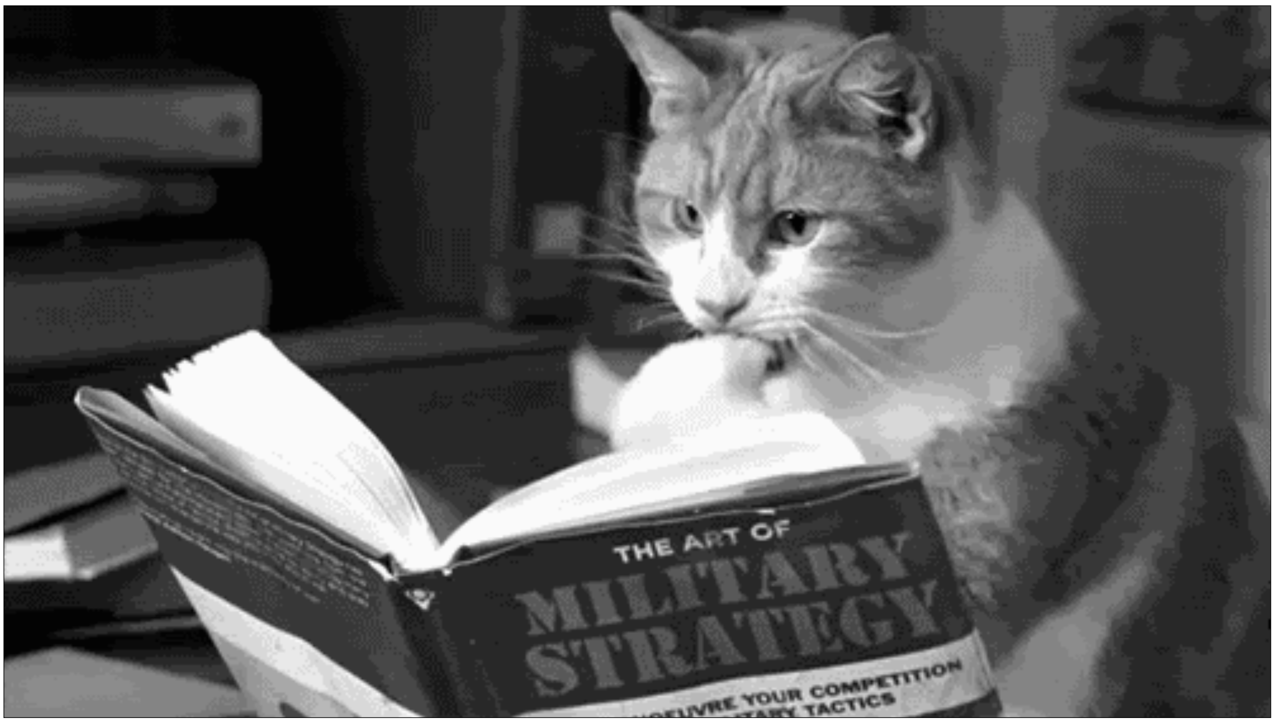


Teaching Service System: Course Resource Sharing Subsystem Software Design Specification

Team Leader: 王逍语

Team Member: 金璐, 辜逸龙, 王瑞, 陈超

4. Juni 2014

TABLE OF CONTENTS

CHAPTER I: INTRODUCTION

1.1 Purpose

This High Level Design (HLD) Document contains the necessary detail to The Teaching Service System Course Resource Sharing Subsystem, representing a suitable model for the developing team to do the coding. This document is also intended to help finding contradictions before the actual building, as well as being used as a reference manual showing the interactions between modules at a high level.

The purposes are:

- Providing the basis for programmers
- Specifying the conditions for modification and maintenance
- Specifying the API and user interface
- The project manager will be supervise the entire development process according to this document.

The Intended Audience:

- Software customers
- Project manager
- Project developers
- Software quality evaluation personnel
- Software maintenance personnel

1.2 Scope

- Name
  Teaching Service System, Resource Sharing Subsystem

- Proposer
  Professor Wang Xinyu, Software Engineering Course, Zhejiang University

- Developer Team
  Students of the Software Engineering Course (22120030) 2013-2014 Summer Semester

- User
  Teaching Staff, Students, System Administrator.

- Network Environment for Deploying
  Local area network with several computers including a server and at least one client.

- Background
  This project is the assignment for students in the Software Engineering Course of Zhejiang University, proposed by Professor Wang Xinyu. The entire system is divided in to 6 parts for the 6 groups of the class to work on.

1.3 Definitions, Acronyms, and Abbreviations

The Product	The product mentioned in this report, Course Resource Sharing Subsystem of The Teaching Service System
User	The users of the system, including three user groups: Teaching Staff, Student and System Administrators
Browser	The application used to access the system.
Certification	Certification is used to restrict the user access to the resources. In details, it identifies the users and stores information of whether or not he has access to certain resources.
Course	Categories of the resources. The basic unit used to sort all the resources stored inside the system in categories.
Course List	Course List is a list of all the courses, including necessary information to identify the course.
Course Resource	A type of resources. The Course Resources resources are available for authorised Students and Teaching Staff, and are expected to be uploaded by the Teaching Staff
Homework	A type of resources. The Homework are expected to be uploaded by the Student.
Homework Assignment	A list for recourses. Homework Assignments is a list of documents required for each Student to upload.
Homework Submitting	Homework Submitting is the process for uploading documents provided in the Homework Assignment.
Deadline	Deadline is a specific date for each Homework Assignment, which requires the Students to do the Homework Uploading for such Homework Assignment before the Deadline.
Teaching Staff	A group of Users. A Teaching Staff is responsible for giving Homework Assignment and uploading Course Resources for Student to download for Courses they have Certification to access.
Student	A group of Users. A Student can download Course Resources and do the Homework Uploading of Courses they have Certification to access.
System Administrator	A group of Users. A System Administrator is responsible for maintaining course information and Certification for courses for Students and Teaching Staff.
Database and File management System	The Database and File management System is the system designed to manage the database which stores all the information related to The Product and the Resources uploaded by the User.
DNTC	Distributed Network Traffic Controller.
Fair Share	An administratively set data rate per time frame that is considered fair.
Throttling	A reduction in maximum transfer rate of data.

Open Pipe	A connection to the Internet with no throttling.
Slow Pipe	A throttled version of an open pipe where all users of that gateway share that reduced pipe’s rate.
Firewall	Functionality that can allow or block certain ports and addresses.
IP Table	A firewall built into the Linux kernel.
IP Forwarding/ Masquerading	The ability to forward traffic.
Postgres SQL Server	A database management system.
Python	A possible programming language to interface between IPTables and Postgres.
JDBC	A possible Java-based interface between IPTables and the Database.
JSP	The language that will be used for displaying user history and administrative functionality.
Tomcat	A free, open-source implementation of Java Servlet and JavaServer Pages technologies developed under the Jakarta project at the Apache Software Foundation.
Apache	An open source Web server.
ER	Entity Relation Diagram
CBQ	Class-Based Queuing. Limits bandwidth at the IP/port level.
Kernel	Core of an operating system, a kernel manages the machine’s hardware resources (including the processor and the memory), and provides and controls the way any other software component can access these resources.
DHCP (Dynamic Host Configuration Protocol)	This is a protocol that lets network administrators centrally manage and automate the assignment of IP Addresses on the corporate network.
Gateway	Bridges the gap between the internet and a local network.

1.4 References

Roger S. Pressman. *Software Engineering: A Practitioner's Approach, Seventh Edition*. Beijing: China Machine Press, 2010.9.

Wang Xinyu. 《教学服务系统软件工程实验(英文)》. Software Engineering Course Resource, Zhejiang University.

1.5 Overview

A complete course resource sharing system should implement the function modules below:

- Resource sharing
  Users should be able to view, search, upload and download resources. The resources should be classified while uploading. The uploader should be able to manage the resources uploaded by himself/herself. The users can get the information before they download the resources.

- Resource managing
  The administrator should be able manage all the resources. The administrator can alter the category's name and content and the displaying position of resources. The administrator can rename or delete the resources or change the description of resources. A message will be sent to the uploader after changing.

- Homework assigning
  The instructors should be able to view the assigned homework of his/her courses. The instructors should be able to download the submitted homework which are uploaded by the students. The instructors can assign a new homework for his/her courses. While assigning, the instructors can set the deadline and give a description for the homework. The students who should finish the homework will receive a message after the homework is assigned.

- Homework submitting
  The students should be able to view the assigned homework of the participated courses. The students should be able to upload their homework before the deadline.

CHAPTER II: OVERALL DESIGN

2.1 Operational Requirement

The Course Resource Sharing Subsystem is a subsystem under the Teaching Service System. It works as a platform for teachers and students to share educational resources. It also allows the teaching staff to assign and collect homework and assignments.

2.1.1 Functional Requirement

The customer should login from a main system and enter our subsystem by clicking the URL link in main interface. The main interface page will also post the basic information about this subsystem.

The teaching resource sharing subsystem has two interfaces, Resource Management Interface and Homework Management Interface.

The Resource Management Interface includes command of viewing resource, command of uploading resource (limited by user's privilege and level of resource (public, group only, class only and school only)), command of resource management, command of downloading resource, command of searching resource.

The Homework Management Interface includes command of homework assignment, command of homework submission, command of checking homework and command of searching homework.

2.1.2 Performance Requirement

The performance of this subsystem would have to depend on the server-side database, data transmitting time of the network and the amount of users online.

For the system itself, good design should be applied. The client-side should avoid repeating meaningless request to server which would greatly affect the performance, especially under a poor internet environment. Also, the UI should appear to be simple and user-friendly.

A stable server should be used so as to maximise the capable number of users and minimise the response time of the system. The server-side should have strategies designed to deal with emergency situations e.g. power cut.

2.1.3 I/O Requirement

A web application is acting as the client to provide a user-friendly UI for user. User can submit forms or click onto links/buttons in the webpage so as to manipulate the information in the database.

The results of whatever changes imposed by user would be shown on the webpage.

2.1.4 Requirement of Data Management Capability

- Security:
  File confidentiality – access of irrelevant individual to the system should be prevented;
  Server security – server should be able to block the common hacks preformed by hackers.

- Performance:
  The server-side of the system should build up a permanent connection to the database, so as to avoid the wastage by reconnecting the database.

2.2 Operating Environment

2.2.1 Minimum System Requirement

Server:

- CPU: ≥2.0GHz (for example Intel Mobile Core 2 Duo T5800)
- Memory: ≥4.0GB
- Keyboard: Usable
- Mouse: Not required if under linux environment
- Monitor: Usable
- Hard Drive: ≥100GB, ≥7200rpm
- Network Interface Card: 100M
- Network Access: Local Area Network

Client:

- OS: Windows Vista or newer; Mac OS X 10.7 or newer; Mobile Platform not supported.
- Network Access: Local Area Network

2.2.2 Supported Software

Server:

- Windows Server 2012 or newer; Windows 8 or newer; Mac OS X 10.9 or newer; Ubuntu 13.04 or newer
- PHP 5 or newer version support
- MySQL support
- Apache support
- Web-Browser: Microsoft Internet Explorer 10 or newer; Safari 7.0.3; Chrome 34.0.1847.137 or newer
- Adobe PDF viewer
- Office 2010 or newer version; Pages 5.2 or newer version, Numbers 3.2 or newer version, Keynote 6.2 or newer version

Client:

- OS: Windows Vista or newer; Mac OS X 10.7 or newer; Mobile Platform not supported.
- Wed-Browser: Microsoft Internet Explorer 10 or newer; Safari 7.0.3; Chrome 34.0.1847.137 or newer

2.3 Basic Design Concepts and Procedure

This system can be divided into client-side and server-side.
For client-side, HTML and CSS is used to build up the website. Bootstrap is adopted for the UI design. Java-Script/JQuery is also used to.
For server-side, Apache is used as web server, PHP is used as the script language and MySQL is used as the database.

