



## NTUAflix - Software Requirements Specification Document

---

SoftEng23-26

## Table of Contents

1 Introduction .....	1
1.1 Introduction: purpose of the software .....	1
1.2 Interfaces .....	2
1.2.1 Component Diagram .....	2
1.2.2 Deployment Diagram .....	2
2 Software Requirements Specifications .....	4
2.1 Use cases .....	4
2.1.1 Use Case 1: Search by genre.....	4
2.1.2 Use Case 2: Search by rating .....	<b>Error! Bookmark not defined.</b>
2.1.3 Use Case 3: Search people .....	<b>Error! Bookmark not defined.</b>
2.1.4 Supplementary Diagrams which are used in all Use Cases .....	20
2.2 Data Organization Requirements.....	26
2.2.1 Entity Relationship Diagram .....	26
2.3 Other requirements .....	27
2.3.1 Software availability requirements .....	27
2.3.2 Security requirements .....	27

## 1. Introduction

### 1.1. Introduction: purpose of the software

NTUAflix is a film database where users can navigate through a big catalogue of Movies and TV-Series (AKA titles). NTUAflix allows its users to preview movies and TV shows by providing their identifying characteristics.

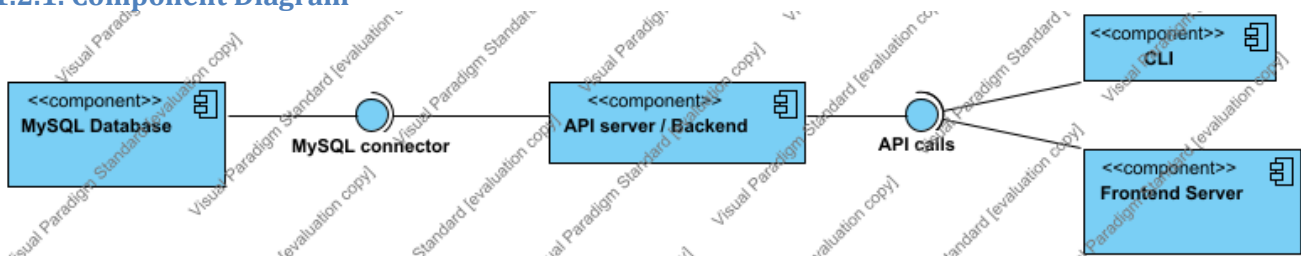
It has three main use cases.

1. The users can browse the provided collection of titles and filter their genre through the "Search By Genre" button.
2. The users can look for a title based on its rating by selecting the "Search By Rating" button and then providing the desired minimum rating value.
3. NTUAflix provides users the option to browse the catalogue through each title's contributing members with the "Search By Person" button.

Once a title or a person has been selected, NTUAflix displays a detailed page containing the key characteristics of the selected item (title or person) as well as active links to related content (similar titles and contributors for titles - title appearances, best rated and latest appearance for people).

## 1.2. Interfaces

### 1.2.1. Component Diagram



Below, a brief description of all components is provided;

External interfaces include the MySQL database server which is accessed through the MySQL connector



#### API calls

The endpoints that are accessible by all users of the application.



#### API server / Backend

The application's backend server, which implements the REST API endpoints.



#### CLI

Command Line Interface for use by Users to access the Endpoints of the application. It runs directly on the application's backend server.



#### Frontend Server

Server for the frontend of the application. It will be used by Users to browse movies, details and people.



