

CINEMINE

Mobile Cinema Reservation System

Design Report

Eren Yılmaz & Emin Özcan

Contents

Contents	1
1 Purpose of the Document	2
2 Project Information	2
2.1 Problem Definition	2
2.2 Project Scope & Aim	2
3 Architecture.....	3
3.1 Overall Architecture	3
3.2 Deployment	4
4 Data Design.....	5
4.1 Entity-Relationship Diagram.....	5
4.2 Data Dictionary.....	6
4.2.1 User	6
4.2.2 Cinema.....	6
4.2.3 Theatre	6
4.2.4 Seating Plan	6
4.2.5 Films	6
4.2.6 Artists	6
4.2.7 Ticket	6
5 Software Design.....	7
5.1 Class Diagrams.....	7
5.1.1 Controller.....	7
5.1.2 DAO Class Diagram	8
5.1.3 DOM Class Diagram	9
5.1.4 Servlet Class Diagram	10
5.1.5 Utility Class Diagram.....	11
5.2 Sequence Diagrams	11
5.2.1 Reservation Sequence Diagram	11
5.2.2 Cinema Save Operation Sequence Diagram.....	12
5.2.3 Theater Save Operation Sequence Diagram	12
6 User Interface Design	13
6.1 Main Page Design	13
6.2 Reservation Page Design	14
7 Schedule	15

1 Purpose of the Document

This document is the final design report of the project “Cinemine – Mobile Cinema Reservation System”. In this document, the reader will find the problem definition, project scope, constraints and detailed UML diagrams of the designed application.

2 Project Information

2.1 Problem Definition

The project is named as “Mobile Cinema Reservation System”, and we named our solution to the project as “Cinemine”. The aim of the project is to create a digital platform to facilitate a fast, user-friendly and robust reservation interface for customers, while giving cinema stakeholders a great opportunity for advertising their offerings.

The system will be an interactive space for visitors, where they will have the opportunity to write down their opinions about the movies, give rating points as they wish, as well as reserving and purchasing tickets for movies. The main difference of the system from the existing solutions is the visual location information of both cinemas and users. The Cinemine system will offer the users to browse and search cinemas according to this location information, in addition to classical search and browse mechanisms.

2.2 Project Scope & Aim

Our project, Cinemine, has a main intention for providing an useful interface about cinemas to people in Ankara. We accept cinema stakeholders as our main customers, who has directly affected by our registered users. Therefore, in order to extend our scope, we aim to enroll all cinema audiences firstly in Ankara, then to people at other big cities in Turkey by reaching more stakeholders in Turkey. All registration and shopping limitations are bound to stakeholders, because we purpose to give more freedom to stakeholders when offer their goods. Since cinema audiences want to be sure about their security when using their credit card at any system, we will guarantee their security by using 128bit SSL certificate. We will also facilitate searching cinema by integrating Google Earth into our system. In addition, we will use Yahoo User Interface to make our system more friendly and useful. If any integrated product is updated by its community, we will automatically novate our system by new interfaces, so our customers will have an oppurtunity to use the latest improvements. As a result, we are open to technological changes in the world to renew our system functionalities and interfaces in order to make it more easy to our users.

3 Architecture

3.1 Overall Architecture

The system is designed as a Java Web Application. We used the famous Model-View-Controller (MVC) pattern in the design process, with our implementation style. The user interaction is handled by JSP pages, which will be enriched by necessary media and JavaScript. In most pages, the page data is gathered from page controllers as Plain Old Java Objects (POJO). For some dynamic content, we designed the system to use AJAX calls within the system. In these cases, the main data transfer media will be XML.

