Design and Implementation of Web front-end based on Mainframe education cloud

Fan Pan
Abstract

Design and Implementation of Web front-end based on Mainframe education cloud

Fan Pan

Mainframe is a server expert in online transaction and batch job and be widely used in different industries especially banking while mainframe skilled specialists are limited. Cloud computing makes it possible to share rare hardware and deliver services by infrastructure, platform and so on. This text explains how the Z Education Cloud can provide stable and high-value education services that support 21st-century mainframe skill development. Additionally, the text outlines design and implementation for the education cloud Web-End that can help college mainframe education.

Firstly, technology mechanism analysis of Web front-end for Z Education Cloud is done with the following aspects: B/S architecture, MVC design pattern, SSH development framework are introduced into this project. The author also proposes a system which is asynchronous communication mechanism between front-end and back-end according to the specialty of mainframe service. Secondly, we do the requirement from Business Requirement and Functional Requirement, define all the function modules and draw the use cases and class diagram with UML. After that, based on the requirements, this text explains how the Z Education Cloud Web-end designs and realizes. There comes up with a mechanism of roles and permissions management for the system, the detailed design proposal and implementation for the resource management module and application management module are focused on. After the implementation is done, we test the system on performance and functions and point the weakness founded during testing.

In the finality, the problems requiring further studies are discussed.
# Table of Contents

Chapter 1 Introduction .................................................................................................................9
  1.1 Origin of My Work ............................................................................................................... 9
  1.2 Background and Significance ............................................................................................. 9
  1.3 Research and Development Status across the World ........................................................ 10
    1.3.1 Development of large-scale mainframe education platforms .................................... 10
    1.3.2 Web Front-End Implementation Research Status around the World ......................... 11
  1.4 Contents and Contributions ................................................................................................. 12
    1.4.1 Contents ..................................................................................................................... 12
    1.4.2 Contributions .............................................................................................................. 12
  1.5 Organization of This Text .................................................................................................... 13

Chapter 2 Description of Related Technologies ........................................................................ 14
  2.1 Architecture and Selection ................................................................................................. 14
    2.1.1 C/S architecture ......................................................................................................... 14
    2.1.2 B/S architecture ........................................................................................................ 14
  2.2 MVC Design Mode ............................................................................................................. 15
    2.2.1 MVC design mode ..................................................................................................... 15
  2.3 Development Platform and SSH Development Framework .............................................. 16
    2.3.1 Struts ........................................................................................................................ 16
    2.3.2 Spring ....................................................................................................................... 17
    2.3.3 Hibernate .................................................................................................................. 17
  2.4 Database Tools and Selection ............................................................................................ 17
  2.5 Asynchronous Message Mechanism for Communication between Front and Back Ends ... 18
  2.6 Mainframe-Related Technologies .................................................................................... 18
  2.7 Chapter Summary .............................................................................................................. 19

Chapter 3 Web Front End Requirements of the Mainframe Teaching Cloud Platform ............ 20
  3.2 Business Requirements ..................................................................................................... 20
    3.2.1 Business Description .................................................................................................. 20
    3.2.2 Use Case View .......................................................................................................... 21
  3.3 Functional requirements .................................................................................................... 23
    3.3.1 User management ..................................................................................................... 23
    3.3.2 Course management ................................................................................................. 24
    3.3.3 Application management ......................................................................................... 25
5.1 Development environment

5.1.2 Deployment environment

5.2 Implementation of three-layers architecture based on SSH

5.2.1 Data persistence layer realized by Hibernate

5.2.2 Business logic layer realized by Spring

5.2.3 Presentation layer realized by Struts

5.3 Implementation of function module

5.3.1 User interactive interface of Z Education Cloud

5.3.2 Implementation of mainframe resources management module

5.4 Security policy

5.5 Chapter summary

Chapter 6 Web Front-end System Testing of Mainframe Education Cloud

6.1 System performance test

6.2 Black box testing of application management module

6.2.1 Black box testing

6.2.2 System functional test

6.3 Chapter summary

6.4 Security

6.5 Chapter summary

Chapter 7 Conclusion and Prospect

7.1 Conclusion

7.2 Prospect

7.3 Chapter summary

References
Chapter 1 Introduction

In the context of growing application of mainframes to various domains, there is an increasing demand for employees with a high degree of mainframe expertise in the job market, highlighting the great importance of producing mainframe talents for universities. Due to the high cost and difficult maintenance of mainframes, colleges can enormously improve the mode and efficiency of mainframe teaching by establishing a Web-based mainframe education platform and sharing mainframe resources.

1.1 Origin of My Work

The work in this text originates from Z Education Cloud, a project about mainframe teaching cloud platform collaboratively designed and developed by the college of the author and IBM. Taking into account teaching and management requirements of mainframe teaching at universities, my work is focused on rapidly implementing the mainframe environment, sharing high-quality teaching resources as well as better monitoring and managing the teaching environment.

1.2 Background and Significance

The supply and demand pertaining to computer professionals nowadays is very asymmetrical. On one hand, many college graduates cannot find a job that matches with their subjects. On the other hand, many employers complain of the difficulty in finding proper talents. The main reason is that the set-up of majors and curriculums at universities does not match with the market, thereby failing to produce talents needed by the market. Therefore, the majors offered at universities should be adapted dynamically to the market needs, i.e. creating new majors against the background of social development and changes of needs as well as updating or even eliminating old majors.

Mainframes find widespread uses in large financial, insurance, and securities organizations. Thanks to its great reliability, applicability and security, it gains dominance over various servers. Meanwhile, it is a large-scale data center and the chief choice of platform for energy conservation, emission reduction and green computing. Mainframes are finding favor with more and more enterprises with the growing size of enterprises and extension of application systems. But until now, the talents with a high degree of expertise in mainframe are lacking, especially in China, where there are almost no professionals who specialize in mainframe. In this context, mainframe majors begin to be offered in universities.

The university education about mainframe still has the following problems:

(1) For universities that have mainframes
   ➢ Mainframe experts need 2-3 days or even longer to implement a new mainframe environment. This entails enormous workloads and impedes teaching.
   ➢ Due to objective factors, it is difficult to update the existing system. The use of the old system will prevent many new techniques from being taught.
(2) For universities that have no mainframes

- Because the mainframe resources are shared among many universities, a DB2 or CICC is sometimes used by more than one university, thereby causing serious interference.
- Teaching is handicapped by the delay of several working days before the mainframe teaching resources are available at the request of the university.
- Teaching is impeded by low system applicability, because the operating authority is usually found at class to be so low that consultation has to be made with the mainframe provider.

(3) For all universities

- Only theoretical courseware is used by teachers in the High Quality Courses collaborated offered by Ministry of Education and IBM Course. Unfamiliarity with the mainframe environment prevents the experiments in the High Quality Courses from being implemented and used quickly. Consequently, the students are inept at actually operating mainframe, and many high qualities teaching resources cannot be shared among many colleges.
- Due to limited skills among teachers, the system problems (non-hardware problems) cannot be fixed, thereby hindering teaching.
- A platform that facilitates exchanges between teachers and students is lacking.

Therefore, to eliminate the bottleneck in the mainframe education, and spread the mainframe expertise more effectively, it is very important to establish a Z/OS-based mainframe education platform. We hope that Z Education Cloud can solve the above problems and achieve the following purposes:

- Prompt the mainframe education environment to be implemented among colleges at the expense of less time, human and financial and material resources
- Enable colleges to use the state-of-the-art platform and teach latest mainframe techniques.
- Enable each course to have a separate platform, including DB2, CICS, MQ and WAS so that a course can be offered without being interfered by other courses.
- Automatically implement the High Quality Courses experiments and Cases, share good teaching resources.
- Grant authorizations needed by all courses.
- Monitor and manage the teaching environment

In addition, the rapid development in system development, coupled with the technical support from IBM, puts us in a good position to develop and apply the mainframe teaching cloud platform.

1.3 Research and Development Status across the World

1.3.1 Development of large-scale mainframe education platforms

Several leading universities in China have begun to offer the courses on mainframe since 1997. Until now, there are 8 universities offering mainframe courses, some of which establish the
mainframe major, making China one of the few countries in the world that offer mainframe majors at universities.

Software College, Tongji University obtained a Z900 mainframe in March, 2005, and was granted to use it free of charge for the first period of 5 years. IBM Mainframe Teaching System Center was established in Shanghai. After six years of development, the cooperation between Tongji University and IBM has produced remarkable results in terms of discipline construction, curriculum construction, international cooperation, high quality course development, internship & employment and mainframe technology application contest. IBM Mainframe Teaching System Center makes full use of talent pools at the university and IBM technological resources, adopts the teaching mode that integrates theory with practice, and has become one of the important bases for producing IBM mainframe talents in China. As one of the first batch of universities that participate in the IBM mainframe cooperative project, Tongji University has been studying the issues pertaining to mainframe curriculum development. However, mainframe teaching has many problems; for example, it is difficult to initiate mainframe teaching, the teaching resources are insufficient, and the mainframe expertise is very theoretical. Thus, the existing teaching mode can be substantially improved by constructing a resource-sharing platform and spreading mainframe expertise among more students.

1.3.2 Web Front-End Implementation Research Status around the World

In terms of technical implementation, the educational cloud platform has been developed (i.e. Web application-based development) in many practical applications. There are currently many technologies for implementing Web applications in the world, the most common and typical examples of which are the .NET platform and the J2EE platform.

The .NET platform refer to the software components that can used to develop the Web service applications and Windows desktop applications, including the establishment of the tools and basic frameworks for the .NET services and device software.

J2EE is a standard architecture which is mainly employed to develop and implement Web-based enterprise applications with Java.

Similarly, both platforms have three-layer or multi-layer architectures: at the client, both .NET and J2EE provide the development kit based on Web browser application; at the middle layer, .NET components rely on the novel COM+ service, but EJB of J2EE is a well-established large-scale enterprise component framework; at the back-end data layer, both platforms offer a customized data access model for connection with the database. The main difference between them lies in portability and openness. Platform independence is the major feature of J2EE. In fact, J2EE is almost the only choice to develop a platform that can operate on several OS (e.g. UNIX and Windows). In terms of openness, J2EE is a series of open specifications instead of being products, enabling any product that complies with these specifications to be J2EE compatible. Many vendors (e.g. BEA, IBM and Oracle) have developed J2EE-compliant products. This text is focused on J2EE-based Web design and development.

Unfortunately, EJB has flaws in it. It is a heavy framework that pursues perfection during its
design. Consequently, the model is very complex and difficult to learn. As a persistence mechanism, the entity bean attempt is a total failure. To approach these problems, the non-EJB light framework achieves popularity in J2EE, with Struts, Spring and Hibernate being the typical examples. Thanks to the continuous investigation and practice of numerous Web application developers, the combination of these three frameworks (i.e. the SSH framework) has become the most widely used framework for developing Web applications in J2EE. This text will use this framework for Web front-end development of the mainframe teaching cloud platform, and closely investigate the mainframe-based Web technologies in order to develop Web innovations.

1.4 Contents and Contributions

1.4.1 Contents

Basic principles of Web service end is learned, core Java techniques are analyzed, and the Web front end of the mainframe education platform is designed and developed according to the requirements of the Z Education Cloud project. All business processes are followed via the proposed Web front end, including common user management, teacher management and cloud administrator management. Details of the work in this text are given below.

(1) Design of the system roles and HCI
   Design of roles and authorizations as well as GUI in the platform is studied for high ease of use.

(2) Design and implementation of functional classes
   Each functional module is partitioned and implemented according to design classes of business processes. Functional modules include user management, course management, cloud platform resource management and order form management.

(3) Design and implementation of system data
   The DB2 relation database is used to store system data and design data tables for physical implementation.

(4) Mechanism for exchange between Web front end and mainframe back end
   Many schemes for exchange between Web front end and mainframe back end are compared, such as Web Service, FTP and CTG. The most appropriate method is chosen for exchange between front end and back end.

1.4.2 Contributions

Although mainframe resource services are currently available in the market, many constraints are imposed on users when they request or use the resources due to the lack of professional and perfect management schemes or other reasons. Furthermore, the mainframe resources are underutilized. The proposed platform enables mainframe education platform resources overall to be allocated and managed more comprehensively. Contributions are as follows.

(1) A cloud-based rapid, simple, secure and flexible service is offered by integrating
advantages of mainframe and cloud computing. This service enables users to customize services based on their needs, saves hardware investment and maintenance costs, and makes full use of existing mainframe resources.

(2) The system is modularized for great openness and scalability, allowing future functional modules to be modified and extended.

(3) A scheme for flexibly managing user authorizations is designed according to system needs. User, role, data range, function and authorization are interrelated in this scheme.

(4) Complicated businesses in the mainframe teaching cloud platform can be implemented automatically by the system. For example, the mainframe environment can be implemented automatically.

(5) Asynchronous communication scheme is used for interaction between front and back ends, enabling the back end mainframe to create and allocate resources in a unified manner. Mainframe resources can also be managed more effectively.

1.5 Organization of This Text

This text is partitioned into seven chapters.

Chapter 1 provides the introduction. The origin of this text, research background and significance, research and development status in the world, contents and contributions are discussed.

Chapter 2 descripts technologies related to the mainframe cloud platform. The project architecture, design mode, and the selection of development platform and database tool are described. The asynchronous scheme for communication between front and back ends as well as mainframe platform-relevant technologies are also discussed, forming the technical foundation for work in this text.

