**Software** **Requirements Specification**

**Movie Tracking System**

**Submitted by:**

**Fnu,Sundara Raj Sreenath Sahana(s\_s1008)**

**Vinay Lokesh()**

**Prarthana Gopal kulkarni(p\_k69)**

Table of Contents

1.Introduction.……………………………………………………………………………………………………………………………………4

1.1 Purpose…………………………………………………………………………………………………………………………….4

1.2 Background and related work…………………………………………………………………………………………...5

1.3 Intended audience and reading suggestion……………………………………………………………………….5

1.4 Product Scope……………………………………………………………………………………………………………….…..5

1.5 DEFINITIONS, ACRONYMS, AND ABBREVIATIONS……………………………………………………………...6

2. Overall Description…………………………………………………………………….………………………………………………..…7

2.1 Product Perspective………………………………………………………………………………………………………..…8

2.2 Product functions………………………………………………………………………………………………………….…..9

2.3 User Classes and Characteristics…………………………………………………….………………………………..10

2.3.1 Customer Class……………………………………………………………………….…………………………………….10

3. External Interface Requirements

3.1 External Interface Requirements …………………………………………………………………………………….12

3.2 User interfaces………………………………………………………………………………………………………………..12

3.3 Hardware Interfaces………………………………………………………………………………………………………..18

3.4 software interfaces………………………………………………………………………………………………………….18

3.5 Communication Interface………………………………………………………………………………………………..18

4. Architecture……………………………………………………………………………………………………………………………..…..19

5.Design and Implementation Constraints………………………………………………………..…………………………..…..21

6.Functional Requirements………………………………………………………………………………………..……………………..21

6.1 Menu Page ………………………………………………………………………………………………………………..……21

6.2 Reservation ……………………………………………………………………………………………………………..……..21

6.3 Find Table…………………………………………………………………………………………………………………………21

6.4 Shopping Cart ………………………………………………………………………………………………………………….21

6.5 Check Out Page ……………………………………………………………………………………………………………….22

6.6 Sign In Request ……………………………………………………………………………………………..………………..22

6.7 Admin Reservation Functionality………………………………………………………………………………….....22

6.8 Admin Event Upload ………………………………………………………………………………………………………..22

6.9 Cancel Reservation……………………………………………………………………………………………………………22

6.10.1 User Authentication Response…………………………………………………………………………………….22

6.10.2 Admin and customer Credentials…………………………………………………………………………..……23

6.10.3 Check out Card payment………………………………………………………………………………..…………..23

6.10.4 Storing the order details…………………………………………………………..…………………………………23

6.10.5 Ordered items after login……………………………………………………………………………………………23

6.10.6 Table Reservation details after login…………………………………………………………………………..24

6.10.7 Reservations and Order details on Admin Credentials………………………………………………..24

6.10.8 Cancel Order…………………………………………………………………………………………………….…………24

7. Assumptions and Dependencies…………………………………………………………………………………….……………..24

Dependency Graph……………………………………………………………………………………………………………….25

8. Non-Functional Requirement……………………………………………………………………………………………………….26

8.1 Performance Requirements…………………………………………………………………………………………….26

81.1 Responsiveness…………………………………………………………………………………………………………….26

8.1.2 Efficiency……………………………………………………………………………………………………………..……….26

8.1.3 Accountability…………………………………………………………………………………..……………………….26

8.1.4 Fault Tolerance………………………………………………………………………………………………………….26

8.1.5 Durability………………………………………………………………………………………………………….……….27

8.1.6 Compatibility…………………………………………………………………………………………….……………….27

9.UML Diagrams…………………………………………………………………………………………………………………………….27

8.1 Use case diagram………………………………………………………………………………………………………….27

8.2 Sequence Diagram………………………………………………………………………………………….…………….28

8.3 Deployment Diagram…………………………………………………………………………………………………….30

8.4 Activity Diagram…………………………………………………………………………………………………………….30

10. References……………………………………………..…………………………………………………………………………………..31

**1.Introduction**

* 1. **Purpose**

The document describes the software requirements and specifications for Movie Tracking System. It is expected by the web user of the Movie Tracker System to have basic knowledge about operations performed in the system. The document also provides overview of our software product, its parameters, and goals. This document describes the project's target audience and its user interface, hardware, and software requirements.