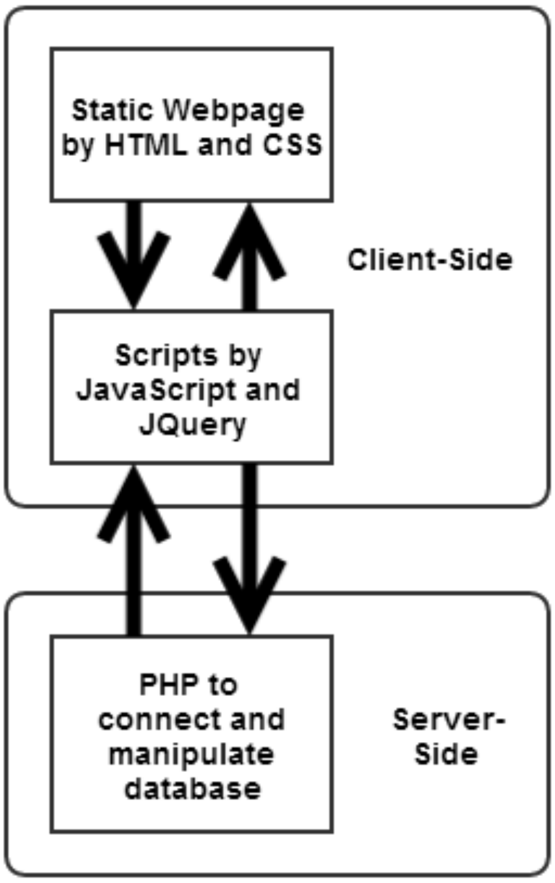
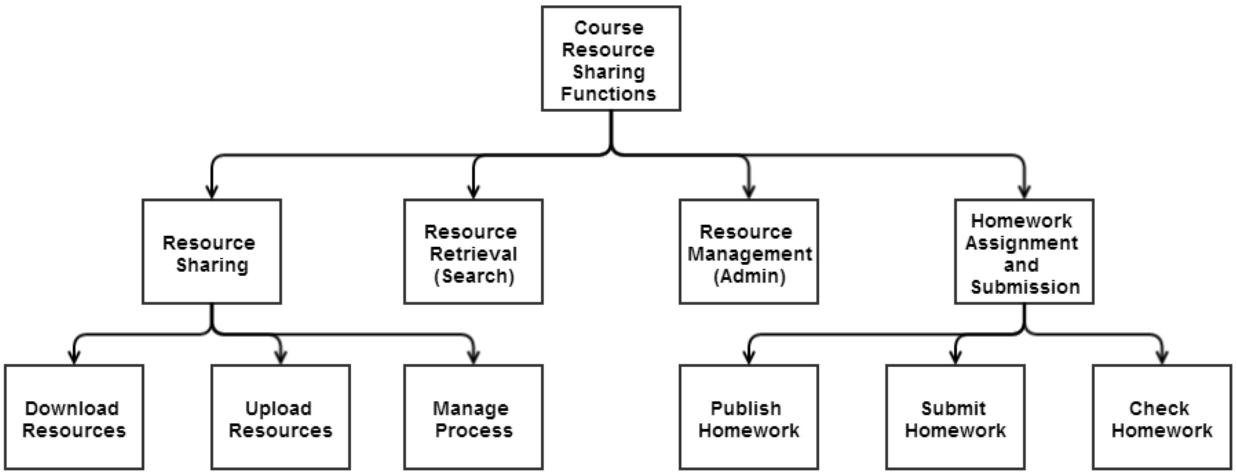

Static Webpage by HTML and CSS
Client-Side
Scripts by JavaScript and JQuery
PHP to connect and manipulate database
Server-Side

Figure 2.1

2.4 Structure

2.4.1 Structural Partitioning by Function (Horizontal partitioning)



Course Resource Sharing Functions
Resource Sharing
Resource Retrieval (Search)
Resource Management (Admin)
Homework Assignment and Submission
Download Resources
Upload Resources
Manage Process
Publish Homework
Submit Homework
Check Homework

Figure 2.2 HIPO of the Course Resource Sharing Subsystem

INPUT	PROCESS	OUTPUT
Click the Download Link Button on the Course Resource page.	Send the request to the Server; Get the requested file ready for downloading.	The Requested Course Resource.

Click the Upload Button on the Course page;
 	
File; Title;
	Send the request to the Server;
	
	Process the Uploading.
		The result.

Select a Course Resource on the Course Resource page.	Send the request to the Server; Receive Course Resource information.	The Course Resource page.

Figure 2.3-1

INPUT	PROCESS	OUTPUT
Select a Course.	Send the request to the Server; Receive Course information.	The Course page.
Click the Homework Assigning Button; Title; Deadline;	Send the request to the Server; Add the assignment to the Database	The result.
Click the Submit Homework Button; Choose the Assignment; Files for upload.	Send the request to the Server; Process the Uploading.	The result.
System Administrator clicks the Modify button; Fill the form.	Send the request to the Server; Modify the data according to the form submitted.	The result.
Click the Modify Button on the Course Resource page; Modify the form.	Send the request to the Server; Modify the data in the database.	The Modified Course Resource page.

Figure 2.3-2

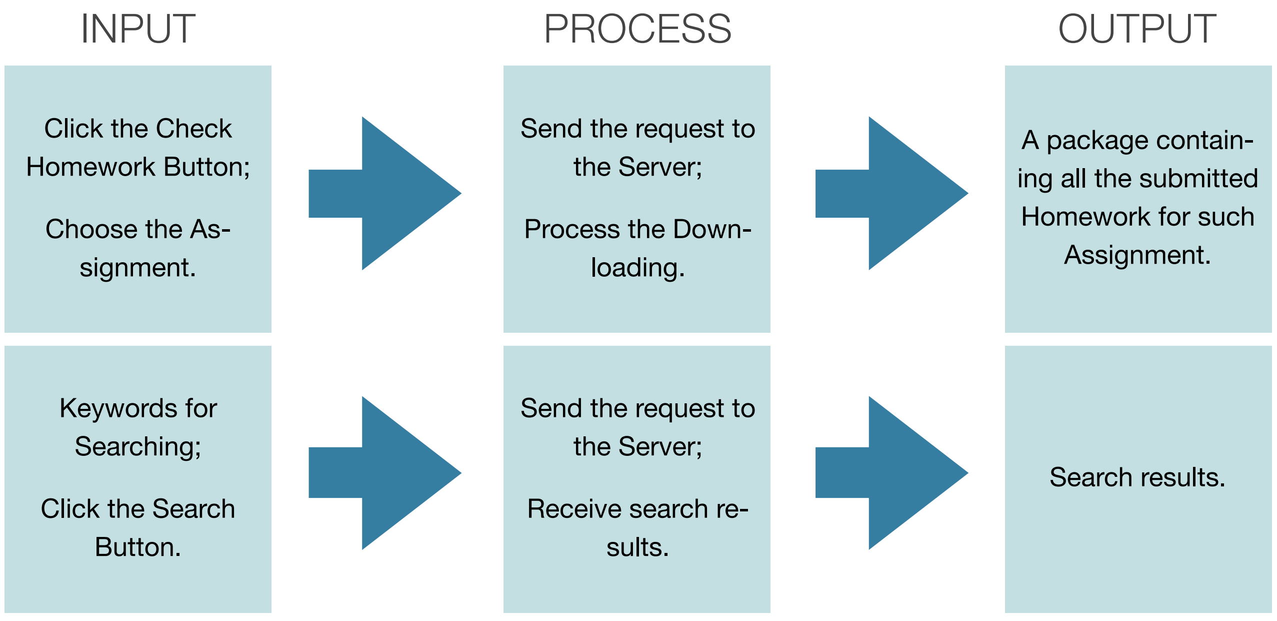

INPUT
PROCESS
OUTPUT
Click the Check Homework Button;
Choose the Assignment.
Send the request to the Server;
Process the Downloading.
A package containing all the submitted Homework for such Assignment.
Keywords for Searching;
Click the Search Button.
Send the request to the Server;
Receive search results.
Search results.

Figure 2.3-3

2.4.2 Structural Partitioning by Function by procedure (Vertical partitioning)

2.4.2.1 Client

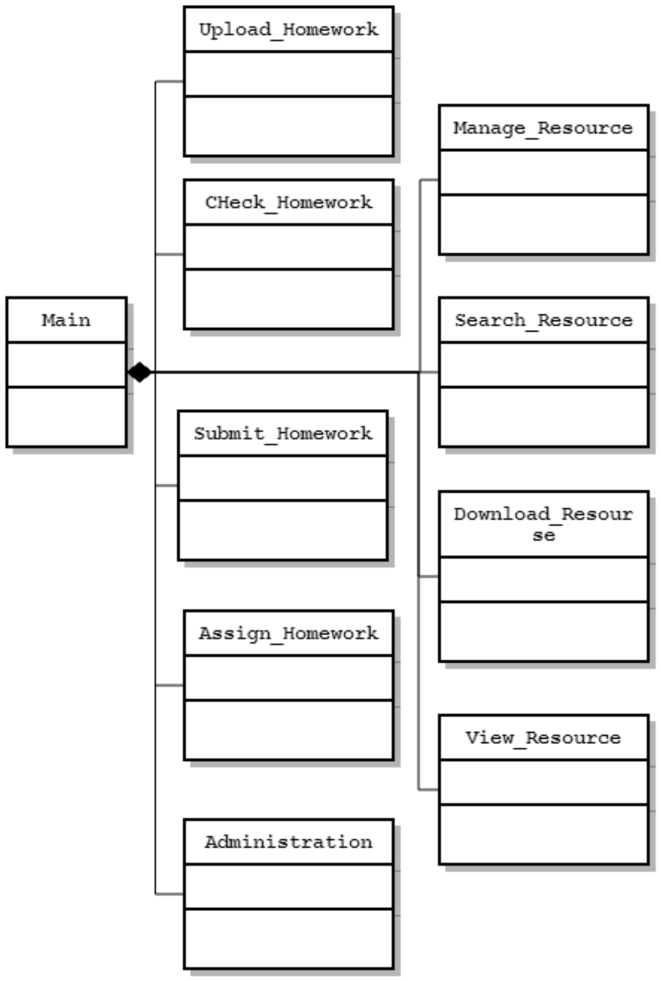


Figure 2.3 Class diagram of client-side programme

2.4.2.2 Server

This part is mainly programmed with PHP.

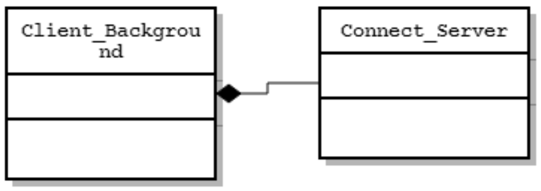


Figure 2.4 Class diagram of server-side programme

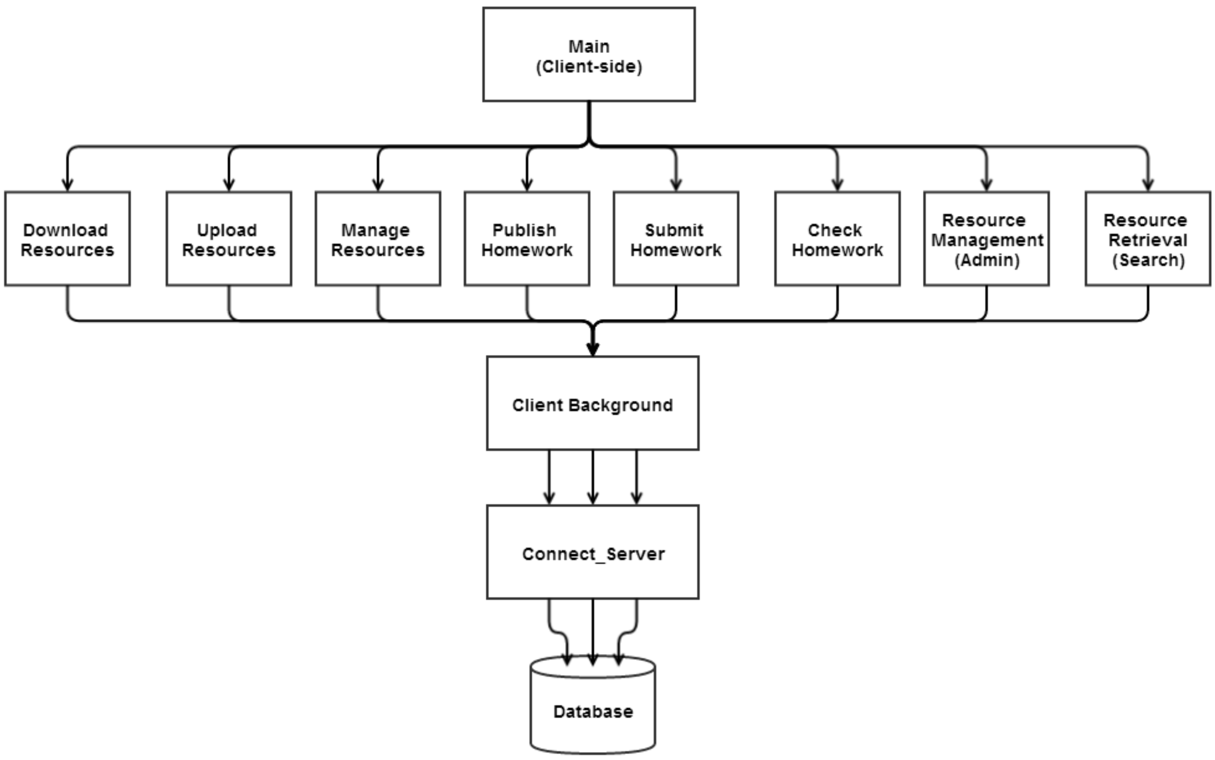


Figure 2.5 Relationship between different modules

2.5 Interface

	Resource Sharing	Homework Assignment & Submission	Resource Management (Administration)	Resource Retrieval (Search)
Client	Share the `Resource` database	Share the `Homework` database	Share the `Resource` and the `Homework` database	Share the `Resource` database

Table 2.1 Database sharing across different modules in table

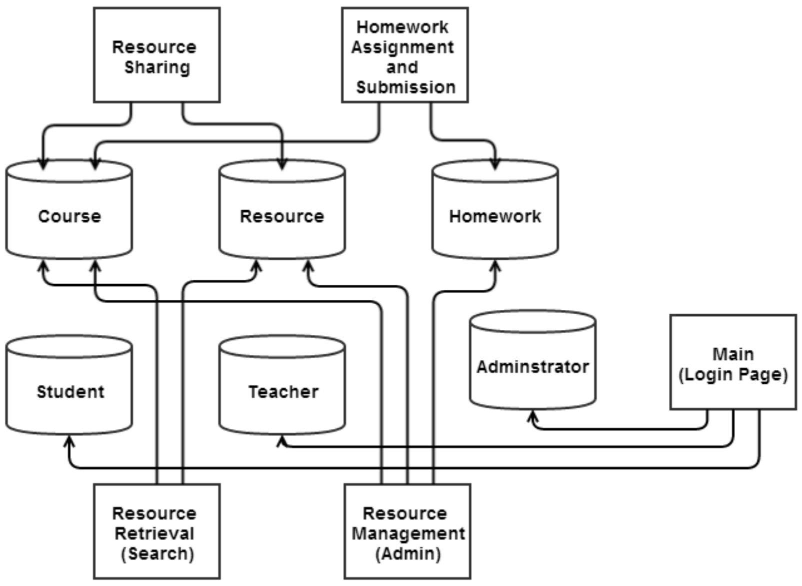

Resource Sharing
Homework Assignment and Submission
Course
Resource
Homework
Student
Teacher
Adminstrator
Main (Login Page)
Resource Retrieval (Search)
Resource Management (Admin)

Figure 2.6 Database sharing across different modules in graph

2.6 Manual Error Handling

If an error occurs and cannot be resolved by the programme itself, it would be handled manually.