Chapter 3 analyzes the Web front end requirements of the mainframe cloud platform. First, an overview of business needs is given. Next, the functional requirements of the proposed system are described. Finally, non-functional requirements and operation modes are also depicted.

Chapter 4 provides the design of the Web front end for the mainframe cloud platform. Taking into account existing business requirements, system architecture, user authorization management scheme, and object model is designed to facilitate data sharing, communication and service delivery as well as efficient operation. The mainframe resource management module and the request management module that are developed by the author of this text are used as the example to elaborate on the design scheme.

Chapter 5 implements the Web front end of the mainframe teaching cloud platform. The environmental configurations, SSH architecture and the interface implementation are discussed, with the focus being on the implementations of major system modules. Some security strategies adopted by the proposed system are also described.

Chapter 6 tests the Web front end of the mainframe teaching cloud platform. A brief account of system performance and functional tests is given.

Chapter 7 discusses conclusions and future work. Research achievements and problems in this text are summarized. The issues that will be the focus of future work and the major tasks are specified.
Chapter 2 Description of Related Technologies

2.1 Architecture and Selection

Client/Server (C/S) and Browser/Server (B/S) architectures are the prevailing development modes nowadays. An analysis of these two architectures is given below.

2.1.1 C/S architecture

Fig. 2.1 illustrates the C/S architecture, where tasks are properly allocated to the client and server to reduce system communication overheads and make full use of the hardware environment at both ends. But this architecture is inflexible, inextensible and difficult to maintain and manage. So it is usually applied to small-scale local area networks.

![C/S architecture diagram](image)

Fig. 2.1 C/S architecture

### 2.1.2 B/S architecture

Fig. 2.2 shows the B/S architecture. In this architecture, users send HTTP requests to network servers via browsers. The servers will respond to the requests from browsers and return the information needed by users to the browsers.
Over C/S, B/S has the following advantages:

- The B/S standards are open, universal and cross-platform.
- It suffices to install the universal browser at the client, because maintenance and updating take place at the server. So, the client does not need to change at all, enormously reducing development and maintenance costs.
- The user interface only appears at the browser, is friendly to users and easy to use, without the need to learn how to use it like the applications.
- A separate layer of Web server is added between the client and the database server to prevent the client from directly manipulating the database, resulting in greater system security.

Because the users of the Web-based mainframe teaching cloud platform are widely dispersed across a large area and the system demand for data volume is small, the B/S architecture is adopted by the proposed system, taking into account its advantages above and the actual requirements of users in this project.

### 2.2 MVC Design Mode

Model-View-Controller (MVC) is a widely used design mode for Web project development. It can separate the input layer, business processing layer and control flow in a view-model-controller manner. These three layers are independent of each other but can also work collaboratively.

#### 2.2.1 MVC design mode

In the MVC mode, different tasks are allocated to the view layer, model layer and the control layer. Fig. 2.3 shows the structure of the MVC mode, describing the functions of these three components and their interrelations.
Fig. 2.3 MVC structure

(1) View is the interface that the user sees and can interact with the user. View can display data to users and receive data input by users, but it is incapable of perform business processing. View can inquire of the model about business status but it cannot change the model. It can also receive the data updating event from the model to update the user interface synchronously.

(2) Model is mainly responsible for processing business flows and changing business status. It needs to inform View of data changes.

(3) Controller is mainly responsible for instructing Model to process businesses and View to return the results needed by the users at the request of users. Instead of processing data, Controller only sends user information to Model, select proper View and returns it to the users. Therefore, a Model can correspond to more than one View, and one View can correspond to more than one model.

To sum up, the advent of MVC separates the functional module from the display module, making the program more readable, maintainable and scalable.

2.3 Development Platform and SSH Development Framework

As described in Chapter 1.3.2, J2EE and SSH are chosen to develop the Web front end of the mainframe teaching cloud platform. Functions and features of each framework in SSH (Struts+Spring+Hibernate) are discussed below.

2.3.1 Struts

Struts itself is a MVC framework. Its View is a group of JSP files, without involving any business logic codes and model information. No specific model component is provided in Struts. For enterprise application development, business logics are usually implemented via JavaBean and
EJB. As the core of Struts, Controller is implemented by ActionServlet and Action. ActionServlet plays a scheduling role in Struts, receiving user requests and allocating them to proper Action classes for processing. Action calls the methods from Model to update the status of Model and control program flows. On completion of the task by Action, ActionServlet chooses a View according to the values returned by Action to display results.

For the configuration file Struts-config.xml: user requests are processed and forwarded by ActionServlet, and it specifies which Action class ActionServlet should forward the request to.

2.3.2 Spring

The core of Spring is a lightweight container. It is the framework that implements IoC container and non-invasiveness, providing support for persistence layer and events. It also provides the implementation of the MVC Web framework and is a well-rounded application framework. Spring offers schemes for integrating with other existing frameworks (Struts and Hibernate, etc.).

2.3.3 Hibernate

Hibernate is a Java environment-oriented object-relation database mapping tool. It is capable of mapping the object represented by the object model into the SQL-based relation database structure in order to achieve data persistence. It offers methods for querying and obtaining data.

2.4 Database Tools and Selection

The mainstream database development tools nowadays include Access, SQL Server, Oracle and MySQL.

- Microsoft Access is one of the Office automation software suites. As a medium- and small-sized database management system, it is characterized by outstanding analysis ability, widespread application and high ease of use. However, it is a single user system, does not support simultaneous multi-user access and can only run on Windows.

- SQL Server is currently a very popular relation database. While inheriting advantages of Back Office series, it is appropriately extended to win popularity among the public due to its graphical interface, various programming interface tools, dramatic scalability and the support for large Web sites.

- Oracle database comes from Oracle Corporation, and it is characterized by high compatibility, portability, connectivity and versatility.

- MySQL is a relation database management system from the Swedish MySQL AB Corporation which now belongs to Oracle. MySQL is a relational database management system that stores data into different tables rather than a large warehouse, resulting in higher speed and flexibility. The SQL language from MySQL is the most common standard language for accessing the database. Generally, medium- and small-scale websites choose MySQL as their database, due to its small size, fast speed, low TCO.
and especially openness of source codes.

From above analysis, it can be observed that MySQL is a good fit for Web-based mainframe teaching cloud platform. Meanwhile, it entails low development cost, short development period and is easy to maintain. The mainframe education platform is not a system that requires a super large database in the near future. So, MySQL is chosen in this text.

### 2.5 Asynchronous Message Mechanism for Communication between Front and Back Ends

In the context of advent and popularity of Web 2.0, various rich client techniques are widely used. The asynchronous communication scheme enables interaction between front end client and back end server to take place at the background, sparing users the pain of refreshing pages and waiting long for server responses. Therefore, user requests can be responded more quickly and the pressure on servers are dramatically alleviated.

In asynchronous communication, data transfer between the sender and the receiver needs no public clock signal; instead, the communication between the sender and the receiver is achieved via asynchronous response. That is, once the data is ready to be sent, then the sender can send the data promptly. Once it is ready, then the receiver can receive the data when it arrives. Asynchronous communication is suited for the situation where not much data needs to be transmitted usually and the time when the response will be made by the other party is unknown.

The message-based asynchronous communication mechanism is widely used. In this mechanism, a user message is defined, the caller initiates the message processing thread, and the called party sends message to the caller as soon as the results are produced. The message processing thread keeps checking whether the called party returns the message. If it does, then it is up to the caller to process. Doing this allows the caller to continue with other codes after sending the calling commands instead of being obstructed, resulting in higher efficiency.

For the mainframe teaching cloud platform in this text, the request for mainframe resources from universities, teachers and students can be regarded as the user message. And these messages are random and widely dispersed. In the case of synchronous communication message mechanism, after users succeed in requesting for resources at the Web front end, the mainframe back end needs to promptly create resources and allocate to users. The use of this scheme may overload the server and impedes unified management and allocation of mainframe resources. Thus, the asynchronous communication message mechanism is adopted in this text. Instead of promptly responding to user request for mainframe resources, after being validated by the cloud administrator, the request is submitted to the mainframe administrator for unified management and allocation of mainframe resources.

### 2.6 Mainframe-Related Technologies

An utterly novel idea is proposed for the mainframe cloud educational platform, i.e. constructing the education cloud at the mainframe, enabling the Web to operate the mainframe through the use of the well-established FTP and JES techniques, applying REXX as the scripting
language to dynamically configure mainframe resources.

(1) IBM mainframe
The proposed platform is oriented to mainframe instruction, so there is no doubt that the mainframe will be used. Unlike the traditional applications that use mainframe as the server, the proposed system employs the mainframe to construct the educational cloud platform. Due to its virtualization and high scalability, it is capable of offering numerous instruction resources and serves as a robust educational platform.

(2) FTP technique
The mainframe communicates with the outside environment via the file transfer protocol (FTP). The open platform can use FTP to submit a job to the mainframe platform.

(3) REXX language
Restructured Extended Executor is a programming language from IBM. It is mainly used by mainframe programmers to design programs that can run on mainframe. The high applicability of REXX and IBM mainframe, coupled with the REXX-based implementation of mainframe functions, yields great availability and ease of modification.

(4) Mainframe security management
The RACF management mechanism in the mainframe enables us to define authorization management mechanism based on the needs, guaranteeing data security of the cloud platform.

(5) Mainframe resource configuration
Consider DB2. We provide the mainframe with a program for automatically installing DB2. The idea for automatic installation is to make a copy of the corresponding DB2 module by using the existing DB2 resources, and then completing DB2 installation by creating necessary resources via script. This installation mode can be duplicated. In this context, great flexibility in configuring and managing resources can be achieved as long as the disk volume is sufficient.

2.7 Chapter Summary

This chapter discusses the technologies relevant to the mainframe teaching cloud platform, including the selection of architectures, MVC design mode, the selection of system development platform and framework, database tools, and the asynchronous message mechanism for communication between front and back ends, with the focus on technologies related to the mainframe cloud platform. This forms the technical foundation for development the Web front end of the mainframe teaching cloud platform.
Chapter 3 Web Front End Requirements of the Mainframe Teaching Cloud Platform

Structured analysis is done in this chapter to provide a detailed account of the major business processes of the mainframe teaching cloud platform and the logical models of each subsystem, through the use of flow charts and UML modelling tools.

3.1 Terminology

Several special terminologies are defined in the mainframe teaching cloud platform.
- **Z Education Cloud**: mainframe education cloud platform, i.e. the name of the project to which the proposed Web front end implementation belongs.
- **Pattern**: resource pattern, i.e. the definition of the mixture of hardware, operating system and middleware, specifying a certain type of mainframe resources (e.g. DB2, CICS, and ACCOUNT) and its version number.
- **Pattern Instance**: resource pattern instance, i.e. the instance of the mixture of hardware, operating system and middleware, or an instance of Pattern.
- **Application**: a request of the teacher for the course, which can involve multiple courses, multiple teacher and student accounts. It also corresponds to the class and the order in the database below.
- **Courseware**: a complete set of courseware for a course, including Lecture, Lab, Study Case, Homework/Test and Instruction.
- **Course Code**: the codes of courseware (e.g. Lab and Study Case) that need to be implemented on the mainframe for a course.
- **Resource Pool**: mainframe resource buffer pool, which can include DB2, CICS, Storage Group, teacher accounts, student accounts and other resources.
- **Package**: Mainframe resources are offered to the users with Package being the unit to achieve unified allocation and management of mainframe resources. A Package consists of a CISC, two DB2s, as well as 100 teacher and student accounts.
- **Organization**: the organization that defines a shared mainframe education platform. It can be a school or company.

3.2 Business Requirements

3.2.1 Business Description

The mainframe education cloud platform is designed for four types of users: students, teachers, cloud administrators, and mainframe administrators. Their business requirements are described below:

(1) For students, they can log on the Web front end of the mainframe education cloud platform to download courseware (non-instant response) that is helpful in understanding contents of courses. They can also apply for a mini account in mini pattern instance to do the homework.
This account is valid for a month. The resources will be deleted directly when the account expires. Students are allowed to visit the history page to view operations like the applications made in the past.

(2) For teachers, they can log on the Web front end of the mainframe education cloud platform to download courseware (non-instant response) that is helpful in understanding contents of courses. They can request for mainframe resources, course pattern or resource pattern, based on their needs. They are also allowed to apply multiple teacher and student accounts. While requesting for resources, teachers need to specify the starting and ending time for the use of the resources, so that the mainframe administrators can create and allocate mainframe resources in advance. The resources will not be deleted until a week after the period of validity ends. If the school or the organization is the VIP of the cloud platform who owns separate mainframe storage space, then the teachers can choose their own separate disk space. Through the Web end, teachers can view the progress of the request and the previous operations. In the case of a resource expiring, teachers can renew it for further use.

(3) For cloud administrators, they can log on the Web end to manage courses and requests, reclaim resources, view information, and issue warnings about the shortage of resources. By managing courses, the mainframe administrators can create, update and edit courses as well as share course resources within the platform. The cloud administrators can view, modify, approve or reject applications of teachers or students. On the expiration of an application, the cloud administrators can directly reclaim its resources. If all resources in the resource pool are being used, a warning on resource shortage will be issued to promptly inform the mainframe end of the need to expand resources. Furthermore, the cloud administrators can view information within the system, such as the resource applications being processed by the system and their status, as well as the statistical information on resources allocated to individuals, teachers or schools.

