

**Detailed Design Document**

Table of Contents

1. Overview
   1. Introduction
   2. Scope
   3. Goal
   4. User Needs
   5. Intended Audience
2. Architecture
   1. Client-Server
   2. Three-Tier
   3. Data-Centric
   4. Event-Driven
3. Security
   1. Operating System Security
   2. Database Security
   3. Application Security
4. Database
   1. Tables
   2. Database Access
   3. Database Maintenance
5. Hardware
6. User Interface
   1. Home Page
   2. Navigation
   3. Reservation
7. Internal Interfaces
   1. Browser and Server
   2. Server and Database
   3. Admin and Database
8. External Interfaces
9. Top Classes
   1. Admin
   2. Movie
   3. Ticket
   4. Reservation
   5. Confirmation
10. Data Flow
11. Reports
12. Other Output
13. Low Level Design: Classes
    1. Admin Class
    2. Movie Class
    3. Ticket Class
    4. Reservation Class
    5. Confirmation Class
14. Low Level Design: Database Tables
    1. Admin Table
    2. Seat Table
    3. Payment Table
    4. Price Table
    5. Movie Table
    6. Screening Table
    7. Auditorium Table
    8. Ticket Table
15. Web Browser Wireframes
    1. Home Page
    2. Movies Page
    3. Menu Page
    4. Promotions Page
    5. About Us Page
    6. Find Us Page
    7. Order Form
16. Resources
17. **Overview**
    1. **Introduction**

This Detailed Design Document has been created as a reference for the design of Triangle Cinema’s “Triangle Ticketing” web application. Through the design, implementation, and testing of this application, which is based on the specifications listed in the requirements document, Six Pack Consulting will have created an experience that allows the reservation of movie tickets to be efficient and convenient for its target users. Users of the web application will be able to place a ticket reservation from the convenience of their homes or on the go through an internet-enabled device. Users will no longer deal with the risk of driving to the movie theater and not having any seats available for the movie they’d like to watch. The app will also offer contact information, directions to the theater, a menu, and a promotions page to provide additional information to its users.

* 1. **Scope**

Our document was created with the intent to guide Six Pack Consulting’s Team in implementing a production ready web application. If new features are added in future releases, this document will continue to assist those involved in those release dates. This is a living document that will be changed as new features are added, modified, or removed.

* 1. **Goal**

Our goal is to create a web application for a local movie theater called Triangle Cinema and providing efficiency, convenience, and information to its customers. Users will be able to browse movies, reserve tickets on the go, and get directions to the theater as well as get contact information and a look at the menu and promotions.

