# Objectives

You are required to assimilate what you have learned in lectures and tutorials, and apply that knowledge to the design and implementation of an *n*-tier commercial application prototype.

Upon successful completion of this assignment, you should be able to:

* (***CILO 1***) Understand the fundamental concepts and procedures of the major server-side Internet application architectures and services;
* (***CILO 2***) Setup web sites that take advantages of server-side processing;
* (***CILO 3***) Write server-side processing scripts;
* (***CILO 4***) Design advanced web-based applications using Java models and frameworks;
* (***CILO 5***) Understand other advanced techniques of web servers, including security and service-oriented architecture.

# Scope

In this assignment, you are required to choose **one** topic out of the following three topics and develop the corresponding system. The system you developed needs to be *highly flexible*, *robust*, and *user-friendly*, yet *unique* from other existing Internet-based systems with similar functions.

1. Internet-based ticketing system for a cinema. This ticketing system, in certain aspect, will look like some existing Internet-based ticketing systems, such as [Cityline](http://www.cityline.com.hk).
2. Internet-based bookstore. This bookstore, in certain aspect, will look like some existing Internet-based bookstores, such as [Amazon](http://www.amazon.com/books-used-books-textbooks/b?node=283155).
3. Internet-based airplane ticket booking. This ticket-booking website, in certain aspect, will look like some existing online ticket booking websites, such as [TravelChinaGuide](http://www.travelchinaguide.com/china-trains/ticket-service.htm).

You are required to deliver a working prototype using appropriate Java technologies.

# Internet-based cinema ticketing system--Use Cases

The major functional requirements of the Internet-based ticketing system are shown in the use case diagram below.



***Three actors: customer, ticketing officer and ticketing manager.***

* ***Use case: browse movie show time.*** A cinema patron may browse the show time of movies at any time. The show times may be arranged in a reasonable logical order.
* ***Use case: purchase movie tickets.*** A customer may, at one time, purchase multiple tickets for one movie’s show time only. A customer who is a registered member may use *some* or *all* accumulated loyalty points to pay for movie tickets purchased. Movie tickets purchased using loyalty points are not refundable. A ticketing officer may, at one time, sells movie tickets for more than one movies showing at different times on behalf of walk-in customers.
* ***Use case: request for refund of purchased tickets.*** A customer who is a registered member may submit request for refund of purchased movie tickets for movies *before 3 hours* of the marked showing time. Upon receive the request, the system will release the tickets held by the customer for public purchase even the request is not authorized by a ticketing officer yet. The request for refund needs to be authorized by a ticketing officer.
* ***Use case: authorize refund request.*** A ticketing officer needs to authorize each request for refund of purchased movie tickets before the ticket price (sans any associated administrative fees) to be paid back to cinema patron.
* ***Use case: manage movie show time and pricing.*** A ticketing manager may add, remove, and change movie show time to the ticketing system. A ticketing manager may change the ticket price for particular movie show time.
* ***Use case: manage seating.*** A ticketing manager may mark seats as available/unavailable for purchase.

# Internet-based bookstore--Use Cases

***Two actors: customer and book manager.***

* ***Use case: browse books.*** A customer can browse books at any time.
* ***Use case: purchase books.*** A customer may, at one time, purchase any number of books. A customer who is a registered member may use *some* or all accumulated loyalty points to pay for books purchased. Books purchased using loyalty points are not refundable.
* ***Use case: request for refund of purchased books.*** A customer who is a registered member may submit request for refund of purchased. The request for refund needs to be authorized by a book manager.
* ***Use case: enquiry loyalty points.*** A customer who is a registered member may enquire the loyalty points accumulated.
* ***Use case: authorize refund request.*** A book manager needs to authorize each request for refund of purchased books.

***Use case: manage books quantity and price.*** A book manager may add, remove, and change book details and available quantity and price.