#### MySQL connector

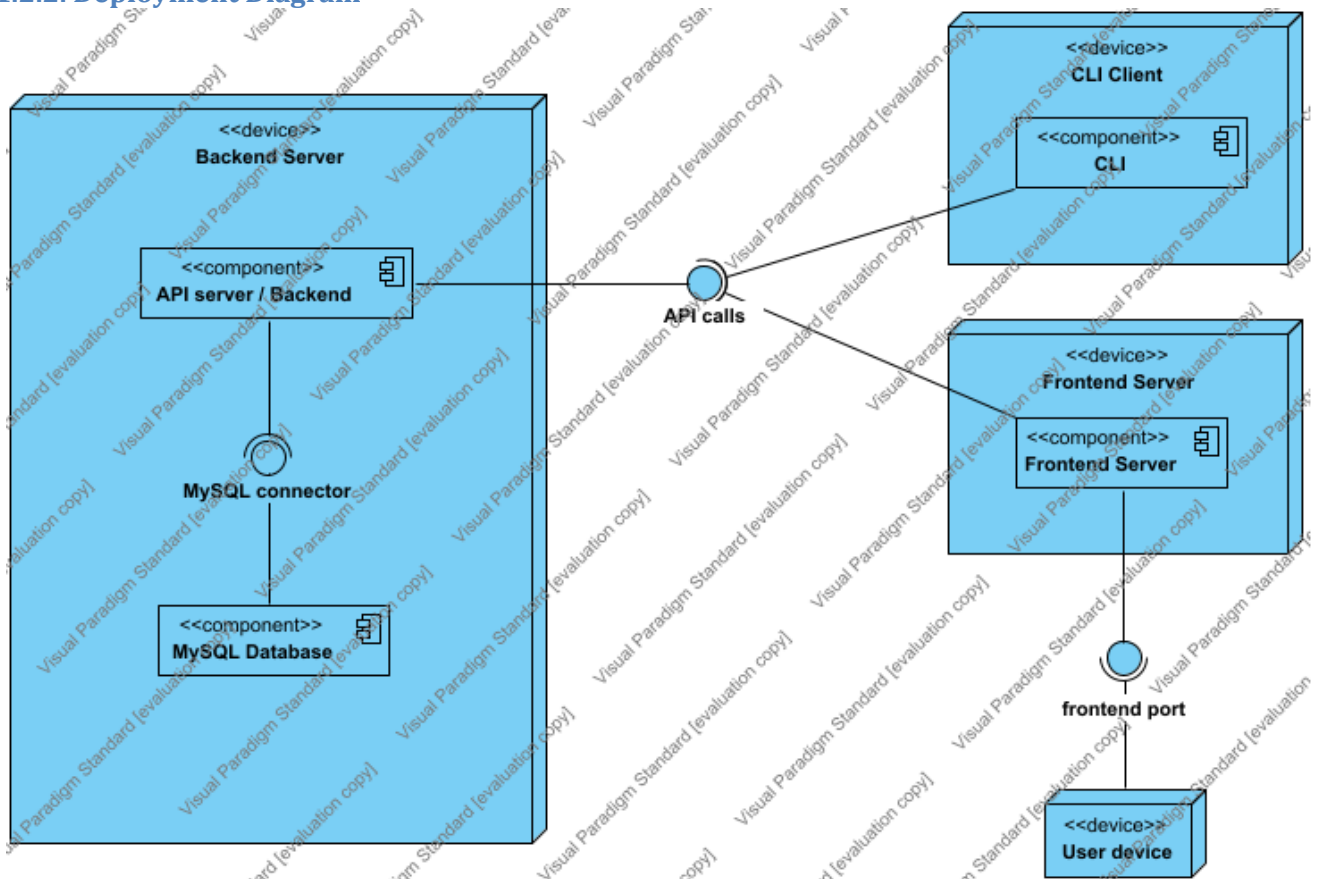
MySQL Connector for connecting to the database



#### MySQL Database

Holds all the actual information in the form of tables compliant to the MySQL schema and is where users' actions translated into queries end up searching.

### 1.2.2. Deployment Diagram



Below, a brief description of all components is provided;  
User interfaces include the the CLI as well as the Frontend server both of which access the database through RESTful API as shown above.



#### **API calls**

The endpoints that are accessible by all users of the application.



#### **API server / Backend**

The application's backend server, which implements the REST API endpoints.



#### **Backend Server**

Is comprised of the MySQL database as well as the API server as the backbone of the application that serves the incoming requests



#### **CLI**

Command Line Interface for use by Users to access the Endpoints of the application. It runs directly on the application's backend server.

## CLI Client

Encapsulates the corresponding component(CLI) as the broader device

## frontend port

Is included as the medium between the user's device and the Frontend Server

## Frontend Server

Encapsulates the corresponding component(Frontend) as the broader device

## Frontend Server

Users interact with the Frontend Server to browse Movies, Series, People and access details about important information about them.