2.7 Unsolved issues

The user cannot change his/her password. Also, there's no means to deal with the situation when a user forgets password.

CHAPTER III: DETAILED DESIGN

3.1 Login Module Design

3.1.1 Module Description

This module will be displayed as a webpage form. It is the first interface that user will interact with. Only after passing the authentication, the other functions can be used. This module is to verify the identity of users. It will do the strict security verification when the user first login.

3.1.2 Function

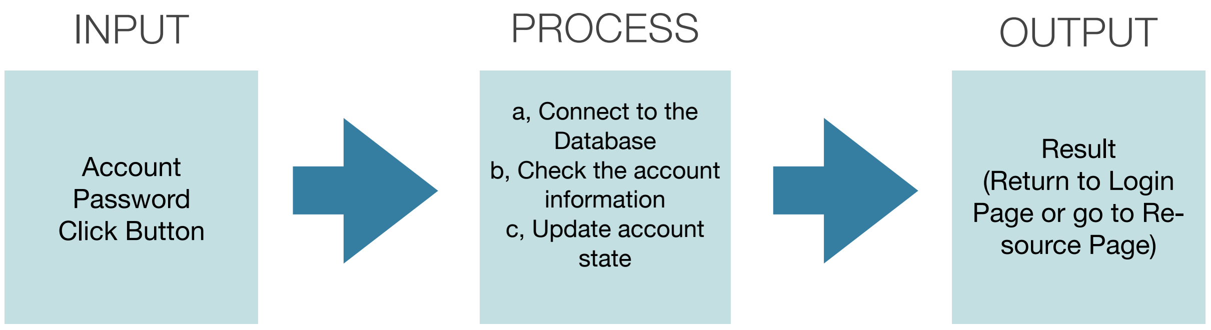

INPUT
Account
Password
Click Button
PROCESS
a, Connect to the Database
b, Check the account information
c, Update account state
OUTPUT
Result
(Return to Login Page or go to Re-source Page)

Figure 3.1.1 IPO of the login module

3.1.3 Property

After clicking on the sign in button with inputting the user ID and the password, the system will do the verification.

3.1.4 Inputs

Name	Identification	Type and form	Input method
Account	User_ID	Int	Type in
Password	Password	String	Type in
Login	Login	Button	Click the button

Table 3.1.1 Input list of user login module

3.1.5 Outputs

Name	Identification	Type and form	Output method
Result of verification	Login_result	Enum{success,fail}	By script

3.1.6 Design Approach

When the user click on the sign in button, the script function Login() will be triggered. The data will be posted to the serve and execute the php code below:

```
<?php
  If(CheckStringSafety(form.user_id)
             &&CheckStringSafety(form.password))
  //check the security of data
  {
             ConnectionDB Db;
             Db.open();
             //Connect to the database
             If(Db.LoginQuery(form.user_id,form.password))
             {
                        //if find the account
                        If(SecurityCheck())
                        //if pass the verification
                        {
                           Session("user_id")=form.user_id;
            Session("user_type")=getUsertype();
                           Session("login_state")=true;
                           Session("time")=time();
                           //store the login state
            $data['result']="Success";
        }
                        else
                        {
                                   $data['result']="Fail";
                        }
             }
             else
             {
                        $data['result']="Fail";
             }
     Db.close();
     //close database connection
  }
  else
  {
             $data['result']="Warn";
  }
  echo json_encode($data);
             //return the result of verification
?>
```

After executing, the result will be send back, the script of the front-end will display the result of verification.

3.1.7 Flow Logic

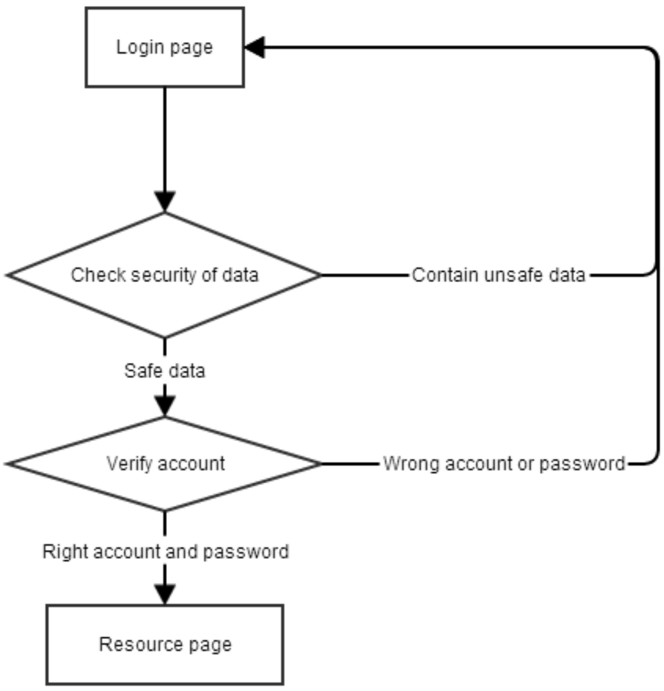
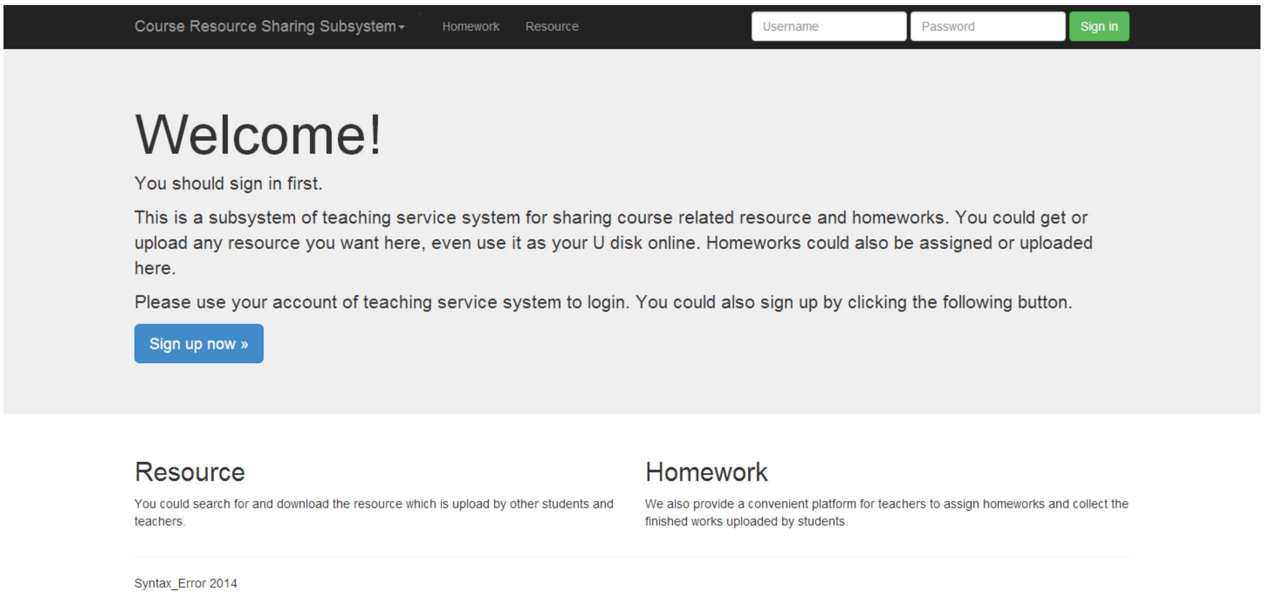


Figure 3.1.2 Flow chart of login module

3.1.8 User Interface



Welcome!

You should sign in first.

This is a subsystem of teaching service system for sharing course related resource and homeworks. You could get or upload any resource you want here, even use it as your U disk online. Homeworks could also be assigned or uploaded here.

Please use your account of teaching service system to login. You could also sign up by clicking the following button.

Resource

You could search for and download the resource which is upload by other students and teachers.

Homework

We also provide a convenient platform for teachers to assign homeworks and collect the finished works uploaded by students.

Syntax_Error 2014

Figure 3.1.3 User interface of the login module

3.1.9 Test Plan

The test requires creating corresponding database and test data on the remote server.

Input	Expected result
Existed user ID and password	Gain access of other functions
Existed user ID and wrong password	Return error message
Existed user ID and password but not corresponding	Return error message
User ID or password that contain special characters or sql command	Return warning

Table 3.1.2 Test plan of login module

3.2 Personal Resource Management Module Design

3.2.1 Module Description

This module is designed for the customer to view, download and change the resources online. Divided by privilege, customer with different account could apply different operation to specific resource which is available to their account. In addition, any customer could view and deal with the resources uploaded by him in a special page. Which means that this module could also be used as a personal cloud.

3.2.2 Function

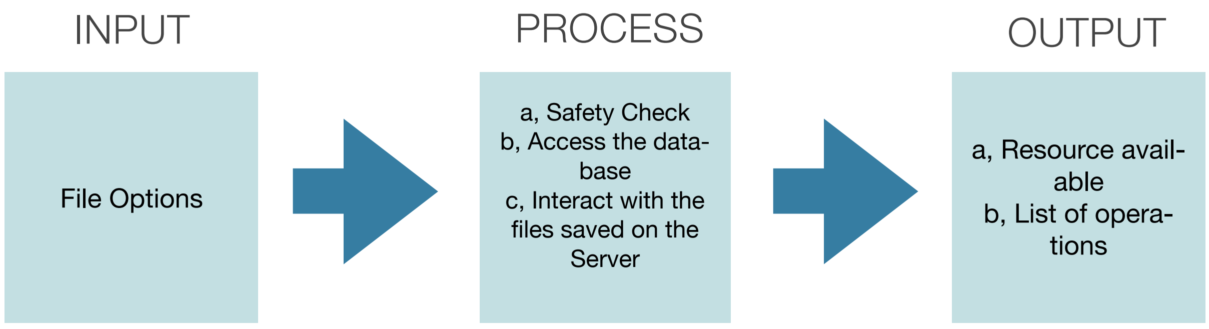


Figure 3.2.1 IPO of module 2

3.2.3 Property

View resources online and click the download link to download.

3.2.4 Inputs

Name	Signal	Type and format	Input format
Upload date of resource	Resource_Date	Date	Textbox
The uploader's name	Resource_Uploader	Varchar	Textbox
The illustration of resource	Resource_Illustration	Varchar	Textbox
The level of resource	Resource_Level	Varchar	Textbox

Table 3.2.1 Input list of module 2

3.2.5 Outputs

Name	Signal	Type and format	Input format
Course Name	Course_Name	Varchar	Textbox
The illustration of course	Course_Illustration	Varchar	Textbox

Name	Signal	Type and format	Input format
Resource	Resource	File	Download link
The name of resource	Resource_Name	Varchar	Textbox
Upload date of resource	Resource_Date	Date	Textbox
The uploader's name	Resource_Uploader	Varchar	Textbox
The illustration of resource	Resource_Illustration	Varchar	Textbox
The level of resource	Resource_Level	Varchar	Textbox

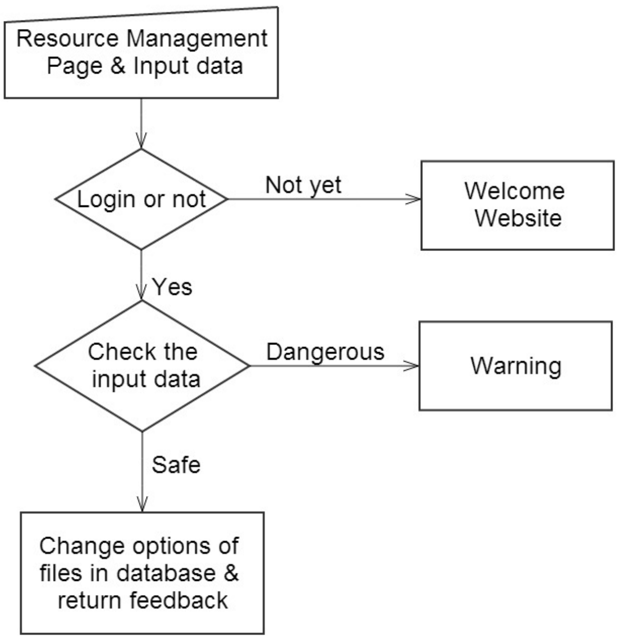
Table 3.2.2 Output list of user login module

3.2.6 Design Approach

```
<?php
   if(CheckStringSafety(form.Resource_Name)
             &&CheckStringSafety(form.Resource_Date)
             &&CheckStringSafety(form.Resource_Uploader)
             &&CheckStringSafety(form.Resource_Illustration)
             &&CheckStringSafety(form.Resource_Level)){
//Safety monitoring
             ConnectionDB Db;
             Db.open();
             //Connect to the database
             if(Session("login"==True))&&(form.Resource_Name!="")
             //Login authentication
             {
                       //If already login

                       if(form.Resource_Date!=""){
                                  $query = "update file set Resource_Date=".$Resource_Date." where (select *
form file where Resource_Name=".$Resource_Name")";
                                  $result = $db->query($query);
                       }

                       if(form.Resource_Uploader!=""){
                                  $query = "update file set Resource_Uploader=".$Resource_Uploader." where
(select * form file where Resource_Name=".$Resource_Name")";
                                  $result = $db->query($query);
                       }

                       if(form.Resource_Illustration!=""){
                                    $query = "update file set Resource_Commit=".$Resource_Illustration."
where (select * form file where Resource_Name=".$Resource_Name")";
                                  $result = $db->query($query);
                       }

                       if(form.Resource_Level!=""){
                                    $query = "update file set Resource_Level=".$Resource_Level." where
(select * form file where Resource_Name=".$Resource_Name")";
```

```
                                        $result = $db->query($sql);
                              }
//Update data to database
                    }
                    else
                    {
                              //not login correctly
                              $data['result']="Please login";
                              header("location: ./welcome.html");
                    }
          Db.close();
          //close database connection
     }
     echo json_encode($data);
     //Return the result
?>
```

