5: Student use case diagram

The communication between modules, module continuity, module protection, module solvability, and module combination should be designed. If you want to improve the continuity principle, you need to maintain the black box characteristics between modules, which are transparent to other modules. If the protection of the module needs to be improved, it is necessary to protect the variables inside the module to prevent the misuse of other modules, and special consideration should be given to the exception handling of the module.

When carrying out the modular design of the system, it is necessary to meet the loose coupling between modules and the high cohesion characteristics within the modules. Modules must be able to function independently to complete the design, but the size of the modules needs to be controlled. The nature of modules can be summarized as interchangeability, pluggability, and boundedness, which mainly include three aspects: (1) When the internal requirements of a module change, the changes do not affect the normal operation of other modules; (2) When a module needs to be deleted, only the functions handled by this module are affected; (3) If a new module implements the same function and has the same operation interface, it will not affect the operation of the entire system after replacement.

Functional Structure Design
The music-assisted teaching management system is mainly composed of modules such as information release, official document management, teaching resource management, auxiliary teaching management, and system management. The information release module is further divided into school news, notice announcements, and BBS. Document management, document receipt management, and filing management; teaching resource management includes courseware management such as music videos, homework management, and music knowledge management; auxiliary teaching management is divided into course management, online examinations, Q&A discussions, and performance analysis; system management is divided into teachers student management, system settings management, log management, and auxiliary decision-making management. The functional structure of the music-assisted teaching management system is shown in Figure 6.

Architecture Design
The design of the music teaching assistant management system adopts a three-tier system structure. The architecture of the system is mainly divided into three layers: Web Server layer (display layer), Application Server layer (control layer), and Database Server (data access layer). The display layer is mainly composed of the Web UI Layer and Web

Service Layer, using JSP and other technologies to achieve interaction with the client; the control layer mainly uses the core frameworks such as Spring to complete the business logic processing of the auxiliary teaching system, calls the data access layer to process the business request sent by the client, and displays the processing result to the user. The specific form of interaction is displayed to the client for users to view and browse; the data access layer mainly realizes the interaction between the business logic layer and the database, preventing business requests from directly accessing the database, causing data inconsistency, and ensuring data security and integrity.

Database Design
In the process of information service, information management, and resource development, system users and design developers have concluded an important experience that database technology is the most effective way to manage data. With the advancement of network technology and computer technology, data management through databases has become an important consensus. In order to complete the sharing, integrity, and consistency of system data, both large-scale management systems and small-scale transaction processing are using database technology to complete data management. At present, the important criteria for measuring the degree of a country's informatization construction are the frequency of database use, the amount of database information, and the scale of the database.

Database Design Principles
The database design of the music-assisted teaching management platform mainly serves the business knowledge base of music-assisted teaching management. Through the management of the database, the processing of knowledge data and business data is optimized. When designing a database, the following aspects are considered important criteria:
(1) Verification is based on database design specifications, and data structure design is carried out in a standardized form to ensure the consistency and normal operation of data operations.

(2) Normative naming: the naming specification is helpful for unified management and upgrades maintenance in the later stage. Therefore, when designing databases and tables, the naming must strictly follow the normative standards and annotate all column information.

(3) Data redundancy and the standardization of data paradigms will affect the retrieval speed of later data. Therefore, when designing data tables, it is necessary to master various degrees in order to achieve the highest value of retrieval and reduce system response time.

(4) Security: strict identity authentication management is implemented, and users with different permissions have different degrees of access to data and operations to improve data security.

(5) Concurrency control: through the use of triggers and stored procedures, the simultaneous operation of the table is strictly controlled to ensure the control of simultaneous modification access, and reduce data inconsistency, and the query can be exempted from this control. Through the above analysis, it can be seen that when designing the database of the music teaching assistant management platform, to 8 Scientific Programming realize the BC paradigm mode, the third paradigm should be used as the main basis.

Conceptual Structure Design
The conceptual schema design of the database is to abstract the existing data. Abstraction refers to the man-made processing of concepts, affairs, and relationships between people, extracting common features that are needed, and ignoring unnecessary parts. The extracted features are described in detail and finally form a certain model structure. This system adopts SQL Server2005 as the database development tool to realize the design and realization of a relational database.

| e-ISSN: 2278 – 8875, p-ISSN: 2320 – 3765| www.ijareeie.com | Impact Factor: 7.282|