**1.2 Background and Related Work**

Lot of end users now in recent days are moving towards watching movies online, writing blogs and critiques about movies, tagging, or saving some as their favorites. Applications that give you an option of watching movies does not usually provide options to user to rate and review or have their personalized pages all together in a single application system. Providing recommendations to watch and welcoming users with their user active session and identifying their likes, dislikes, similar recommendations and accurately providing search results that closely match their preferences is very crucial as end users expect accurate and quick response. We propose to develop the application which solves this problem.

**1.3 Scope**

The project will come up with options to watch movies online, the Movie page will be refreshed often, users can login and have personal recommendations, user can rate a movie, see the top reviews, users can see movies watched by other users and search for a movie. The following is non exhaustive list of topics we consider out of scope: Some of the features in OTT platforms (Netflix) like Continue watching, sort by genre, trending top 10, multiple users accessing one account etc.

**2.Overall Description**

**2.1 Product Perspective**

The application also shall provide user login with valid user authentication, if not a user an option to sign up. The main page intends to capture recently released movies (updated every week from available API’s). The movies that have been reviewed by frequent users. The application also intends to show the number of frequent users and the number of movies they have watched, and the number of reviews written on the available movie list in the application system. The product system also aims to provide an overview of the top reviewers based on the review count.

**2.2 Product Functions**

* Provide secure user authentication to login.
* Use an API to fetch recent movies and its data to display.
* Should allow users to perform a text-based search closely matching movies searched by user using specific keywords
* Will allow user to mark available movies as favorites
* Should allow a valid user to write reviews of their liking
* Should have REST API endpoint for CRUD operations
* Will allow user to create personalized movie dairies
* It should enable users who are logged in to review other user’s dairies
* The application should display users by the number of top reviews provided on various movies available in the Web UI (fetched from the REST API)
* Our application should also aim at providing personalized user page for each user in the active session who has logged in with valid credentials.
* Personalized page of each user shall include their favorite movies watched; reviews provided by that specific user as well as recently watched movies by the end user.
* Web application should provide movie recommendation based on the favorited movie.
* The application should also ensure that certain actions like posting reviews, and adding movie favorites are only allowed If users are logged in.
* Application should also allow user to manage their own user diary once logged into the web application and unable to edit other user details/reviews/favorites.

**3. External Interface Requirements**

**3.1 User Interfaces**

The application GUI provides easy access to Login, recently added movies, see top reviewers, recent reviews with containers, grids allowing for easy control by a keyboard and a mouse. Preliminary knowledge of knowing the tabs available is self-sufficient to navigate across pages in our application. [write technical stuff]

**3.2 Hardware Interfaces**

To optimize the memory utilization, our system can interact with any hardware component as it can be used from any web browser with any operating system. The application cannot be accessed from mobile devices. We do not have any hardware interfaces other than a customer using our software on their systems which saves us memory and it is light weight for moving the application or hosting it on the cloud or even taking backup.

**3.3 Software Interfaces**

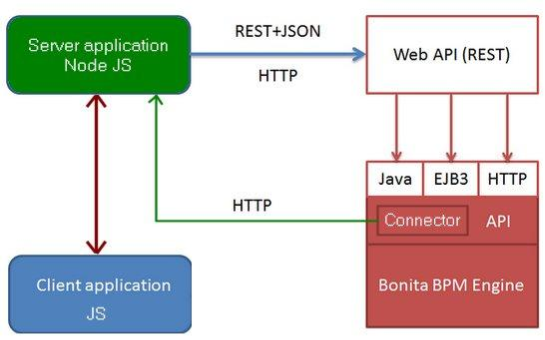
Software interfaces required for the project are open-source web hosting platforms where our web application binaries and packages can be pushed, built, and deployed. There is interaction from our Node js middle ware and Ejs package component which internally processes requests and process information which are retrieved from the database. Node acts as a central repository to handle function calls and modify the object to view or the model layer which is here the view and the data model.