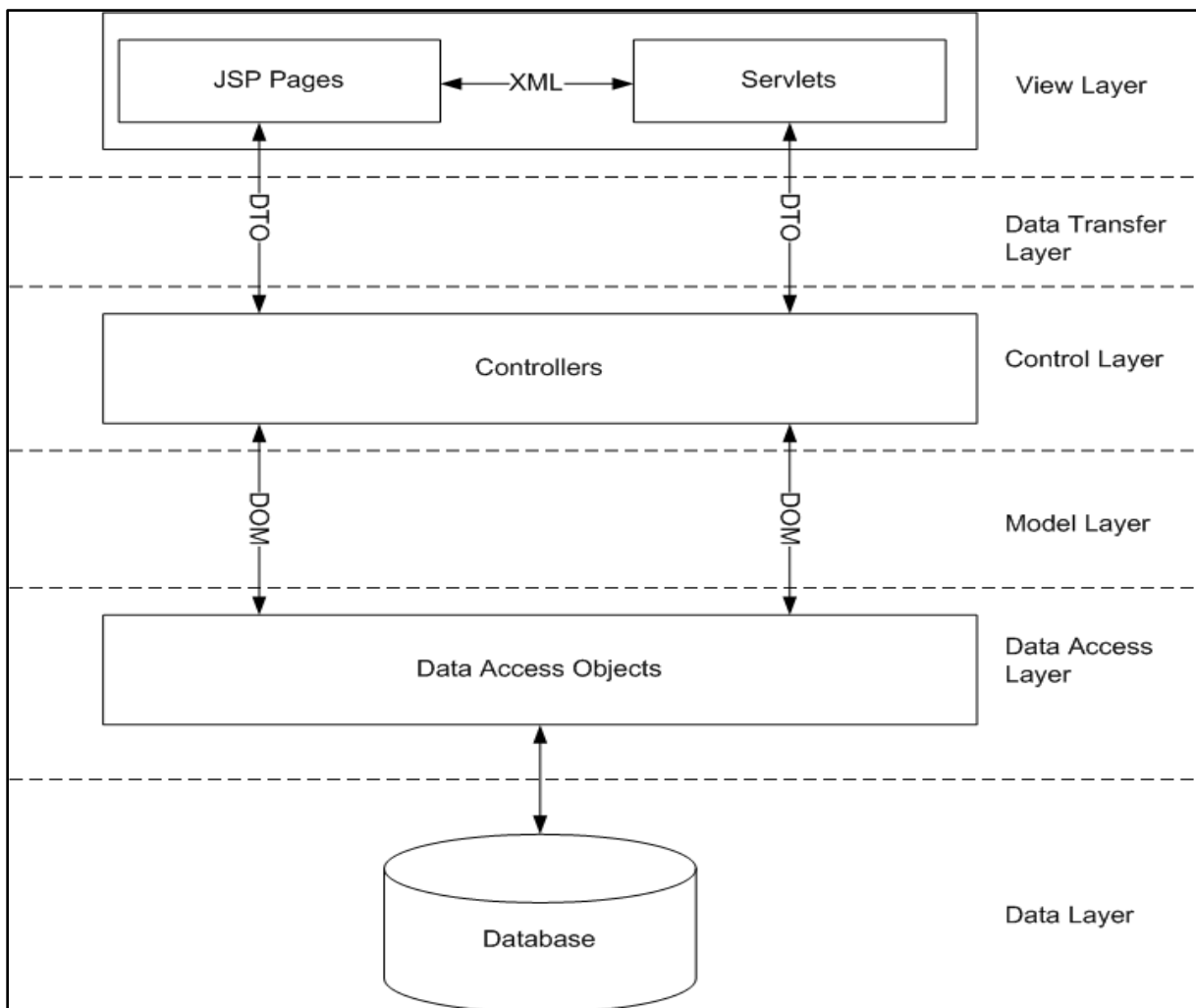


Image 1 - General Architecture

Each interface unit (JSP page or Servlet) has a related controller class. The interface unit will pass the data to the controller class by the Data Transfer Objects (DTO). The controller class will read the “operation” field in the incoming DTO and process according to the operation. If the process involves any operation with the database, it is done through instances of Data Access Objects (DAO). The data transfer between controller and DAO is done by the main Model objects (DOM).

3.2 Deployment

The designed application has all properties of a simple Java Web Application. The interface contents (JSP pages, HTML pages, images, JavaScript, etc.) will reside in the WebContent folder. The Java classes will reside in the src folder and the package structure will be as described in Image 2 - Package Structure.