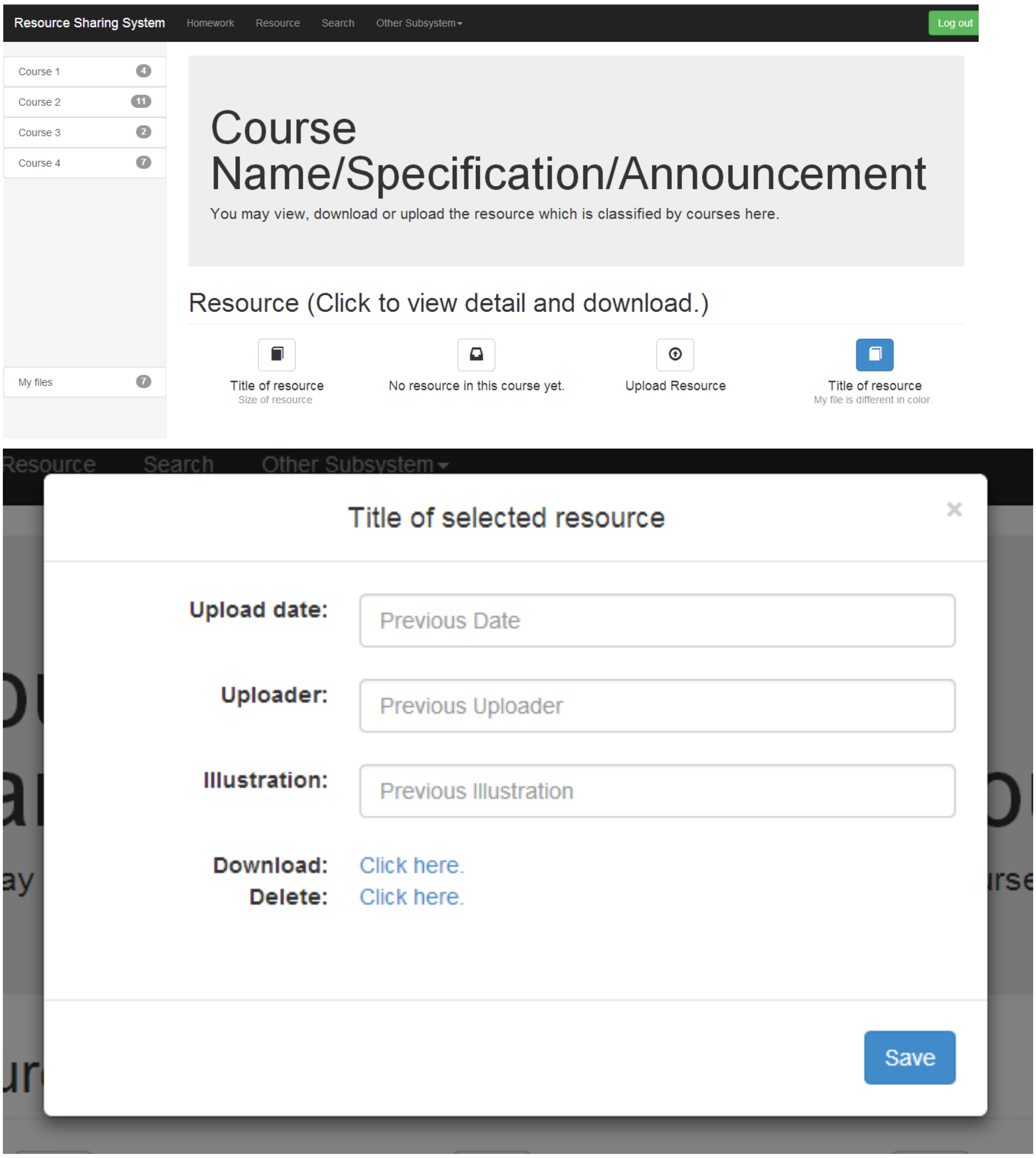
3.2.7 Process and Logic



Resource Management Page & Input data
Login or not
Not yet
Welcome Website
Yes
Check the input data
Dangerous
Warning
Safe
Change options of files in database & return feedback

Figure 3.2.2 Flow chart of login module

3.2.8 User Interface



Course Name/Specification/Announcement

You may view, download or upload the resource which is classified by courses here.

Resource (Click to view detail and download.)

Title of selected resource

Upload date:	Previous Date
Uploader:	Previous Uploader
Illustration:	Previous Illustration
Download:	Click here.
Delete:	Click here.

Figure 3.2.3 User Interface of the Module 2

3.2.9 Test Plan

Input data	Expected Result
Normal options of file.	None.(The options of file successfully changed)
Text including SQL sentence and illegal symbols.	Warning message.
The user didn't log in.	Jump to welcome website.

Table 3.2.3 Test plan of Module 2

3.3 Personal Resource Management Module 2

3.3.1 Module Description

This module is designed for the customer to upload a file as a resource to the server. It shares the same website with module 2.

3.3.2 Function

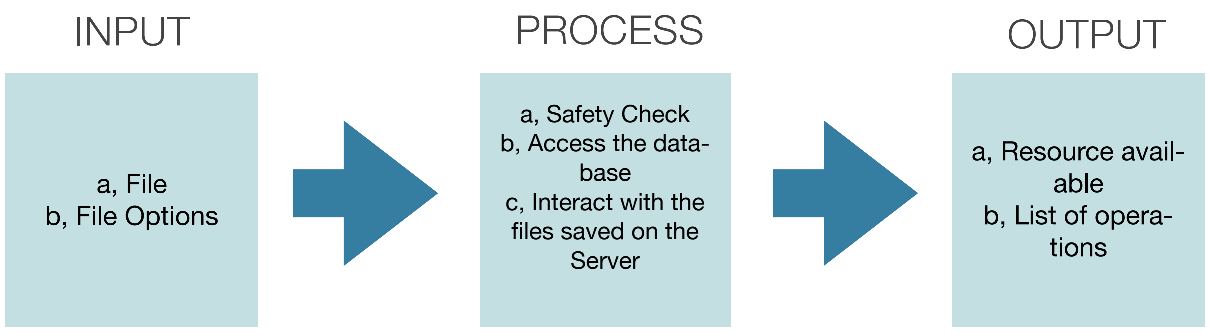


Figure 3.3.1 IPO Graph

3.3.3 Property

Select file and input file options to upload a file as the resource to the cloud.

3.3.4 Inputs

Name	Signal	Type and format	Input format
Resource	Resource	File	Upload button
The illustration of resource	Resource_Illustration	Varchar	Textbox

Table 3.3.1 Input list

3.3.5 Outputs

None output items are involved in this module.

3.3.6 Design Approach

```
<?php
    if ((($_FILES["file"]["type"] == "image/gif")
    || ($_FILES["file"]["type"] == "image/jpeg")
    || ($_FILES["file"]["type"] == "image/pjpeg"))
    && ($_FILES["file"]["size"] < 20000))
    && CheckStringSafety(form.Resource_Illustration)
//Safety monitoring
                {
                        if (Session("login"==True))
                        //Login authentication
```

```
                                        {
                                        //If already login
                                                    ConnectionDB Db;
                                                    Db.open();
                                                    //Connect to the database
                                                    $query  =  "insert  into  file  values  (".$_FILES["file"]["tmp_name"].",".
$_FILES["file"]["type"].",".$_FILES["file"]["size"].",".form.Resource_Illustration.")";
                                                    $result = $db->query($query);
                                            Db.close();
                                            //close database connection
                                                    if ($_FILES["file"]["error"] > 0)
                                                        {
                                                                $data['result']="Return Code: " . $_FILES["file"]["error"] . "<br />";
                                                        }
                                                    else
                                            {

                                            if (file_exists("upload/" . $_FILES["file"]["name"]))
                                                {
                                                    $data['result']=$_FILES["file"]["name"] . " already exists. ";
                                                }
                                            else
                                                {
                                                move_uploaded_file($_FILES["file"]["tmp_name"],
                                                "upload/" . $_FILES["file"]["name"]);
                                                }
                                            }
                                        //Upload file
                                        }
                            }
                else
                {
                            $data['result']="Invalid file or input";
                }
            echo json_encode($data);
            //Return the result
            ?>
```

3.3.7 Process and Logic

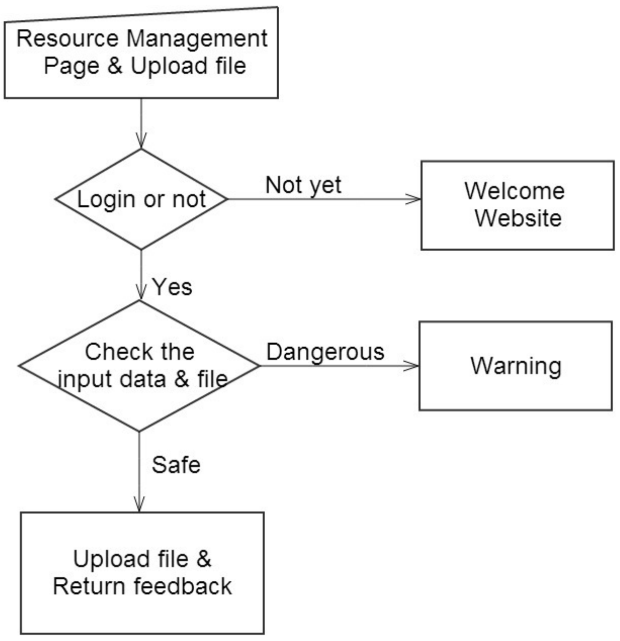

Resource Management Page & Upload file
Login or not
Not yet
Welcome Website
Yes
Check the input data & file
Dangerous
Warning
Safe
Upload file & Return feedback

Figure 3.3.2 Flow chart of login module

3.3.8 User Interface

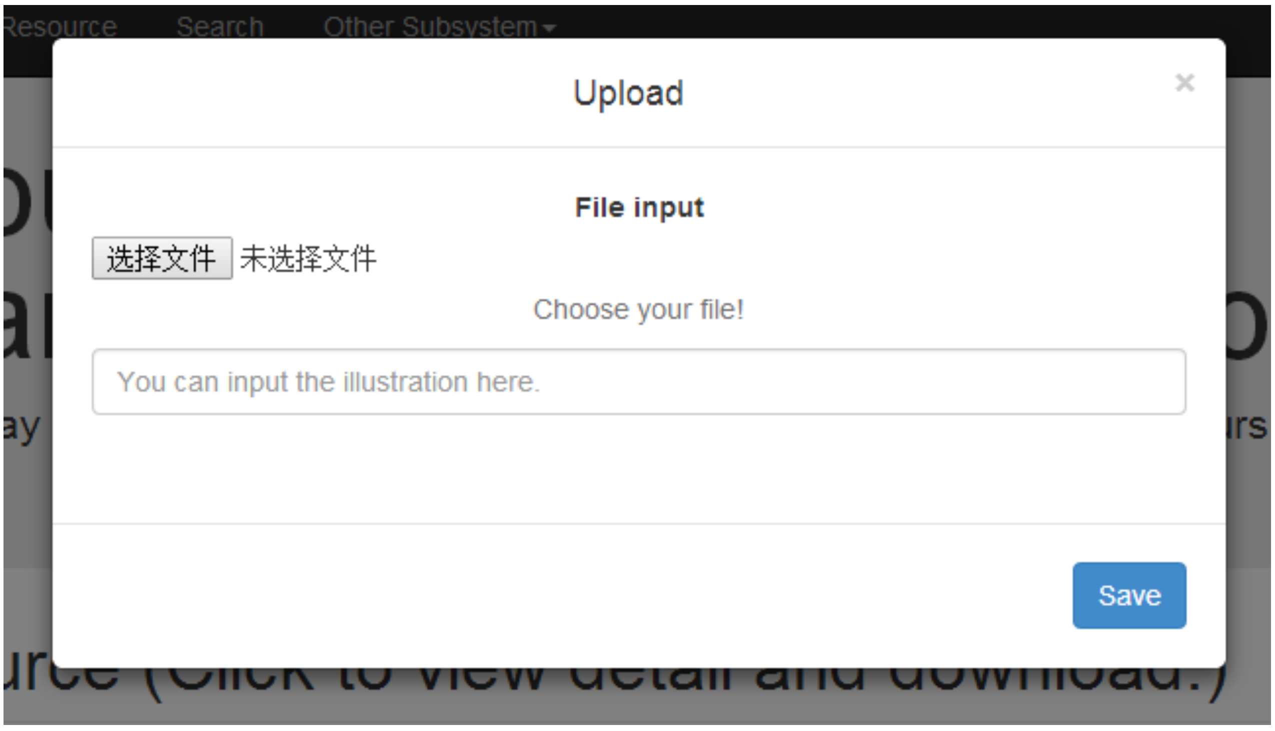


Figure 3.3.3 User Interface

3.3.9 Test Plan

Input data	Expected Result
Normal files and illustration.	A message of successfully uploaded.
Text including SQL sentence and illegal symbols.	Warning message.
The user didn't log in.	Jump to welcome website.

Table 3.3.2 Test plan

3.4 Search Module Design

3.4.1 Module Description

This module is designed for the customer to search a file from all the homework and resources. This module provides two way to search and several optional search items to help the customer to perform a search with high-accuracy.