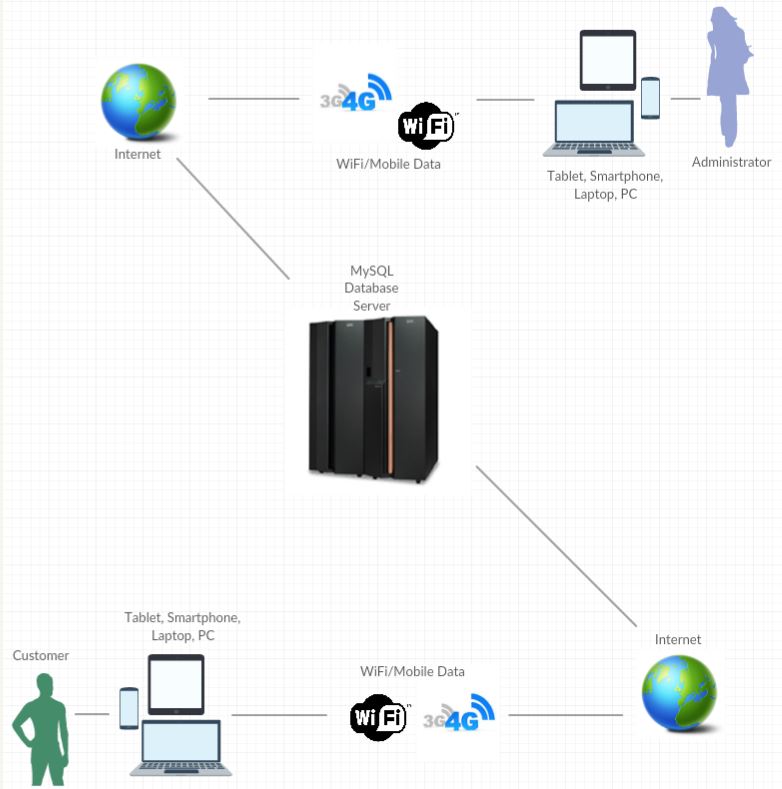
* 1. **User Needs**

Users of this application need a convenient way to reserve tickets through their smartphones, tablets, laptops, and other mobile data or WiFi enabled devices. Regarding the core aspects of the app, users must be able to easily navigate the Triangle Cinema website and browse through the available movies and reserve their tickets on the go. Navigating through the website must be straightforward and user friendly. Once a movie is selected (clicked on), all information about the movie must be displayed as well as a button to start the ticket reservation process. Upon clicking the ticket reservation button, users must be able to select the number of tickets and the group they belong to such as child, senior, student, adult for the appropriate pricing for the purpose of calculating the reservation cost. Users should also be able to access different pages that contain contact information for the Triangle Cinema, a description of the cinema, a page for the food and drink menu, and a page to check current promotions offered by Triangle Cinema.

* 1. **Intended Audience**

This web application is designed to be used by the general public, specifically those in the area in which Triangle Cinema is located. Upon visiting the website users will be able to easily browse through the latest movies being screened at the Triangle Cinema, view the details for each movie such as title, director, synopsis, etc. and reserve their tickets in a streamlined fashion without having to navigate through several webpages. People who like the convenience of getting their tickets in advance without having to stand in long lines and risking having the tickets be sold out will enjoy the ease of use and convenience that the Triangle Ticketing web application will provide.

1. **Architecture**



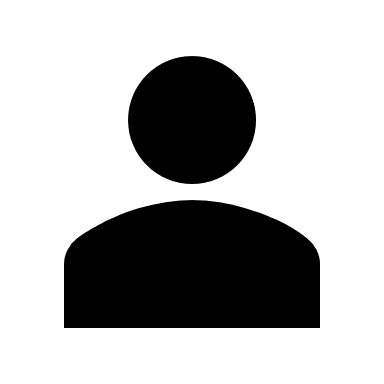
* 1. **Client-Server**

Our architecture focuses on decoupling the various aspects of our web application. Keeping the client separate from the server allows the different pieces of the project to be worked on separately. This results in an application that is easier to maintain over time.

* 1. **Three-Tier**

This model allows for further decoupling of the client-server functions. Having a middle-tier adds an extra layer of security when dealing with data that should not be accessible by non-admin users such as users interacting with the database and deleting or changing data that should remain inaccessible to them.

* *Presentation Layer*: This layer will use a web browser to display data to the users through HTML pages styled with CSS. These pages will include the browsing page, the contact us page, the about us page, the menu page, and the promotions page, as well as the ticket reservation page.
* *Logic Layer:* Django will be used to process requests from the web browser and respond with the correct data from the database. For example, users needing to check movie information such as the film title or the film’s director and admins of the site needing to update movie information.
* *Data Layer:* PostGreSQL will be used to store movie data (title, director, synopsis, duration, etc.), auditorium data, etc. This data will only be readable by non-admin users while admins will be the only ones with writing rights.



**Web Browser**

**Django**

**PostGreSQL**

**Google Maps API**

* 1. **Data-Centric**

Our web application requires admins to access and update data in the database regularly to update movie information, etc. It includes all data associated with the movies displayed on the website available for ticket reservation.

* 1. **Event-Driven**

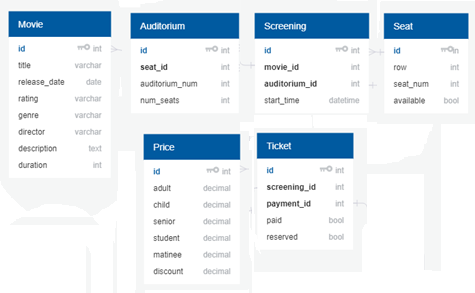
Once a user places a ticket reservation, a confirmation should automatically be displayed containing movie data as well as the total cost of the tickets. When a user clicks on the embedded Google Maps’ directions button, the user will get directions to get to the movie theater.