(4) By logging on the mainframe via Web front-end ISPF, the mainframe administrators can manage users and mainframe resources, as well as compute the price. By managing users, the mainframe administrators can directly create, modify, delete or authorize common users. Management of mainframe resources involves querying allocation of mainframe resources, creating, updating, and deleting resources. The created resources can be DB2, CICS, teacher accounts, student accounts, mainframe DASD storage group, and LAB codes on the mainframe. As for computing the price, it means that while the users use the mainframe education cloud platform, the prices for their use of the mainframe resources need to be computed. This function is implemented by the mainframe administrator end.

3.2.2 Use Case View

Use case view defines what functions the system should provide, and describe what problems the system should help users to solve, i.e. who uses which case, in order to model system functions. This is also known as requirement modelling. The use case diagram consists of such model elements as system, role and case; it also needs to specify the relations between elements, between role and case, as well as between roles.

The use case view of the mainframe education cloud is mainly responsible for describing the relation between role and case. In the system, there are four roles: student, teacher, cloud
administrator, and mainframe administrator, each of which has their own case. The use case view of the proposed system is shown in Fig. 3.1.
3.3 Functional requirements

According to the analysis on the mainframe education cloud platform, Z Education Cloud is partitioned into six functional modules: user management, course management, application management, resource management, charge, view and report. The general structure of the system functions is shown in Fig. 3.2.

Fig. 3.2 Z Education Cloud system functions

3.3.1 User management

(1) Functional description:

The user management module is mainly responsible for registering users, editing and querying user profile, as well as allocating role to each user. When a user logs on, the system will identify his or her role with the username, and display the webpage corresponding to the authorization of this role, meaning that the user can only view webpages under his or her authority. Teachers are allowed to view course application information and status. Cloud administrators can add users, determine user rights and query users.

(2) Use case diagram of this module is shown in Fig. 3.3.
3.3.2 Course management

(1) Functional description

The course management module allows operations on courses. For example, students and teachers can view course description and download courseware. The cloud administrator can manage these courses, maintain course information, upload or download courseware. Furthermore, the modes of some courses can be set up such that users can apply for courses more easily.

(2) Use case diagram of this module is shown in Fig. 3.4.
3.3.3 Application management

(1) Functional description:

Application management involves creating applications, updating applications, restore expired applications, cancel applications, approve and reject applications of other users. Different roles have different rights over application management. In the proposed system, the applications are made for resources or teacher rights. While the ordinary user can apply for the role as a teacher, the teacher is allowed to apply for mainframe resources via the Web in the mainframe platform. All users can view the records and status of their applications. School Administrators can view not only the records of their applications but also records and status of all applications of the school for resources. Furthermore, school administrators have the right to approve or reject users’ applications.

(2) Use case diagram of this module is shown in Fig. 3.5.
3.3.4 Resource management

(1) Functional description:

The resource management module provides the following sub-functions: adding resources, deleting resources, allocating resources, reclaiming resources, issue warnings on resource shortage, and view resources. In the proposed system, resources are managed by cloud administrators and mainframe administrators. The cloud administrators can view the use of resources in the resource pool. If all existing resources are occupied, a delay may be caused in responding to new user applications. In this case, the cloud administrators need to issue warnings on resource shortage to inform the mainframe of the need for resource pool expansion. The mainframe administrator can create new resources on user application, like DB2, CICS, teacher account, student account, mainframe DASD storage group, and the LAB codes in the mainframe. The mainframe administrators are also entitled to allocate, reclaim and delete resources.

(2) Use case diagram of this module is shown in Fig. 3.6.
3.3.5 Charge

(1) Functional description

The charge module has two sub-functions: computing the price and making the payment. Z Education Cloud is not free. The price is computed taking into account the types and numbers of resources requested by users. The price per user account per 1M disk space is stored in the price configuration file. Currently, the price is computed as:

$$ F = \sum (R \times P) $$  \hspace{1cm} (3.1)

where $F$ denotes the overall price for the platform service, $R$ denotes the number of a specific resource; $P$ denotes the price of this resource. This equation shows that the overall price for the platform service is equal to the sum of the price of each requested resource in the school.

(2) Use case diagram of this module is shown in Fig. 3.7.
3.3.6 View and report

(1) Functional description:

View and report is a very important function, because it enables users to understand the use of mainframe resources by their schools. This is really essential for administrators to maintain the system, because it determines when the administrators should create new resources. Furthermore, the proportion of occupied resources is a consideration that should be taken into account when the school’s administrators decide whether to accept user applications.

(2) Use case diagram of this module is shown in Fig. 3.8.

![Use case diagram of the view and report module](image)

3.4 Non-functional Requirements

(1) System structure: B/S structure, MVC design mode, J2EE development environment under the SSH framework, and MySQL as the background database.

(2) Performance requirements:

- Response time: during interaction between each of the subsystem and users, the system must promptly respond to each user application, ensuring that the response time is no more than 0.5s.
- System reliability and stability: the system needs to ensure its reliability, stability and data security in the case of high concurrency.
- Ease of use of the system: the system should encapsulate complex operations as many as possible, enabling users to operate easily and with express purposes.
- System scalability: the system should be highly cohesive and loosely coupled so that it can be easily extended to accommodate future changes and needs of the cloud platform.

(4) Failure processing requirements: the system can inform users of any software and hardware failures that occur. Error logs can be created to help quickly locate the errors.

(5) Operation requirements: the system can be operated in ways similar to common Windows operating modes in order to facilitate friendly interaction with users.

3.5 Run Mode

Users of Z Education Cloud are classified into three types: students, teachers and
administrators. The client can access the cloud platform using the ordinary browser and the mainframe-connected PCOM. This is the so-called thin client. The mainframe education cloud provides clients with services they need by using the following three basic modules: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS). Users can apply for any resource with the PC browser and PCOM connected to the mainframe. After applying for the mainframe resources, the client can log on the mainframe system via PCOM for background operation. The resource allocation and mainframe environment installation can be done automatically in the cloud. Logical topology of the run mode for the proposed system is shown in Fig. 3.9.

![Z Education Cloud logical topology](image)

3.6 Chapter Summary

By following the object-oriented analysis method and the UML modelling concept, the businesses of the Z Education Cloud system are analyzed; four user roles of the proposed system are determined: teachers, students, cloud administrators and mainframe administrators. Use cases are allocated to each role based on their business requirements. The requirements of user management, course management, application management, resource management, charge, as well as view and report modules are specified. Furthermore, non-functional requirements (system structure, performance requirements, failure processing requirements, and operating requirements) and run mode during system development and application are analyzed. Requirement analysis forms the foundation for system design below.
Chapter 4 Web Front-End Design of the Mainframe Education Cloud Platform

4.1 System design principles

(1) Principles for optimizing performance overall

Performance optimization principle is one of the set of important principles for system design. Not only the performance of a single machine, but also the overall performance of the entire system should be taken into account during performance optimization.

Performance considerations involve memory management, database management, load balance, system interface and key algorithms.

(2) Scalability

Being upgraded is an instinct nature of the system. The system should enable software to be upgraded and increase its capacity without modifications.

(3) Reliability

To achieve high reliability, the system structure should be designed to ensure that the system can run reliability and the system data can be restored reliability in the case of system failure.

(4) Maintainability

System maintainability is closely related to architecture design. Workloads associated with development and maintenance can be enormously reduced if bugs are easily and accurately located during system development, debugging and maintenance. Therefore, system design should be done following the maintainability principle.

(6) Reusability

Reusability means reusing what has existed. It is helpful in improving product quality, enhancing production efficiency and reducing costs.

4.2 System Design Scheme and Structure

To implement functions of Z Education Cloud, the currently well-established three-layer framework combination of Struts + Spring + Hibernate that is prevailing among mainstream Web applications is used as the system development framework.

Struts, Spring and Hibernate have their respective functions. According to functional layers of Web applications, their applications can be applied to different layers to implement functions of these layers. Doing these integrates functions of the frameworks. Its inheritance strategy in the project is given in Fig. 4.1.
The working flow of the integrated SSH framework is shown in Fig. 4.2.

Understanding of the principles of SSH-based combined framework can be deepened with Figures 4.1 and 4.2. It can be observed that the Web application layer has three functional layers; Struts, Spring and Hibernate are at the presentation layer, business logic layer, and the data persistence layer, respectively.

Presentation layer: this layer is responsible for managing user applications, responding to user applications and displaying response to users. Struts is capable of processing user applications at the presentation layer, proving a controller that can invoke lower-layer business
logic modules, processing exceptions and validating user inputs.

Business logic layer: this layer is responsible for implementing the system’s business logics. It also enables Spring to process business logics at this layer for applications, check businesses, manage transactions, manage interdependence between objects in the business layer, and implement business services by executing corresponding business programs.

Persistence layer: this layer is responsible for system access to database. At this layer, Hibernate provides Java with the object-oriented persistence mechanism and query service. The object-oriented methods can be used to store, update, delete database records and query detailed data records.

4.2.2 System architecture design

Because the SSH combined framework is chosen for architecture design of the mainframe education cloud platform, the proposed system follows the hierarchical design based on the multi-layer architecture of J2EE.

The layered structure implemented in Z Education Cloud is shown in Fig. 4.3.

As shown in this Fig., the system has four layers from top to bottom:

(1) User layer: this layer runs in the user browser to display the user interface for interaction with the user.

(2) Presentation logic layer: this layer can be regarded as a Web layer, and operates in the Web container of J2EE. It receives the user input from the user layer, sends it to the components in the business logic layer, and then forms the user interface by outputting the results from the
business logic layer to the client. The entire presentation logic layer relies on Struts and mainly consists of JSP and Servlet.

(3) Business logic layer: this layer process the user input from the presentation logic layer, and returns the results to the presentation logic layer. This layer is mainly responsible for defining the logic implementation of the core logics for each sub-system, i.e. the provided service (e.g. the implementation of the data acquisition interface). It needs to manipulate the database via the data access layer. The loading of these business logics is managed by the Spring container. The interaction between each sub-system is associated with each other through the interface service layer and the injection mode.

(4) Data persistence layer: this layer is responsible for mapping Java persistence objects to the table via Hibernate, adding, deleting, modifying and querying objects.

4.3 User right management mechanism

User management and right management forms the foundation for normal operation of the system, and is the prerequisite for the ability of the system to provide accurate information services according to different business scopes. Uses of the proposed system include students, teachers, cloud administrators and cloud administrators. Different types of users have different rights over system functions and data ranges, depending on their functions. Therefore, how to provide accurate right management for different users is an important aspect of user right management. Basic concepts in the proposed system are shown in Table 4.1.

<table>
<thead>
<tr>
<th>Concepts</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User</td>
<td>User can access system subject independently, is created by the system via application interface and labelled by a unique ID.</td>
</tr>
<tr>
<td>Role</td>
<td>Role is a set of users, meaning that the users in this role share the same business cope. Each role is labelled by a unique serial number</td>
</tr>
<tr>
<td>Right</td>
<td>Right is the permission granted to the user to use a particular system function. It is the most basic unit of authorization granted by the system to the user.</td>
</tr>
<tr>
<td>Permission level</td>
<td>Permissions of the proposed system are classified into five levels: 1. read, 2. modify, 3. add, 4. delete and 5. execute. Level 1 means the permission to read; level 2 means the permission to read and modify, and so on. The permission levels of the proposed system have been specified when administrators allocate different roles to users. These levels indicate at what levels the users can operate all functional modules authorized in their roles.</td>
</tr>
<tr>
<td>Permission allocation</td>
<td>Permission allocation is a binary relation between role and permission.</td>
</tr>
</tbody>
</table>
Role and permission can be used together to indicate whether a role is permitted to access a particular function. Meanwhile, the relation between role and permission is a many-many relation, because a role is allowed to have many permissions and permission can be allocated too many roles.

Data range

Data range means the operating range of the basic data (e.g. the data of teachers and students in the mainframe education platform, and the data of teachers and students in a particular school). Data range is specified by the system based on the role.

4.3.1 System roles and data range

Taking into account characteristics of users, the mainframe education cloud platform defines five roles: ordinary user, student, teacher, cloud administrator, mainframe administrator.

When a new user is added to the system, the system administrator needs to allocate a role to this user. A user is allowed to have multiple roles. On allocating the role, the administrator will specify the permission level of the user role. Therefore, for permission allocation later on, the right of users over the functional modules is dependent on the permission level allocated to the role when the user is created. Role allocation and corresponding data range is shown in Table 4.2.

<table>
<thead>
<tr>
<th>Serial number</th>
<th>Roles</th>
<th>Data ranges</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Mainframe administrator</td>
<td>All user data and mainframe resources in Z Education Cloud</td>
</tr>
<tr>
<td>2</td>
<td>Cloud administrator</td>
<td>All user data and course resources in Z Education Cloud</td>
</tr>
<tr>
<td>3</td>
<td>Teacher</td>
<td>Courses managed by teachers and student data in Z Education Cloud</td>
</tr>
<tr>
<td>4</td>
<td>Student</td>
<td>Profile and corresponding course resources</td>
</tr>
<tr>
<td>5</td>
<td>Ordinary user</td>
<td>Users who have not been allocated a role</td>
</tr>
</tbody>
</table>

Note: Ordinary users refer to the newly registered users who have not been allocated a role by the administrator. They can manage their profile; view the public information in Z Education Cloud.

