Nicholas Richard Harding

20200344

IT Ticketing Solutions

Team Members the following members contributed to this document  
Nicholas Harding  
Mathew Brencyz  
Pradip Panthi  
Brijesh Kachhadiya

**System Design  
 DOCUMENT**

Task 1

Contents

[1      Introduction 2](#_Toc51516637)

[1.1     Purpose 2](#_Toc51516638)

[1.2     Description 2](#_Toc51516639)

[1.3     Scope 2](#_Toc51516640)

[1.4     Target Users 2](#_Toc51516641)

[2      Solution Design 3](#_Toc51516642)

[2.1     Architecture overview 3](#_Toc51516643)

[2.2     Model Design (Data Model) 4](#_Toc51516644)

[2.3     Functional Decomposition 1](#_Toc51516645)

[2.3.1 Functional Entity Descriptions 1](#_Toc51516646)

[2.3.1 Functional Decomposition Diagram 2](#_Toc51516647)

[2.4     View Design (UI List) 3](#_Toc51516648)

[2.4.1     User Story 1 3](#_Toc51516649)

[2.4.2     User Story 2 3](#_Toc51516650)

[2.4.3     User Story 3 3](#_Toc51516651)

[2.5     Design Considerations 4](#_Toc51516652)

[2.5.1     NF requirement 1 4](#_Toc51516653)

[2.5.2     NF requirement 2 4](#_Toc51516654)

[2.5.3     NF requirement 3 5](#_Toc51516655)

[2.5.4     NF requirement 4 5](#_Toc51516656)

System Design Document

# 1      Introduction

## 1.1     Purpose

The Eye Candy Cinemas Ticketing System is a direct replacement for the existing manual system. Presently, customers are required to complete purchases with assistance from the ticketing staff, either in-store or over the phone.

This new system enables customers to book tickets directly online. It reduces wait times and improves customer satisfaction.

## 1.2     Description

*The system allows the customers to book the ticket directly online. Customers can purchase one or more tickets. It provides the optional to select the seating locations of each. The system is accessible from mobile, tablet, and desktop devices. Customers can make bookings with or without a member account.*

*Ticketing staff can also process transactions on behalf of the customer, using the same online system from in-store devices. Bookings generate an E-Ticket. E-Tickets contain both a QR code and serial number. This enables ticketing staff or self-service booths to confirm the ticket when the customer arrives at the theatre to attend the relevant movie screening.*

## 1.3     Scope

*The first release will focus on online booking of tickets for movie screenings.*

The **core user stor**y is:

* ***US1. As a customer, I want to be able to book movie tickets with my phone or home computer, so that I can do so quickly and from any location.***

The following user stories supplement the core user story with related functionality:

* *US2. As a customer I also want to be able to choose seating arrangement and screening time when making bookings.*
* *US3. As a manager, I want the ticketing system to be able to confirm e-tickets with a manual serial number on the ticket, in so staff can confirm it even if the scanner is not working.*
* *US4. As a customer, I want my tickets to not be tied to my name, so that I can transfer them to someone else.*

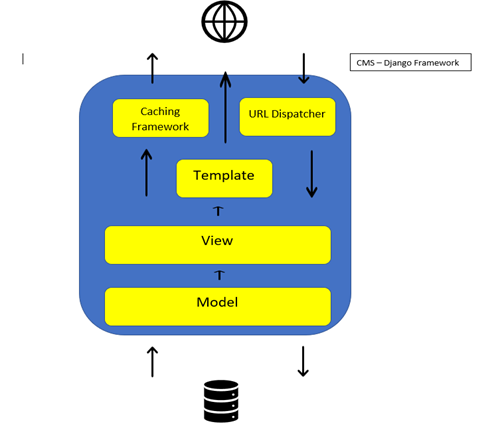
## 1.4     Target Users

The users of the system for this release will be:

* The **Customer:** They book tickets to attend movie screenings at a chosen theatre.
* The **Cinema Manager:** They update the list of current and upcoming movies. They also update screening schedule for movies by theatre.
* The **Ticketing Staff:** They confirm customer E-Tickets. They also process in-store ticket purchases on behalf of customers.

