# Sports Center Management System
# High Level Design

## *Objective:*

This document is the Architecture Blue Print and High Level design document for the Sports Center Management System built on Flask framework and Python. It explains the high level design decisions taken for converting the user requirements into technical design specifications.

The objective of this document is to provide:
1.Application Architecture
2.Application Objects and their interaction
3.Interface architecture with external systems

## *Summary of proposed solution*

All users need to register an account with their email address. However, staff and customers will have different web interface to support for different function.

Our first target user is all citizens who live in Leeds. People will use this website to join in membership, view all the timetables and information of facilities and activities which supported by sports center, view the price list of each activity, book each facility online and they will receive a receipt.

Our second user is staff in the sports center, they will use this website to manage the sports center. They can update the latest information such as timetables and availability of facilities and they are expected to book regular sessions for the customers in the sports center. They are also expected to handle card and cash payment and print receipt and tickets.

## *Data Modelling*

| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | **Role Model** | | | | |
| 2 | Column Name | Description | Column Data Type | Column Null Option | |
| 3 | id | role id | integer | Not null | |
| 4 | name | role name | string | Not null | |
| 5 | default | | boolean | Not null | |
| 6 | permissions | | integer | Not null | |
| 7 | users | | | | |
| 8 | | | | | |
| 9 | **User Model** | | | | |
| 10 | Column Name | Description | Column Data Type | Column Null Option | |
| 11 | id | user id | integer | Not null | |
| 12 | email | user register email | string | Not null | |
| 13 | username | | string | Not null | |
| 14 | role_id | user's role | integer | Not null | |
| 15 | password_hash | user's set password | string | Not null | |
| 16 | confirmed | status of user 's account | boolean | Not null | |
| 17 | | | | | |
| 18 | **Facility Model** | | | | |
| 19 | Column Name | Description | Column Data Type | Column Null Option | |
| 20 | id | facility id | integer | Not null | |
| 21 | name | facility's name | string | Not null | |
| 22 | capacity | facility's capacity | integer | null | |
| 23 | description | description of a facility | string | Not null | |
| 24 | courts | | | | |
| 25 | activities | | | | |
| 26 | | | | | |
| 27 | **Court Model** | | | | |
| 28 | Column Name | Description | Column Data Type | Column Null Option | |
| 29 | id | court id | integer | Not null | |
| 30 | number | court No. | integer | null | |
| 31 | availability | Is court free for use | string | Not null | |
| 32 | facility_id | belongs to which facility | integer | Not null | |
| 33 | times | | | | |

| | | | | | |
|---|---|---|---|---|---|
| 35 | **Activity Model** | | | | |
| 36 | Column Name | Description | Column Data Type | Column Null Option | |
| 37 | id | activity id | integer | Not null | |
| 38 | activity_name | activity name | string | Not null | |
| 39 | repeat_every | | integer | null | |
| 40 | facility_id | belongs to which facility | integer | Not null | |
| 41 | | | | | |
| 42 | **PriceList Model** | | | | |
| 43 | Column Name | Description | Column Data Type | Column Null Option | |
| 44 | id | price id | integer | Not null | |
| 45 | price | | integer | Not null | |
| 46 | type | child, adult,elderly | string | null | |
| 47 | activity_id | belongs to which activity | integer | Not null | |
| 48 | | | | | |
| 49 | **Booking Model** | | | | |
| 50 | Column Name | Description | Column Data Type | Column Null Option | |
| 51 | id | booking id | integer | Not null | |
| 52 | status | unpaid,process,complete | string | Not null | |
| 53 | payment | payment method:cash or card | string | Not null | |
| 54 | fees | booking fees | integer | Not null | |
| 55 | account_id | belongs to which account | integer | Not null | |
| 56 | | | | | |
| 57 | **Time_management Model** | | | | |
| 58 | Column Name | Description | Column Data Type | Column Null Option | |
| 59 | id | booking id | integer | Not null | |
| 60 | start_time | session for booking start time | datetime | null | |
| 61 | end_time | session for booking end time | datetime | null | |
| 62 | booking_id | belongs to which booking | integer | Not null | |
| 63 | court_id | belongs to which court | integer | null | |
| 64 | | | | | |