## MySQL connector

MySQL Connector for connecting to the database

## MySQL Database

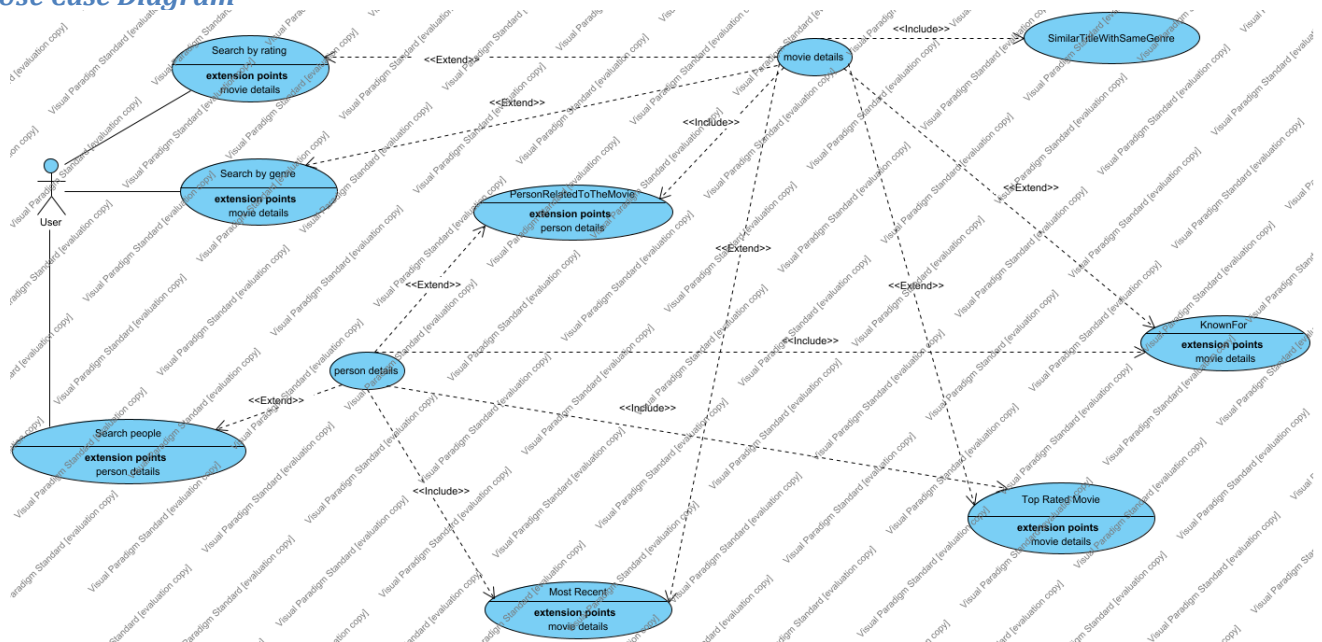
Holds all the actual information in the form of tables compliant to the MySQL schema and is where users' actions translated into queries end up searching.

## User device

# 2. Software Requirements Specifications

## 2.1. Use cases

### Use Case Diagram



### 2.1.1. Use Case 1: Search by genre

This use case corresponds to the first function of the browser which is title catalogue filtering via genres.

#### 2.1.1.1. Users-Roles

The roles involved in this particular use case are the user and the information archive from which the database extracts information and is updated.

#### 2.1.1.2. Execution Requirements

The only requirements are that the user looks up a valid genre which i.e. means that it must be a real genre and a genre-category that the said archive contains.

