Cinema Management System

Software Architecture Document

Version 1.2

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 12/12/2020 | 0.1 | Initial document | Ngô Thị Ngọc Thơ |
| 24/12/2020 | 1.0 | Updated document | Ngô Thị Ngọc Thơ |
| 01/01/2021 | 1.0 | Reviewed and updated document | Ngô Thị Ngọc Thơ |
| 03/01/2021 | 1.2 | Reviewed and updated document | Phạm Nguyễn Hữu Phương |

**Table of Contents**

[**1.** **Introduction** 4](#_Toc60606119)

[**1.1.** **Purpose** 4](#_Toc60606120)

[**1.2.** **Scope** 4](#_Toc60606121)

[**1.3.** **Definition, Acronyms and Abbreviations** 4](#_Toc60606122)

[**1.4.** **References** 4](#_Toc60606123)

[**2.** **Architectural Goals and Constraints** 4](#_Toc60606124)

[2.1. Implementation strategy 4](#_Toc60606125)

[2.2. Architectural and design patterns 5](#_Toc60606126)

[2.3. Usability 5](#_Toc60606127)

[2.4. Transaction 5](#_Toc60606128)

[2.5. Security 5](#_Toc60606129)

[**3.** **Use-Case Model** 6](#_Toc60606130)

[**4.** **Logical View** 6](#_Toc60606131)

[**4.1.** **Overview** 6](#_Toc60606132)

[**4.1.1.** **Systems** 6](#_Toc60606133)

[**4.1.2.** **Layering** 6](#_Toc60606134)

[**4.2.** **Architecturally Significant Design Packages** 7](#_Toc60606135)

[**4.2.1.** **Class Diagram** 7](#_Toc60606136)

[**5.** **Deployment View** 8](#_Toc60606137)

[5.1. Overview 8](#_Toc60606138)

[5.2. Topology (single computer/distributed 3-tier) 8](#_Toc60606139)

[5.3. Hardware requirements 8](#_Toc60606140)

[5.4. Platform requirements 8](#_Toc60606141)

[**6.** **Implementation View** 8](#_Toc60606142)

[6.1. Overview 8](#_Toc60606143)

[6.2. Layers 9](#_Toc60606144)

[6.2.1. Presentation Layer 9](#_Toc60606145)

[6.2.2. Business Logic layer 9](#_Toc60606146)

[6.2.3. Data Access layer 9](#_Toc60606147)

**Software Architecture Document**

1. **Introduction**
   1. **Purpose**

The subject "Cinema management system" is intended to fully describe the features of the Cinema management system. It provides project development participants as well as the customer understanding what the system is capable of and how the process is done.

In fact in the software development process, at each stage a different document is built corresponding to that stage. To minimize the complexity of documents in the teaching process, this material is consistently developed throughout the project development process.

* 1. **Scope**

This document covers all stages of the CMS application development process up until the beginning of the programming process.

* 1. **Definition, Acronyms and Abbreviations**

|  |  |
| --- | --- |
| **Term** | **Description** |
| **SAD** | Software Architecture Document |
| **RCP** | Cinema Management System |

* 1. **References**

[1] CMS Software Architecture Document

[2] Documents of lecturer Nguyen Trong Minh Hong Phuoc - Saigon International University.

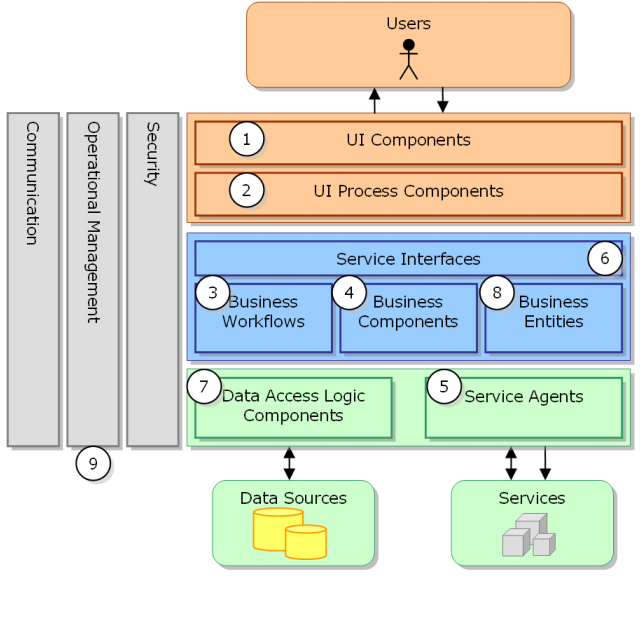
1. **Architectural Goals and Constraints**

There are some key requirements and system constraints that have a significant bearing on the architecture. They are:

1. The system is meant as a proof of concept for a more complete project prediction system to be built in the future. Therefore one of the primary stakeholders in this document and the system as a whole are future architects and designers, not necessarily users as is normally the case. As a result, one goal of this document is to be useful to future architects and designers.
2. The system will be written using Microsoft .NET technologies but will use an open source RDBMS system (SQL Server) for data persistence. These special deployment requirements require additional consideration in the development of the architecture.