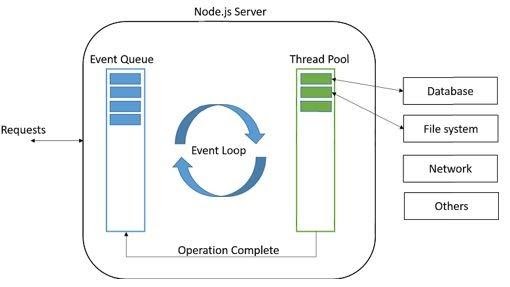
**3.4 Communications Interfaces**

Internally there are various interactions with different interfaces, communication interface used with webserver is HTTP and HTTPS secure protocols to the application server which is 8080 and 443 ports using TCP/IP protocolsAs our project is hosted on local host the Node js project will by default run on port 3000

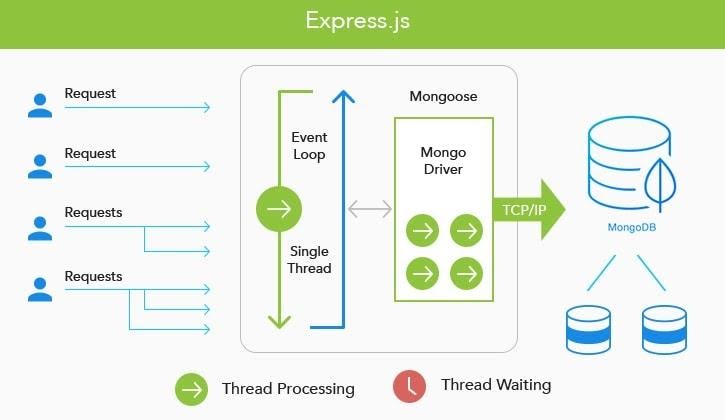
**4. Architecture**

MovieTracker software system follows modern architecture of MVC in Node Js, where the Model corresponds to business validation and modelling of data using Mongodb, The View layer consists of the user interface using HTML,CSS for effective interactions, better accessibility and enhancing visual effects, where as Controller layer is configured as routes for request and response handlers. The invocation starts from app.js which in turn invokes index.js to find the path to various pages linked to button events. There is always a need to make a web server call from our web application to API endpoints which is to be accessible to the browser for which we use ExpressJs. The process then will flow to the database as a request object for DML or DDL operations to fetch or store necessary information for further validation and use cases.





Express is a routing and middleware web framework which is essentially used for function calls. Middleware has access to request and response object functions in the application. It makes changes to these objects such as GET,PUT,POST methods which are processed concurrently through Mongo drivers called Mongoose - This driver installation and call is made from Node Js through Express using connection string details(admin access credentials) in db.js file. The process helps in storing or retrieving information from or to the database  using a secure TCP/IP protocol which uses a three way handshake mechanism to ensure there is no data loss and the copy of the data is always available and accessible through our web application.



**5.Design and Implementation Constraints**

Major constraints of the system are:

* Dependency on the database
* Internet connection is a constraint for this system because system is available from cloud therefore customer needs to have good network connection to connect to our web interface
* Secure connection layer like SSL or TLS needs to be enabled and maintained for the active sessions where purchases are made using confidential information
* As it is a central repository, there might be network connection delays to fetch from the edge locations or storage locations from the database hosted on the cloud cluster.
* Operating System Constraint-
* Device Constraint – Movie Tracker’s user interfaces should be compatible with computers. However, running on small android/IOS mobile devices is not implemented.
  1. **Functional Requirements:**

The following are the core functions of the Movie tracking system:

**Homepage** – Our movie tracking home page will have the option to see the recent movies, Login, sign up, about and user tab. Users can also see recent revies on the home page without logging in.