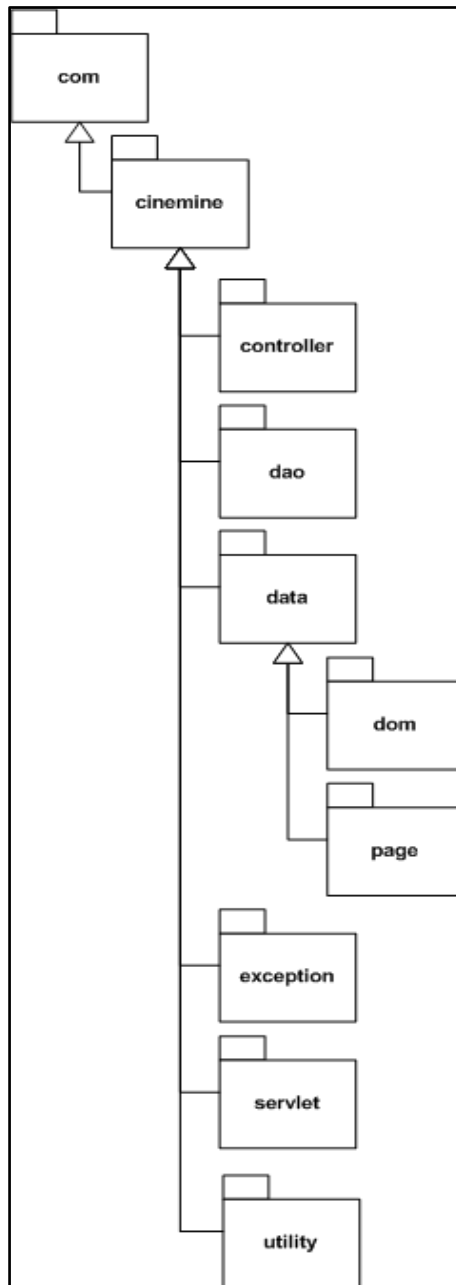


Image 2 - Package Structure

Referenced libraries and other tools will be linked as standard Web Application Libraries. The application will be deployed as a Java Web Archive (war) file to the application server. Referenced libraries will be included in the deployment unit.