| | | | | |
|---|---|---|---|---|
| 65 | **Credit_card_info Model** | | | |
| 66 | Column Name | Description | Column Data Type | Column Null Option |
| 67 | id | credit card id | integer | Not null |
| 68 | card_number | credit card number | integer | Not null |
| 69 | expire_month | | integer | Not null |
| 70 | expire_year | | integer | Not null |
| 71 | security_code | credit card security code | integer | Not null |
| 72 | holder_name | credit card holder's name | integer | Not null |
| 73 | account_id | belongs to which account | integer | Not null |
| 74 | | | | |
| 75 | **Staff Model** | | | |
| 76 | Column Name | Description | Column Data Type | Column Null Option |
| 77 | id | staff id | integer | Not null |
| 78 | name | staff name | string | Not null |
| 79 | role | staff's job | string | Not null |
| 80 | facilities | | | |
| 81 | | | | |
| 82 | **Account Model** | | | |
| 83 | Column Name | Description | Column Data Type | Column Null Option |
| 84 | id | account id | integer | Not null |
| 85 | user_id | belongs to which user | | |
| 86 | bookings | | | |

## *Solution architecture*

**Key Decisions and Recommendations**

**KEY DECISION #1:**

What will be the platform of Sports Center Management System?

CHOSEN OPTION – Develop the application in Python using the Flask framework.

.

**KEY DECISION #2:**

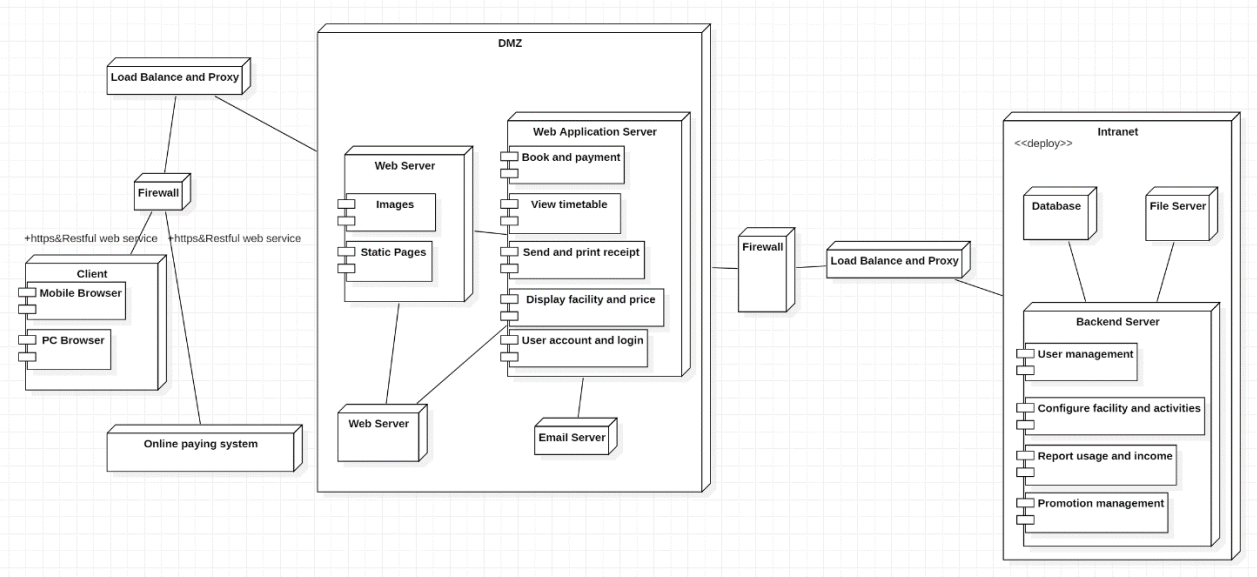
What will be the Database for Sports Center Management System?

The following are the drivers for the decision.

CHOSEN OPTION – SQLIte will be used as the database for the Sports Center Management System. There will be tight controls and separate Schema for the

Sports Center Management System database to ensure that no one except the Centralized Membership IT team has permissions to access the database.

## *Implementing the proposed solution*




The hardware contains smart phones, various server machines, firewall and workload balance. The software contains client application/browser, web servers in DMZ, web application server, a database, application management backend server and email server.