#### 2.1.1.3. Execution Environment

- Frontend Server
- API Server/Backend
- MySQL Database

#### 2.1.1.4. Input

The user's choice of genre from the list provided

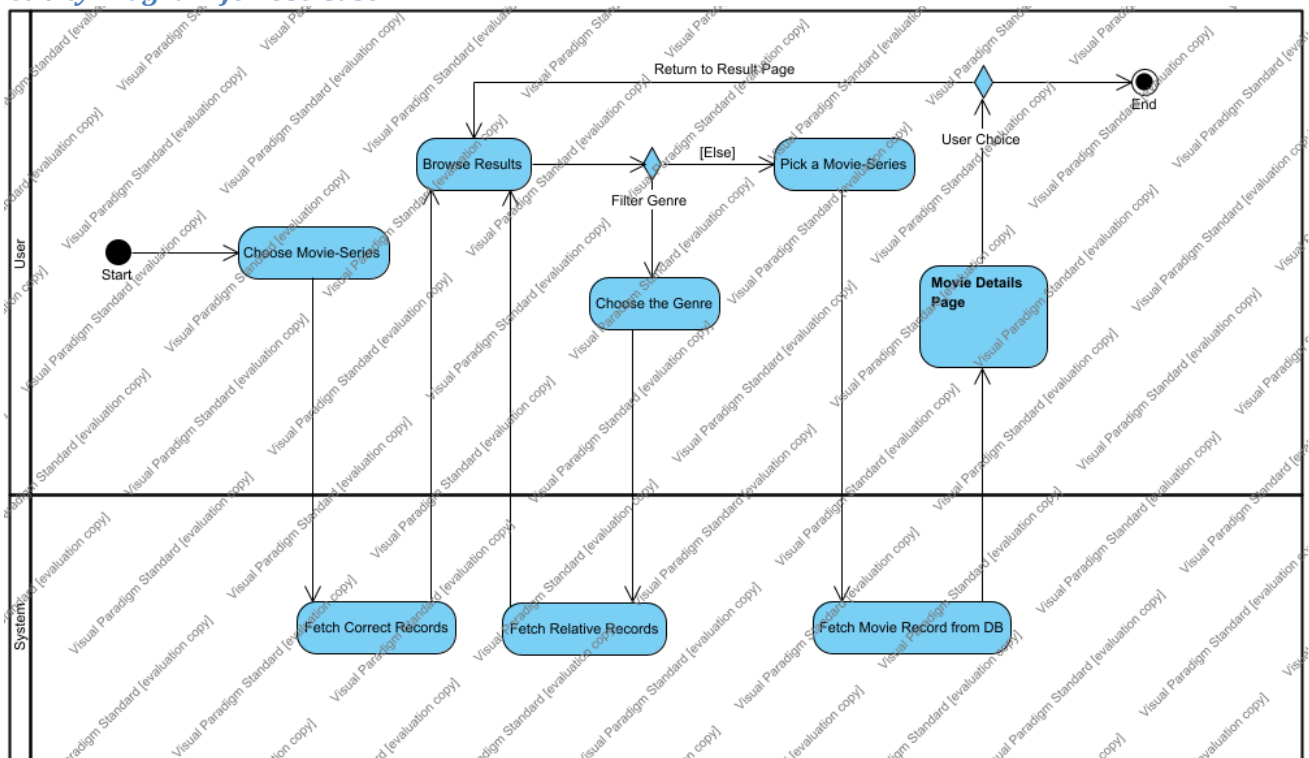
#### 2.1.1.5. Chain of Events

- The user visits the website which lands on the welcome page
- The user clicks the "Search by Genre" button
- The user clicks on the "Select a Genre" field and a list expands
- The user selects one of the genres from the list
- The user clicks the "Filter" button
- The results are now filtered and the user can only browse the titles with the specified genre

#### 2.1.1.6. Output

The output is the list of titles that comply with the chosen genre

#### Activity Diagram for Use Case 1



#### Browse Results

The user browses the displayed movies and series. In this page, the user can decide on what movie or TV series to click on based of the cover, the title and the rating.

#### Choose Movie-Series

The user clicks on the movies-series page in order to browse the movies and series in the database.

#### Choose the Genre

The user is able to specify the genre they want the movies and the series to be in.

#### End

The user leaves "NTUAflix".

#### Fetch Correct Records

The system fetches the data for the movies and the series from the database in order to display them to the user.

#### Fetch Movie Record from DB

The system fetches the data from the database for the movie or the series the user picked in order to display them to the user.

#### Fetch Relative Records

The system fetches the filtered data for the movies and the series from the database in order to display them to the user.

#### Filter Genre

The user is able to filter the results based on the genre. They can decide if they want to filter the results or not.

#### Movie Details Page

Navigate to Movie Details Page.

#### Pick a Movie-Series

The user clicks on the movie or series they want to learn more details for.