# 2      Solution Design

## 2.1     Architecture overview



*Figure 8 Shows how the content management system operates*

The Django framework provides a Content Management System (CMS) that will provide a fast and simple way to manage the content of the system and construction of the design.

The diagram illustrates one example of CMS architecture. This architecture operates on the Model-View – Controller schema. The Controller layer is further comprised of the caching framework, URL dispatcher, and the CMS templates.

Templates allow developers to customize both:

* The **model (**the organized schema of stored database entities)
* The **views** (the graphical user interfaces displayed on the screen).

The management system will provide the system with:

* A **controller** in the framework and URL dispatcher

Our development team will customize the **Views** and the **Models.** The Django framework will provide the rest.

2.2     Model Design (Data Model)

Based on the business domain model, we have created a detailed data model with the entities we will record in the database. This model includes their attributes and relationships.

**Database Entities**

* **CustomerMembership**
* **PK** customerID
* firstname
* lastName
* contactEmail
* phoneNumber

* **Movies**
* **PK** movieID
* movieTitle
* duration
* genre

* **Cinema**
* **PK**cinemaID
* **FK**theatreID
* **FK** seatingplanID
* maxCapacity

* **Theatre**
* **PK** theatreID
* theatreName
* theatreLocation

* **Bookings**
* **PK** bookingID
* **FK** movieShowTimeID
* **FK**ticketID
* **FK** customerID

* **OnlinePurchases**
* **PK**purchaseID
* **FK** bookingID
* **Weekday**
* **PK**date
* weekday

* **MovieShowTimes**
* **PK**movieShowTimeID
* **FK**cinemaID
* **FK**movieID
* **FK** date
* time
* **Ticket**
* **PK**ticketSerialID
* **FK**bookingID
* **FK**seatingMapID
* ticketQRCode

* **SeatingPlan**
* **PK**seatingPlanID
* **FK**cinemaID

* **Seating**
* **PK**seatingID

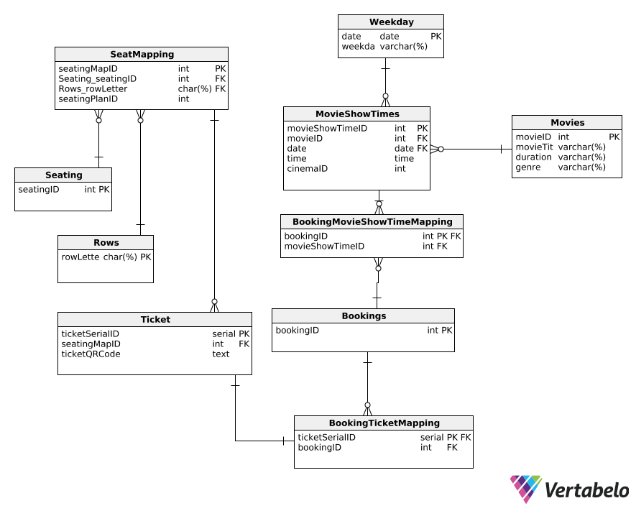
* **Row**
* **PK**seatingRowID

* **SeatingMap**
* **PK**seatingMapID
* seatingID
* RowID
* SeatingPlanID

**Entities Excluded as Out-of-Scope**

* OnlinePurchases
* Bookings
* MovieShowTimesMapping
* BookingTicketMapping
* Ticket
* SeatMapping
* Seating
* Rows
* MovieShowTImes

**1st Release ERD**



*Figure 9 Entity Relationship Diagram shows how the data interacts within the first release database system*

## 

## 2.3     Functional Decomposition

The users that view and use the system are:

* Customer
* Ticketing Staff
* Cinema Manager

### 2.3.1 Functional Entity Descriptions

**Booking**

* (Selection by seat selection, theatre location, cinema number, screen time.)