3.4.2 Function

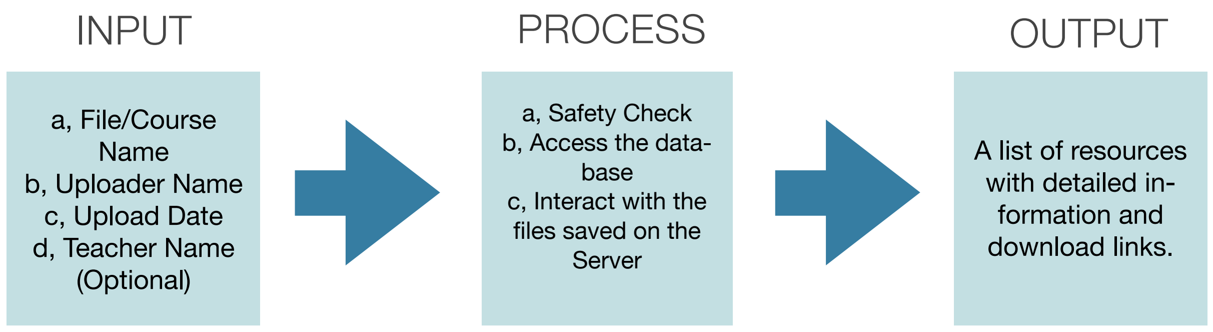


Figure 3.4.1 IPO Graph

3.4.3 Property

Input keywords to search for resources and homework in the server and return the related files with its download link.

3.4.4 Inputs

Name	Signal	Type and format	Input format
The name of resource	Resource_Name	Varchar	Textbox
The name of homework	Homework_Name	Varchar	Textbox
The name of course	Course_Name	Varchar	Textbox
The name of uploader	Uploader_Name	Varchar	Textbox
The upload date	Upload_Date	Varchar	Textbox
The name of teacher	Teacher_Name	Varchar	Textbox

Table 3.4.1 Input list

3.4.5 Outputs

Name	Signal	Type and format	Input format
Resource	Resource	File	Download Link

Name	Signal	Type and format	Input format
Homework	Homework	File	Download Link
The name of resource	Resource_Name	Varchar	Textbox
The name of homework	Homework_Name	Varchar	Textbox
Course Name	Course_Name	Varchar	Textbox
The illustration of course	Course_Illustration	Varchar	Textbox

Table 3.4.2 Output list

3.4.6 Design Approach

```
<?php
  if(CheckStringSafety(form.File_Name)
          &&(form.Course_Name)
          &&(form.Uploader_Name)
          &&(form.Upload_Date)
          &&(form.Teacher_Name)){
          //Safety monitoring
          ConnectionDB Db;
          Db.open();
          $File_Name=form.File_Name;
          $Course_Name=form.Course_Name;
          $Uploader_Name=form.Uploader_Name;
          $Upload_Date=form.Upload_Date;
          $Teacher_Name=form.Teacher_Name;
          //Connect to the database
          if(Session("login"==True))&&(form.Resource_Name!="")
          //Login authentication
          {
                  //If already login
                  if(form.File_Name!=""){
                  $query = "select * from file where (file_name like '%".$File_Name."%') and
(Course_Name like '%".$Course_Name."%') and (Uploader_Name like '%".$Uploader_Name."%') and
(Upload_Date like '%".$Upload_Date."%')";
                  $result = $db->query($query);
                  }

                  if(form.Course_Name!=""){
                  $query = "select * from course where Course_Name like '%".$Course_Name."%' and
(Teacher_Name like '%".$Teacher_Name."%')";
                  $result = $db->query($query);
                  }
                  //Search in the Database
                  $num_results = $result->num_rows;

                  for ($i=0; $i <$num_results; $i++) {
                    $row = $result->fetch_assoc();
                    $data['array_file_name']=$row['file_name'];
                    $data['array_course_name']=$row['course_name'];
```

```
                        $data['array_course_illustration']=$row['course_illustration'];
                    }
            }
            else
            {
                    //not login correctly
                    $data['result']="Please login";
                    header("location: ./welcome.html");
            }
        Db.close();
        //close database connection
    }
    echo json_encode($data);
    //Return the result
?>
```

3.4.7 Process and Logic

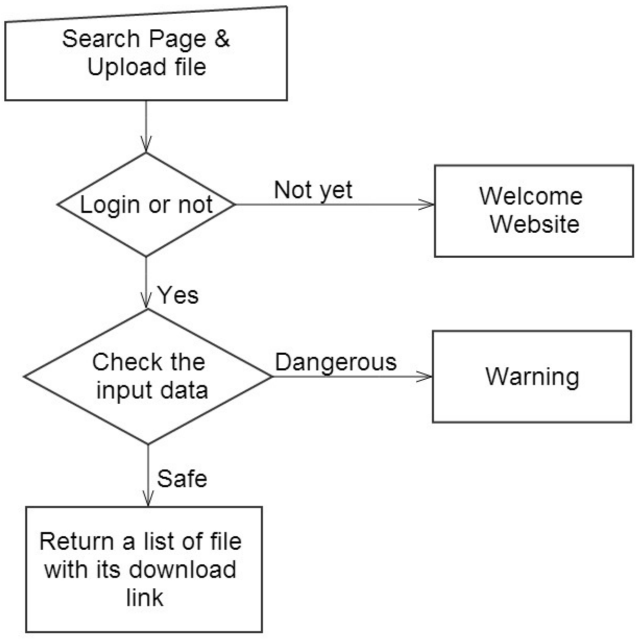
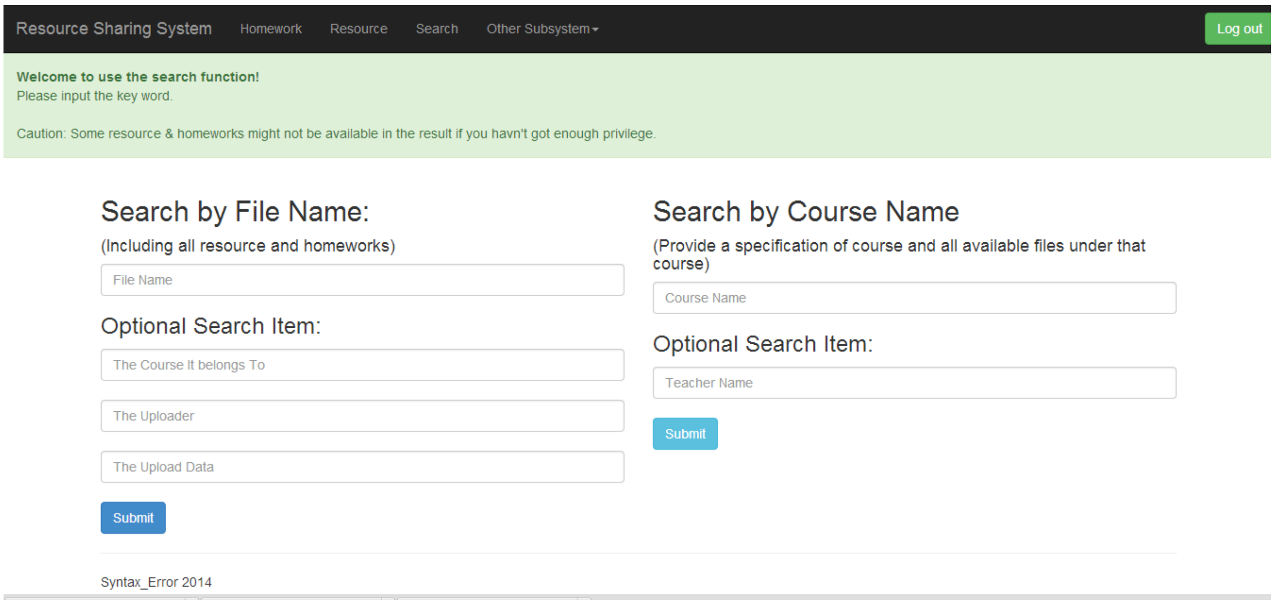


Figure 3.4.2 Flow chart

3.4.8 User Interface



Welcome to use the search function!
Please input the key word.

Caution: Some resource & homeworks might not be available in the result if you havn't got enough privilege.

Search by File Name:
(Including all resource and homeworks)

File Name

Optional Search Item:

The Course It belongs To

The Uploader

The Upload Data

Submit

Search by Course Name
(Provide a specification of course and all available files under that course)

Course Name

Optional Search Item:

Teacher Name

Submit

Syntax_Error 2014

Figure 3.4.3 User Interface

3.4.9 Test Plan

Input data	Expected Result
Normal input of keyword.	Jump to the result of search.
Text including SQL sentence and illegal symbols.	Warning message.
The user didn't log in.	Jump to welcome website.

Table 3.4.3 Test plan

3.5 Homework Assign Module Design

3.5.1 Module Description

This module is for the Teaching Staff to assign homework for the Courses. Only after logging in as the Teaching Staff, shall they have access to this module.

3.5.2 Function

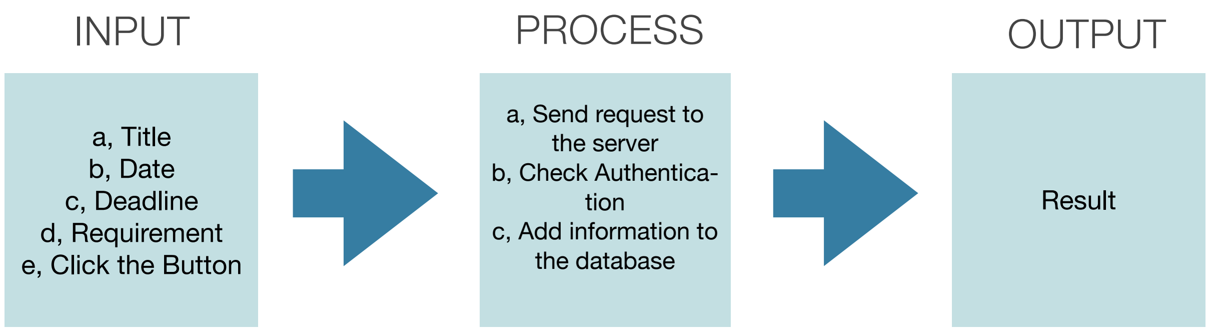


Figure 3.5.1 IPO Graph

3.5.3 Property

This module will show the courses the instructor are teaching on the left. After choosing one course, the assigned homework will be displayed on the right. And there will be a button for the instructor to assign new homework. After clicking on the button, a form will be popped up. The input data will be posted to serve.

3.5.4 Inputs

Name	Identification	Type and form	Input method
Title	Title	String	Type in
Assign data	Assign_data	Date	Type in
Deadline	Deadline	Date	Type in
Requirement	Requirement	String	Type in
Save	Save	Button	Click

Table 3.5.1 Input list

3.5.5 Outputs

Name	Identification	Type and form	Output method
Result of submitting	Assign_result	Enum{success,fail,not_teacher}	By script

3.5.6 Design Approach

After filling the form and submitting, the php code below will be executed:

```
<?php
   If(CheckStringSafety(form.title)
            &&CheckStringSafety(form.requirement))
   //check the security of data
   {
            ConnectionDB Db;
            Db.open();
            //Connect to the database
            If(Session("login_state")==True&&Session("user_type")=="teacher")
            //check authentication
            {
                     //if already login
                     If(AddAssignment(form))
                     //if add assignment successfully
                     {
           $data['result']="Success";
       }
                     else
                     {
                              $data['result']="Fail";
                     }
            }
            else
            {
                     $data['result']="Not_teacher";
            }
      Db.close();
      //close database connection
   }
   else
   {
            $data['result']="Warn";
   }
   echo json_encode($data);
            //return the result of assignment
?>
```

After executing, the result will be send back, the script of the front-end will display the result of assignment.