# Internet-based airplane ticket booking--Use Cases

***Two actors: customer and airplane*** ***company.***

* ***Use case: enquire airplane*** ***ticket information.*** A customer can browse available tickets based on the search requirements such as date and departure/destination places.
* ***Use case: book tickets.*** A customer can book tickets only after he registered in the system. Necessary passenger information is needed for registration, e.g. name, ID, credit card #.
* ***Use case: modify booking.*** A customer can modify his bookings, i.e., change the date, departure/destination places, or cancel the booking. The success of booking modification depends on the availability of tickets.
* ***Use case: manage tickets.*** The ticket company can update the ticket information, including the original price and the total number of issued tickets.
* ***Use case: manage booking.*** The ticket company can approve or reject the refund when customer asking for booking cancellation.
* ***Use case: view statistics.*** The ticket company may view the sales statistics of tickets.
* ***Use case: automatically discount***. The ticket company may set a function for sold tickets, that the price of a ticket for a particular airplane will automatically discounted with the number of remaining tickets. E.g. the discount is 80% off, if the remaining tickets is 100%; the discount is 10% off, if the remaining tickets is 10%.

# Bonus

Extra functions are encouraged and welcome in the developed system to ensure a smooth online experience. You can get bonus if some extra functions are implemented, such as:

1. You can determine a reasonable set of business rules and the appropriate application logic that maintain customer’s online experience as well as maintaining the safety of online transactions.
2. The system can support the multi-lingual service.
3. You implement creative/innovative way of applying java technology.

# Design Considerations

You are advised to segregate[[1]](#footnote-0) the presentation and application logic in all dynamic generated pages so that the user interface can be modified without affecting the application logic. The objective is to facilitate the division of work between user interface designers and application programmers.

You may consider incorporating an appropriate Model-View-Controller (MVC) framework in your overall design.

# Technical Expectations

Your prototype is expected to be deployed in a physical environment in CSLab as shown in the diagram below.

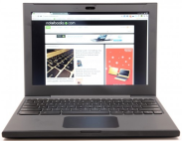












* ***Client Tier (Presentation)***
* Support at least *Microsoft Internet Explorer*, but may also support small screen device, device with limited bandwidth and processing power, etc.
* Main interface a person interacts with the system.
* ***Processing Tier***
* Java application server, e.g., *Apache Tomcat 6.x*.
* Accept user request.
* Implement application logic and enforce business rules.
* ***Database Tier***
* Database server: *Microsoft SQL Server*.
* Allows persistence of business objects.
* Enforces integrity of business data.

You are required to implement your prototype using relevant technologies which you may already know (or may not know) as well as those introduced in lectures and tutorial sessions. The most fundamental technologies, to name a few, include, but not limit to, XHTML (HTML 5), cascade style sheet, JavaScript, Java applet, Java Servlet, JavaServer Pages, SQL query language, etc.

# Implementation Considerations

You need to implement appropriate authorization control to restrict access to functions only applicable to certain users.

You may integrate some social networking features in your prototype, but you should not use your prototype to collect real personal data.

You may implement a simple payment processing procedure using your own code, but you should not collect real credit card information. Do not connect to any real system for payment.

The following note **must** appear on the header and footer of *all* pages:

**This web site exists to fulfill the coursework requirement of** [**CS4280**](http://eportal.cityu.edu.hk/bbcswebdav/institution/APPL/Course/Current/CS4280.htm)**.**

**Do not use your real personal data as input.**

# Deliverables

Each group is required to prepare the followings:

* A 10 – 15 minutes presentation (in designated time slot out of tutorial sessions) that focuses on:
* The key features of your prototype (***WHAT*** have you implemented and ***WHO*** did it?);
* Your design choices/techniques (***WHY***?) and development process (***HOW***?).
* The additional creative features / innovative solutions.
* A written report[[2]](#footnote-1) (in no more than 10 pages) that include details about:
* Detailed design of your prototype;
* How you have addressed the assessment guidelines outlined in the overview document?
* Strengths and weakness of your design;
* Contribution of each group member;
* URL of your prototype (hosted in CSLab environment);
* Reflection, i.e., what have you learned in this assignment? what is the most difficult part of this assignment?
* Source code of your prototype.