4 Data Design

4.1 Entity-Relationship Diagram

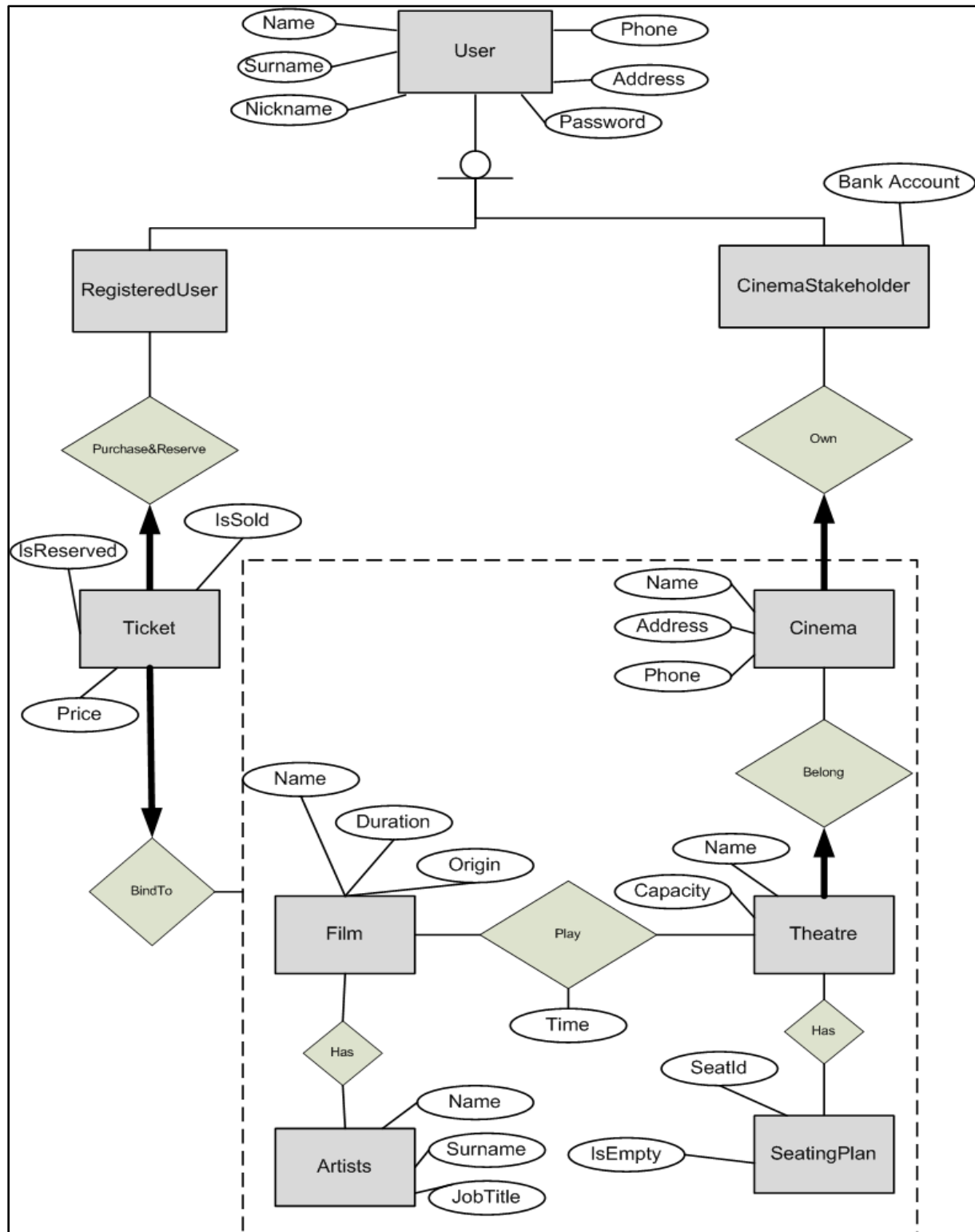


Image 3 - Entity-Relationship Diagram

4.2 Data Dictionary

4.2.1 User

Cinema stakeholders and registered users are stored in “user” entity. It includes name, surname, nickname, password, address and phone attributes. All these attributes are hold as varying character type and all of them are common for both registered users and cinema stakeholders. There is a usertype attribute in order to seperate cinema stakeholders and registered users. Cinema stakeholders have also bank account information.

4.2.2 Cinema

This entity is used for storing cinemas. All cinemas in the system must have an owner. Each cinema must have only one owner. Each cinema entity has name,address and phone attributes. All of these attributes are stored as varying character type.

4.2.3 Theatre

This entity is used for storing theatres of cinemas. All theatres in the system must belong to a cinema. Each theatre in the system must belong to only one cinema. This entity consists of theatre name, capacity attributes and a relationship with the entity “seating plan”.

4.2.4 Seating Plan

This entity is used for stroring seating plans of theatres. It consists of seat identity and isEmpty attributes.

4.2.5 Films

This entity stores all movies in the system. Films play in different theatres at different times. Therefore this entity includes time and theatre information. It also contains name of the film, duration and origin country of the film. Films have several artists. So, this entity has a relationship with the “artist” entity.

4.2.6 Artists

This entity is used for storing the artist information of movies. It consists of name and surname of the artist and jobtitle of the artist(i.e director,scenarist,actor...)

4.2.7 Ticket

This entity is used for ticket information storing. Ticket entity has price,isReserved and isSold attribute. Ticket entity depends on film,theatre and cinema information. Therefore each ticket entity has a relationship with the specified entities.

5 Software Design

5.1 Class Diagrams

5.1.1 Controller

Controllers are the basic operation handler classes. Every interface unit has a related controller to process the operations. All of the controllers are derived from an abstract base class “BaseCtrl”. This class defines some basic method interfaces to make sure every controller behaves in the same way.