**Decomposition**

* **Create (Customer)**
* **Delete (Customer)**
* **View (Customer)**

**Ticket**

* (Contains details of the booking for confirmation.)

**Decomposition**

* **View (Customer, Ticketing Staff)**

**Movies**

* (Manager changes the schedule.)
* (Customer searches and views movies.)

**Decomposition**

* **View (Customer, Management, Ticketing Staff)**
* **Create (Manager)**
* **Update (Manager)**
* **\*Delete (Manager)**

*\*Management can only delete movies when all screening sessions expire. This is to enforce referential integrity during future iterations.*

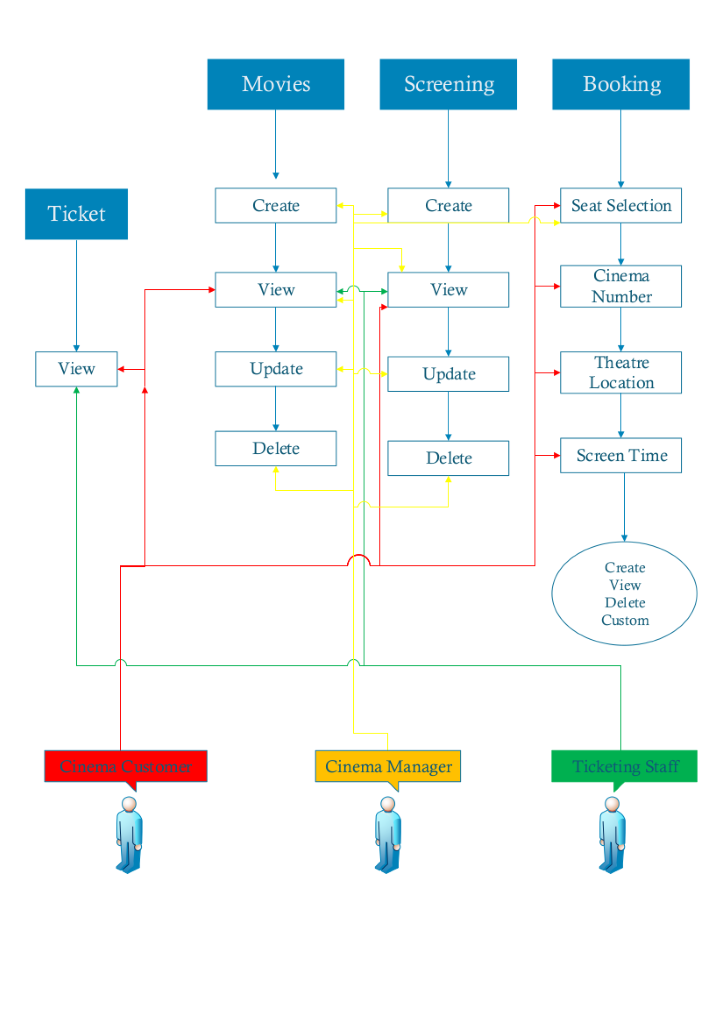
**Screening**

* (Manager edits the screen times, within referential constraints.)
* (Manager can remove movies at will, with no referential constraint.)

**Decomposition**

* **View (Customer, Management, Ticketing Staff)**
* **Create (Manager)**
* **Delete (Manager)**

### 2.3.1 Functional Decomposition Diagram



*Figure 10 Shows the tasks performed by the users*

2.4     View Design (UI List)

These are views and interfaces needed to fulfill each user story:

* Home Page
* Booking Page
* Now Showing Page
* Contact Us

Further pages are out of scope for now. They may be developed in future iterations. The supplied minimum viable product (MVP) may include page links to them. This will exist only as placeholder to indicate further areas of development.