#### Start

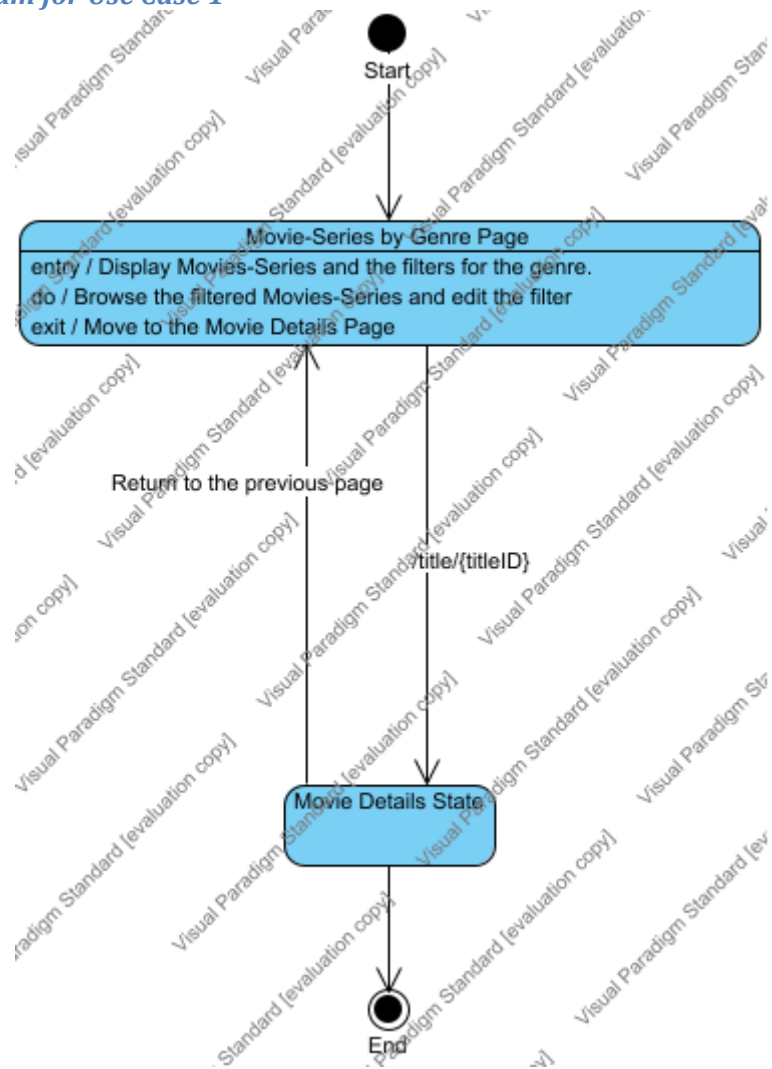
The user opens "NTUAflix"

#### Swimlane4

#### User Choice

The user has the option to go back to the previous page in order to continue browsing the collection.