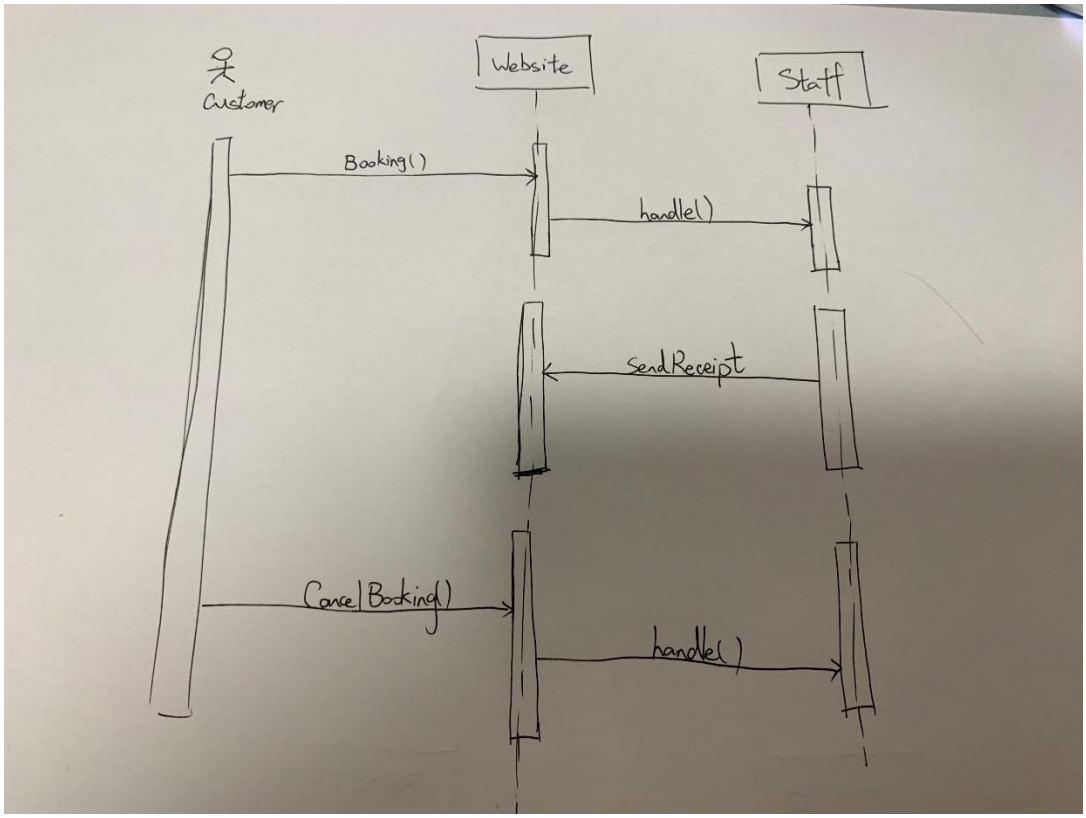
Our implementation is based on Python Flask framwork. Restful Webservice which is adopt to implement server-side functionality. Plus, in this project, both mobile and website interface should be implemented so that Restful webservice will be a good choice. VUE.js will be applied to coding in the front end so that the web pages will be more attractive. Load balancing and server clustering techniques are unavoidable to ensure reliability and prevent from failing over. Two firewalls are implemented to help enhance the security and privacy. Plus, responsive design will be applied to allow users can use the application from different devices including smart phones and tablets, which will improve usability.

## *Software*

For the development and deployment of the Web application following software components will be used.