1. **Security** 
   1. **Operating system security**

* Depends on operating system’s security, which is not handled by Six Pack Consulting.
  1. **Database Security**
* Separate from the web client, Django-enabled admin username and password.
* Username and password of the admins in charge of the database must be strong and secure.

1. **Database** 
   1. **Tables**

This application will use a relational database to store data, specifically, PostGreSQL. Our initial set of tables will be Movie, Auditorium, Screening, Seat, Price, and Ticket.



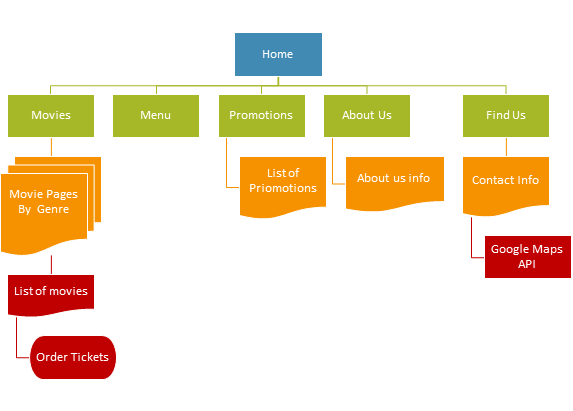
* 1. **Database Access**
     + Admins will have read/write access to the database in order to view and update data about the available movies and their prices.
  2. **Database Maintenance**
     + The database will be updated anytime there is a new movie to screen, a change in movie information, a change in price, or when an error (such as a typo) has been found.
     + The database will be tested and reviewed every time changes are made to its content to ensure the data displayed on the website is correct.

1. **Hardware**

As a web-based application, the following pieces of hardware will be needed:

* + Desktop, laptop, tablet, or smartphone with access to the internet
  + Web server hardware
  + Database server hardware

1. **User Interface**



* 1. **Movies Page (Home Page)**
     + The user can navigate to all other pages from the any page by using a navbar.
     + The navbar should include links to the Movies, Menu, Promotions, Find Us, About Us, and Contact Us pages
     + The Home Page will feature the latest movies.
     + Each movie has its own clickable poster image that redirects users to that movie’s details page.
     + The details page contains the movie title, a synopsis, its rating, and the year of release.
     + Details page contains a reservation form and a button to submit the data inside the form.
  2. **Navigation**
     + The application must be easy to navigate and understand (user friendly) so no training should be necessary to go from one page to another.
     + Buttons and tabs will be clearly labeled.
     + Web client will utilize a navbar menu at the top of the screen for easy navigation.
  3. **Reservation Process**
     + When the “Reserve Tickets” button is clicked on the user will be provided with a form with movie information and textboxes to specify the number of tickets being reserved and a drop-down list to select the age group they belong to. After making these selections and pressing submit the amount to be paid will be shown and the users will see a confirmation page.

1. **Internal Interfaces**
   1. **Browser and Server**

This section is for interactions between the browser and server

* + - Client will send data written into html forms to the server
    - Other data to be displayed to the user include movie title, movie screening times, number of tickets selected, and total ticket cost.

* 1. **Server and Database**

This section is for interactions between the server and the database

* Database can receive SQL commands from server concerning movies selected, movie times, seats available/taken
* Database processes the queries and sends the requested info to the server
  1. **Admin and Database**