||Volume 11, Issue 1, January 2022||

|DOI:10.15662/IJAREEIE.2022.1101021|

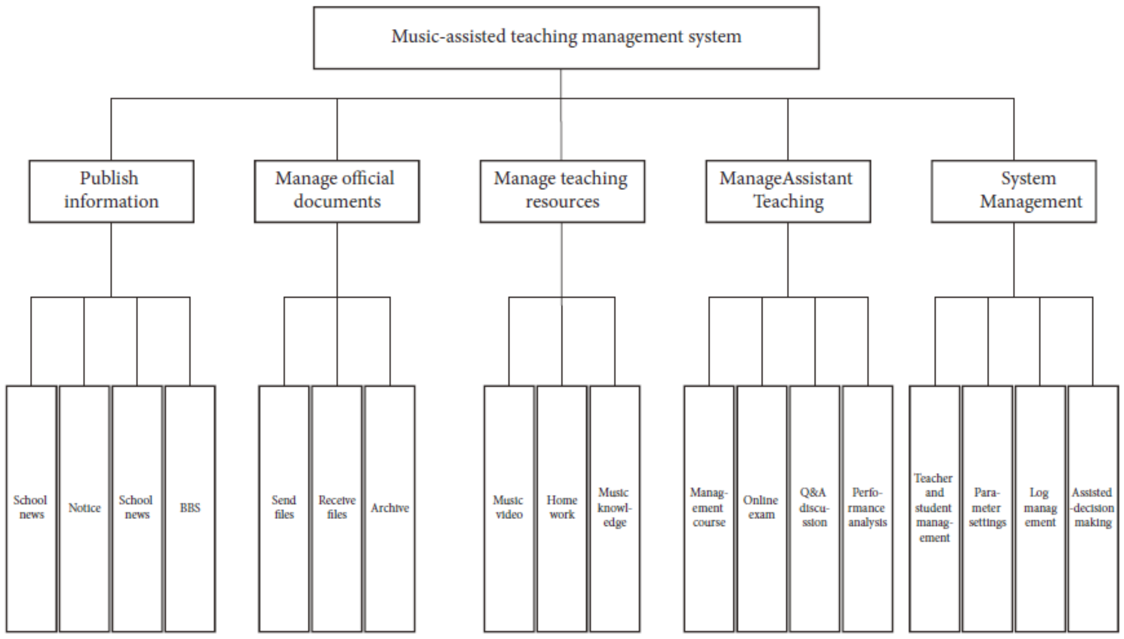

Figure 6: System function structure diagram

V. CONCLUSION

This study focuses on the design and development of a music teaching assistant system. Initially, it discusses the importance, applications, and feasibility of creating an interactive multimedia network teaching system, supported by background research and feasibility analysis.

During the requirements analysis phase, the system's needs are thoroughly examined and documented to outline all user requirements for the proposed system. This phase involves creating a comprehensive model of the music teaching assistant system and detailing its functional design and implementation. This clear understanding of required functionalities enables the software design team to precisely identify their tasks, facilitating better control over development progress and quality. Furthermore, it provides a foundation for future upgrades, allowing for quick identification and resolution of deficiencies and streamlining the software upgrade cycle.

Although the developed system demonstrates a high level of functionality and technological advancement, it has some shortcomings. Specifically, certain module functions are incomplete, and some features are not fully considered. However, with ongoing improvements by the development team, the system is expected to be eventually ready for practical application.

REFERENCES

1. H. Yan, “Design of online music education system based on artificial intelligence and multiuser detection,” Algorithm[J]. Computational Intelligence and Neuroscience, vol. 2022, Article ID 9083436, 11 pages, 2022.
2. M. Liu, “Research on the application value of off music education system in college music education based on computer new media technology[J],” Journal of Physics: Conference Series, vol. 2021, no. 3, 1992.
3. L. Vitchatalum, C. Yootthana, L. Monsikarn, and S. Skowrung, “An investigation into the status of Thailand’s music education systems and organisation[J],” British Journal of Music Education, vol. 38, no. 2, 2020.
4. A. A. Kallio and M. Heimonen, “A toothless tiger? Capabilities for indigenous self-determination in and through Finland’s extracurricular music education system,” Music Education Research, vol. 21, no. 2, pp. 150–160, 2019.
5. C. G. Espada, “El sistema institucional de la Uni´on en el Tratado que establece una Constituci´on para Europa (2004) [J],” Anuario Español de Derecho Internacional, vol. 20, 2018.
6. A. O. Adeogun, “A historical review of the evolution of music education in Nigeria until the end of the twentieth century,” Journal of the Musical Arts in Africa, vol. 15, no. 1-2, pp. 1–18, 2018.

| e-ISSN: 2278 – 8875, p-ISSN: 2320 – 3765| www.ijareeie.com | Impact Factor: 7.282|

||Volume 11, Issue 1, January 2022||

|DOI:10.15662/IJAREEIE.2022.1101021|

7. G. Baker, “El sistema, “the Venezuelan musical miracle”: the construction of a global myth,” Latin American Music Review, vol. 39, no. 2, pp. 160–193, 2018.
8. Orfilia Saiz Vega, “Lectura a primera vista en los instrumentos de cuerda. Situaci´on educative,” Dedica. Revista de Educação e Humanidades, no. 13, 2018.
9. C. Ruiz, “Reflexiones sobre pol´ıticas educativas de reforma y educaci´on musical,” Revista Internacional De Educaci´on Musical, vol. 3, no. 1, pp. 69–73, 2015.
10. P. Limited, Patent Application Titled “System and Method for Music Education, Politics & Government Week, 2014.
11. X. Yu and X. Gui, “Design and implementation of macroeconomic situation analysis application system based on J2EE technology[J],” Journal of Physics: Conference Series, vol. 2021, no. 3, 1881.
12. F. Wu, “Based on J2EE research and realization of national fitness sports convenience service platform[J],” Journal of Physics: Conference Series, vol. 1693, no. 1, 2020.
13. Computing, Study Findings on Computing Are Outlined in Reports from University of Jinan (Tfa: An Efficient and Precise Virtual Method Call Resolution for Java)[J], Computer Technology Journal, 2020.
14. Software Research, New Findings Reported from Amrita Vishwa Vidyapeetham Describe Advances in Software Research (Finite-State Model Extraction and Visualization from Java Program Execution)[J], Computer Weekly News, 2020.
15. Sustainability Research - Sustainable Food and Agriculture, Sebelas Maret University Researchers Further Understanding of Sustainable Food and Agriculture (-e Contradiction of Sustainable Food Agricultural Land Protection of Sukoharjo Regency, Central Java)[J], Food Weekly News, 2020.