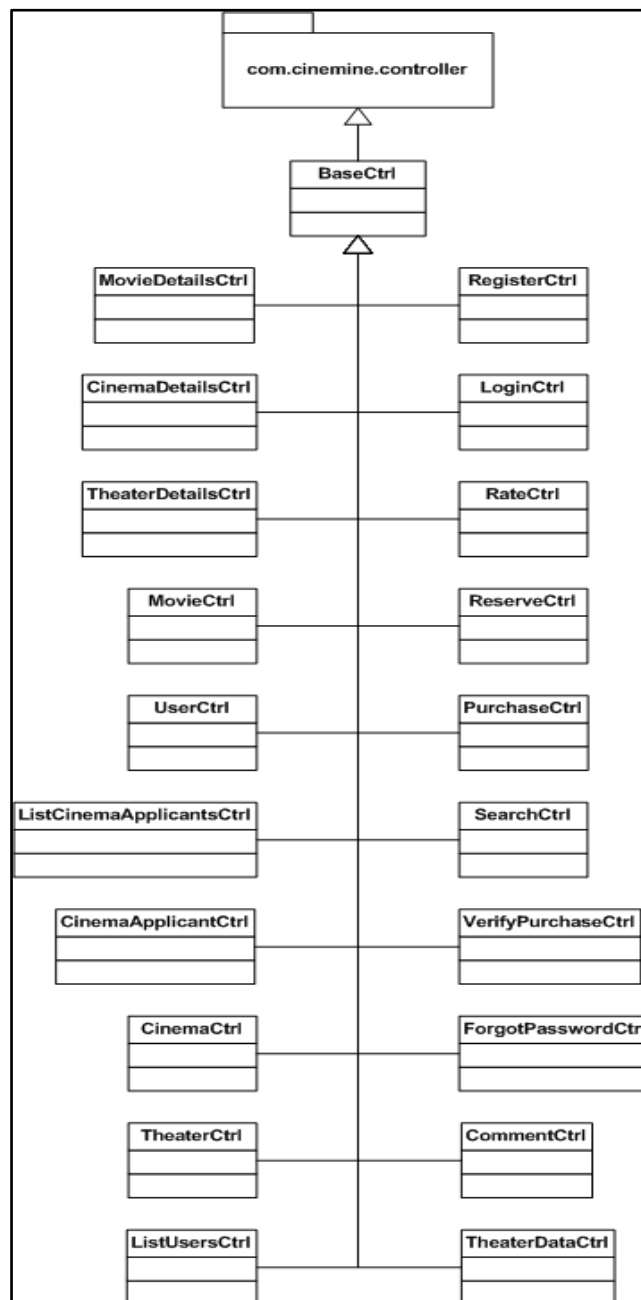


Image 4 - Controller Class Diagram

5.1.2 DAO Class Diagram

DAOs are the main objects which has the rights to communicate with the database layer. Any controller, which needs to access the database should instantiate the related DAO object and use the methods provided by the DAO. This way, the database structure is made transparent to the controllers and all the levels above.

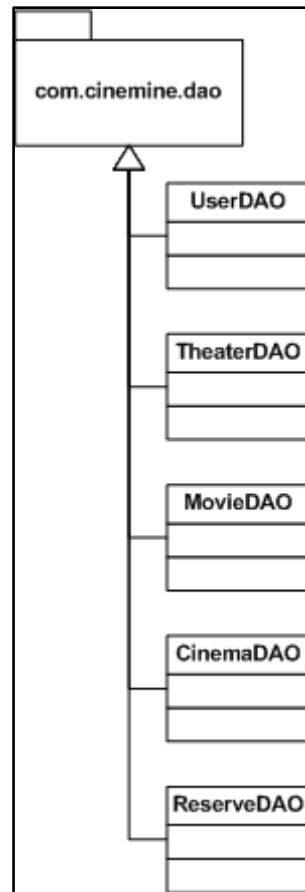


Image 5 - DAO Class Diagram

5.1.3 DOM Class Diagram

DOM is the acronym for the Domain Object Model, and is one of the three legs of the MVC pattern. The data is structured by these objects and used in background processing.

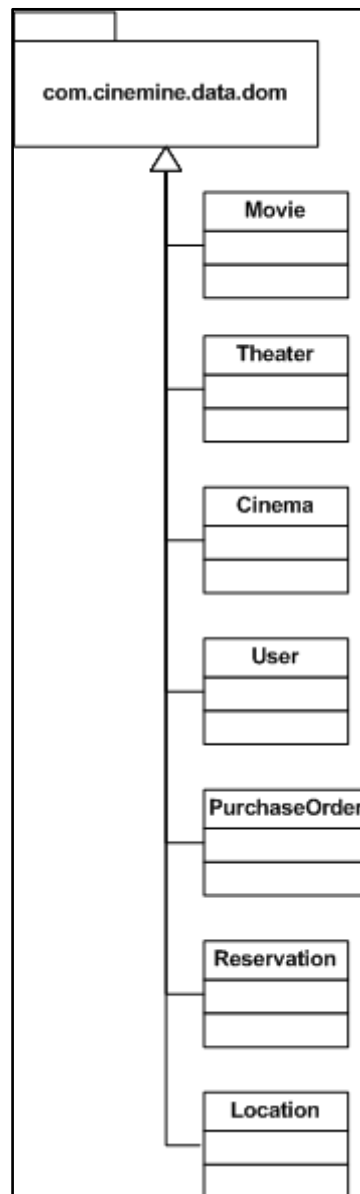


Image 6 - DOM Class Diagram

5.1.4 Servlet Class Diagram

Servlets are used for handling AJAX calls in this application. The reason to use Servlets instead of JSPs is the following: The JSP pages write extra whitespace characters to the response stream, which are inappropriate for XML data.