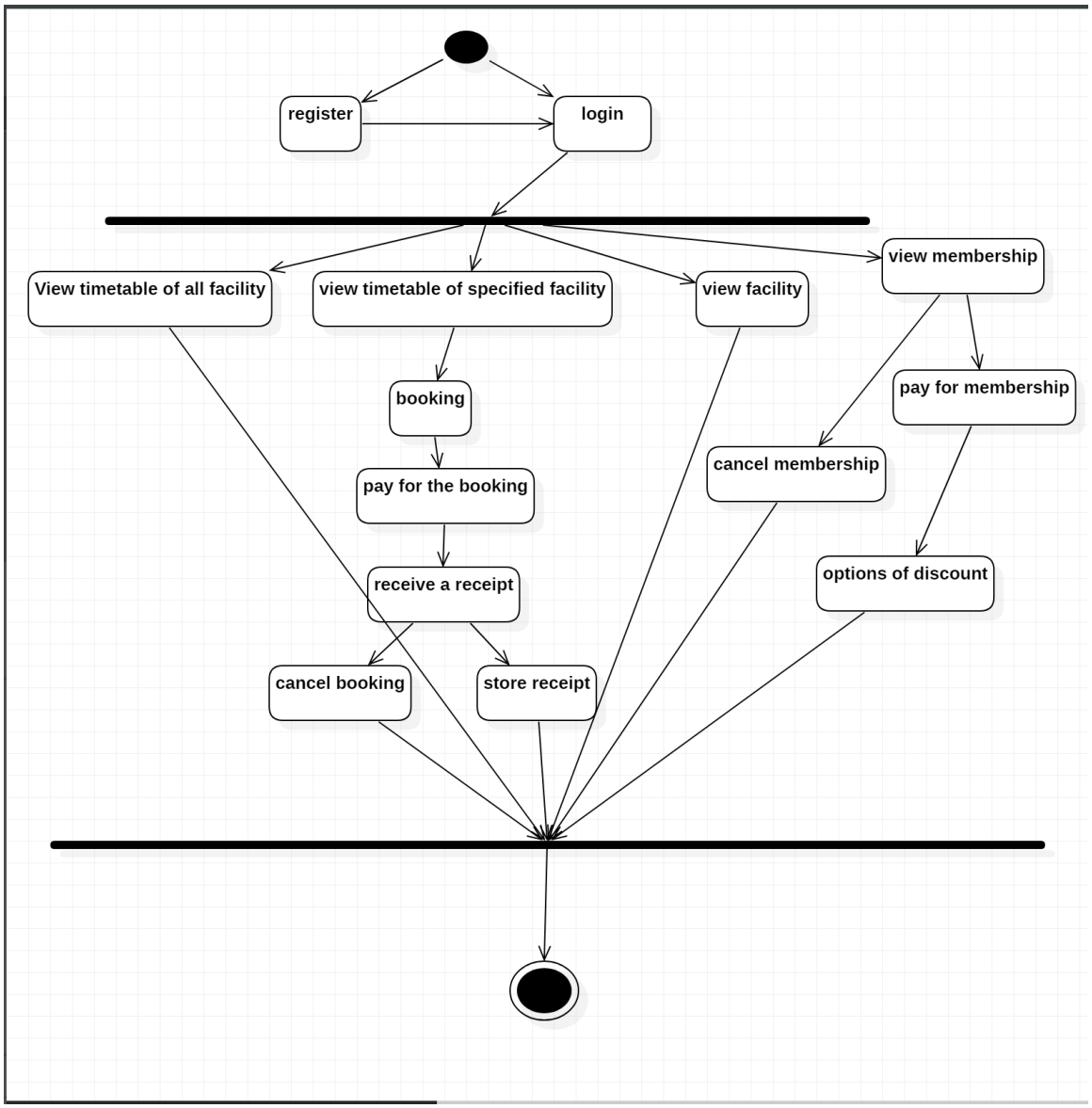
| Type | Software Component |
|---|---|
| Web Browser | Microsoft edge, Google chrome<br>Resolution Supported: 800 X 600 and 1024 X 768 |
| Application Server | The built-in servers offered by Flask and Django<br>A RESTful API |
| Python Development Environment | JetBrains PyCharm 2.3 |
| Python Version | Python3 |
| Database | SQLite |
| Configuration Management Tool | VSS (Offshore) |
| Defect Logging Tool | IPM+, Excel sheets |
| Version control | Gitlab |
| Components | Struts 2.0 (including Ajax )<br>EJB 3.0<br>JPA<br>Log4j |