This section is for interactions between the Admin and the Database

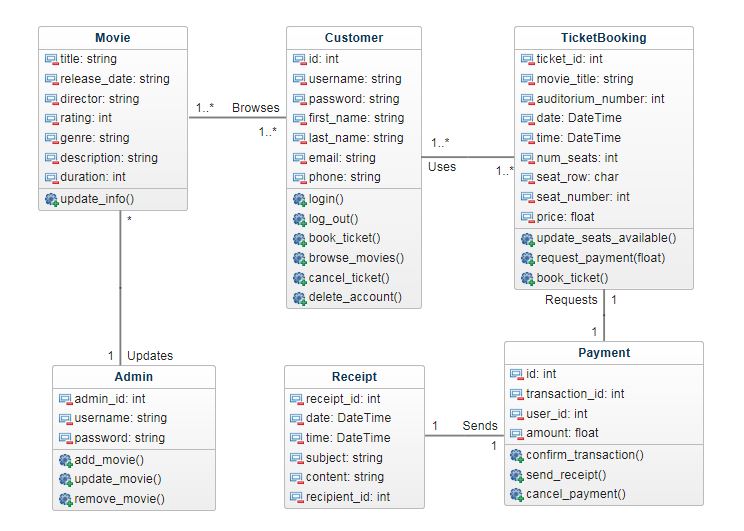
* Admin has login and log out access.
* Admin can read and write to the database in order to add/change/delete: movies available, movie screening times, movie details, etc.

1. **External Interfaces**

For our app’s external interfaces, we will:

* Embedded Google Maps map to allow users to get directions to the Triangle Cinema

1. **Top Classes**

****

UML Class Diagram

* + **Admin**
* Admins can log in and out of their accounts freely
* Admins can add, update, and remove movies
  + **Movie**

The movie class has instance variables for the movie’s title, year of release, director, rating, genre, screening dates, and auditorium of the screening.

* + **Ticket**

Keeps track of the number of tickets available, the movie’s title, and the date and time for the screening.

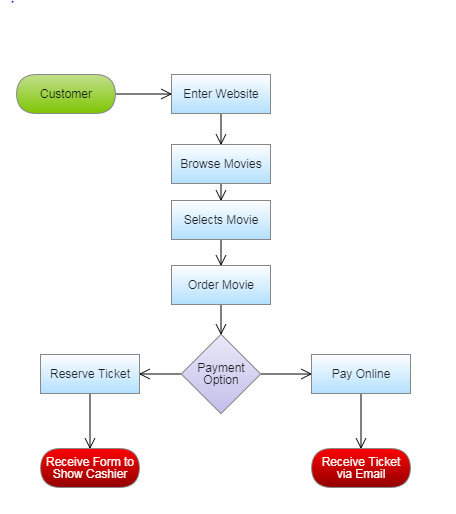
* + **Reservation**

This class contains an id to identify the reservation, and the total cost of the tickets which depends on user selection in the ticket reservation page.

* + **Confirmation**

The confirmation displays data to the user that placed a ticket reservation, this data includes the receipt id, the date and time of the movie’s screening, movie details (title, director, year of release, etc.), and total cost of the reserved tickets.

1. **Data Flow**



Shows a simple ordering process for ordering movie tickets

1. **Reports**

In the future, release reports will be written up and reviewed by administrators to ensure the program and reservations are working correctly. They will be able to gauge how successful the website is by how much traffic it receives each month.

* Testing\Error Reports (weekly)
* Application’s usage(monthly)

1. **Other Output**
   * In a future release reservation confirmation info and promotions will be sent to users via email if they decide to provide their email.

1. **Low Level Design: Classes**

* 1. **Admin Class**

|  |
| --- |
| Admin |
| -admin\_id: int  -username: string  -password: string |
| +add\_movie()  +update\_movie()  +remove\_movie() |

* 1. **Movie Class**

|  |
| --- |
| Movie |
| -title: string  -release\_date: string  -director: string  -rating: int  -genre: string  -description: string  -duration: int |
| +update\_info() |

* 1. **Ticket Class**

|  |
| --- |
| Ticket |
| -ticket\_id: int  -movie\_title: string  -auditorium\_number: int  -date: DateTime  -time: DateTime  -num\_seats: int  -seat\_row: char  -seat\_number: int  -price: float |
| +update\_seats\_available() +request\_payment(float) +book\_ticket() |