4.3.2 System rights management mechanism

Permission allocation is a one-to-one binary relation between role and permission, which can
indicate whether a role is permitted to access a module. The allocation of roles only determine whether the role has right over a particular functional module, and its corresponding right is dependent on the default permission level specified when the system administrator allocates a role to the user. Therefore, the right of a user over a particular functional module relies on the role of the user in the system, the corresponding data range; the permission level specified when the user role is determined, and the correspondence between role and right. That is, the relation between the role and the allocated right determines whether the user has the right to a particular functional module. The permission level that is specified when the role is allocated to the user determines at what level the user is allowed to operate the module. The data range specified by the user role determines the range of data that can be accessed by the user in the functional module.

The relation between the user management mechanism and the rights allocation mechanism is shown below (Fig. 4.4).

![Fig. 4.4 User management and rights allocation mechanism](image)

From this figure, it can be observed that the top layer represents the user, the middle layer represents the role, and the bottom layer represents the functional module. The line connecting user and role denotes the relation between user and role, and the number on the line denotes the permission level. The line connecting role and module denotes the rights allocation relation between user and functional module. A user can belong to many roles simultaneously, and each role has access to different functional modules accordingly. The permission level determines the level of rights of the role over the functional module. For example, role 5 has access to module 5, users 3 and 5 in role 5 have access to module 5, user 3 has a permission level of 2 over module 5, and user 5 has a permission level of 3 over module 5. Furthermore, when a user belongs to two roles, if the permission levels over the same functional module are different, then the highest permission level is chosen by default. For example, user 2 belongs to roles 1 and 4 and has rights over module 2, the permission levels being 5 and 1, respectively. Hence, the right of this user over module 2 is level 5, i.e. the highest of both levels. Note that the permission level of the user over the module is determined by default when the user is assigned a role. But after the rights are granted to the role, the administrator can modify default permissions, grant permission to or withdraw permission from the user.
4.4 Design of object module

This section is focused on the description of the principles for designing classes in the object model. The inter-class relation is determined by extracting and analyzing classes after instantiating the system use case model. UML is used again here to design the system object model.

4.4.1 Principles for class design

The class should be designed by following the principles below:

(1) Open closed principle (OCP)

A module should be open for scalability and closed for alterability. Therefore, the encapsulation mechanism, the abstraction mechanism, and polymorphism should be fully taken into account during object-oriented design. As one of the important principles for software engineering, this principle is also applicable to non-object-oriented design.

(2) Liskov substitution principle (LSP)

The subclass can substitute for the parent and appear at any place where the parent may be. This principle was proposed by Liskov in 1987 and can be inferred from the Design by Contract (DBC) concept from Bertrand Meyer.

(3) Dependency inversion principle (DIP)

On designing businesses, the dependency of specific businesses should rely on interface and abstract classes rather than specific classes. The specific classes are only responsible for implementing relevant businesses so that the specific classes can be modified without affecting dependency of specific businesses.

It can be observed that in structured design, the underlying module is an implementation of the upper-layer abstract module, meaning that the abstract module is dependent on other module that implements it. Therefore, changes of the implementation in the underlying module will severely affect the upper-layer abstract module. Obviously, this is a major drawback of the structured methodology.

(4) Interface segregation principle (ISP)

Using multiple interfaces relevant to a specific client class is preferable to adopting a universal class that covers many business methods. ISP is another enabling technique that supports componentization (e.g. COM). Applicability and portability of components and classes will be seriously compromised without ISP.

4.4.2 MVC-based object model design

In Z Education Cloud, to enable users and administrators to analyze and use mainframe resources more easily, the same business data usually needs to be displayed in multiple views. For example, a table object and a columnar object can represent the message of the same application data object in different ways. The table object and the columnar object are transparent to each other, allowing users to separately reuse the table or histogram based on their needs. The histogram can immediately represent the changes that users make to the information in the table,
meaning that both the table and the histogram are dependent on data object. Early design strongly relies on event-driven user interface, resulting in a tangle of data processing codes, program function codes and interface display codes. The MVC mode originated from small talk 80 GUI design in the 1970s, and described the communication mode between objects from different parts. Doing this prevents them from being involved in each other’s data modelling methodology, making the program structure more clear and flexible. 

The three components of the MVC design mode, i.e. Model, View and Controller, correspond to internal data, data representation and input/output control. The role of MVC in the design of system object model for the proposed system is as follows:

(1) Model is a logic abstraction of problem-related data, represents internal properties of the object and is also the core of the entire model. It follows the object-oriented method, and abstracts the object in the problem into the application object. These abstracted objects encapsulate the properties of the objects and the logics hidden in them. Hence, in the proposed system, the object properties and methods encapsulated in each functional module fully represent the component of Model. The design of object models (e.g. course class, order class and resource class) is the design of Model.

(2) View is the visual representation of the Model. A model can correspond to one or more views. It can be classified into ordinary user view, student view, teacher view and administrator view, according to the types of users. View is capable of interacting with the outside world and serves as the interface between the application system and the outside world. On the one hand, it provides the outside world with input means and invokes application logics. On the other hand, it displays the results of application logics to the outside world. Hence, the subclasses (e.g. student, teacher, cloud administrator and mainframe administrator) derived from the user class in the proposed system forms the foundation for the implantation of the design mode where more than one view corresponds to the same model. The system invokes different views for different roles. Some JSP files are used in the system to design various user interfaces.

(3) Controller is the link between model and view. It extracts the outside information sent by view, transforms the interaction between user and view into the application behavior-based standard business events, and then interprets the standard business events into the actions that should be followed by model (e.g. activating business logics or modifying the status of model). Meanwhile, controller informs view of model update and modifications to guarantee consistency of each view with model. In the proposed system, this is mainly represented by the design of the Action class. Request action in the page corresponds to Action. Both setter and getter are responsible for processing and storing data or status, executing business logics, and returning processing results to users. Basic contents of the Action class are as follows:

- Provide basic properties as well as the getter and setter methods of the properties. Basic properties include request parameters and their corresponding information.
- Provide a default Action processing method, i.e. public String execute(), which can be used to process business logics and return the string obtained by processing characters.
- Provide other Action processing methods.
4.4.3 Design of classes in Z Education Cloud

The analysis shows that the major classes in Z Education Cloud are as follows:

User class: personal information is encapsulated in this class and several subclasses are also derived from it: student, teacher, cloud administrator and mainframe administrator.

Resource class: all properties of mainframe resources, including resource names, statuses and logs, are encapsulated in this class.

Order class: all properties of orders (i.e. user application for resources), including order ID, ID of users who request resources, starting and ending time of orders as well as statuses, are encapsulated in this class.

Course class: all properties of mainframe courses, including course ID, name, teacher and school, are encapsulated in this class.

Organization class: properties of schools and organizations in the mainframe education cloud platform, including school ID, name, address, contact information, whether it is VIP and active, are encapsulated in this class.

There are other classes that are not listed here, such as Course_file (courseware) class, Course_lab class (course laboratory), message class (message), order_course class (course order), order_resource class (resource order), and resource_pattern class (resource pattern).

Analysis classes are extracted by instantiating the use case model. The analysis class encompasses the boundary class, control class and entity class. The boundary class is mainly responsible for interacting with users and implementing user interface. The control class is mainly responsible for controlling the process and implementing the processes in the use case. The entity class is responsible for extracting real-world entities based on requirements and representing objects in the system. The extracted analysis class may later on evolve into one or more specific classes, or even a subsystem. Use cases of application management, resource management and course management are taken as an example for illustration.

1) Analysis of use cases of application management and resource management yields the finally extracted analysis classes as follows:

Entity class: teacher class, student class, cloud administrator class, resource class

Boundary class: interfaces that enable application orders to be created updated and approved.

Control class: apply for orders

The class diagram of this use case is shown in Fig. 4.5, which only gives the names and most important properties of classes. Further analysis is required to know the inter-class relation, detailed properties and operations of all classes.
Determine inter-class relation: after determining analysis classes, the use case needs to be analyzed further to understand the inter-class and inter-object relations. Fig. 4.6 illustrates the class diagram of application management and resource management use case, involving two relations (i.e. dependency and correlation).
(2) Analysis of course management use case

The class diagram of this use case is shown in Fig. 4.7, which only gives the names and most important properties of classes. Further analysis is required to know the inter-class relation, detailed properties and operations of all classes.

Determine inter-class relation: after determining analysis classes, the use case needs to be analyzed further to understand the inter-class and inter-object relations. Fig. 4.8 illustrates the class diagram of course management use case, involving two relations (i.e. dependency and correlation).
Although class diagrams are used throughout software model development, they describe abstraction at different layers. At the stage of design, the class diagram describes the inter-class interface. The class diagram can be used to describe the following three models: system vocabulary, straightforward writing model and logic database pattern model.

Fig. 4.9 shows the class diagram of the system. This class can be used for modelling database pattern.

4.4.4 Sequence diagram of Z Education Cloud

In previous subsection, major classes in Z Education Cloud are discussed, and the inter-class
relations are determined via use case analysis. Here, to further specify how the system implements use case via object interaction, this subsection will display major functional sequence diagrams. The interaction between different user classes and system function classes is used as an example for illustration.

(1) Student users

Fig. 4.10 shows the sequence diagram of student users viewing course information and downloading courseware:

![Sequence Diagram of Student Users Viewing Course Information and Downloading Courseware](image)

We can see from the above diagram that the student class first applies to request the course resources. After the system decides that the corresponding role and permission are in line with the requirements, it will return to the course page from course class; Then after the students click to download the courseware, the course class will invoke RequestCourseware() method to find out the corresponding course information in the database, and return the download link to the student class. In this way, the course information querying and downloading is completed, thus realizing the interaction between the user class and the course class.

(2) Teacher users

Fig. 4.11 indicates the sequence diagram of teacher applying for mainframe course resources.
Fig. 4.11 Sequence Diagram of Teacher Users Applying for Mainframe Course Resources

The above figure shows the interaction between teacher class and course application class. As shown in the figure, teacher class sends Request to Main Page to apply for course application page. After the system approves the user role and permission, it will immediately return the course application page to the browser of the front-end users. Then, after the teacher class selects the courses using select course () method and modifies the course information using modify course source () method, a new record of course resource application will be added to the database by the course application class (add course application (course, resource)). After the database is updated successfully, the system will return a message to the course application class.

(3) Cloud administrator

- Fig. 4.12 shows the sequence diagram of cloud administrator inspecting resource utilization.
In the interaction between cloud administrator class and resource class, the cloud administrator first sends Request to Main Page for mainframe resources. After the system approves the role and permission, the system will continue to send Request to response the mainframe resource page (URL). Then selection operation will be carried out for the database (select resource info ()) in order to locate the required mainframe resources. After the database returns the corresponding mainframe resource information, the resource class will return the mainframe resource page that contains this information to the cloud administrator.

- Fig. 4.13 shows the sequence diagram of cloud administrator approving resource application.

Fig. 4.13 Sequence Diagram of Cloud Administrator Approving Resource Application

In Z Education Cloud system, all users (students, teachers etc.) can apply for the mainframe resources and the cloud administrator is responsible for the examination and approval of these applications. After the users succeed in applying for the mainframe resources, the cloud administrator will approve the application in the application management interface. To do so, the cloud administrator first requests application management information and the relevant page. Then the application class will carry out query operation for the database (select application info ()), and return the application management that contains relevant information. Next, the cloud
administrator can click the button “approve” after viewing relevant applications, update the application records in the back-end database by invoking approving method (approve application () through the application class, and return a successful message.

Due to the limited space, the sequence diagrams of interaction among other classes in this system are presented in Appendix 1. The author will not give detailed explanation here. Please refer to Appendix 1 for more information.

The above four sections focus on introducing the design principles, design scheme and architecture of Z Education Cloud, setting of roles and authorization management mechanism, and design of object model. Next the author will explain in detail the implementation of some function modules in this system.

The next section will focus on the introduction and explanation of some major function modules in Z Education Cloud, including mainframe resource management module and application management module.

4.5 Design of mainframe resource management module

The mainframe resource management in this thesis refers to installation, deployment and protection of DB2 for z/OS, CICS and Storage Group resources in the mainframe.

4.5.1 DB2 for z/OS resource management module

(1) Design description:

- Purpose: A certain amount of DB2 for z/OS resources that have been installed and deployed are reserved in the system. These resources can be applied by the VIP users in the course experimental resource application or by the teachers in the course application. After the applications are approved, the system will automatically complete DB2 for z/OS authorization to meet the requirements of the users. When the system is short of DB2 for z/OS resources, the administrator will add new DB2 for z/OS resources to the system.
- Feature: Automatic installation. As long as the administrator provides the relevant information of DB2 for z/OS to be installed, such as the quantity, the system will carry out automatic deployment and protection without manual operation.
- Binding character: Only the cloud administrator has the right to execute the operation.
Functional description:

- Front-end submits a JES to the mainframe through ftp in order to install DB2 for z/OS.
- DB2 for z/OS protective measures: Only those who have permissions can access DB2 for z/OS, while those without permissions will have no access to DB2 for z/OS.

Input items:

Table 4.3 Input Items of DB2 for z/OS Resource Application

<table>
<thead>
<tr>
<th>Information Content</th>
<th>Input Method</th>
<th>Field Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>DB2 for Z/OS Quantity</td>
<td>Front-end Input</td>
<td>Textfield</td>
</tr>
<tr>
<td>Starting Name of DB2 for Z/OS</td>
<td>Front-end Input</td>
<td>Textfield</td>
</tr>
</tbody>
</table>

Output items:

- If the installation is successful, the result will be returned to the front-end.
- If the installation fails, the result as well as the cause of failure will be returned to the front-end.