## *Using the proposed solution*



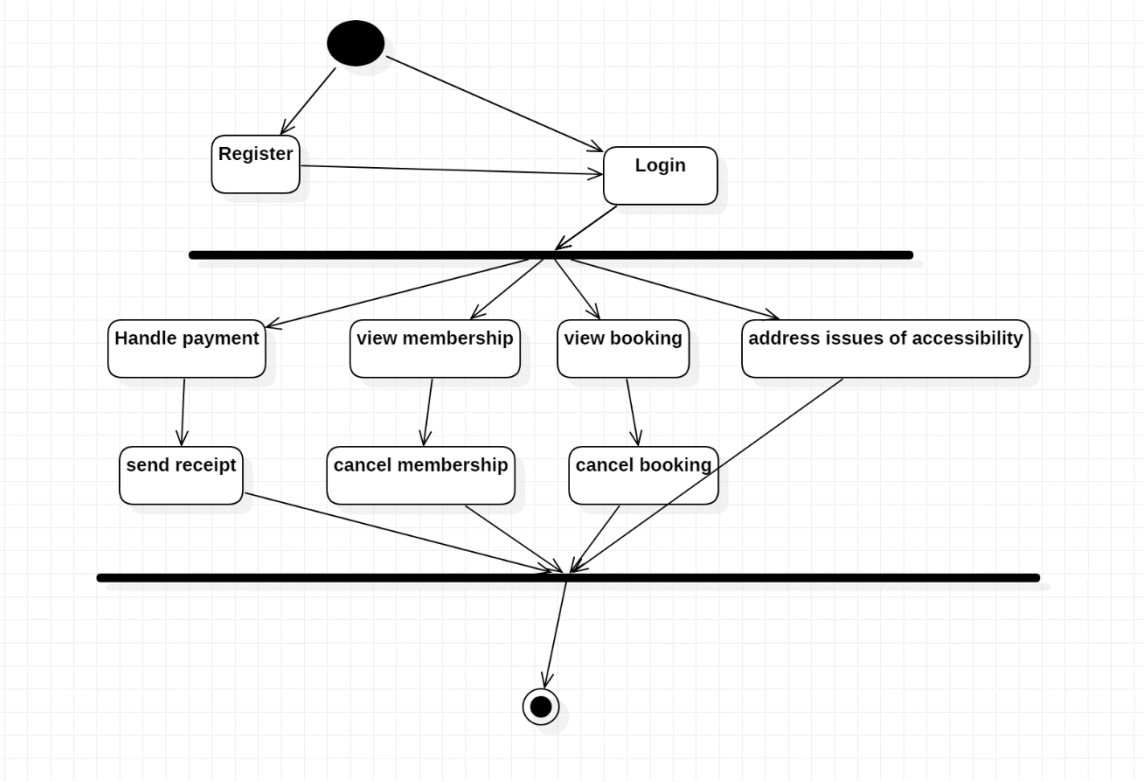

I am Peter and I am a citizen in Leeds and I have the membership of the sports center. I want to play basketball so that I visit the website of the sports center. I want to view the timetables of basketball court so that I can know what time can I go to play basketball. I want to book a basketball court so that I can play basketball on one day.

**Activity diagram: customer view**



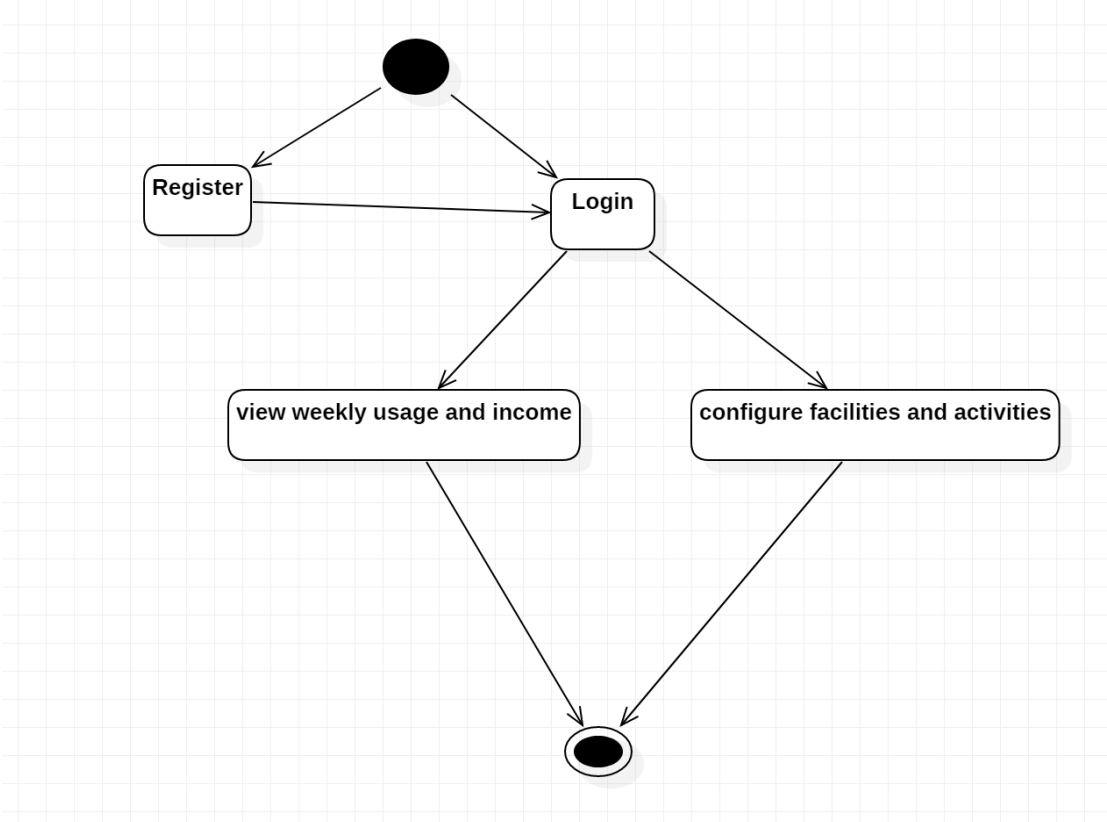

I am peter and a customer of sports center. I want to view the timetable of all facilities so that I can choose what time and what kind of sports I can play. I want to view timetable of specific facility so that I can choose the time to book a court. I want to cancel booking because I have some stuffs so that money will not be wasted. I want view facility so that I can make up my mind to join in the membership. I want to be the membership so that I can have some discount rights.