* 1. **Reservation Class**

|  |
| --- |
| Reservation |
| -id: int  -transaction\_id: int  -user\_id: int  -amount: float |
| +confirm\_transaction()  +send\_receipt()  +cancel\_payment() |

* 1. **Confirmation Class**

|  |
| --- |
| Confirmation |
| -receipt\_id: int  -date: DateTime  -time: DateTime  -subject: string  -content: string  -recipient\_id: int |

1. **Low Level Design: Database Tables**
   1. **Admin Table**

|  |
| --- |
| Admin |
| id:int<<PK>>  username: varchar  login: bool |

* 1. **Seat Table**

|  |
| --- |
| Seat |
| id:int<<PK>>  row:int  seat\_num:int  available: bool |

* 1. **Reservation Table**

|  |
| --- |
| Reservation |
| id:int<<PK>>  price\_id:int<<FK>>  total:decimal  confirmation:bool |

* 1. **Price Table**

|  |
| --- |
| Price |
| id:int<<PK>>  child:decimal  adult:decimal  senior:decimal  student:decimal  matinee:decimal  discount:decimal |

* 1. **Movie Table**

|  |
| --- |
| Movie |
| id:int<<PK>>  title:varchar  release\_date:date  director:varchar  rating:varchar  genre:varchar  description:text  duration:int |

* 1. **Screening Table**

|  |
| --- |
| Screening |
| id:int<<PK>>  movie\_id:int<<FK>>  auditorium\_id:int<<FK>>  start\_time:datetime |

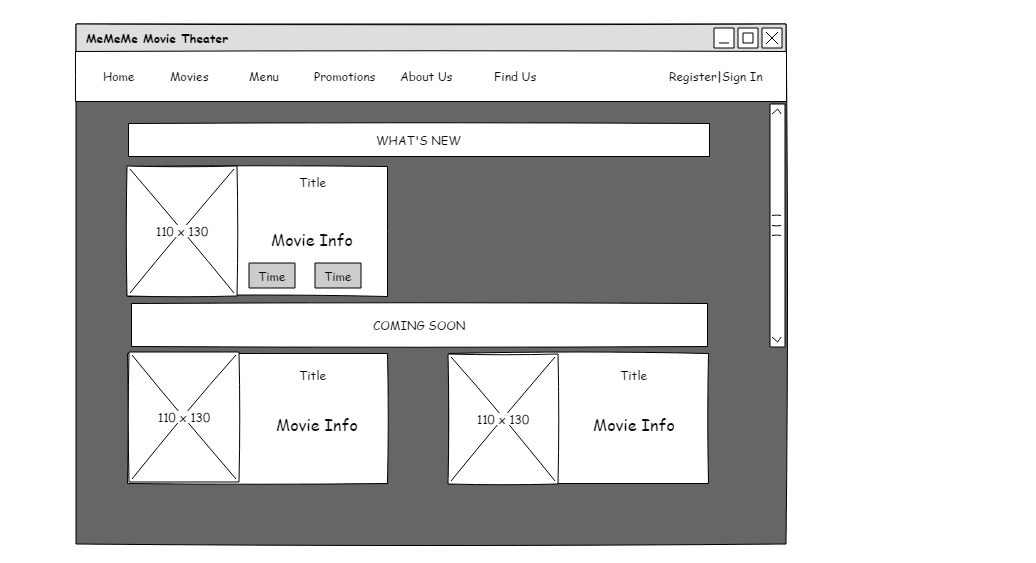
* 1. **Auditorium Table**

|  |
| --- |
| Auditorium |
| id:int<<PK>>  seat\_id:int<<FK>>  auditorium\_num: int  num\_seats: int |