Algorithm:

- To get the quantity of DB2 for z/OS to be installed and a name of DB2 (Dxxx) from the front-end through ftp, which is taken as the starting name of DB2 for z/OS. The subsequent names are obtained by increasing progressively from 1.
- To call the corresponding REXX program in order to install DB2 for z/OS.

Flow chat (as shown in Fig. 4.14)
Fig. 4.14 Flow Chat of DB2 for z/OS Resource Management Module

4.5.2 CICS resource management module

(1) Design description:

- Purpose: similar to DB2 for z/OS, the system also prepares a certain amount of CICS resources for application by the VIP users and teachers. The CICS resources will be supplemented by the administrator if the quantity is not enough.
- Feature: Automatic installation. As long as the administrator provides the relevant information of CICS to be installed, such as the quantity, the system will carry out automatic deployment and protection without manual operation.
- Binding character: Only the cloud administrator has the right to execute the operation.

(2) Functional description:

- Front-end submits a JES to the mainframe through ftp in order to install CICS.
- Protection for CICS. Only those who have permissions can log in CICS and use Transaction.

(3) Input items:

Table 4.4 Input Items of CICS Resource Application
(4) Output items:
   - If the installation is successful, the result will be returned to the front-end.
   - If the installation fails, the result as well as the cause of failure will be returned to the front-end.

(5) Algorithm:
   - To get the quantity of CICS to be installed and a name of CICS (CICSxxx) from the front-end through ftp, which is taken as the starting name of CICS? The subsequent names are obtained by increasing progressively from 1.
   - To call REXX and install CICS.
   - To build multiple CICS Region, protect CICS through RACF, and let different user group (such as CICS001) log in different CICS Region (such as CICSR001).
   - To protect the TRANSACTION of CICS through RACF, enable the users with access right to use TRANSACTION (READ permission), while the users without access right or users who use ESC (CICSUSER) are not allowed to use TRANSACTION (NONE permission).
   - To associate the obtained account to the group with CICS permission in the process of CICS resource application.

(6) Flow chat (as shown in Fig. 4.15)
4.5.3 Storage group resource management module

(1) Design description:
- **Purpose:** to create a new Storage Group to associate with the accounts added to the system.
- **Feature:** automation: after the administrator adds new accounts, the system will automatically submit JES to the mainframe to create a new Storage Group in order to associate with these accounts and also automatically associate volume with Storage Group.
- **Binding character:** all the public accounts in this system share one Storage Group named SGPUBLI; for the VIP accounts VIPxxxx, each has its own Storage Group: SGVxxxx; the teacher/student accounts in a course application are equipped with Storage Group: SGCPxxx. Only the administrator has the right to execute account adding and trigger this module.

(2) Functional description:
- Front-end submits a JES to the mainframe through ftp in order to create a Storage
Group that is associated with the new accounts.

- To associate volume with Storage Group.

3) Input items:

Table 4.5 Input Items of Storage Group Resource Application

<table>
<thead>
<tr>
<th>Information Content</th>
<th>Input Method</th>
<th>Field Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Storage Group Name</td>
<td>Front-end Input</td>
<td>Textfield</td>
</tr>
<tr>
<td>Volume Quantity</td>
<td>Front-end Input</td>
<td>Textfield</td>
</tr>
</tbody>
</table>

4) Output items:

- If the installation is successful, the result will be returned to the front-end.
- If the installation fails, the result as well as the cause of failure will be returned to the front-end.

5) Algorithm:

- To get the name of Storage Group to be created and Volume quantity from the front-end through ftp.
- To call the corresponding REXX program in order to create Storage Group and increase the volume through JCL bulk operation SMS.

6) Flow chat (as shown in Fig. 4.16)

![Flow Chat of Storage Group Resource Management Module](image)

**Fig. 4.16 Flow Chat of Storage Group Resource Management Module**
4.5.4 Resources monitoring

(1) Design description:

➢ Purpose: the users can query MEMORY utilization through this module.
➢ Feature: the mainframe calls corresponding JCL to execute the query request sent by
the front-end, and returns the processing results to the front-end for the users or
administrator.

(2) Functional description:

➢ After receiving the monitoring application from the cloud administrator, the
mainframe will call corresponding JCL to view Memory information: utilization rate
and free space etc., and return the results to the front-end.

(3) Input items

(4) Output items:

Table 4.6 Output Items of Memory Monitoring

<table>
<thead>
<tr>
<th>Information Content</th>
<th>Field Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memory Type</td>
<td>Textfield</td>
</tr>
<tr>
<td>Utilization Rate</td>
<td>Textfield</td>
</tr>
</tbody>
</table>

(5) Algorithm:

➢ To obtain the name of input SG, submit corresponding JCL, generate the SG
REPORT files, analyze the REPORT files using REXX statements, and send the
final analysis results to the front-end through FTP.

(6) Flow chat (as shown in Fig.4.17)
Fig. 4.17 Flow Chat of Resources Monitoring

To sum up, the mainframe administrator has access to resources utilization in various mainframes, and decides whether to approve the users’ resource application according to the remaining resources. If approved, the mainframe administrator will allocate enough resources to the cloud administrator and return the result (inform whether to pay for the application or not). If rejected, the mainframe administrator will give reasons for the rejection and return the result to the applicant.

4.6 Design of application management module

4.6.1 Mainframe account application module

(1) Functional description: Users who successfully log in the system can apply for a mainframe account. Ordinary users and teacher users have different permission in account application.
(2) Precondition: The users should have an account in Z Education Cloud.
(3) Input items: The ordinary users are unable to modify the resource quantity due to the limited permission. In this case, the system will directly allocate an account to them. Therefore, the resource allocation is transparent for the ordinary users. On the contrary, the teacher users have the right to modify DB2, CICS and Volume quantity while applying for a personal account.
(4) Output items: After the application is submitted, the system will return an outstanding message to the user. After account operation, the system will send a message that contains the
account information to the applicant.

(5) Flow chat (as shown in Fig. 4.18)

![Flow chat diagram](image)

Fig. 4.18 Flow Chat of Mainframe Account Application Module

### 4.6.2 Mainframe course application module

(1) Functional description: The teacher users can apply for various curricular resources and information through this module, and define the number of student accounts.

(2) Precondition: For teacher users only.

(3) Input items:

<table>
<thead>
<tr>
<th>Information Content</th>
<th>Input Method</th>
<th>Field Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Course Name</td>
<td>Front-end Input</td>
<td>Textfield</td>
</tr>
<tr>
<td>DB2 Quantity</td>
<td>Front-end Input</td>
<td>Int</td>
</tr>
<tr>
<td>CICS Quantity</td>
<td>Front-end Input</td>
<td>Int</td>
</tr>
<tr>
<td>Volume Quantity</td>
<td>Front-end Input</td>
<td>Int</td>
</tr>
<tr>
<td>Student Account Number</td>
<td>Front-end Input</td>
<td>Int</td>
</tr>
<tr>
<td>Teacher Account Number</td>
<td>Front-end Input</td>
<td>Int</td>
</tr>
</tbody>
</table>

(4) Output items: After the application is submitted, the system will return an outstanding message to the user. After course account operation and environment deployment, the cloud
service subsystem will send a message that contains the account information to the applicant. The users can download the excel document that contains the detailed information to their local computer.

(5) Flow chat (as shown in Fig. 4.19)

![Flow Chat](image)

**Fig. 4.19 Flow Chat of Mainframe Course Application Module**

### 4.7 Database design

Database is the core and foundation of information system. It organizes the mass data in the information system according to a certain model, provides data storage, maintenance and retrieval functions, and helps the information system to conveniently, immediately and exactly access the information required by the users. The key factor affecting whether various parts of an information system can be closely combined or not is database. Therefore, only rational logical design and effective physical design for the database can guarantee the successful development of a complete and efficient information system.

#### 4.7.1 Principles of database design

(1) Consistency and normalization of database: The serial numbers of user, school, resource, course and order should be generated by the program according to the norm in order to ensure
that these serial numbers are unique. For example, the user IDs are automatically allocated in ascending order.

(2) Authenticity: Authenticity plays a very important role in the design of application system. The database design should be able to accurately reflect all links in the practical application. Therefore, in early design, apart from the discussion of user demand with the clients, it is also necessary to carry out a series of research and survey, have a thorough understanding of the operation procedure of Mainframe education cloud, and ensure the authenticity by comprehensive study and collecting data from various sources.

(3) Rational redundancy: If data redundancy of the system is too low, it will greatly increase the connection query operations among the tables though the data integrity can be easily ensured. To increase the system response time, proper data redundancy design is necessary. Besides, we should also consider how to achieve a balance between normalization and non-normalization and how to improve system performance using appropriate data redundancy in database design. We can use rational design redundancy to exchange space for time so that we can maximize the system efficiency.

(4) Proper selection of data type: The proper selection of data type has great influence on the performance and efficiency of the database. If the primary key uses integer type, the query efficiency will be largely improved. On the contrary, if it chooses the character type, data insertion, update and query efficiency will decline. Therefore, the keywords of the tables that demand much query generally use integer type in the database design.

### 4.7.2 Design of database tables

The basic tables required in the database design of Z Education Cloud System are developed according to data modeling after thorough analysis of the system models. Details as per Appendix 2.

The user management module mainly contains the user information sheet, as listed in Table 1 and Table 2; the course management module primarily contains the curriculum schedule 5, course_documents table 6 and course_labs table 7 etc.; the application management module mainly contains the user order-related tables, such as order table 8, order_curriculum schedule 9, order_package list 10, and order_resource table 11 etc; the resource management module mainly contains resource table 3, and resource_pattern list 4 etc.
4.7.3 Relationships among Data Tables

From the above database analysis and table design, we can obtain the relationships among the data tables in Z Education Cloud, as shown in Fig. 4.20.

![Fig. 4.20 EER Diagram of System Database](image)

4.8 Chapter Summary

This chapter elaborates the design process of the entire system based on expandability, maintainability, and reliability, including design of system scheme and system architecture, setting of system role and authority management mechanism, design of object model, design of primary function modules and database.

Z Education Cloud uses Web application & development architecture based on SSH frame, so that the service logic is able to be independent on the user interactive interface in the development process, thus improving the efficiency of system design and facilitating the functional extensions in the future.

The setting of system role and authority management mechanism is to assign different system roles to different users. Each of the system roles has own permission to access the function modules. We define in this thesis a series of system roles and their data scope, and specify the roles’ right to access the system function modules, which establishes a set of flexible and uniform mechanism to manage the roles and authority.

The design of object model mainly involves class design. In this thesis we apply MVC
design mode, determine the core class for the system, extract classes for the use case, identify the relationships among the classes, and finally present the use case diagram and sequence diagram.

From the numerous function modules of Z Education Cloud system, the author selects several modules he has dealt with in the project design for analysis, including mainframe resource management module and application management module, which mainly describe how the front-end users can submit the resource applications to the mainframe administrator at the back-end for approval, and execute management operations such as resource creation, allocation, and recovery.

Database design occupies a significant position in the system. Only a good database can construct a sound and highly-effective system. The database in this system is designed based on the database design principles.
Chapter 5 Web Front-end Implementation of Mainframe Education Cloud

Z Education Cloud is a complicated system. Its functions can only be realized based on the function demand analysis, relevant concept design, logic design, and page design, and by connecting the Web front-end to the database. The established platform should have a user-friendly interface which is characterized by good interaction and easy to operate. The system should have comprehensive functions in order to meet the demands of the users. This chapter mainly introduces how to implement the Web front-end of Z Education Cloud, with focus on the implementation of mainframe resource management module that is unique to this system.