3.5.7 Process and Logic

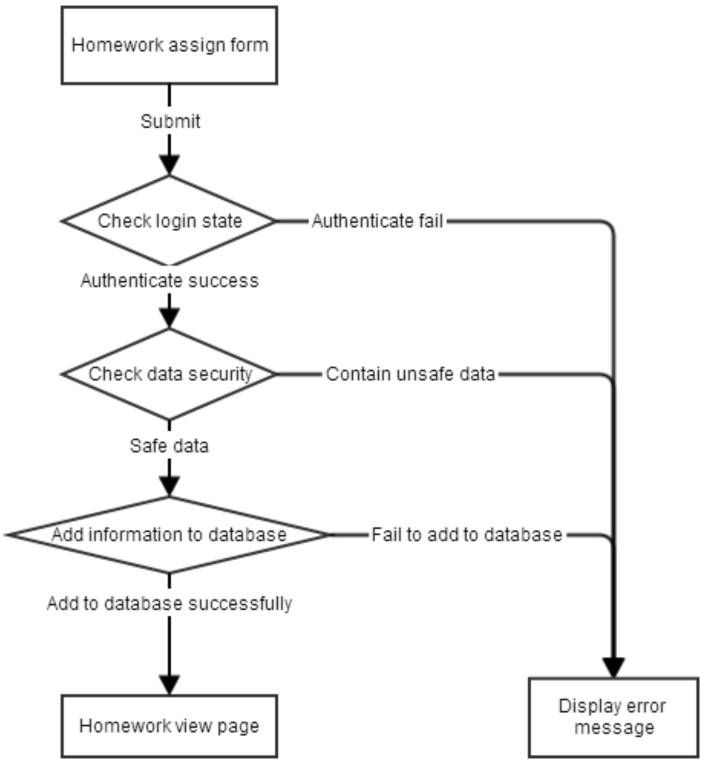
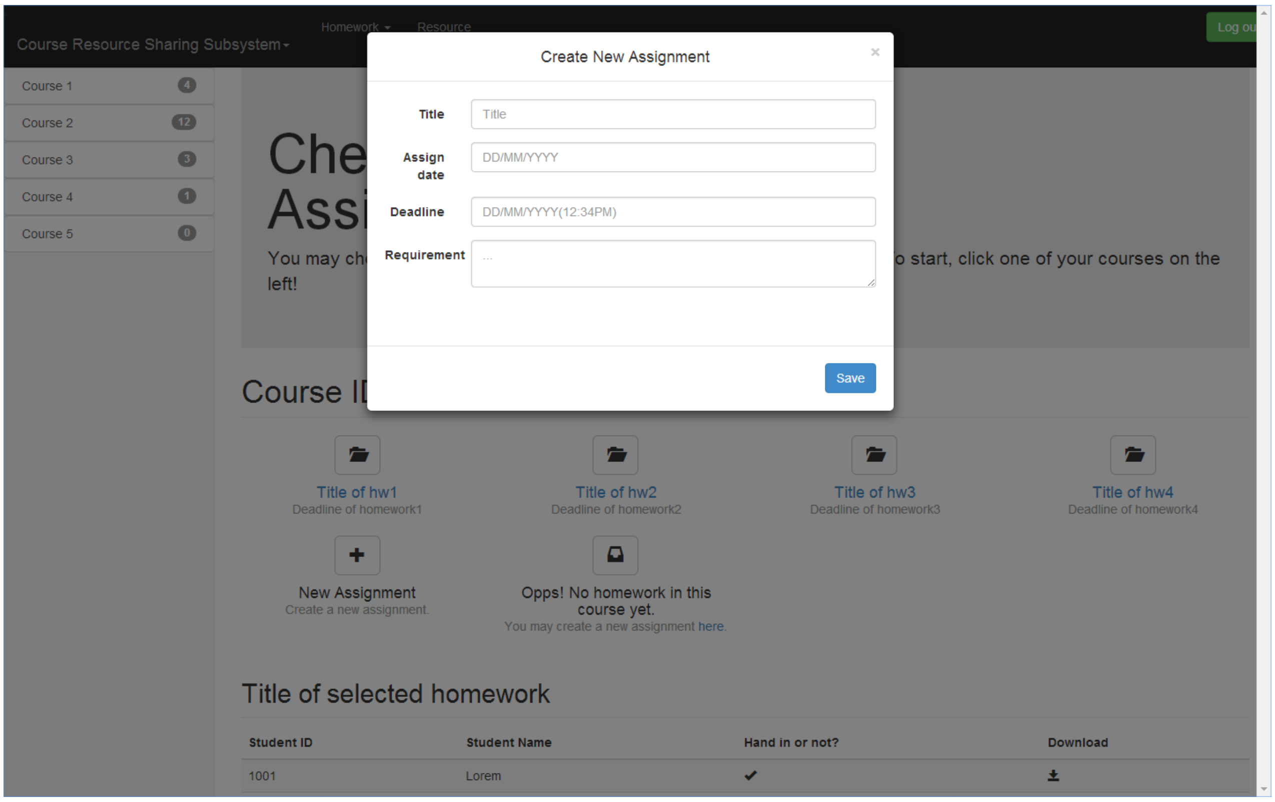


Figure 3.5.2 Flow chart

3.5.8 User Interface



Che
Assi
You may ch   o start, click one of your courses on the left!

Title: Title

Assign date: DD/MM/YYYY

Deadline: DD/MM/YYYY(12:34PM)

Requirement: ...

Course I

Opps! No homework in this course yet.
You may create a new assignment here.

Title of selected homework

Student ID	Student Name	Hand in or not?	Download
1001	Lorem	✓	⤓

Figure 3.5.3 User Interface

3.5.9 Test Plan

Input	Expected result
Correct title, date and requirement	Add assignment successfully
Correct title and requirement with wrong date format	Return error message
Assign date or deadline is illogic	Return error message
Title or requirement that contain special characters or sql command	Return warning
Correct title, date and requirement but log information timeout	Return error message

Table 3.5.3 Test plan

3.6 Homework Upload Module Design

3.6.1 Module Description

This module is to upload files to serve from Students' computers. After logged in, Students can view their courses and homework and select a homework to upload.

3.6.2 Function

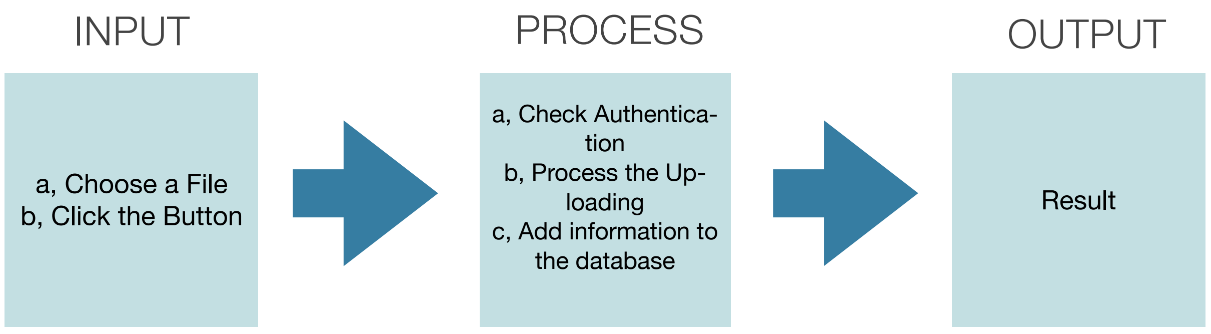


Figure 3.6.1 IPO Graph

3.6.3 Property

After clicking on the submit link, a upload form will be displayed. Students can choose the file to upload and click the save button to submit.

3.6.4 Inputs

Name	Identification	Type and form	Input method
File	File	File	Choose from browser
Save	Save	Button	Click the button

Table 3.6.1 Input list

3.6.5 Outputs

Name	Identification	Type and form	Output method
Result of upload	Upload_result	Enum{success,fail,not_student, invalid_file}	By script

Table 3.6.2 Output list

3.6.6 Design Approach

When the user click on the save button, the script function Upload() will be triggered. The data will be posted to the serve and execute the php code below:

```
<?php
```

```
ConnectionDB Db;
Db.open();
//Connect to the database
If(Session("login_state"==True&&Session("user_type")=="student")
//check authentication
{
        //if already login
        if ($_FILES["file"]["size"] < MAXFILESIZE)
        //check file size
        {
                if ($_FILES["file"]["error"] > 0)
                //if fail to upload
                {
                        $data['error']=$_FILES["file"]["error"];
                }
                else
                {
                //if upload successfully
                        $data['error']=0;
                        $tmpname=generateTmpName($_FILES["file"]["name"]);
                        //generate a new name for the file

                        while (file_exists("upload/" . $tmpname))
                        //avoid overwrite the existed file
                        {
                                $tmpname=generateTmpName($_FILES["file"]["name"]);
                        }

                        if(AddFileInfoToDB($_FILES["file"]["name"],$_FILES["file"]["type"],
  $_FILES["file"]["size"], $tmpname, $_POST["course_ID"]))
                        {
                        //add information to database
                                move_uploaded_file($_FILES["file"]["tmp_name"],
                                        "upload/" . $tmpname);
                                //move the uploaded file to upload folder
                        }
                        else
                        {
                                $data['error']=-3;
                        }
                }
        }
        else
        {
        //if the file is invalid
                $data['error']=-1;
        }
}
else
{
//not login correctly
        $data['error']=-2;
}
Db.close();
//close database connection
```

```
    echo json_encode($data);
    //return the result of uploading
?>
```

After executing, the result will be send back, the script of the front-end will display the result of uploading.

3.6.7 Process and Logic

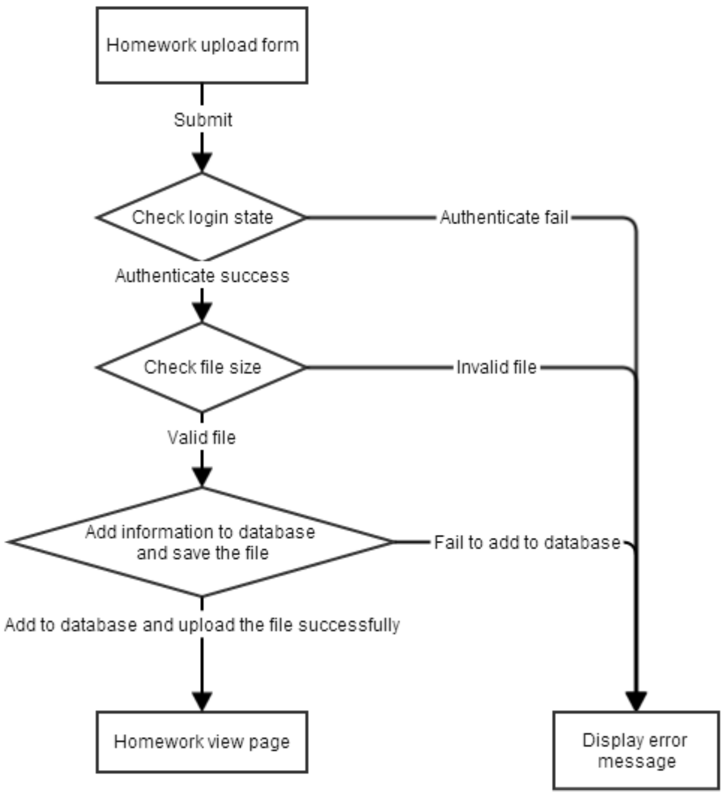
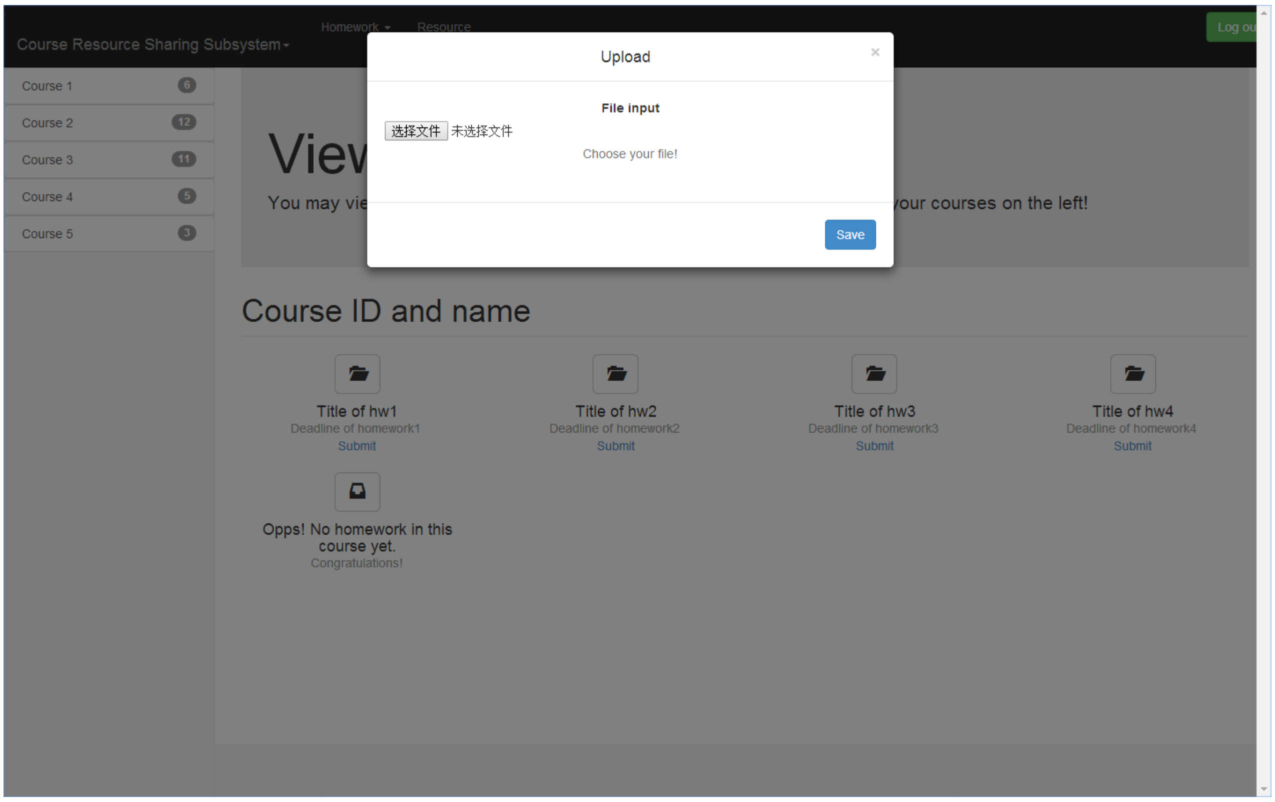


Figure 3.6.2 Flow chart

3.6.8 User Interface



Viev

You may vie … your courses on the left!

Course ID and name

Title of hw1
Deadline of homework1

Title of hw2
Deadline of homework2

Title of hw3
Deadline of homework3

Title of hw4
Deadline of homework4

Opps! No homework in this course yet.
Congratulations!

Figure 3.6.3 User Interface

3.6.9 Test Plan

Input	Expected result
Valid file	Return success message
Invalid file	Return error message
Valid file but log information timeout	Return error message

Table 3.6.3 Test plan

3.7 Homework Check Module Design

3.7.1 Module Description

This module is for the Teaching Staff to check their Students' homework. It will display the homework submitting information and provide the download function.