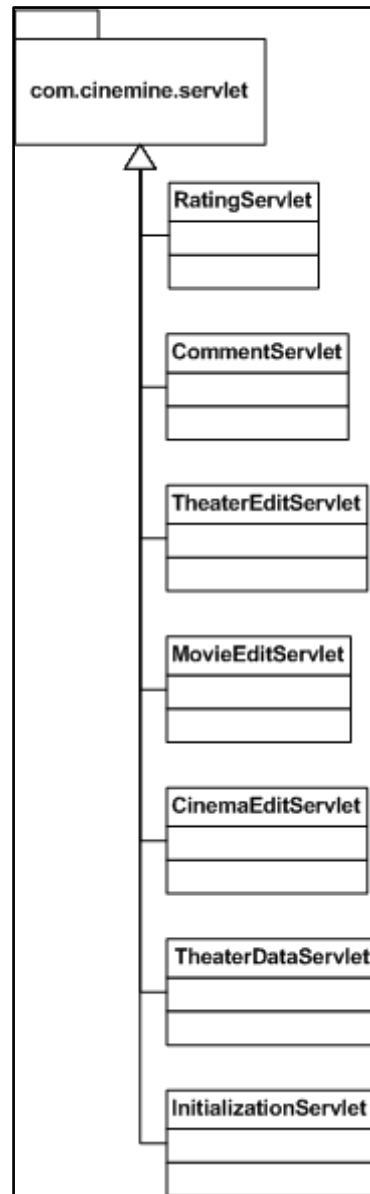


Image 7 - Servlet Class Diagram

5.1.5 Utility Class Diagram

Utilities are small classes to handle some common operations. Most of their methods are static.