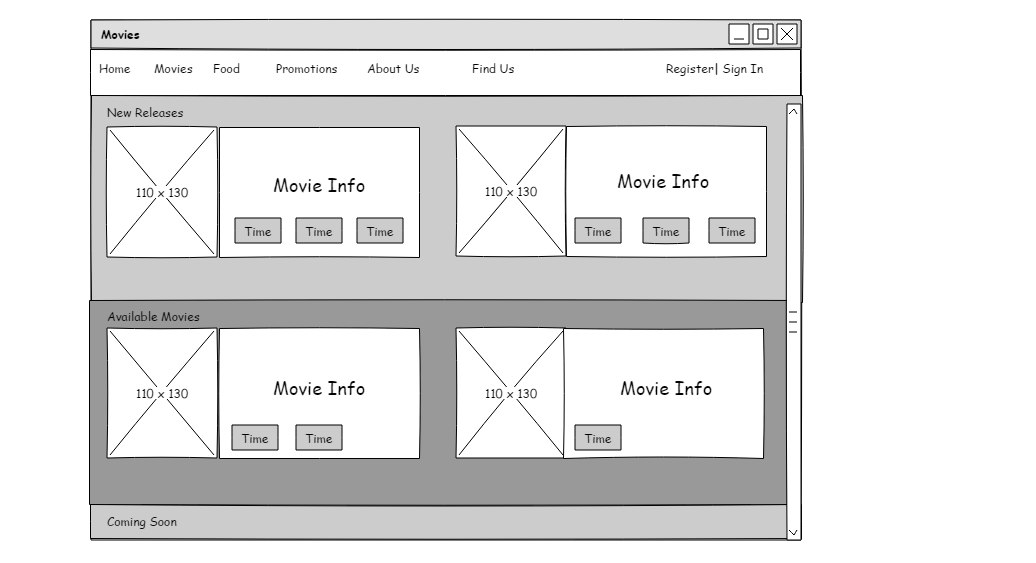
* 1. **Ticket Table**

|  |
| --- |
| Ticket |
| id:int<<PK>>  screening\_id: int <<FK>>  payment\_id: int <<FK>>  paid: bool  reserved: bool |

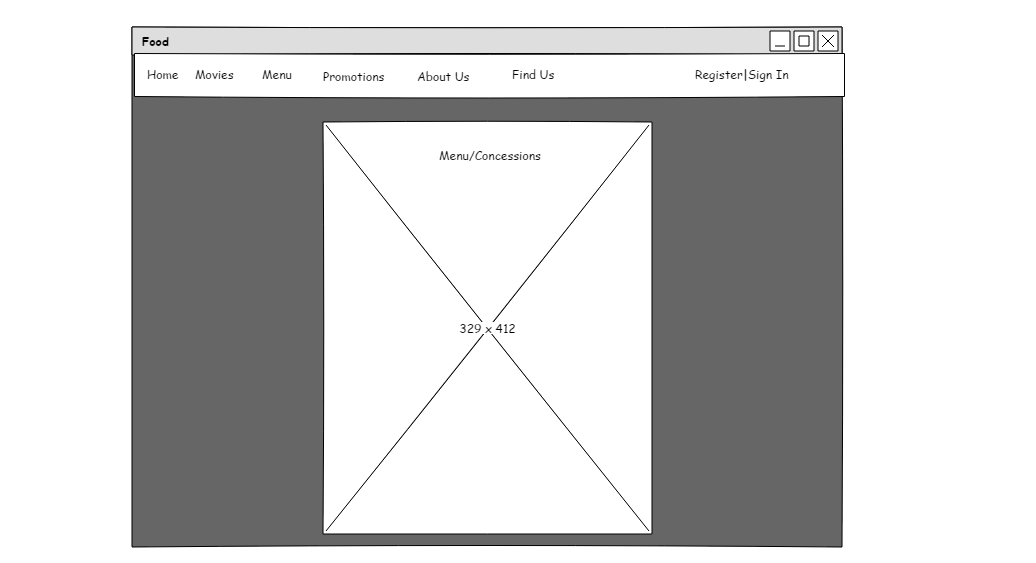
1. **Web Browser Wireframes**
   1. **Home Page**