5.1 System environment configuration

5.1.1 Development environment

The development environment for the Web front-end of Z Education Cloud is configured as follows:

- Web server: Tomcat 6
- Database server: MySQL 5.1
- Development Platform: MyEclipse 9.0
- Java version: JDK 1.6.0
- Struts version: Struts 1.3
- Spring version: g2.0
- Hibernate version: Hibernate3.0

5.1.2 Deployment environment

The deployment environment of the constructed system is as follows:

- Operating system: Microsoft Windows XP/Windows 7
- Software support: Browser: Google Chrome/Firefox/Internet Explorer
5.2 Implementation of three-layers architecture based on SSH

5.2.1 Data persistence layer realized by Hibernate

Database operation is rather frequent in Z Education Cloud. In order to make it more rapid and efficient, we have brought the data persistence layer into between the database and the application programs in the development process. The data persistence layer stores the database information, which can be added, deleted, altered and queried. Here we explain the implementation process of Hibernate by taking the example of Z Education Cloud’s database resource (mainframe resource) tables and their objects:

1) Data Persistence Layer Objects

In the database, resource tables have the following columns: ResourceId, PackageId, ResourcePattern, ResourceName, Prefix, Password, and Status, which will be mapped into the system as objects and thus form Resource.java. Its code is as follows:

```
public class Resource implements java.io.Serializable {
    private Integer resourceId;
    private Packages packages;
    private ResourcePattern resourcePattern;
    .......
    public Integer getResourceId () {
        return this.resourceId;
    }
    public void setResourceId (Integer resourceId) {
        this.resourceId = resourceId;
    }
    .......
}
```

2) Hibernate Frame Mapping File
The mapping relation between the Resource tables in the database and the objects of Resource.java is realized by Hibernate mapping file: Resource.hbm.xml, whose code is as follows:

```xml
<hibernate-mapping>
  <class name="com.zcloud.dto.Resource" table="resource" catalog="zcloud">
    <id name="resourceId" type="java.lang.Integer">
      <column name="ResourceId" />  
      <generator class="native" />
    </id>
    <many-to-one name="packages" class="com.zcloud.dto.Packages" fetch="select" cascade="delete">
      <column name="PackageId" not-null="true" />
    </many-to-one>
    <property name="resourceName" type="java.lang.String">
      <column name="ResourceName" length="20" not-null="true" />
    </property>
    ......  
  </class>
</hibernate-mapping>
```

(3) Hibernate DAO Operation

The objects of persistence layer and database operation are realized through DAO (Data Access Object). Each object of the persistence layer has its own DAO. The public package provides the interface class of DAO: HibernateDaoSupport, and defines the public operations including adding, deleting, altering and querying. All the interfaces of DAO in the system are derived from this public interface class.
DAO interface of Resource objects is ResourceDAO.java, whose code is as follows:

```java
public class ResourceDAO extends HibernateDaoSupport {
    private static final Logger log = LoggerFactory.getLogger(ResourceDAO.class);
    // property constants
    public static final String RESOURCE_NAME = "resourceName";
    public static final String PREFIX = "prefix";
    ..... 
    //find mainframe resource according to ResourceId
    public Resource findById(java.lang.Integer id) {
        log.debug("getting Resource instance with id: " + id);
        try {
            Resource instance = (Resource) getHibernateTemplate().get("com.zcloud.dto.Resource", id);
            return instance;
        } catch (RuntimeException re) {
            log.error("get failed", re);
            throw re;
        }
    }
    ..... 
}
```

### 5.2.2 Business logic layer realized by Spring

The business logic layer (BLL) is designed using Spring frame, whose main tasks are: processing the application's business logic and business validation; processing management transaction; providing interface to interact with other layers; dependency management for the BLL objects; adding a flexible mechanism between the presentation layer and the data persistence layer in order to reduce their coupling; program execution management.

Between the presentation layer and BLL, Spring realizes an interface which enables BLL to simply provide business logic for its upper layer, so that it reduces the coupling between BLL and the presentation layer, and improves the reusability of the software.

Between BLL and the data persistence layer, Spring realizes the object-oriented thought of “Program to Interface” and IoC mechanism. Setter methods of business processing objects accept DAO interface rather than the specific implementation of DAO. In this way, it will be more convenient to implement the defined objects and then realize injection.

In this thesis, we explain the Spring implementation process by taking the application management module of Z Education Cloud as an example: AdApplyManagerImpl.java is an
interface implementation class of the application management. There are multiple attribute objects in this class and its code is as follows:

```java
public class AdApplyManagerImpl {
    //DAO interface
    OrderDAO orderDAO;
    UserDAO userDAO;
    OrderResourceDAO orderResourceDAO;
    OrderCourseDAOImpl orderCourseDAOImpl;
    OrderDAOImpl orderDAOImpl;
    SessionFactory sf;
    private BackgroundManager backgroundManager;
    private TaskExecutor taskExecutor;

    public AdApplyManagerImpl(TaskExecutor taskExecutor) {
        this.taskExecutor = taskExecutor;
    }

    public OrderResourceDAO getOrderResourceDAO() {
        return orderResourceDAO;
    }
    ...... 
}
```

We can see that, OrderDAO, UserDAO, and OrderResourceDAO are objects of AdApplyManagerImpl. In Spring frame, we need to inject them into AdApplyManagerImpl using Setter methods. Their dependency is shown in the configuration file: applicationContext.xml. Spring configuration file is as follows:

```xml
//Apply manage business bean
<bean id="adApplyManagerImpl" class="com.zcloud.manager.impl.AdApplyManagerImpl">
    <property name="orderDAO"><ref bean="OrderDAO" /></property>
    <property name="userDAO"><ref bean="UserDAO" /></property>
    <property name="orderResourceDAO"><ref bean="OrderResourceDAO" /></property>
    <property name="sf"><ref bean="sessionFactory" /></property>
    <property name="orderDAOImpl"><ref bean="orderDAOImpl" /></property>
    <property name="orderCourseDAOImpl"><ref bean="orderCourseDAOImpl" /></property>
    <property name="backgroundManager"><ref bean="BackgroundManager" /></property>
    <constructor-arg ref="taskExecutor" />
</bean>
```
5.2.3 Presentation layer realized by Struts

The presentation layer uses Struts frame which realizes MVC design pattern. Here we will introduce the concrete implementation method from four aspects including Struts configuration file, MVC design pattern’s View, Controller, and Module.

(1) Struts configuration file: It plays an important role in the overall framework. Corresponding configurations should be carried out for Struts model, web presentation and control in the configuration file.

(2) MVC pattern’s View: This part is the system user interface designed in the Eclipse project, namely index.jsp, which is returned to the corresponding JSP page by Controller according to the results provided by Module. To make the user interface more pleasing to the eye, many JSP pages will add CSS style, making the interface more user-friendly.

(3) MVC pattern’s Controller: It is generally recognized that Struts controller mainly includes the following components: ActionServlet, Action, and RequestProcessor. In actual use, it is defined by ActionMapping class in the configuration file.

(4) MVC pattern’s Module: It is generally recognized that the MVC pattern-based Module is composed of system internal state and business logic. The internal state is expressed using ActionForm, while the business logic is provided by Spring in SSH frame. Therefore, the internal state is considered as Module, namely ActionForm.

5.3 Implementation of function module

5.3.1 User interactive interface of Z Education Cloud

For security reasons, two Web front-end interfaces are designed in Z Education Cloud. One is the platform’s common gateway (cloud service and managerial subsystem), which is available for the teachers, students and cloud administrators in the mainframe education cloud. On the contrary, the other is only used by the mainframe administrator for back-end resource operation and system management (mainframe managerial subsystem). In this way, the security of mainframe operation is guaranteed and the mainframe safety problems due to outside invasion are also prevented.

(1) Cloud Service and Managerial Subsystem Interface of Z Education Cloud

Cloud service subsystem is user-oriented and its major function is to bridge the connection between the users and the cloud service subsystem. With this subsystem, the users do not interact
directly with the mainframe. Instead, they should send different requests to the managerial subsystem through different applications. Only when these requests are approved by the cloud managerial subsystem, can the users get the mainframe resources.

Cloud managerial subsystem is administrator-oriented. With this subsystem, the administrator is able to manage the entire mainframe cloud education platform, and process the resource applications from various users more conveniently and efficiently. Besides, the administrator can also interact with the mainframe at the Web-end, thus basically realizing automatic back-end operations.

Fig. 5.1 indicates the cloud service and managerial subsystem interface of Z Education Cloud. The users (mainframe administrator excluded) of mainframe cloud education platform can log in the system using username and password to apply for mainframe resources and share course resources. The cloud administrator can also log in the system to manage the users, courses, applications and resources.

![Cloud Service Subsystem Interface of Z Education Cloud](image)
Mainframe Managerial Subsystem of Z Education Cloud

Mainframe managerial subsystem is designed for back-end maintenance. Its design thought is to complete some complex back-end maintenance work for better management of mainframe cloud education. Meanwhile, it also aims to provide better user experience by allocating required resources for the cloud education platform and deploying experimental environment for different courses.

Fig. 5.2 shows the entrance Web interfaces of mainframe administrator, who can log in the system through this entrance to update, alter, or delete the users, and ads, delete or update the mainframe courses. Besides, the mainframe administrator can also manage the applications, view the detailed information of all users’ mainframe applications, and allocate the mainframe resources. Its primary function is unified management of mainframe resources, including resource creation, allocation and recovery.

Fig. 5.2 Cloud Managerial Subsystem Interface of Z Education Cloud

The environment configurations for the implementation of Z Education Cloud, implementation of SSH frame, and user interactive interfaces are emphasized in the above two sections. Next the author will focus on the implementation of some function modules in the
system.

In the following sections, the author will explain in detail the major function modules designed in Z Education Cloud, including resource creation, allocation and recovery in the mainframe resource management module.

The function modules in this system are implemented by submitting JES job to the mainframe through ftp at the front-end, and calling relevant REXX to realize application, recovery, creation and protection for the mainframe accounts and installation and protection for system resources (such as DB2 for z/OS, CICS etc). The mainframe resources are protected by RACF, and DataSet storage management is realized through DFSMS.

5.3.2 Implementation of mainframe resources management module

The mainframe resource management module contains three parts: resource creation, resource allocation and resource recovery. Next the implementation process of these three parts will be described in detail.

(1) Resource Creation

Resource creation can be done in three different ways: DB2, CICS and user. The resources created by DB2 and CICS both share the Load Module (system program of mainframe), but data isolation also exists. The resources created by user are divided into two types: one (Sxxxxxx/Txxxxxx) uses the default Storage Group and the other uses custom Storage Group.

If the resource applications from the front-end users are approved by the cloud administrator, the resources can be created at mainframe back-end. First, the system will store the information of mainframe resources applied by the users into an xml file, connect to the mainframe through ftp, and upload this file to the designated directory in the mainframe. Next the system will call REXX program to create new mainframe resources according to the configuration information in the xml file. In this process, if the task fails, the failure log will be downloaded to the Web in file format for the analysis by the administrator.

Fig. 5.3 shows the operation interface of mainframe resources creation.
Fig. 5.3 Operation Interface of Mainframe Resources Creation

The algorithm flow chart is as follows (Fig. 5.4):

Fig. 5.4 Algorithm Flow Chart for Resource Creation
The key code is as follows:

```java
public void installResources(List<PackageBean> packageBeanList, Integer userId) throws
FileNotFoundException, IOException {
    XMLBuilder xmlBuilder = new XMLBuilder();
    if (xmlBuilder.buildInstXML(packageBeanList, "C:\XML\pkginst.xml") ) {
        log.logPrint(logName, "Generate XML input successfully \n\n Start connection, uploading file.....\n");
    } else {
        log.logPrint(logName, " Fail to generate XML input, could not apply resource\n");
    }
    FtpToMainFrame ftp = new FtpToMainFrame("10.60.37.129");
    int execID = getExecID();
    if (ftp.uploadFileToMainFrame("C:\XML\pkginst.xml", "'zcloud.input(pi
 + execID+')'" )){
        log.logPrint(logName, "Upload file successfully\n\n Start to connect mainframe, Run tso command......");
    } else {
        log.logPrint(logName, "Fail to upload file, could not apply resource\n");
    }
}
```

(2) Resource Allocation

Mainframe back-end resources are pre-allocated and stored in the buffer pool. The mainframe resources are allocated in package for reasonable allocation and management. One package generally contains one Instance required by DB2 course and CICS course separately, 10 teacher accounts and 100 student accounts. After the user confirms the required resources, the system should associate the resources with the account, namely, carry out the following authorizations:
Authorization for teacher accounts include: start/stop DB2 and CICS, use DB2 and CICS, modify student files;

Authorization for student accounts include: start/stop DB2 and CICS, use DB2 and CICS, access teacher files;

Add Volume to the corresponding Storage Group

Different users (ordinary/VIP) use different Storage Group, as shown in Table 5.1.

Table 5.1 Example of Mainframe Resources Allocation

<table>
<thead>
<tr>
<th>Application ID</th>
<th>DB2</th>
<th>CICS</th>
<th>Teacher Account</th>
<th>Student Account</th>
<th>Storage Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>A00000001</td>
<td>D001</td>
<td>C0000001</td>
<td>T00001-T000010</td>
<td>S00001-S000010</td>
<td>SG0</td>
</tr>
<tr>
<td>A00000002</td>
<td>D002</td>
<td>C0000002</td>
<td>T000011-T000020</td>
<td>S00001-S000020</td>
<td>SG0</td>
</tr>
<tr>
<td>A00000003</td>
<td>D003</td>
<td>C0000003</td>
<td>A00101-T001010</td>
<td>B00101-B001010</td>
<td>SG001</td>
</tr>
<tr>
<td>A00000004</td>
<td>D004</td>
<td>C0000004</td>
<td>A002001-A002010</td>
<td>B002001-B002010</td>
<td>SG002</td>
</tr>
</tbody>
</table>

Each time when a user submits a resource application, the mainframe administrator will properly allocate the mainframe resources to the corresponding user according to the resource utilization in the buffer pool.

If a resource application from a front-end user is approved, the resources will be created first at the mainframe, that is to say, the required environment is created for the user in advance, and the mainframe administrator will allocate the resources one month before the resources are needed. The following steps are similar to those in the resource creation: First, the system will upload the xml file that contains the information of mainframe resources applied by the users to the designated directory in the mainframe. Then the system will call REXX program to automatically submit the jobs to the mainframe and allocate the resources. After the jobs are completed, the results will be returned to the front-end administrator for further operations.

Fig. 5.5 shows the operation interface of mainframe resources allocation.