**Sign up /Login:** User from the home page can navigate to sign up or Login where new users can register to login or returning users can Login.

**Recommendations:** This function will recommend movies to user based on there recently matched movie category. For example, a user who has watched movie which belong to comedy genre will have suggestions of movies from the comedy genre.

**Logging User Dairy:** Users have the option to see and maintain their personal dairy. Users can mark their favorite movie by clicking an icon on the movie description. He will also have the option to add a movie to his dairy. in the dairy page user can see the movies user has watched and can also his all-time favorite movie.

**My Page:** After login, the user can see the list movies he has recently watched, list of movies that are his favorite and the list displaying the recommendations. Here there will be an option to see the movies user has recently reviewed as well.

**User reviews:** In this page user can see the reviews he has given and will also have the option to delete the reviews.

**Popular Movies this week:** In this page the user can see the popular movies that are trending, these are fetched from our API

**Search for movie:** User can search a movie; recently searched movies can also be seen. The search is fetched with this mechanism

**7.Assumptions and Dependencies**

We have assumed that the user can operate these system's basic functions including but not limited to being able to power on the system, login and open either Internet Explorer or Mozilla Firefox, and navigate the browser to the address of this Movie Tracking system website.

We have dependency on [write here]

**8.Non-Functional Requirements**

**8.1 Performance Requirements**

We plan performance testing to address the bottlenecks of the system and to fine tune the system by finding the root cause of performance issues. Performance testing answers to the questions like how many users the system could handle, how well the system could recover when the no. of users crossed the maximum users, what is the response time of the system under normal and peak loads.

**8.1.1: Responsiveness**

In case of scrolling through the menu there should be no delay of more than 10-15 seconds before the next page of menu items is displayed. The information will be ensured to be fetched within 30-60 seconds by using indexing at document level.

**8.1.2: Efficiency**

It uses LRU algorithm to process the requests with minimum delay. Efficiency of the planning modules is deplorable to function and retain the quality of functions and data.

**8.1.4: Fault tolerance**

we are maintaining backup of the database and the tables to recover if any case of sudden crash or any human error. We also have ensured to keep the copy of our application and binaries in the cloud infrastructure as well on local drive if any case of media recovery is required.

**8.1.5: Durability**

in our project we are creating a security encrypted layer for that particular session in order to save the that particular card information for the user. If the system crashes too, the data base information will always have a backup and ensure high availability to users. The application being able to sustain any accidental incidents have been ensured.

**8.1.6: Compatibility**

Multiple functions are called which work as independent calls to respective pages without effecting the working of other functionalities which can be said as compatibility in our application structure.

**9.UML Diagrams:**

**9.1 Use Case Diagram:**

**9.2 Use cases:**

**9.2.1**

|  |  |
| --- | --- |
| **Use case** | **Manage Users** |
| Brief Description | Admin manages the user accounts and deletes them in case of irrelevant content |
| Pre-conditions | Admin accesses correct web page and is logged in |
| Post-conditions | Admin is shown the list of users who have registered |
| Events Flow | Admin views users list.  Admin views the contents reviewed by user  Admin deletes data |
| Alternative Flow and Exceptions | If the internet is down, Error page is displayed. If no one registered, a page with no user data is displayed. |

|  |  |
| --- | --- |
| **Use case** | **Recommendation** |
| Brief Description | Functionality suggests a movie to user |
| Pre-conditions | User is logged in and has watched at least one movie |
| Post-conditions | Functionality will decide a movie to be suggested based on previous movie genre |
| Events Flow | User logs in  User views the recommendation web page. |
| Alternative Flow and Exceptions | If the internet is down, Error page is displayed. If no one registered, a page with no user data is displayed. |