****

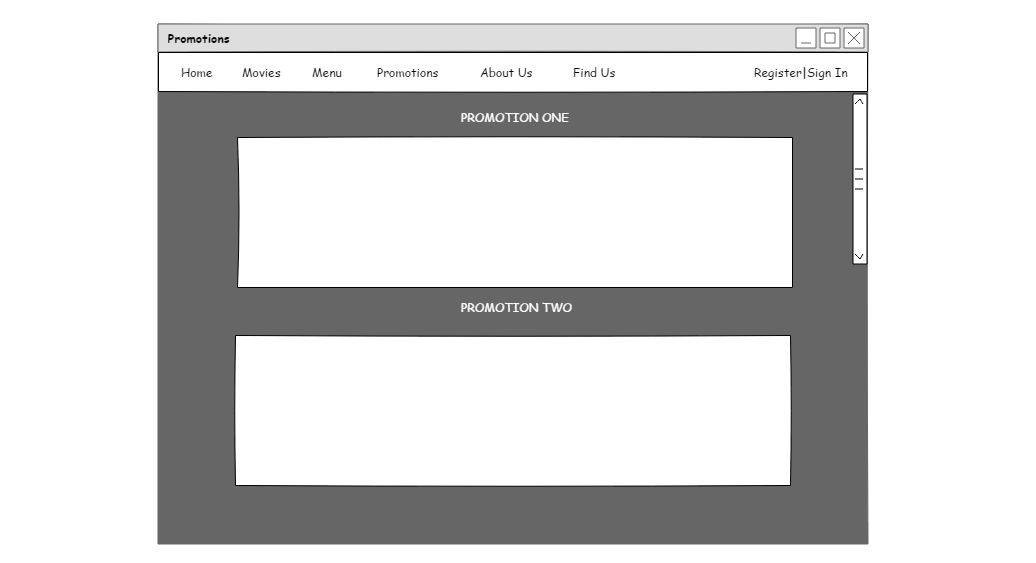
* 1. **Movies Page**

****

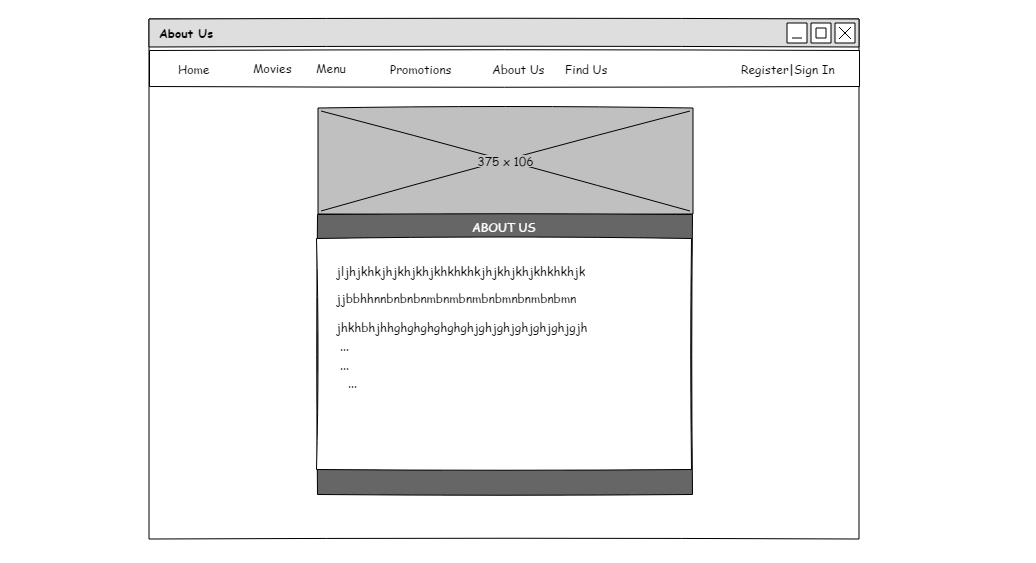
* 1. **Menu Page**



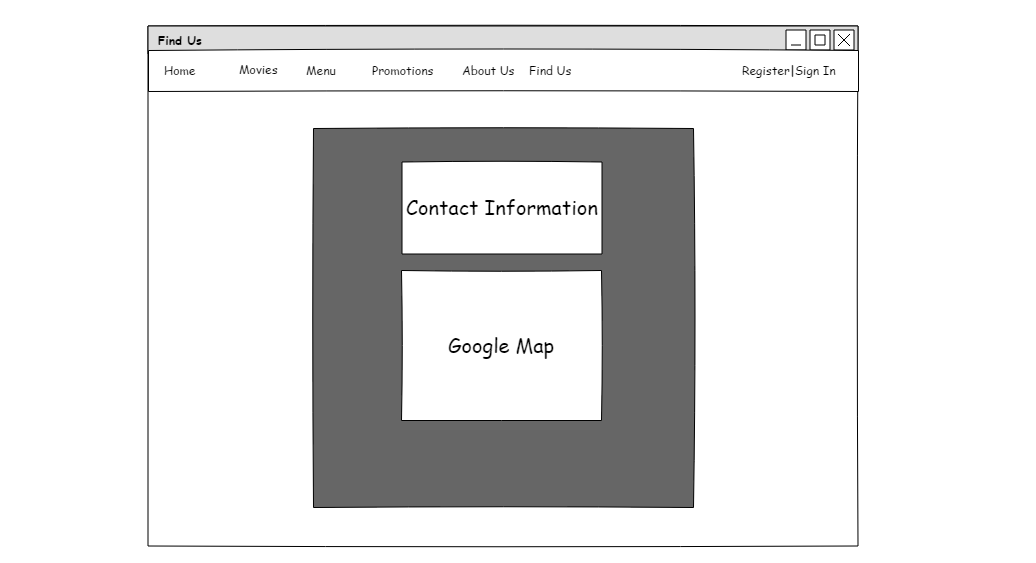
* 1. **Promotions Page**



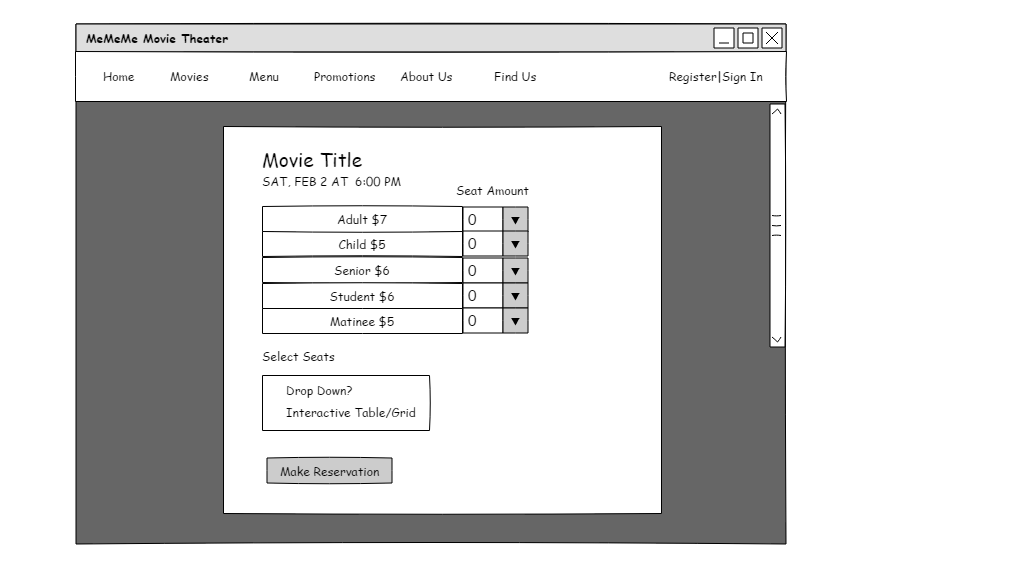
* 1. **About Us Page**



* 1. **Find Us Page**



* 1. **Reservation Form**

****

1. **Resources**
   * Django Tutorials:
   * <https://www.youtube.com/channel/UCWEHue8kksIaktO8KTTN_zg>
   * https://www.codingforentrepreneurs.com/projects/try-django

* Django Reference:
  + <https://djangobook.com/mdj2-models/>
* Diagram Resource:
  + <https://api.genmymodel.com/>