## Implementation strategy

The CMS application is to be developed following the *3-tier* architectural style where the application is specific type of client-server system; and Microsoft.NET related technologies are used as the development and deployment environment (as shown in the following figure).



## Architectural and design patterns

In order to develop the CMS application, the main architectural pattern is based on 3 layers and service-oriented architecture. You can also use other styles such as component-based and aspect-oriented architecture.

## Usability

The solution must follow the latest user experience standards and supporting technologies, in order for the human actors to use easily the system and reduce the learning curve. You can use mockups for the screen analysis and design as well as RIA technologies such as Ajax, jQuery, ExtJS, etc.

## Transaction

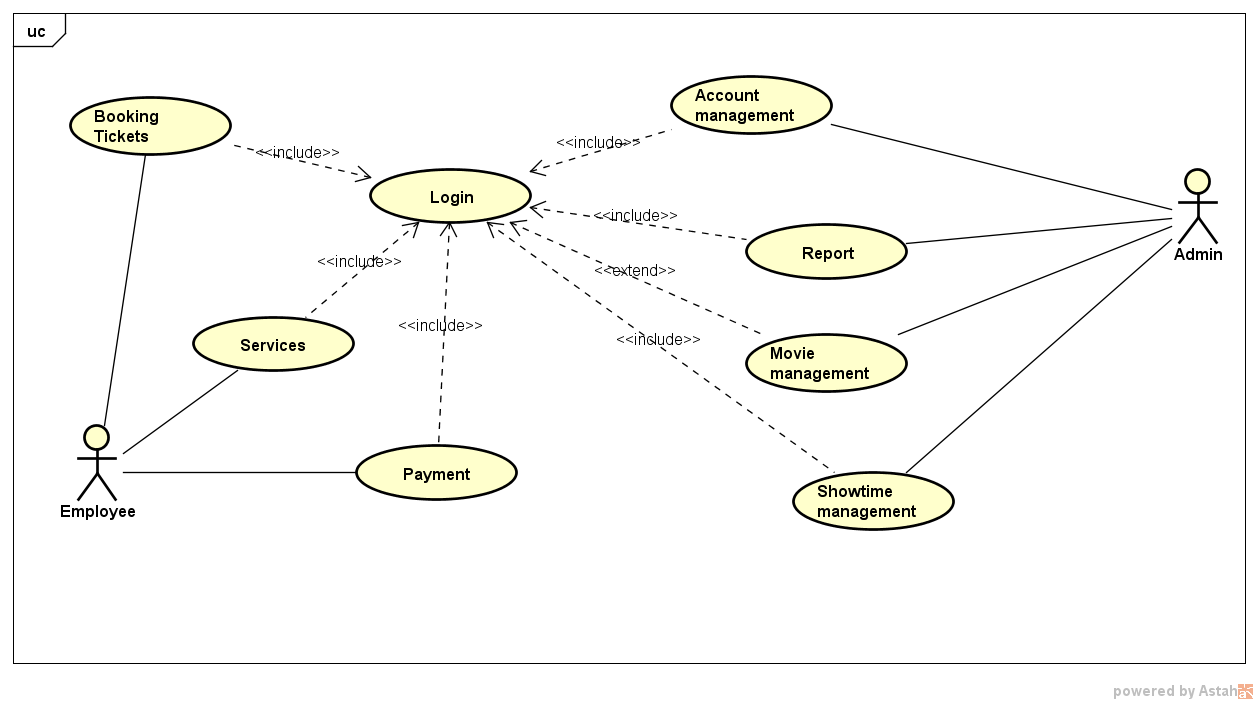
The CMS application is transactional, leveraging the technical platform capabilities. Transaction management model of the Microsoft.NET platform will be reused intensively.

## Security

Software must be highly confidential so that employees cannot access theater statistics, disclose customer information. It must support the following security services:

* Authentication: Login using at least a username and a password.
* Employee cannot enter Admin section.