3.7.2 Function

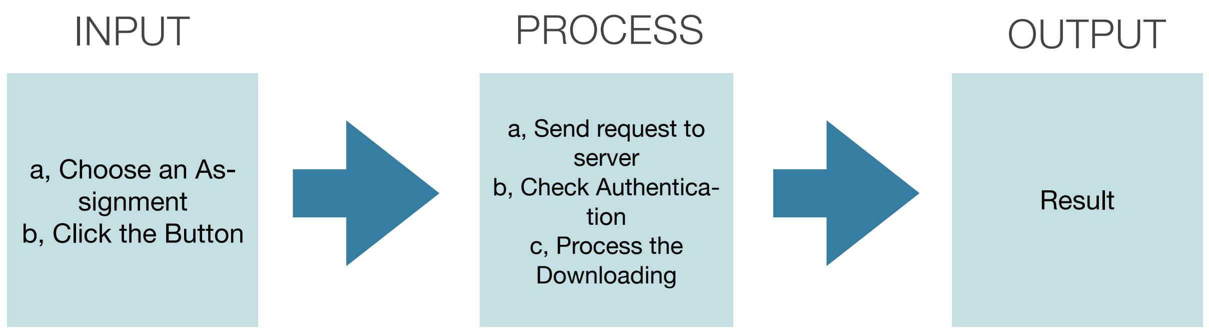


Figure 3.7.1 IPO Graph

3.7.3 Property

After choose a specific homework, the page will display the students list and the submit states. For the students who have already submitted, the instructor can click the download button on the right.

3.7.4 Inputs

Name	Identification	Type and form	Input method
Download	Button	Button	Click the button

Table 3.7.1 Input list

3.7.5 Outputs

Name	Identification	Type and form	Output method
Homework	Down_file	file	By script
Message	Down_msg	String	By script

Table 3.7.2 Output list

3.7.6 Design Approach

When the user click on the download button, the script function Download() will be triggered. The data will be posted to the serve and execute the php code below:

```
<?php
```

```
    ConnectionDB Db;
    Db.open();
    //Connect to the database
    If(Session("login_state"==True&&Session("user_type")=="teacher")
    //check authentication
    {
                //if already login
                $file=getFromDB($_POST["File_ID"],$);
                //get file's information from database
                $file_path="upload/" . $file.['tempname'];
                if(!file_exists($file_path)
                //check the file
                {
                            echo "File not existed";
                }
                else
                {
                            $fp=fopen($file_path,"r");
                            $file_size=filesize($file_path);
                            //The header of file
                            Header("Content-type: application/octet-stream");
                            Header("Accept-Ranges: bytes");
                            Header("Accept-Length:".$file_size);
                            Header("Content-Disposition: attachment; filename=".$file_name);
                            $buffer=1024;
                            $file_count=0;
                            //Send file to browser
                            while(!feof($fp) && $file_count<$file_size)
                            {
                                        $file_con=fread($fp,$buffer);
                                        $file_count+=$buffer;
                                        echo $file_con;
                            }
                            fclose($fp);
                }
    }
    else
    {
    //not login correctly
                echo "Please login";
    }
    Db.close();
    //close database connection
?>
```

After executing, the file or the error message will be sent back.

3.7.7 Process and Logic

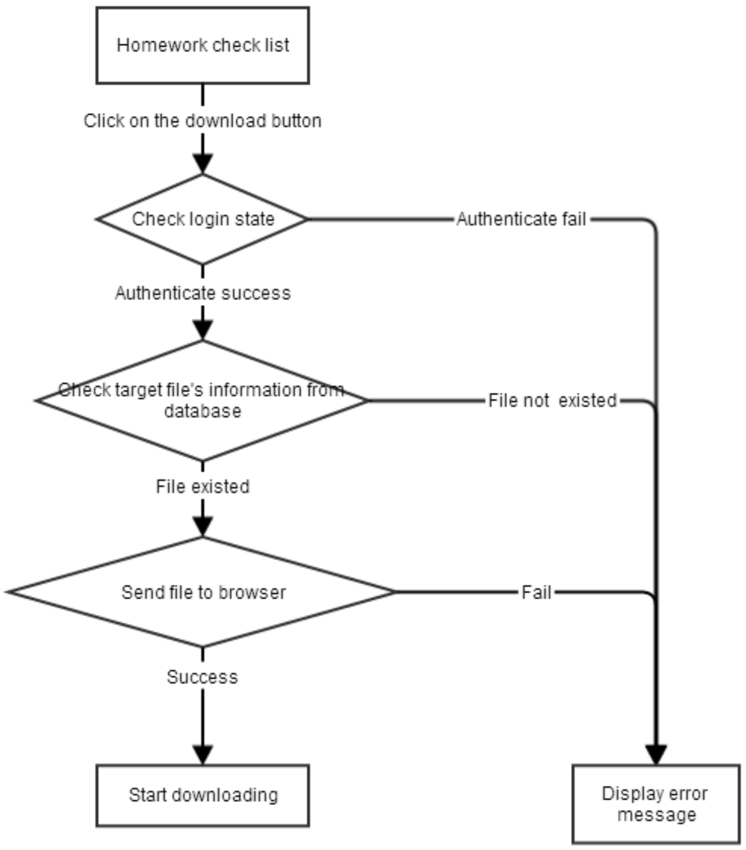
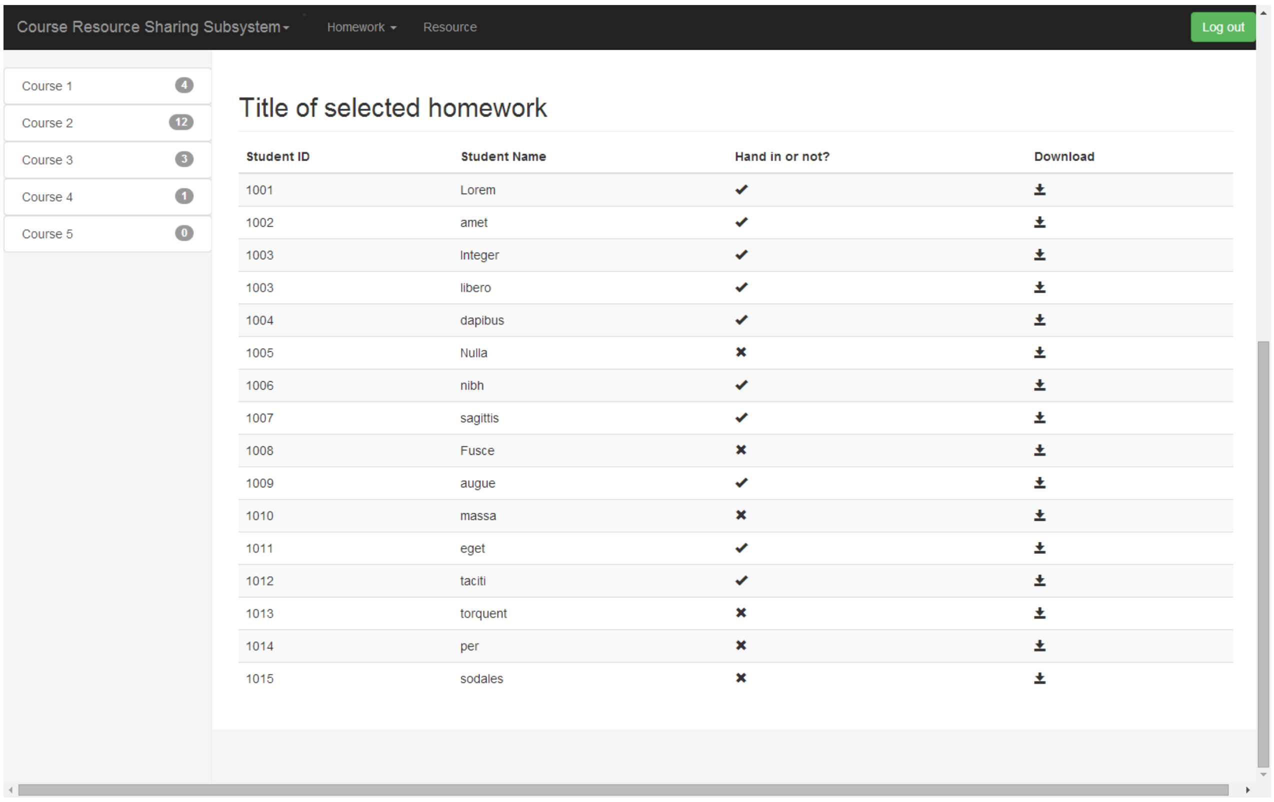


Figure 3.7.2 Flow chart

3.7.8 User Interface



Title of selected homework

Student ID	Student Name	Hand in or not?	Download
1001	Lorem	✓	⤓
1002	amet	✓	⤓
1003	Integer	✓	⤓
1003	libero	✓	⤓
1004	dapibus	✓	⤓
1005	Nulla	✗	⤓
1006	nibh	✓	⤓
1007	sagittis	✓	⤓
1008	Fusce	✗	⤓
1009	augue	✓	⤓
1010	massa	✗	⤓
1011	eget	✓	⤓
1012	taciti	✓	⤓
1013	torquent	✗	⤓
1014	per	✗	⤓
1015	sodales	✗	⤓

Figure 3.7.3 User Interface

3.7.9 Test Plan

Input	Expected result
Click on the download button	Get the file successfully
Click on the download button but login timeout	Return error message

Table 3.7.3 Test plan

3.8 ConnectDB design

3.8.1 Module description

This module is about the connection to the database. All operations to the database need this module, and other modules just invoking the functions in this module.

3.8.2 Profile

The name of profile is **ConnectConfig.php**, and these are what in the file:

```
<?
  $database=Course_Resource
  $acount=root
  $password=root
  $server=localhost
  $port=3306
?>
```

3.8.3 Performance

In each module, the connection to the database is need, and each operation related to the database is finished in this module, including connecting, disconnecting, querying, and even changing data.

3.8.4 The definition of ConnectDB

Function Name	Function
open()	Use the function mysql_connect() inside the php to create a connection to the database. If succeed, return true, else return false.
close()	Use the function mysql_close() inside the php to close the connection to the database. If succeed, return true, else return false.
query (sql)	Use the function mysql_query(sql) inside the php to make a SQL query, and the SQL statement is in the constant string 'sql'. If succeed, return the query result, else return an error.
fetch_assoc()	Use the function mysql_fetch_assoc() inside the php to get an associative array from the result. If succeed, return the associative array, else return an error.
LoginQuery (user_id, password)	Make a SQL query to find out if there is a user whose id is 'user_id' with a password 'password'. If there is such a user, store the login status in the session and return true, else return false.
SecurityCheck()	Verify the user's information. If the user passes the verification, return true, else return false.
getUsertype()	Get the type of the user, and return the result with a string.
AddAssignment (form)	All information of assignment is in the form. Add the assignment to the database, if succeed, return true, else return false.

Function Name	Function
generateTmpName (filename)	Get a new name for the file ‘filename’, and return the new name.
getFromDB (fileID)	Query the information of the file with an ID ‘fileID’, and return all the information.
AddFileInfoToDB (filename,filetype,filesize, tmpname,course_id)	Add the information of the file to the database. The original name of the file is ‘filename’, the type of the file is ‘filetype’, the size of the file is ‘filesize’, the name in the database of the file is ‘tmpname’, and the ID of the course the file related to is ‘course_id’.

3.9 Data Security Detection Design

3.9.1 Module Description

This module is used to detect the data via the client and make sure the data security. Some users may user SQL injection or some other methods to get some data or change the data in the database illegally. To avoid this condition, security detection is needed.

3.9.2 Function

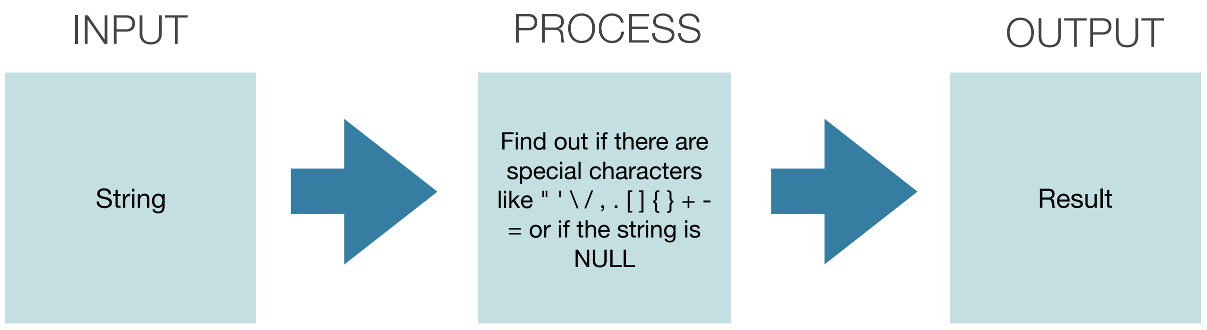

INPUT
PROCESS
OUTPUT
String
Find out if there are special characters like " ' \ / , . [ ] { } + - = or if the string is NULL
Result

Figure 3.9.1 IPO Graph

3.9.3 Property

After getting a string, to make sure that the string is legal, make a detection on the string and give out the result.

3.9.4 Inputs

Name	Signal	Type and Format	Input Format
String to be detected	StrCheck	string	Function parameter

Table 3.9.1 Input list

3.7.5 Outputs

Name	Signal	Type and Format	Output Format
Detection result	CheckResult	string	Function return value

Table 3.9.2 Output list

3.9.6 Design Approach

```
<?
function bool CheckStringSafety(string StrCheck)
{
   char example[]={“,’,/,\,,,.,[,],=,-,+};
   if (none of example in StrCheck)
             return true;
   else
             return false;
}
?>
```

3.9.7 Test Plan

Input data	Expected Result
NULL	Return warning
A string with illegal character	Return warning
A legal string	Operate succeed and return result

Table 3.9.3 Test plan

CHAPTER IV: INTERFACE DESIGN

4.1 User Interface

Because we expect the application to be developed as a WebApp in the browser, therefore users are expected to click the buttons and fill-in the forms on the web page provided by the application to access the features.
The detailed design of the User Interface can all be found in Chapter 3 of this document under each module description.

4.2 Outer Interface

The system use the php function **mysql_connect** to establish the connection with the MySQL database.