* About Us
* Terms and Conditions
* Privacy Policy

### 2.4.1     User Story 1

***As a customer, I want to be able to book movie tickets with my phone or home computer, so that I can do so quickly and from any location.***

* Create: Booking
* Delete: Booking
* View: Booking
* View: Ticket
* View: Movies
* Make Payment (insert credit card details)

### 2.4.2     User Story 2

***As a customer I also want to be able to choose seating arrangement and screening time when making bookings.***

* Create: Booking
* Delete: Booking
* View: Booking
* View: Ticket

### 2.4.3     User Story 3

***As a manager, I want the ticketing system to be able to confirm e-tickets with a manual serial number on the ticket, in so staff can confirm it even if the scanner is not working.***

* \*View: Ticket

*\*Although this user story originates from the manager perspective, it specifies that ticketing staff will be viewing the system.*

2.5     Design Considerations

Non-functional requirements attendant to the first iteration are described below.

### 2.5.1     NF requirement 1

**Customers, Ticketing Staff, and Managers need a secure means of logging into their accounts**

* Staff will need to login securely
* Manager will need to login securely
* User ID’s and passwords must be stored encrypted
* Measures must be taken to minimize the risk of stolen credentials.

It is required that traffic between client devices and the web server be encrypted to guard against eavesdropping. Django CMS supports HTTPS using SSL.

**Security of Login Credentials**

***Note:*** *The first release will not deal with login credential security.*

Customers must be informed that they are responsible for their own devices and/or credentials. In future iterations the deal with account management, Django offers the ability mechanism to remotely log a user out of all devices, to lock accounts remotely, and enable two-factor authentication via phone text.

Ticketing staff use the system to book tickets, in a similar manner to customers. They only do so during work hours and at the physical theatre counter. Their login may be limited to a set of static IP addresses representing their workstations or devices.

Manager login requires additional considerations. Manager login may be limited to a set of static IP addresses representing their office or personal devices.

**Secure Password Storage (next iteration scope)**

***Note:*** *The first release will not require secure password storage.*

Usernames and password must be stored encrypted on the server. They cannot be stored as plain text.

Django CMS ensures that all usernames and passwords are encrypted via a 1-way algorithm with a unique salt.

### 2.5.2     NF requirement 2

**Protect Customer information Common Security Threats and Attacks**

* **Cross site scripting (XSS) protection**
* **Protection Methods:** Django templates sanitize malicious scripts by escaping specific characters.

* **Cross site request forgery (CSRF) protection**
* **Protection Methods:** POST requests cannot be duplicated. Django POST requests generate a secret element. This secret is unique to a request.

* **SQL injection protection**
* **Protection Methods:** Queries are constructed using parameterization. SQL code is defined separately from parameters. A malicious user cannot submit their own parameters.

* **Clickjacking protection**
* **Protection Method:** Django can be configured to prevent any web pages from being rendered inside a frame. This requires the use of supporting browsers.

* **SSL/HTTPS**
* **Protection Methods:** Django CMS supports the use of SSL certificates for web pages. It also provides the following supporting features:
* **HTTPS Redirection:** The CMS can be configured to route all HTTP requests over secure HTTPS.
* **Strict Transport Security (HSTS):** This is a header. It informs client browsers to always use HTTPS when connecting to a website in the future.

* **Host header validation**
* **Protection Methods:** Django validated host headers against a list of allowed headers that are defined in the configuration settings. Invalid host entries are ignored, preventing page redirects.

* **Session security**
* **Protection Methods:** Django websites prohibit untrusted users from accessing subdomains. This includes session.

### 2.5.3     NF requirement 3

**Secure Network Traffic**

Protect customer data by using the HTTPS and Transport Layer Security. All user data directly goes to the system database and system administer can only get access to these data.

**Usability**

Application is easy to use, it supports and fulfil all the client requirement with effectiveness, efficiency and satisfaction.

### 2.5.4     NF requirement 4

**Portability**

**For customers:**

Our system mainly focus on the customers’ needs and their satisfaction. So, our main priority is desktop and mobile phone interface and for the customers we need to provide wireframes for:

1. Desktop
2. Mobile
3. Tablet

Customers can access to our system via those three options.

**For ticketing staff and management:**

Ticketing staff and management have similar needs, so our main priority is desktop interface. For the ticketing staff and management, we need to provide wireframes for

1. Desktop
2. Tablet