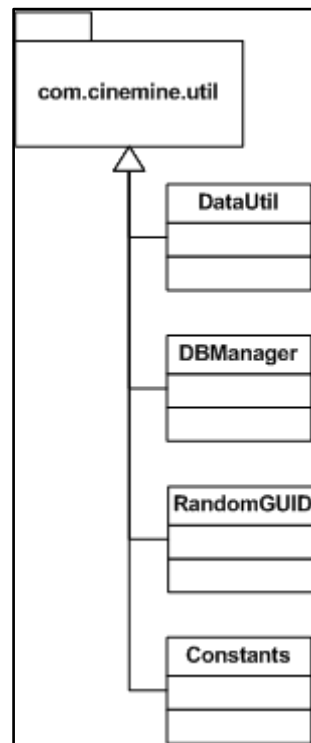


Image 8 - Utility Class Diagram

5.2 Sequence Diagrams

5.2.1 Reservation Sequence Diagram

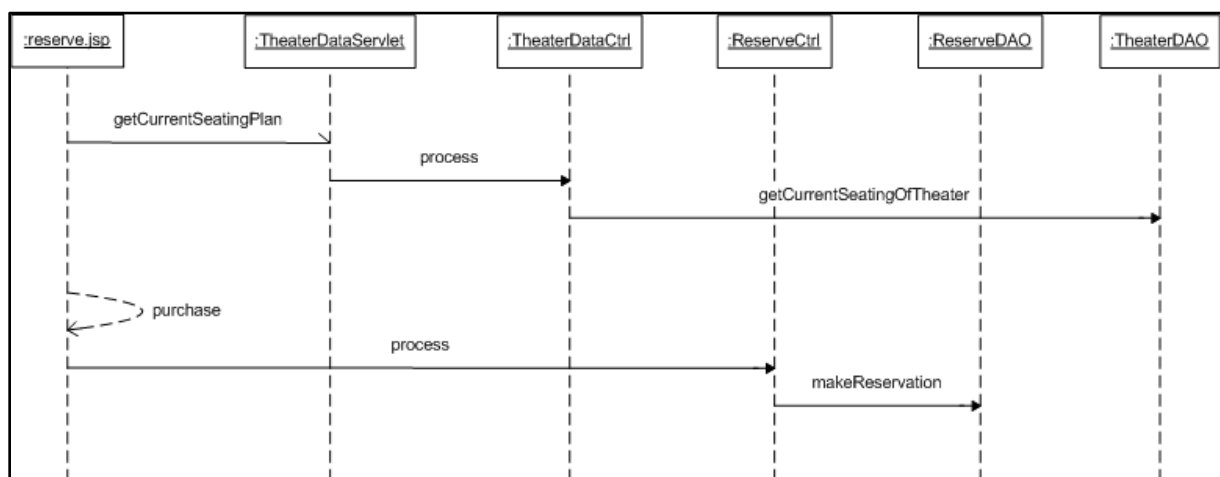


Image 9 - Reservation Sequence Diagram

5.2.2 Cinema Save Operation Sequence Diagram

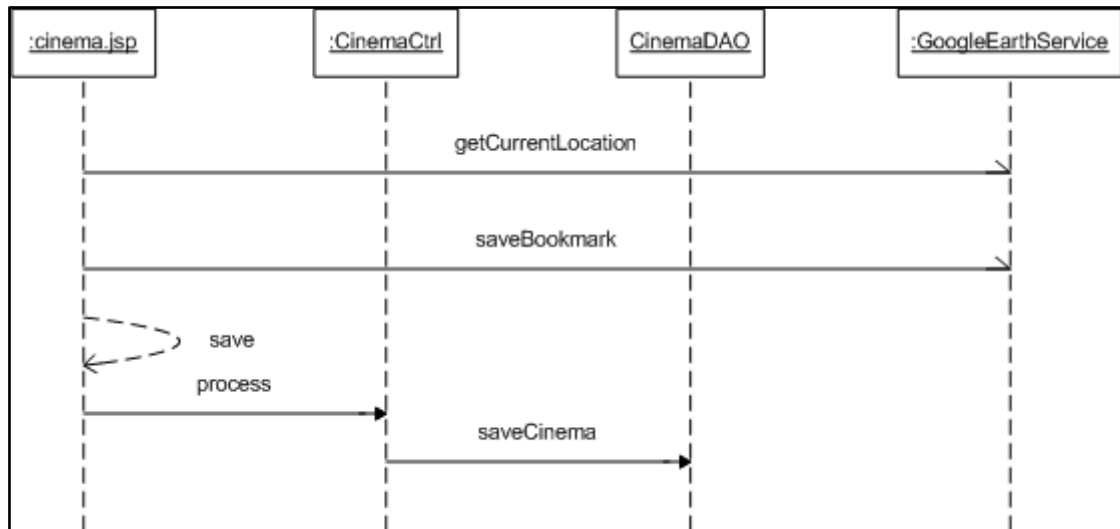


Image 10 - Cinema Save Operation Sequence Diagram

5.2.3 Theater Save Operation Sequence Diagram

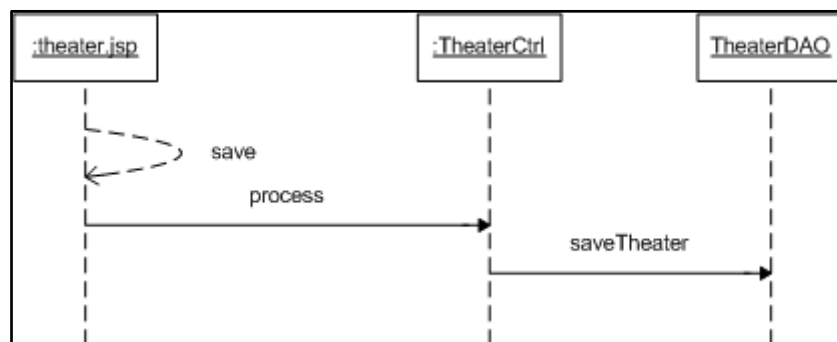


Image 11 - Theater Save Operation Sequence Diagram

6 User Interface Design

6.1 Main Page Design

Site Logo																	
Banner	Promotion & Advertiesment Field 1																
<table border="1"><tr><td>Movie of the Week</td><td>Movie of the Week</td></tr><tr><td>Movie</td><td>Movie</td><td>Movie</td><td>Movie</td><td>Movie</td></tr><tr><td>Movie</td><td>Movie</td><td>Movie</td><td>Movie</td><td>Movie</td></tr></table>	Movie of the Week	Movie of the Week	Movie	Movie	Movie	Movie	Movie	Movie	Movie	Movie	Movie	Movie	<table border="1"><tr><td>User Name</td></tr><tr><td>Password</td></tr><tr><td>Login</td></tr></table> <table border="1"><tr><td>Promotion & Advertiesment Field 2</td></tr></table>	User Name	Password	Login	Promotion & Advertiesment Field 2
Movie of the Week	Movie of the Week																
Movie	Movie	Movie	Movie	Movie													
Movie	Movie	Movie	Movie	Movie													
User Name																	
Password																	
Login																	
Promotion & Advertiesment Field 2																	
Legal Notice Field																	