### State Machine Diagram for Use Case 1



● End

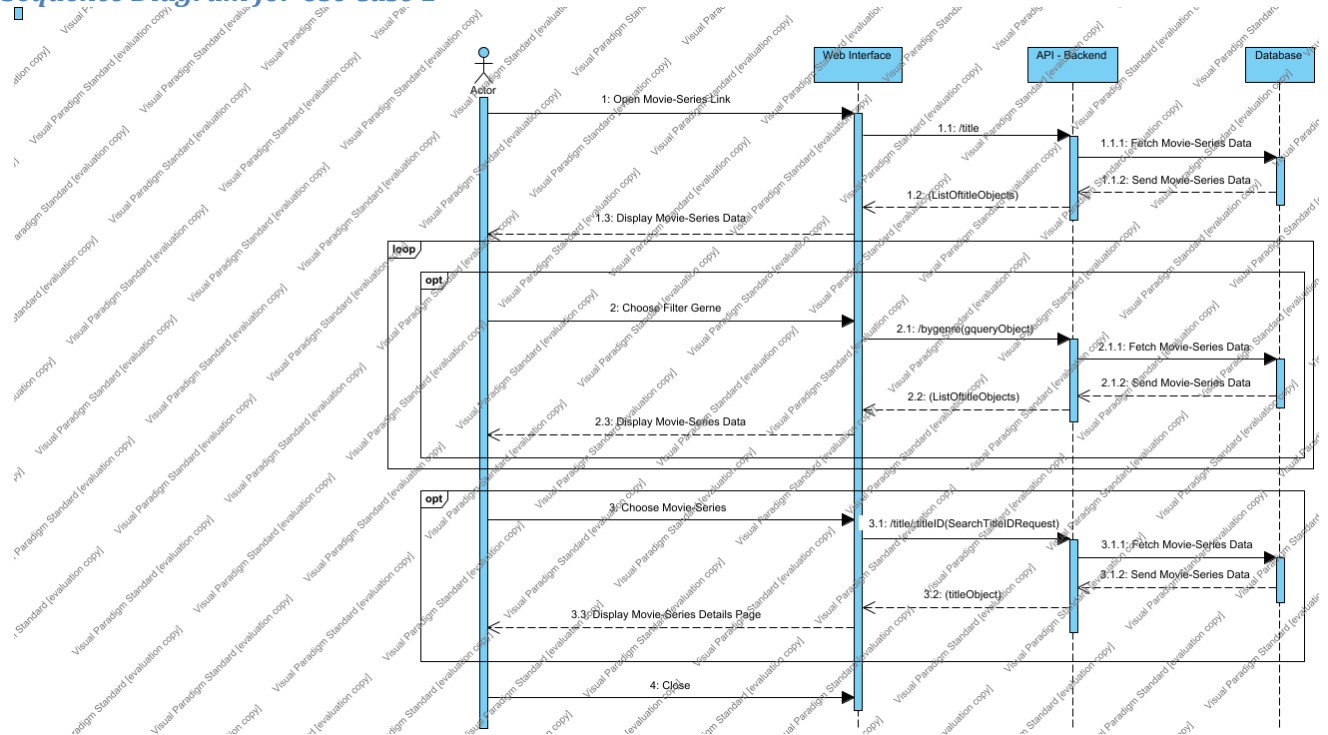
▭ Movie Details State

▭ Movie-Series by Genre Page

In this page, the user can browse the collection of movies and series in NTUAflix. Also, they can filter the results based on the genre.



## Sequence Diagram for Use Case 1



Actor

API - Backend

CombinedFragment







































CombinedFragment2

CombinedFragment3

Database

Web Interface

Messages

From	No.	Name	Type	Action Type	To	Async.
 Actor	1	Open Movie-Series Link	→	Unspecified	 Web Interface	
 Web Interface	1.1	/title	→	Unspecified	 API - Backend	
 API - Backend	1.1.1	Fetch Movie-Series Data	→	Unspecified	 Database	
 Database	1.1.2	Send Movie-Series Data	→	Reply	 API - Backend	
 API - Backend	1.2	unnamed	→	Reply	 Web Interface	
 Web Interface	1.3	Display Movie-Series Data	→	Reply	 Actor	
 Actor	2	Choose Filter Genre	→	Unspecified	 Web Interface	
 Web Interface	2.1	/bygenre	→	Unspecified	 API - Backend	
 API - Backend	2.1.1	Fetch Movie-Series Data	→	Unspecified	 Database	
 Database	2.1.2	Send Movie-Series Data	→	Reply	 API - Backend	
 API - Backend	2.2	unnamed	→	Reply	 Web Interface	
 Web Interface	2.3	Display Movie-Series Data	→	Reply	 Actor	
 Actor	3	Choose Movie-Series	→	Unspecified	 Web Interface	
 Web Interface	3.1	/title/:titleID	→	Unspecified	 API - Backend	
 API - Backend	3.1.1	Fetch Movie-Series Data	→	Unspecified	 Database	
 Database	3.1.2	Send Movie-Series Data	→	Reply	 API - Backend	
 API - Backend	3.2	unnamed	→	Reply	 Web Interface	
 Web Interface	3.3	Display Movie-Series Details Page	→	Reply	 Actor	
 Actor	4	Close	→	Unspecified	 Web Interface	

### 2.1.2. Use Case 2: Search by Rating

This use case corresponds to the second function of the browser which is title catalogue filtering via average rating.

#### 2.1.2.1. Users-Roles

The roles involved in this particular use case are the user and the information archive from which the database extracts information and is updated.

#### 2.1.2.2. Execution Requirements

The only requirements are that the user inserts a valid number from 0 to 10.