Each group needs to upload the entire project (in form of a zip[[3]](#footnote-2) archive) and report at the CS4280 [Canvas](https://canvas.cityu.edu.hk) course web site.

Go to the “Assignments” => “Assignment 2” => “Submit Assignment” and upload your file

* URL of your prototype.

Put the URL of your developed system prototype in https://personal.cs.cityu.edu.hk/~<CityU EID>/cs4280/asgp2/index.html.

* Due date

The expected delivery date of your prototype is 19th-April 2017. To ensure a high quality system can be delivered on time, there will be one intermediate check point in week 11. You will be asked to demonstrate your work-in-progress system.

***Note****:  
Demonstration must be done inside CSLab premises.*

# Assessment Criteria

CILOs 1 – 5 will be assessed in this assignment.

|  |  |  |  |
| --- | --- | --- | --- |
| CILO | Criteria | Major Assessments | Weight |
| 1 | * Well organize and present the key features of your prototype and justify your design choices. * Flexible design – future changes and new requirements requires minimal effort to implement. | * Report | 10% |
| 2 | * Robustness – tolerance of errors. | * Program codes; * Presentation; and * Demonstration | 60% |
| 3 | * Comprehensiveness (in terms of the specified functional requirements). * Uniqueness – features which distinguish your implementation from others. |
| 4 | * Informative, well organized, and presentable. * Able to highlight the key features of your prototype and justify your design choices. | * Report | 15% |
| 5 | * Innovation – creative use of server-side technologies; implementation of extra functions as mentioned in the bonus part. | * Program codes; and * Demonstration | 15% |

# Submission Deadline and Late Penalties

|  |  |
| --- | --- |
| Key Dates | What is Expected? |
| Week 11, Tutorial session | Intermediate check points  (Fill-in the intermediate check list form before attending tutorial) |
| April 19, 2017, 2:00 PM | Final submission (including written report, system source code, and system URL) |
|  | Live system demonstration  (Fill-in the final check list form before attend the tutorial) |

* Late penalties will be applied at the rate of *10%* per day (less than 24 hours also count as one day).
* Please make your system ready within the first 15 minutes of your tutorial session. Mark may be deduced for the group failed to setup their system on-time.
* Your system must run on the Linux server cs4280-01.cs.cityu.edu.hk. Demo on NetBean (or other Java IDE’s) is *not* allowed.
* **Penalties** will be applied to the group who cheats on the checklist.

# Collaboration and Plagiarism

This is a group assignment. Each group consists of ***two-three (2-3)*** students. Each group member should make an even contribution to each submission.

Individual performance will be assessed. Therefore, you need to inform your instructor about which part(s) of the assignment that you have made most contribution.

You are encouraged to discuss your design with other groups. However, you should not:

* Share your work / source code with other groups.
* Use others’ work[[4]](#footnote-3) (including any on-line resources) and pretend to be your own work without proper citation.

Penalties for inappropriate collaboration and violation of plagiarism can range from:

* ***Zero (0)*** mark on submission;
* A ***reduction*** in the overall mark; to
* Grade [***F***](http://www6.cityu.edu.hk/arro/content.asp?cid=70#gradetable) for the course.

Discuss with your instructor if you have any question about collaboration and plagiarism. You can also consult the University’s guidelines on academic honesty, which is available at <http://www6.cityu.edu.hk/arro/content.asp?cid=73>.

1. Tips: user interface in JSP’s, logic in Java Servlets, data representation in Java Beans. [↑](#footnote-ref-0)
2. Your report will be compared through [Turnitin](http://www.turnitin.com) database to check for improper citation or potential plagiarism. [↑](#footnote-ref-1)
3. Other formats (such as .dmg, .rar, .tgz, .7z, etc.) will NOT be accepted. [↑](#footnote-ref-2)
4. Use of third-party library/API is allowed and encouraged. [↑](#footnote-ref-3)