|  |  |
| --- | --- |
| **Use case** | **Logging user diary** |
| Brief Description | User maintain their personal dairy by marking their favorite movie. |
| Pre-conditions | User accesses correct web page and is logged in |
| Post-conditions | User is shown the list of movies with date watched, title, favorite movie and an option to delete the movie from their diary. |
| Events Flow | User logs in.  User views his ‘my dairy’ data  User checks his favorites or deletes a movie from page |
| Alternative Flow and Exceptions | If the internet is down, Error page is displayed. If no one registered, a page with no user data is displayed. |

|  |  |
| --- | --- |
| **Use case** | **My Page** |
| Brief Description | User finds their movie page |
| Pre-conditions | User accesses correct web page and is logged in |
| Post-conditions | User is shown the different lists such as recently watched, all-time favorites, movie recommendations and recently viewed movies, along with total number of reviews and movies watched |
| Events Flow | User logs in.  User views his ‘my page’  User checks list from the page |
| Alternative Flow and Exceptions | If the internet is down, Error page is displayed. If no one registered, a page with no user data is displayed. |

|  |  |
| --- | --- |
| **Use case** | **My Reviews** |
| Brief Description | User can review one or more movies |
| Pre-conditions | User accesses correct web page and is logged in |
| Post-conditions | User is shown the list of movies with along with its poster, an option to delete. |
| Events Flow | User logs in.  User views his ‘my reviews’ page  User can delete a movie from his review list.  User can post his review which will be saved.  User can view his review along with other reviews by different user. |
| Alternative Flow and Exceptions | If the internet is down, Error page is displayed. If no one registered, a page with no user data is displayed. |

|  |  |
| --- | --- |
| **Use case** | **Search** |
| Brief Description | User can search a movie of his choice |
| Pre-conditions | User accesses correct web page and is logged in |
| Post-conditions | User is shown the list of movies and a search box |
| Events Flow | User logs in.  User views ‘Movies’ page  User searches for a movie.  It will be listed if it is available otherwise necessary message will be provided. |
| Alternative Flow and Exceptions | If the internet is down, Error page is displayed. If no one registered, a page with no user data is displayed. |

|  |  |
| --- | --- |
| **Use case** | **Popular movie** |
| Brief Description | User can find all the movies available |
| Pre-conditions | User accesses correct web page and is logged in |
| Post-conditions | User is shown the list of movies |
| Events Flow | User logs in.  User views ‘Movies’ page  User can search for a movie.  User can check top reviewers. |
| Alternative Flow and Exceptions | If the internet is down, Error page is displayed. If no one registered, a page with no user data is displayed. |

**9.3 Sequence Diagram**

**9.4 Deployment Diagram:**

**9.5Activity Diagram:**

**10.References**

Developed with best practices for node js**<https://apiko.com/blog/node-js-architecture-tips/>**

 We are using Mongo dB Atlas to host our mongo DB database instance and using mongoose to connect from node :  [**https://docs.atlas.mongodb.com/getting-started/**](https://docs.atlas.mongodb.com/getting-started/)

Software Development Practices, Software Complexity, and Software Maintenance Performance: A Field Study Rajiv D. Banker, Gordon B. Davis and Sandra A. Slaughter

A Control Theory Perspective on Agile Methodology Use and Changing User Requirements. Likoebe M. Maruping, Viswanath Venkatesh and Ritu Agarwal

[**https://www.researchgate.net/publication/243766301\_When\_Concepts\_Point\_at\_Other\_Concepts\_The\_Case\_of\_UML\_Diagram\_Reconstruction**](https://www.researchgate.net/publication/243766301_When_Concepts_Point_at_Other_Concepts_The_Case_of_UML_Diagram_Reconstruction)

[**https://creately.com/blog/diagrams/sequence-diagram-tutorial/**](https://creately.com/blog/diagrams/sequence-diagram-tutorial/)

[**https://www.altexsoft.com/blog/business/functional-and-non-functional-requirements-specification-and-types/**](https://www.altexsoft.com/blog/business/functional-and-non-functional-requirements-specification-and-types/)