4.3 Internal Interface

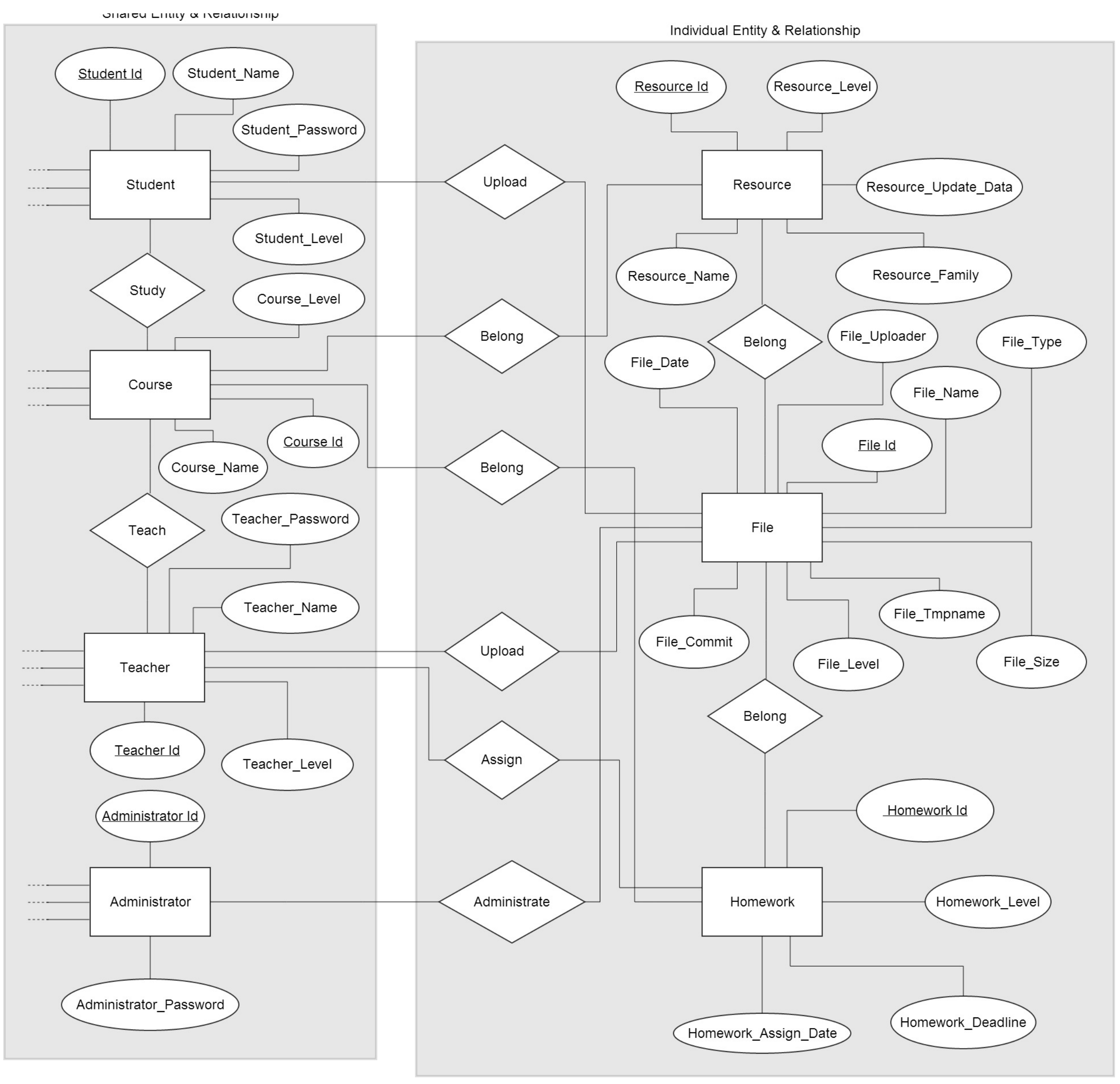
The Subsystem works pretty much as an stand along application in the system, sharing the main system's database with limited information.

	Resource Sharing Subsystem
Information Management System	User information, course information, user privileges, relations between users and courses.
Automatic Course Arrangement Subsystem	No Communication
Course Selection Subsystem	No Communication
Discussion Forum Subsystem	No Communication
Online Testing Subsystem	No Communication
Score Management Subsystem	No Communication

CHAPTER V: SYSTEM DATA STRUCTURE DESIGN

The Gate is a device to transmit the packages between the server and sensors, so there is no need for the Gate to unpack the package and store any information. Therefore, there is no database used in the development of the Gate.

5.1 Concept Structure Design



5.2 Logical Structure Design

Notes:“...”implies that this schema may have other attributes which won’t be used in the Resource Sharing Subsystem.

5.2.1 Shared Schemata:

- Student Account: Student (Student Id, Student_Name, Student_Password, ...)
- Teacher Account: Teacher (Teacher Id, Teacher_Name, Teacher_Password, ...)
- Course Information: Course (Course Id, Course_Name, Course_Level, ...)
- Administrator Account: Administrator (Administrator Id, Administrator_Password, ...)
- Studying Relationship: Study (foreign key Student_Id, foreign key Course_Id)
- Teaching Relationship: Teach (foreign key Teacher_Id, foreign key Course_Id)

5.2.2 Individual Schemata:

- File Information: File (File Id, File_Date, File_Name, File_Type, File_Size, File_Tmpname, File_Level, File_Commit, File_Uploader)
- Homework Information: Homework (Homework Id, Homework_Level, Homework_Deadline, Homework_Assign_Date, Homework_Requirement)
- Resource Information: Resource (Resource Id, Resource_Level, Resource_Update_Date, Resource_Family, Resource Name)
- Affiliation Relationship between File and Resource: File_Belong_Resource (foreign key File_Id, foreign key Resource_Id)
- Affiliation Relationship between File and Homework: File_Belong_Homework (foreign key File_Id, foreign key Homework_Id)
- Affiliation Relationship between Resource and Course: Resource_Belong_Course (foreign key Resource_Id, foreign key Course_Id)
- Affiliation Relationship between Homework and Course: Homework_Belong_Course (foreign key Homework_Id, foreign key Course_Id)

5.3 Physical Structure Design

Notes:“…”implies that this schema may have other attributes which won’t be used in the Resource Sharing Subsystem.

a, Student Account

Field	Data Type	Can’t be null	Primary key or not	Notes
Student_Id	int	Y	Y	Id of student
Student_Name	varchar(20)	Y	N	Name of student
Student_Password	varchar(20)	Y	N	Password of student
…	…	…	…	…

b, Teaching Staff Account

Field	Data Type	Can’t be null	Primary key or not	Notes
Teacher_Id	int	Y	Y	Id of teacher
Teacher_Name	varchar(20)	Y	N	Name of teacher
Teacher_Password	varchar(20)	Y	N	Password of teacher
…	…	…	…	…

c, Course Information

Field	Data Type	Can’t be null	Primary key or not	Notes
Course_Id	int	Y	Y	Id of course
Couser_Name	varchar(20)	Y	N	Name of course
Course_Level	int	Y	N	Level of course to control the visibility of files under it
…	…	…	…	…

d, Administrator Account

Field	Data Type	Can’t be null	Primary key or not	Notes
Administrator_Id	int	Y	Y	Id of administrator
Administrator_ Password	varchar(20)	Y	N	Password of administrator
...	...	...	...	...

e, Studying Relationship

Field	Data Type	Can’t be null	Primary key or not	Notes
Student_Id	int	Y	N	Id of student
Course_Id	int	Y	N	Id of course

f, Teaching Account

Field	Data Type	Can’t be null	Primary key or not	Notes
Teacher_Id	int	Y	N	Id of teacher
Course_Id	int	Y	N	Id of course

g, File Information

Field	Data Type	Can’t be null	Primary key or not	Notes
File_Id	int	Y	Y	Id of file
File_Date	date	Y	N	The submit time
File_Name	varchar(30)	Y	N	The name of name
File_Type	int	Y	N	1 as resource, 2 as homework
File_Size	int	Y	N	Size of file
File_Tmpname	varchar(30)	Y	N	Temp name of file in server
File_Level	int	Y	N	Level of file to control the visibility

Field	Data Type	Can't be null	Primary key or not	Notes
File_Commit	varchar(100)	N	N	Commit of file, added when it is submitted
File_Uploader	char+int	Y	N	Point out who uploaded this file

h, Homework Information

Field	Data Type	Can't be null	Primary key or not	Notes
Homework_Id	int	Y	Y	Id of homework
Homework_Level	int	Y	N	Level of homework to control the visibility of file under it
Homework_ Deadline	varchar(30)	Y	N	The deadline of homework
Homework_ Assign_Date	Date	Y	N	Date of when the homework is assigned
Homework_Requirement	Varchar(200)	N	N	Commit or requirements from the people who assign it

i, Resource Information

Field	Data Type	Can't be null	Primary key or not	Notes
Resource_Id	int	Y	Y	Id of resource
Resource_Family	varchar(30)	N	N	Symbolize the type of resource for search
Resource_Name	varchar(30)	Y	N	The name of resource
Resource_Update _Date	date	Y	N	The last update date

Field	Data Type	Can’t be null	Primary key or not	Notes
Resource_Level	int	Y	N	Level of resource to control the visibility of file under it

j, Affiliation Relationship between File and Resource

Field	Data Type	Can’t be null	Primary key or not	Notes
File_Id	int	Y	N	Id of file
Resource_Id	int	Y	N	Id of resource

k, Affiliation Relationship between File and Homework

Field	Data Type	Can’t be null	Primary key or not	Notes
File_Id	int	Y	N	Id of file
Homework_Id	int	Y	N	Id of homework

l, Affiliation Relationship between Resource and Course

Field	Data Type	Can’t be null	Primary key or not	Notes
Resource_Id	int	Y	N	Id of resource
Course_Id	int	Y	N	Id of course

m, Affiliation Relationship between Homework and Course

Field	Data Type	Can’t be null	Primary key or not	Notes
Homework_Id	int	Y	N	Id of homework
Course_Id	int	Y	N	Id of course

CHAPTER VI: RUNNING DESIGN

6.1 Module combination

In this subsystem, the modules are divided by their different functions. Each module contains client interface, client script and background program in the server. Different modules don't share an interface except for the main interface, and different client scripts just share a common class XMLHttpRequest. The background programs have connection by the database connection.

6.2 Running control

The user choose different function in the main interface:

- Login Module:
  The user inputs the User_ID and Password to log in, and then other functions are allowed.

- Personal Resource Management Module:
  After logging in, the user are granted to do some operations. In this module, the user can view, download and change the resources online.

- Resource Upload Module:
  After logging in, the user are granted to do some operations. In this module, the user can upload a file as a resource to the server.

- Search Module:
  After logging in, the user are granted to do some operations. In this module, the user can search and download a file from all the homework and resources.

- Homework Assign Module:
  After logging in, the user are granted to do some operations. In this module, a instructor can assign homework for the courses, while a student can do nothing.

- Homework Upload Module:
  After logging in, the user are granted to do some operations. In this module, students can view their courses and homework and select a homework to upload.

- Homework Check Module:
  After logging in, the user are granted to do some operations. In this module, a instructor can check their students' homework and download it.

6.3 Running time

With the using of AJAX, we can use static update technique in the client, and full page operations can also be reduced, so we can achieve a high server bandwidth utilisation. In this way, the running time is greatly reduced.
One thing influence the running time a lot is the connection and disconnection to the database. Each operation needs to link the database will touch off these two steps, and these operations can be found in every module. What’s more, when doing querying operations, for example, when looking up the user’s information in the database, if the table is very big or the input is wrong, the program has to traverse the whole table and cost a lot of time.

CHAPTER VII: SYSTEM ERROR HANDLING

7.1 Errors

Type of Error	The Reason	Handling
Unable to connect the database	The profile is not right and cannot connect the database, or too much users access to the database at the same time.	Correct the profile to make sure the connection. Make a limit on the interval between two requirements of the same user.
SQL error	Some users insert SQL statement in the form and try to destroy the database or change the data illegally, which may lead to error in the database.	Detect the data in the form in the server first, and drop the bad request.
Information missing on the website	Some users try to get access to the website even the background without logging in, and destroy the website or get some information illegally.	Some websites should be banned to access directly.
The server crashed	The server works for a long time with high work load, and meet a lot of bugs in the running, finally the server crashed.	Have regular server maintenance.
Account Information leaked	Computer viruses in the client or other reasons lead to account information leaked, and the password changed. Or hacker attack the server and get everything in the database.	Use antivirus software to protect the computer, and The users should never log in with other people's computer. And the passwords in the database should be encrypted by algorithms like MD5.
The hard disk crashed	Reading and writing too often and too many operations to the database lead to the damage of the hard disk and the lost of data.	Backup regularly.
Illegal instructions cannot be handled.	No resource exist or other bad requests.	Check and make sure the instruction is right before executing the statement.
Get random or mess code from the database	Having reading or writing errors when having database operation.	Use the same code or encode different kind of code.

7.2 Remedy

7.2.1 Backup system technology

- Backup the database regularly to avoid losing.
- Set different database in different computer, so that the data won’t lost at the same time.
- Save the same data at more than one computer, so that we can recover the data from other computers.

7.2.2 Use a less effective technology

- When the client does not work, we can make operations with manual work. For example, we can download the resource directly from the server, or change the resource information manually.

7.2.3 Recover and reboot technology

- If the server or the database crashed, we can rerun it from the breaking point or just start it at the very beginning.

7.3 System maintenance design

- In the database operation, when connecting or disconnecting the database or making an SQL query, catch every exception and get the warning.
- Check the IP accessed to the system, don’t let a IP visit the website too often to avoid attack like DDOS.
- Have a record of administers’ operation to the server and database, so that dangerous operations can be checked out very soon.