![Fig. 5.5 Operation Interface of Mainframe Resources Allocation](image-url)
The algorithm flow chart is as follows (Fig. 5.6):

![Algorithm Flow Chart for Mainframe Resource Allocation](image)

Fig. 5.6 Algorithm Flow Chart for Mainframe Resource Allocation
The key code is as follows:

```java
public void dispatch(List<OrderBean> orderBeanList, Integer userId) throws FileNotFoundException, IOException {
    XMLBuilder xmlBuilder = new XMLBuilder();
    if (xmlBuilder.buildDispatchXML(orderBeanList, "C:\XML\dispatch.xml")) {
        log.logPrint(logName, "Generate dispatch XML
Start to connect to FTP. Upload file to mainframe

    })
        log.logPrint(logName, "Fail to generate XML file, could not allocate resource
    
    FtpToMainFrame ftp = new FtpToMainFrame("10.60.37.129");
    int execID = getExecID();
    if (ftp.uploadFileToMainFrame("C:\XML\dispatch_output.xml", "zcloud.input(" + execID + ")")
        log.logPrint(logName, "Upload file successfully
Start to connect mainframe, Run tso command
    
    Telnet telnet = new Telnet("10.60.37.129");
    try {
        telnet.openConnection();
        String tsoCMD = "exec " + execID + "");
        telnet.execTsoCMD(tsoCMD, logName);
        if (ftp.downloadFileFromMainFrame("ZCLOUD.OUTPUT.RESULT(" + execID + ")", "C:\XML\dispatch_output.xml")) {
            log.logPrint(logName, "Finish download file
        
        }
        
        log.logPrint(logName, "Could not connect to mainframe, fail to download file
    }
}
```

(3) Resource Recovery

The mainframe administrator has the right to recover the expired resources. The concrete implementation steps are:

a. Revoke user actions and delete user DataSet from the mainframe system:
   Physical restore of DB2;
   Physical restore of CICS;

b. Revoke the association between resources and accounts:
Revoke teacher accounts;
Revoke student accounts;
Take back Volume from Storage Group.

The algorithm flow chart is as follows (Fig. 5.7):

![Algorithm Flow Chart for Resource Recovery](image)

**Fig. 5.7 Algorithm Flow Chart for Resource Recovery**

The key code is as follows:

```java
public void reclaimResources(List<OrderBean> orderBeanList,Integer userId) throws FileNotFoundException, IOException {
    XMLBuilder xmlBuilder = new XMLBuilder();
    if (xmlBuilder.buildDispatchXML(orderBeanList, "C:\XML\reclaim.xml")) {
        log.logPrint(logName, "Generate reclaim XML \n\n Start connect mainframe, upload file.....\n");
    } else {
        log.logPrint(logName, "Fail to generate XML, could not dispatch resource \n");
    }
    FtpToMainFrame ftp = new FtpToMainFrame("10.60.37.129");
    int execID = getExecID();
    if(ftp.uploadFileToMainFrame("C:\XML\reclaim.xml", "'zcloud.input(OR" + execID + ")'")){
        log.logPrint(logName, "Upload file successfully\n\n Startto connect mainframe, run TSO command......");
    }
}
```
To sum up, the mainframe resource management module mainly provides an entrance at the back-end for the mainframe administrator to manage resource creation, allocation and recovery. Resource management is within the responsibility scope of the mainframe administrator, so it is implemented in the mainframe managerial subsystem. The resource management interface is shown in Fig. 5.8.

![Fig. 5.8 Resource Management Interface](image)

In Fig. 5.8, the mainframe administrator can add new mainframe resources by clicking “Add”. The administrator is allowed to create package in the buffer pool to provide the users with different mainframe resource combinations, thus providing more convenient and efficient services.
5.4 Security policy

The system security mechanism is shown in the process of analysis, design and implementation of Z Education Cloud. To ensure the security of the system model as a whole, the author specifies the following mechanisms after comprehensive analysis. With these security mechanisms, safe access and management of the model system can be realized effectively. Besides, the security mechanisms will be integrated into the implementation and application of the model.

1. Authorization and authentication: The functions of user authentication and operation authorization for the system module are realized from user administration, mainframe resource management, application management and course management, and they are highly flexible and feasible.

2. Input validation: To enhance user input accuracy, this system carries out bounds checking and pattern matching for user input. All these functions can verify the user input, for example, some fields only permit input of numbers, or Chinese and English characters, or regulate the character length.

3. Configuration management: Database connection statements are set in Hibernate configuration files, and these statements can be directly invoked to realize database connection in the page file. The database connection string in Hibernate.cfg.xml is configured as follows:

```xml
<hibernate-configuration>
  <session-factory>
    <property name="dialect">org.hibernate.dialect.MySQLDialect</property>
    <property name="connection.url">jdbc:mysql://127.0.0.1:3306/zcloud</property>
    <property name="connection.username">root</property>
    <property name="connection.password">1234</property>
    <property name="connection.driver_class">com.mysql.jdbc.Driver</property>
  </session-factory>
</hibenate-configuration>
```

Apart from the above security mechanisms, the following safety techniques are also integrated into Z Education Cloud:

- The overall system model gives different access rights to the users and strictly limits users’ access to the subsystem and database.
- Database operation applies stored procedure and the database can be accessed by directly
5.5 Chapter summary

This chapter introduces in detail the environment configuration of Z Education Cloud, implementation of SSH frame, implementation of major function modules, and security policy. The concrete frame configuration for Struts, Spring, and Hibernate are explained in the implementation of SSH frames. Three subsystems are defined according to system users’ roles: cloud service subsystem and cloud managerial subsystem, which serve as the entrance of public users, and mainframe managerial subsystem, which is only available for the mainframe administrator. Therefore, two Web front-end entrances are realized and their interfaces are presented respectively. Next, the author expounds the implementation method for the key module he designed and implemented in this project---mainframe resource management module, gives detailed operation interfaces, algorithm flow charts, and key codes, and presents corresponding functional interfaces. Finally the author introduces some security mechanisms in the system.
Chapter 6 Web Front-end System Testing of Mainframe Education Cloud

The Web-based systems are characterized by distribution, isomerism, concurrence, and platform independence, and rely on multiple factors, such as technology, components, platform, server and architecture. The inherent complex features expose it to big risks in the using process, making the Web-based system testing different from the conventional software testing, including content test, interface test, functional test, performance test, compatibility test, and safety test etc. Among these testing, performance test and functional test are the focus. The author will introduce the methods of these two testing in this chapter.

6.1 System performance test

Performance test plays an important role in software quality assurance, and is an effective tool for system optimization. Here we mainly check whether the software meets the requirements of all performance indexes in the requirements specification, in order to ensure that the application system has the required properties. China Software Testing Center generalizes performance test into three parts: client performance test, online performance test, and server performance test. Generally speaking, the effective and rational combination of these three parts can guarantee the objectives of performance test.

Performance test involves the evaluation of system performance under normal and heavy use conditions. It examines the application’s response to the user requests under heavy load conditions as a result of various users and a large data volume in order to ensure the system security, reliability, and execution efficiency, and find out feasible methods to maintain a good system performance. In the Web-based application testing, performance test adds load to the server with requests by simulating a large number of user operations. At the same time, it monitors the utilization of database server, application server, and network resource, and inspects the relationship between system response time and these resources, in order to verify the system capacity and identify the defects. Performance test has the following methods:

- Load testing: to examine the change of system performance by gradually increasing system load, and finally determine the system’s maximum permitted load.
- Pressure test: to examine the change of system performance by gradually increasing
system load, and finally identify the load conditions of invalid system performance, in order to acquire the system’s maximum service level.

- Large data volume test: This test focuses on data size, including test for independent data size and integrated data size. The former carries out large data volume tests for storage, transmission, statistics, and querying operations of some systems while the latter is usually combined with pressure test, load testing, and fatigue test.

- Fatigue test: to expose the system to continuous execution for a period of time with the largest number of concurrent users or daily operation users under the stable operation conditions, and determine the system’s maximum strength performance through comprehensive analysis of transaction execution index and resources monitoring index.

Based on the analysis of system testing data, it is recommended to optimize the system from the following aspects: to improve correlated parameters of application server or change a new server with better performance, in order to improve the efficiency of junk data processing, and avoid server pause caused by excessive resource consumption as a result of data access; to optimize the SQL query statement in the case of large date volume in the database; to optimize the parameter configuration for the database in order to improve the overall system performance.

6.2 System functional test

Functional test mainly examines whether the WEB application system has comprehensive functions, meets the design requirements, and caters to the customers’ demands or not. Details are as follows: to verify whether the system has the expectant functions; whether the user management and course management in Z Education Cloud are correct; whether application adding, modifying, deleting and annotation are normal; whether user authentication, user authorization and information management reach set objectives; and whether the mechanism of error processing functions well etc. Black box testing is used for the functional test of mainframe education cloud in this thesis. It should include all the functions and can be carried out together with compatibility test and performance test. The actual test can be done by simulating customer scenarios, and designing documents according to the software requirements. In view of limited space, the author only gives a brief introduction to the black box testing for one function module (application management module) in this section.
6.2.1 Black box testing

Black box testing mainly tests whether the system meets all requirements of the function module according to the functional requirements. Its purpose is to find out the external errors of the codes instead of the internal logical errors. Therefore, the whole system is like a black box for the testing personnel, who do not know the internal structure or the concrete implementation of the codes. The specific methods for black box testing are as follows:

- equivalence partitioning
- boundary value
- cause and effect diagram
- decision table driven

6.2.2 Black box testing of application management module

Here the author gives a brief introduction to black box testing by taking the application management module as an example.

The application management module enables various levels of users to apply for the mainframe resources according to their requirements, such as student application for the mainframe accounts, teacher application for the mainframe courses, cloud administrator approval of user applications, and mainframe administrator management of applications etc.

Test cases and testing process are as follows:

1. Test cases of teacher course application

Log in Z Education Cloud service subsystem as a teacher. The function interface that displays the teacher’s application courses and account should be in Application menu. As is shown in Fig. 6.1:
On this page, the teacher should first set the start date and end date for the applied courses, and choose the required courses, such as Z/OS, DB2, or CICS. Then the teacher should verify whether the system will prompt error message and remind the user to input correct time nodes when the time is improperly set, for example, when StartDate is later than EndDate. The prompt message is as shown in Fig. 6.2:

After inputting correct time and course information, enter the next step by clicking “next” button at the end, as shown in Fig. 6.3:
In this interface, the teacher is able to choose student account number, teacher account number, DB2 environment number and version, and CICS environment number and version; verify whether the system will automatically set the maximum permitted number when the student account and teacher account exceed the caps (20 teacher accounts and 200 student accounts at most); check whether the system will prompt error message for the user when the input account number is minus, decimals, or other illegal characters using equivalence partitioning method; check whether the system will function well when the input student account number is 0 or 200 using boundary value method; and examine whether the corresponding pages will appear by clicking the two buttons “Return” and “Next” at the bottom.

After successful course application, the teacher can view the course application records in “My Application”, check whether the newly applied courses appear in the records, and examine whether the application information is correct.

(2) Test cases of mainframe administrator management of course application

Log in the mainframe managerial subsystem as a mainframe administrator, who is able to manage the users, courses, applications and mainframe resources, as shown in Fig. 6.4:
After the courses are approved by the cloud administrator in the cloud managerial subsystem, the corresponding application information will appear in the mainframe administrator interface. Then the back-end mainframe administrator can manage the course applications, as shown in Fig. 6.5:

In the course management interface, the mainframe administrator can view the information and status of course applications, and make some comments, as shown in Fig. 6.6. In this page, the administrator verifies whether the course application information conforms to the detailed information of teacher course application in the cloud service subsystem, adds administrator platform to the comment form, and check whether the comments are stored in the application information after saving.
Return to the application page, select a course application in “new” status, and click the “Approve” button below the list in order to check whether there is returned information that has been processed successfully (see Fig. 6.7), and return to check whether the status of this course information has been modified into “approve”.

Due to limited space, the author only gives a brief introduction to the black box testing for some function modules. Detailed design and implementation of test cases for various function modules are presented in the practical tests. Besides, the system errors and problems from each
text are modified and improved, enabling the system to meet the design requirements in terms of performance and functions.

6.3 Chapter summary

This chapter analyzes the methods of system performance testing and functional test by focusing on the testing and results of application management module in the mainframe managerial subsystem. Besides, Z Education Cloud system is also improved by solving the problems from the practical testing so that the system can meet higher requirements.
Chapter 7 Conclusion and Prospect

7.1 Conclusion

As an integrated solution in large-scale applications, IBM mainframe has been widely accepted in the industry. Banking, insurance and security industries around the world cannot do without the mainframe. However, the ordinary software employees have few opportunities to learn the mainframe in a systematic way because it is a high-end and expensive machine. Besides, both China and foreign countries run short of mainframe talents. The fundamental reason is that a good environment for studying the mainframe technologies is not available, resulting in the talents shortage. Obviously, this is unfavorable for the further development of mainframe.

The system proposed in this thesis is an effective trial of could-based mainframe education. Since our school (Tongji University) built the IBM Technology Center in 2005, we have been devoted to cultivating excellent mainframe talents. As one of the first colleges and universities that participate in cooperative IBM mainframe projects, our school keeps on exploring and researching the mainframe curriculum system construction. However, mainframe education is still challenged by the problems such as difficulty in getting started, teaching resource shortage, and overemphasized theoretical property. Therefore, it will greatly improve the existing education pattern to establish a platform for sharing resources to popularize the mainframe education.

In this thesis, the author designs and implements the mainframe education cloud and Z Education Cloud Web front end. The specific work is as follows:

1. Analyze the system technology mechanism required in the implementation of Z Education Cloud Web front-end, introduce the B/S-based architecture, MVC design mode, SSH development framework, database tools, and asynchronous information mechanism between front-end web and back-end mainframe.