**Activity diagram: Staff view**




I am Alice and also the staff in sports center. I want to handle payment so that the customers' booking will be processed. I want to send receipt so that customers can use their bookings. I want to view membership of a customer so that I can tell customers' its membership information. I want view booking so that I can cancel booking for a customer. I want to address issues of accessibility so that customer will not have problem with our website.

**Activity diagram: manager view**




I am john and also the manager of sports center. I want to view weekly usage and income so that I can manage the state of business of sports center. I want to configure facilities and activities so that customer can get to know our latest facility and activity.

## *Approaches/Mechanisms*

**Data Management**

The database will be accessed through Flask web application. SQAlchemy will be used to manage the database connection and for database operations such as select, insert, delete and update.

**Transaction Management**

The database transaction manager is used to manage local database transactions. The transaction is controlled by the business layer, while database operation is executed on persistence layer.
The transactions within the Sports center management system will be handled using the database transaction Manager.

**Session Management**

HTTP session will be maintained for each logged in user. All Actions in the application will verify the existence of a valid session via a common method.

For scalability, it will be ensured that the session data is minimal.

Session timeout will be configured on the application server. User will be forwarded to the Login Page after a session timeout.

**Pagination**

Pagination will be used in screens where a list of items are displayed or maintained. This will improve user interface and system performance by retrieving only data required for display. Records displayed per page will be a configurable item.

**Sorting**

The default sorting function uses the column id to be sorted and current sorted column id to display sort indicators and also attaches required JavaScript.

**Error & Exception Handling**

Application can encounter Logical or System errors. Exceptions such as validation errors are classified as logical errors and will be displayed in the same page. System errors such as the one caused by non availability of database will be displayed in a different error page.

Errors will be logged to the error log file using Log4j. Error log file location and name will be configurable in the application properties file.

Business Validations and Error Reporting

**Client side validations**
All data type validations will be done using JavaScript and will be reported back using standards.

**Server side validations**
Business validations need to be done when users do save and submit operation.
Validation/error reporting will be handled in a consistent manner across the application.

**Logging & Tracing**

All application errors and tracing statements will be logged to a file. Log4j component will be used for this purpose. For all exceptions and traces, complete exception and trace will be written to the file. Maximum size per log file and maximum number of log files can be specified in the property file.

Following are the sample configuration entries in application property file for logging.

**Maintainability**

In order to ensure that the application is maintainable, application design/development needs to follow Java Guidelines, Sun Coding Guidelines for Java.

A layered approach (presentation, business and data) will be used for application construction to improve maintainability.

Application will also provide different levels of logging and tracing which can be configured.

**Security**

The System will use the AES encryption method to encrypt the password of every user.

When the user first login, system will encrypt the user's password and compare with the stored encrypted password of this user, if they are the same, the user can login and store the user object in the session, otherwise the user cannot login and log the information, In each JSP page, it will first get the User id from the session, If the user id cannot be found in the session or the user has no authority to view the page, the system will redirect to Login page.

**Internationalize**

In the first login page, we can add a dropdown list where the user can select the language between English and Chinese, and the application will store the language code in the session. So in every page, the application will display the text message from the property file according to the language code stored in the session.

**Webservice**

Because the end user may need to query his points and change his password etc in the sports center website from other website, so the Centralized Membership System provides the webservice and the interface to let other application call.

The Centralized Membership System will use the Metro 1.2 (a Web Services framework that provides tools and infrastructure to develop Web Services solutions for the end users and middleware developers) requiring JDK 1.5 Update 2 or later and is developed by SUN Microsystem.

Design Patterns

MVC - The application will use the Struts2 which is based on the MVC design pattern.
Decorate – The application will use the decorate pattern to organize the classes which have many same methods.
Facade – All the business logic will be encapsulated in EJB3. So the system will provide classes as the façade to many EJB methods.

Factory Method – the system can use the factory methods to create many objects instead of using the constructor methods directly.