Image 12 - Main Page Design

6.2 Reservation Page Design

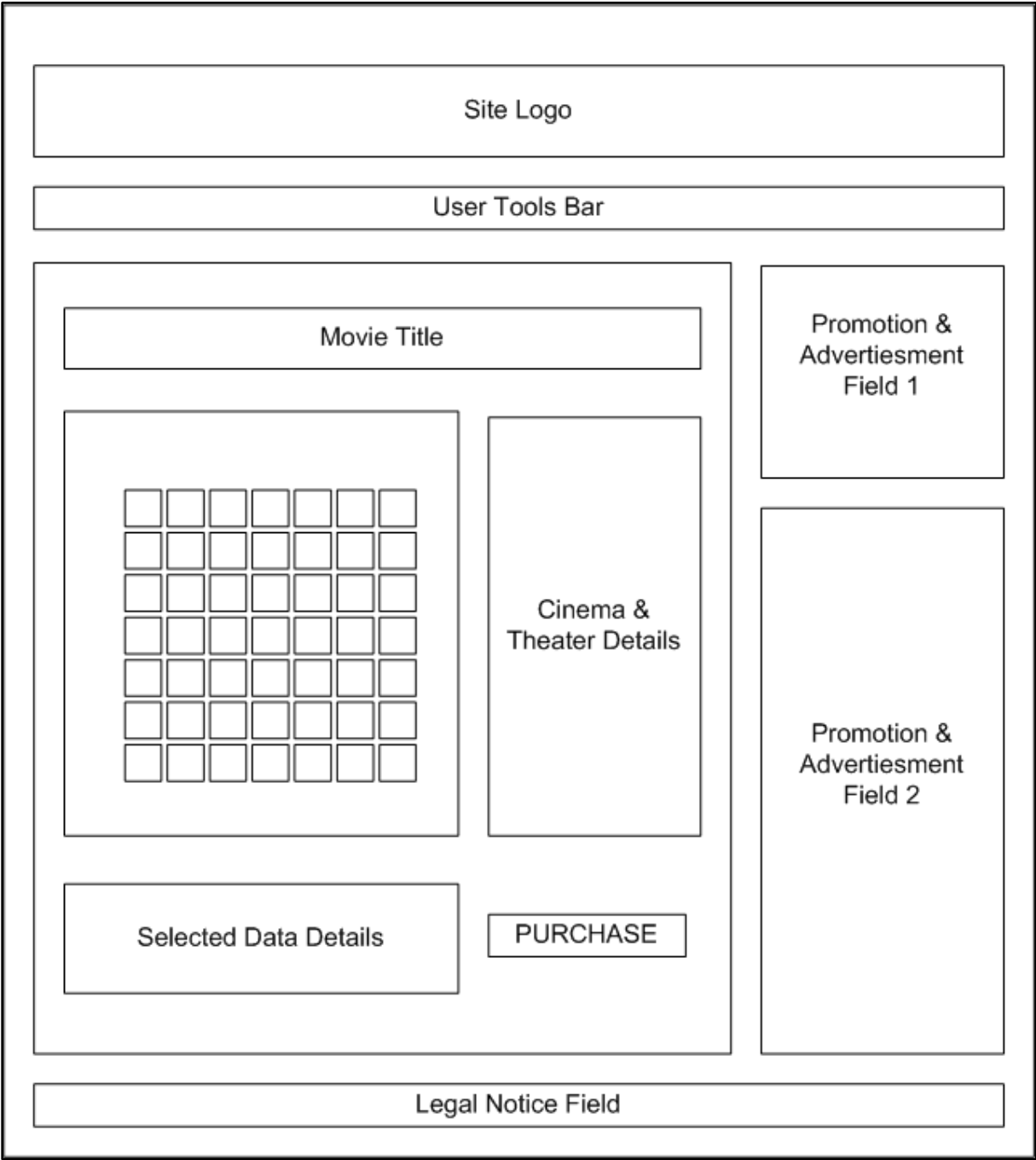


Image 13 - Reservation Page Design

7 Schedule

Number	Task	Resource	Start	End	Duration	2008			2009
						October	November	December	January
1	Communicate With Customer		20/10/2008	24/10/2008	4	■			
2	Problem Definition		20/10/2008	24/10/2008	4	■			
3	Milestone : Communication With Customer		24/10/2008	25/10/2008	1	◆			
4	Initial Analysis		27/10/2008	5/11/2008	7	■			
5	Final Analysis		5/11/2008	16/11/2008	8		■		
6	Milestone : Analysis		17/11/2008	18/11/2008	1		◆		
7	Initial Design		18/11/2008	25/11/2008	5		■		
9	Detail Design		25/11/2008	13/12/2008	12			■	
10	Milestone : Design		13/12/2008	14/12/2008	1			◆	
11	Code Implementation		14/12/2008	2/1/2009	17			■	■
12	Integration Other Software into the System		22/12/2008	2/1/2009	11			■	■
13	Unit Tests		22/12/2008	2/1/2009	11			■	■
14	Milestone : Implementation		2/1/2009	3/1/2009	1				◆
15	Testing		5/1/2009	8/1/2009	3				■
16	Error Correction		5/1/2009	8/1/2009	3				■
17	Milestone : Testing		8/1/2009	9/1/2009	1				◆
18	Packaging		9/1/2009	10/1/2009	1				■
19	Customer Evaluation		9/1/2009	10/1/2009	1				■
20	Milestone : Release		10/1/2009	11/1/2009	1				◆