2. Carry out requirement analysis for Z Education Cloud Web front-end by focusing on business requirement and functional requirement. The design and description of the system use case are presented in the business requirement using UML. The detailed classification and description of function modules are given in the functional requirement, including user management module, course management module, application management module, resource management module, charge module, view and report module. Besides, the author also introduces some non-function requirements and their operation modes.
According to the results of system requirement analysis, design Z Education Cloud Web front end, explain the system design principles, design scheme and architecture, design the user authorization management mechanisms and object models, design the mainframe resource management module and application management module, and design the database.

According to the system requirement analysis and designs, implement Z Education Cloud Web front-end, which is expounded from three aspects: implementation of SSH framework, interface, and some function modules. Discuss the implementation of security policy in the system.

To sum up, based on the requirement analysis of mainframe education cloud, this thesis designs and realizes Z Education Cloud system, enables the users to learn and share by taking the mainframe education cloud as a platform, so that they can share good teaching resources and rapidly deploy mainframe environment for teaching practice on the platform, which will be significant for the development of mainframe education.

7.2 Prospect

The research emphasis of the mainframe education platform in this thesis is: Web front-end application technologies, user roles and authorization management mechanisms on the platform, and asynchronous communication mechanism between front-end web and back-end mainframe. In spite of the preliminary achievements obtained in the research process, we still have a long way to go. Many problems still need further investigating and the important ones are as follows:

- function module division remains to be improved
- system interface remains to be embellished
- system performance remains to be optimized

Obviously, it is now challenging and difficult for us to have a further study of the above tasks and to solve the problems, which will require us to make unremitting efforts in the future. With the continuous advances in science and technology, we firmly believe that all the above problems will be solved and the technologies will be put into practical use one day.
References


[16] Liao Xuefeng. Spring 2.0 Core Technology and Best Practice. Electronic Industry Press. 2007.6

[17] The Education Cloud: Delivering Education as a Service


[21] Paul Rogers. ABCs of z/OS System Programming Volume 1. IBM. 2011.6
Appendix 1 Z Education Cloud System Sequence Diagram

(1) Ordinary User

![Ordinary User Register Sequence Diagram](image1)

Fig. 1 Ordinary User Register Sequence Diagram

![Logon Sequence Diagram](image2)
(2) Teacher User
Fig. 5 Teacher User Renew Application Sequence Diagram

(3) Cloud Administrator
Fig. 6 Cloud Administrator Course Maintain Sequence Diagram

Fig. 7 Cloud Administrator Request Manage Application Sequence Diagram
### User Table Structure

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>If Null</th>
<th>Default Value</th>
<th>Property</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>UserId</td>
<td>Int(10)</td>
<td>No</td>
<td>&lt;auto-increment&gt;</td>
<td>PK</td>
<td></td>
</tr>
<tr>
<td>Email</td>
<td>varchar</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>password</td>
<td>varchar</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FirstName</td>
<td>varchar</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LastName</td>
<td>varchar</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gender</td>
<td>enum(‘M’,’F’)</td>
<td>No</td>
<td>M</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OrgId</td>
<td>Int(10)</td>
<td>Yes</td>
<td></td>
<td>FK</td>
<td></td>
</tr>
<tr>
<td>Phone</td>
<td>varchar</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RegTime</td>
<td>datetime</td>
<td>Yes</td>
<td>0000-00-00 00:00:00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Address</td>
<td>varchar</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TelStatus</td>
<td>varchar</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VipStatus</td>
<td>varchar</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IsVip</td>
<td>enum(‘0’,’1’)</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IsActive</td>
<td>enum(‘0’,’1’)</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Role</td>
<td>enum(‘TA’,’RP’,’AD’)</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

User Table is to store User info.

### Organization

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>If Null</th>
<th>Default Value</th>
<th>Property</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>OrgId</td>
<td>Int(10)</td>
<td>No</td>
<td>&lt;auto-increment&gt;</td>
<td>PK</td>
<td></td>
</tr>
<tr>
<td>OrgName</td>
<td>varchar</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OrgDesc</td>
<td>text</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Contract</td>
<td>varchar</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Phone</td>
<td>varchar</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Address</td>
<td>varchar</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RegUserId</td>
<td>Int(10)</td>
<td>Yes</td>
<td></td>
<td>FK</td>
<td></td>
</tr>
</tbody>
</table>
RegTime | datetime | No | 0000-00-00 00:00:00
---|---|---|---
IsVip | enum('0','1') | No | 0
IsActive | enum('0','1') | No | 0

Organization Table is to store all info of school and organizations.

Table3: Resource

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>If Null</th>
<th>Default Value</th>
<th>Property</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>ResourceId</td>
<td>Int</td>
<td>No</td>
<td>&lt;auto-increment&gt;</td>
<td>PK</td>
<td></td>
</tr>
<tr>
<td>PackageId</td>
<td>Int</td>
<td>No</td>
<td></td>
<td>FK</td>
<td></td>
</tr>
<tr>
<td>ResourcePattern</td>
<td>Int</td>
<td>No</td>
<td></td>
<td>FK</td>
<td></td>
</tr>
<tr>
<td>ResourceName</td>
<td>varchar</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Prefix</td>
<td>varchar</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Password</td>
<td>varchar</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Status</td>
<td>varchar</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Resource Table stores all mainframe resource.

Table4: Resource_Pattern

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>If Null</th>
<th>Default Value</th>
<th>Property</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>PatternId</td>
<td>Int</td>
<td>No</td>
<td>&lt;auto-increment&gt;</td>
<td>PK</td>
<td></td>
</tr>
<tr>
<td>ResourceType</td>
<td>varchar</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Comments</td>
<td>longtext</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Version</td>
<td>varchar</td>
<td>No</td>
<td></td>
<td>PK</td>
<td></td>
</tr>
</tbody>
</table>

Resource_Pattern table stores all mainframe info under specific environment and version.

Table5: Course

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>If Null</th>
<th>Default Value</th>
<th>Property</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>CourseId</td>
<td>Int</td>
<td>No</td>
<td>&lt;auto-increment&gt;</td>
<td>PK</td>
<td></td>
</tr>
<tr>
<td>RegUserId</td>
<td>Int</td>
<td>Yes</td>
<td></td>
<td>FK</td>
<td></td>
</tr>
<tr>
<td>CourseName</td>
<td>varchar</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CourseDesc</td>
<td>longtext</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>Data Type</td>
<td>If Null</td>
<td>Default Value</td>
<td>Property</td>
<td>Comments</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------</td>
<td>---------</td>
<td>------------------------</td>
<td>----------</td>
<td>----------</td>
</tr>
<tr>
<td>CourseType</td>
<td>varchar</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CourseOwnName</td>
<td>varchar</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CourseOrgName</td>
<td>varchar</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RegTime</td>
<td>datetime</td>
<td>Yes</td>
<td>0000-00-00 00:00:00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IsActive</td>
<td>varchar</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Course table stores all courses info of mainframe.

Table6: Course_file

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>If Null</th>
<th>Default Value</th>
<th>Property</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>FileId</td>
<td>Int</td>
<td>No</td>
<td>&lt;auto-increment&gt;</td>
<td>PK</td>
<td></td>
</tr>
<tr>
<td>Courseld</td>
<td>Int</td>
<td>Yes</td>
<td></td>
<td>FK</td>
<td></td>
</tr>
<tr>
<td>RegUserId</td>
<td>Int</td>
<td>Yes</td>
<td></td>
<td>FK</td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>varchar</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FileName</td>
<td>varchar</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FileDesc</td>
<td>varchar</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RegTime</td>
<td>datetime</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Size</td>
<td>varchar</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DownloadTimes</td>
<td>Int</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WithAnswer</td>
<td>varchar</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Path</td>
<td>varchar</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IsActive</td>
<td>varchar</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Course_file stores all file info of courses, such as course introduction and courseware.

Table7: Course_lab

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>If Null</th>
<th>Default Value</th>
<th>Property</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>LabId</td>
<td>Int</td>
<td>No</td>
<td>&lt;auto-increment&gt;</td>
<td>PK</td>
<td></td>
</tr>
<tr>
<td>Courseld</td>
<td>Int</td>
<td>No</td>
<td></td>
<td>FK</td>
<td></td>
</tr>
<tr>
<td>LabTitle</td>
<td>varchar</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Comments</td>
<td>varchar</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Course_lab stores all lab course info of mainframe.
### Table 8: Order Table

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>If Null</th>
<th>Default Value</th>
<th>Property</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>OrderId</td>
<td>Int</td>
<td>No</td>
<td>&lt;auto-increment&gt;</td>
<td>PK</td>
<td></td>
</tr>
<tr>
<td>ReqOrgId</td>
<td>Int</td>
<td>No</td>
<td></td>
<td>FK</td>
<td></td>
</tr>
<tr>
<td>ReqUserId</td>
<td>Int</td>
<td>No</td>
<td></td>
<td>FK</td>
<td></td>
</tr>
<tr>
<td>ReqTime</td>
<td>datetime</td>
<td>Yes</td>
<td>0000-00-00 00:00:00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>StartTime</td>
<td>date</td>
<td>No</td>
<td>0000-00-00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EndTime</td>
<td>date</td>
<td>No</td>
<td>0000-00-00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UserComment</td>
<td>longtext</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OrgReqComment</td>
<td>longtext</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AdmComment</td>
<td>longtext</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Status</td>
<td>enum</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>deadline</td>
<td>enum</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Order table stores all application orders that users request from mainframe.

### Table 9: Order_course Table

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>If Null</th>
<th>Default Value</th>
<th>Property</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Int</td>
<td>No</td>
<td>&lt;auto-increment&gt;</td>
<td>PK</td>
<td></td>
</tr>
<tr>
<td>CourseId</td>
<td>Int</td>
<td>No</td>
<td></td>
<td>FK</td>
<td></td>
</tr>
<tr>
<td>OrderId</td>
<td>Int</td>
<td>No</td>
<td></td>
<td>FK</td>
<td></td>
</tr>
<tr>
<td>ReqLab</td>
<td>varchar</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Order_course table stores all application orders that users request for courses.

### Table 10: Order_package Table

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>If Null</th>
<th>Default Value</th>
<th>Property</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Id</td>
<td>Int</td>
<td>No</td>
<td>&lt;auto-increment&gt;</td>
<td>PK</td>
<td></td>
</tr>
<tr>
<td>----</td>
<td>-----</td>
<td>----</td>
<td>-----------------</td>
<td>----</td>
<td></td>
</tr>
<tr>
<td>OrderId</td>
<td>Int</td>
<td>No</td>
<td>0</td>
<td>FK</td>
<td></td>
</tr>
<tr>
<td>packageId</td>
<td>Int</td>
<td>No</td>
<td>0</td>
<td>FK</td>
<td></td>
</tr>
</tbody>
</table>

OrderPackage table stores all application orders that user request for package.

### Table11: Order_resource Table

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>If Null</th>
<th>Default Value</th>
<th>Property</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Int</td>
<td>No</td>
<td>&lt;auto-increment&gt;</td>
<td>PK</td>
<td></td>
</tr>
<tr>
<td>OrderId</td>
<td>Int</td>
<td>No</td>
<td>0</td>
<td>FK</td>
<td></td>
</tr>
<tr>
<td>ReqNumber</td>
<td>Int</td>
<td>No</td>
<td></td>
<td>FK</td>
<td></td>
</tr>
<tr>
<td>ResourcePatternId</td>
<td>Int</td>
<td>No</td>
<td>1</td>
<td>FK</td>
<td></td>
</tr>
</tbody>
</table>

Order_resource table stores all application orders that user request for mainframe resource.

### Table12: Packages Table

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>If Null</th>
<th>Default Value</th>
<th>Property</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>PackageId</td>
<td>Int</td>
<td>No</td>
<td>&lt;auto-increment&gt;</td>
<td>PK</td>
<td></td>
</tr>
<tr>
<td>Prefix</td>
<td>varchar</td>
<td>No</td>
<td></td>
<td>FK</td>
<td></td>
</tr>
<tr>
<td>PostDate</td>
<td>datetime</td>
<td>No</td>
<td>0000-00-00 00:00:00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PackageExcel</td>
<td>varchar</td>
<td>No</td>
<td></td>
<td>FK</td>
<td></td>
</tr>
<tr>
<td>Status</td>
<td>enum(Free,Locked,InsError,UnInsError)</td>
<td>Yes</td>
<td>Free</td>
<td></td>
<td></td>
</tr>
<tr>
<td>type</td>
<td>enum(fixed,extended)</td>
<td>Yes</td>
<td>Fix</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Package table stores all application orders that user request for mainframe package.

### Table13: Messages Table

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>If Null</th>
<th>Default Value</th>
<th>Property</th>
<th>Comments</th>
</tr>
</thead>
</table>

95
<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Int</td>
<td>No</td>
<td>&lt;auto-increment&gt;</td>
<td>PK</td>
</tr>
<tr>
<td>fromUser</td>
<td>Int</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>toUser</td>
<td>Int</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>title</td>
<td>varchar</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>content</td>
<td>text</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>upTime</td>
<td>datetime</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>isRead</td>
<td>smallint</td>
<td>Yes</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>type</td>
<td>enum(System, Application, fee)</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Message table stores all message info of users’ relationship.