1. **Use-Case Model**

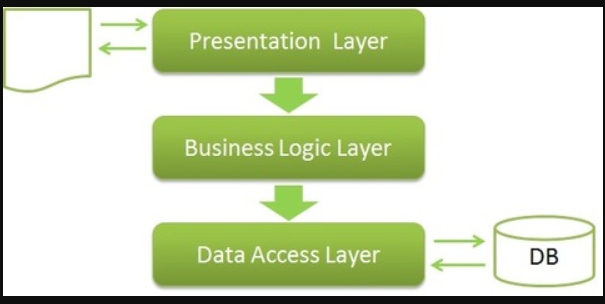


1. **Logical View**
   1. **Overview**
      1. **Systems**

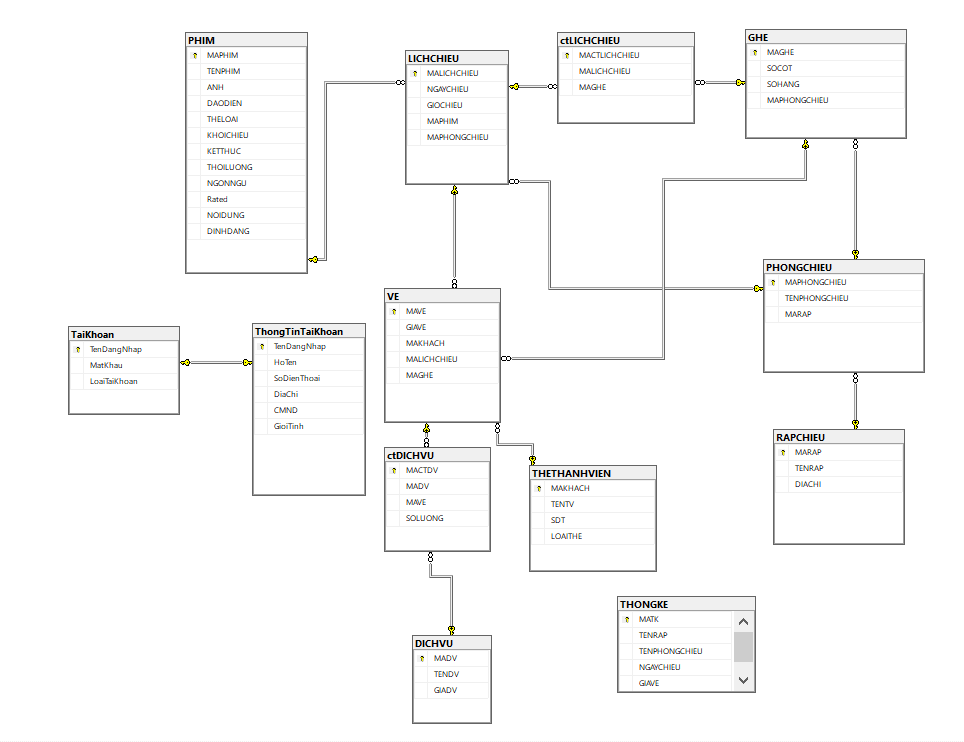
CMS application can be divided into 2 main sub systems.

* Admin management system
* Employee management system
  + 1. **Layering**

CMS application is divided into 3 layers. A 3-tier architecture is a type of software architecture which is composed of three “tiers” or “layers” of logical computing. They are often used in applications as a specific type of client-server system. 3-tier architectures provide many benefits for production and development environments by modularizing the user interface, business logic, and data storage layers. Doing so gives greater flexibility to development teams by allowing them to update a specific part of an application independently of the other parts. This added flexibility can improve overall time-to-market and decrease development cycle times by giving development teams the ability to replace or upgrade independent tiers without affecting the other parts of the system.



* 1. **Architecturally Significant Design Packages** 
     1. **Class Diagram**



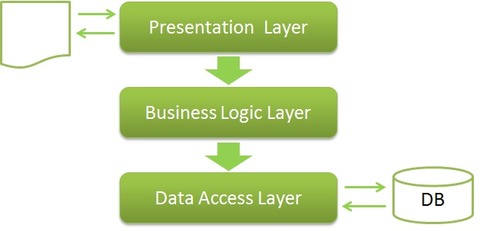
1. **Deployment View**

## Overview

In this section, the deployment is described by identifying the infrastructure in which the CMS application is deployed and run. It is specified any deployment constraints and how the CMS application components are distributed over computing nodes.

## Topology (single computer/distributed 3-tier)

The deployment topology for the CMS application is as shown in the following figure.



## Hardware requirements

According to the high availability, reliability and performance requirements.

## Platform requirements

According to the platform requirements in the sub-section *Implementation Strategy*.

1. **Implementation View**

## Overview

The implementation view depicts the physical composition of the implementation in terms of implementation artifacts such as source code, data, development frameworks, distributable files and executable files.

|  |  |
| --- | --- |
| **Namespaces/Packages** | **Description** |
| *CMS.Presentation* | It´s a Winform application developed using C#. |
| *CMS.DTO* | It´s a package (a folder) inside the *CMS* to hold the objects of the module. Inside this package, you should group the objects by module (similar to the presentation structure). |
| *CMS.BLL* | It´s a Microsoft.NET library developed using C# language and contains the business logic artifacts. |
| *CMS.DAL* | It´s a Microsoft.NET library developed using C# language and contains the data access artifacts. |

## Layers

### Presentation Layer

Presentation Layer is the only layer which is directly connected with the user. So in this matter, it’s also a really important layer for marketing purposes. Presentation Layer is mainly used for getting user data and then passing it to Business Logic Layer for further procedure, and when data is received in Value Object then it’s responsible to represent value object in the appropriate form which user can understand.

### Business Logic layer

The *Business Logic* layer contains all the components that represent the concepts to support the business process. It contains business objects, entities, rules workflows, and so on. The *Business Logic* layer will be developed using C# and the underlying technologies provided by Microsoft.NET.

### Data Access layer

The *Data Access* layer contains all the components that represent the entities mapped to the relational schema and the communication mechanisms with the database engine. It also contains the logic to access other data source as well as the definitions of Web services’ clients (*service agents*). The *Data Access* layer will be developed using C# and LINQ as the ORM technology.