#### 2.1.2.3. Execution Environment

- Frontend Server
- API Server/Backend
- MySQL Database

#### 2.1.2.4. Input

The user's choice of minimum rating for the title

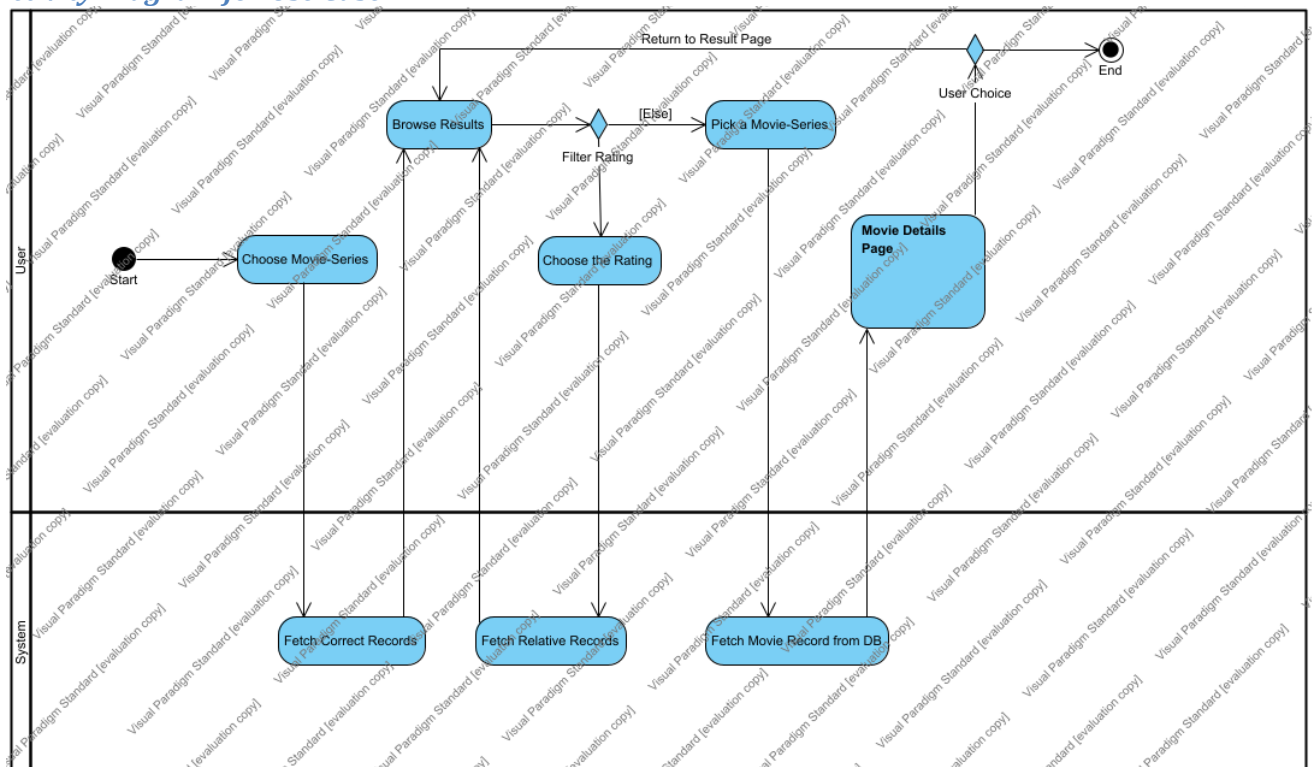
#### 2.1.2.5. Chain of Events

- The user visits the website which lands on the welcome page
- The user clicks the "Search by Rating" button
- The user enters a float in the "Min" field
- The user clicks on "Filter" button
- The results are now filtered and the user can only browse the titles with the specified minimum rating

#### 2.1.2.6. Output

The output is the list of titles that comply with the set minimum rating

#### Activity Diagram for Use Case 2



#### Browse Results

The user browses the displayed movies and series. In this page, the user can decide on what movie or TV series to click on based of the cover, the title and the rating.

#### Choose Movie-Series

The user clicks on the movies-series page in order to browse the movies and series in the database.

#### Choose the Rating

They user is able to specify the range of the rating they want the movies and the series to be in.

#### End

The user leaves "NTUAflix".

#### Fetch Correct Records

The system fetches the data for the movies and the series from the database in order to display them to the user.

#### Fetch Movie Record from DB

The system fetches the data from the database for the movie or the series the user picked in order to display them to the user.

#### Fetch Relative Records

The system fetches the filtered data for the movies and the series from the database in order to display them to the user.

#### Filter Rating

The user is able to filter the results based on the rating.

#### Movie Details Page

Navigate to Movie Details Page.

#### Pick a Movie-Series

The user clicks on the movie or series they want to learn more details for.

#### Start

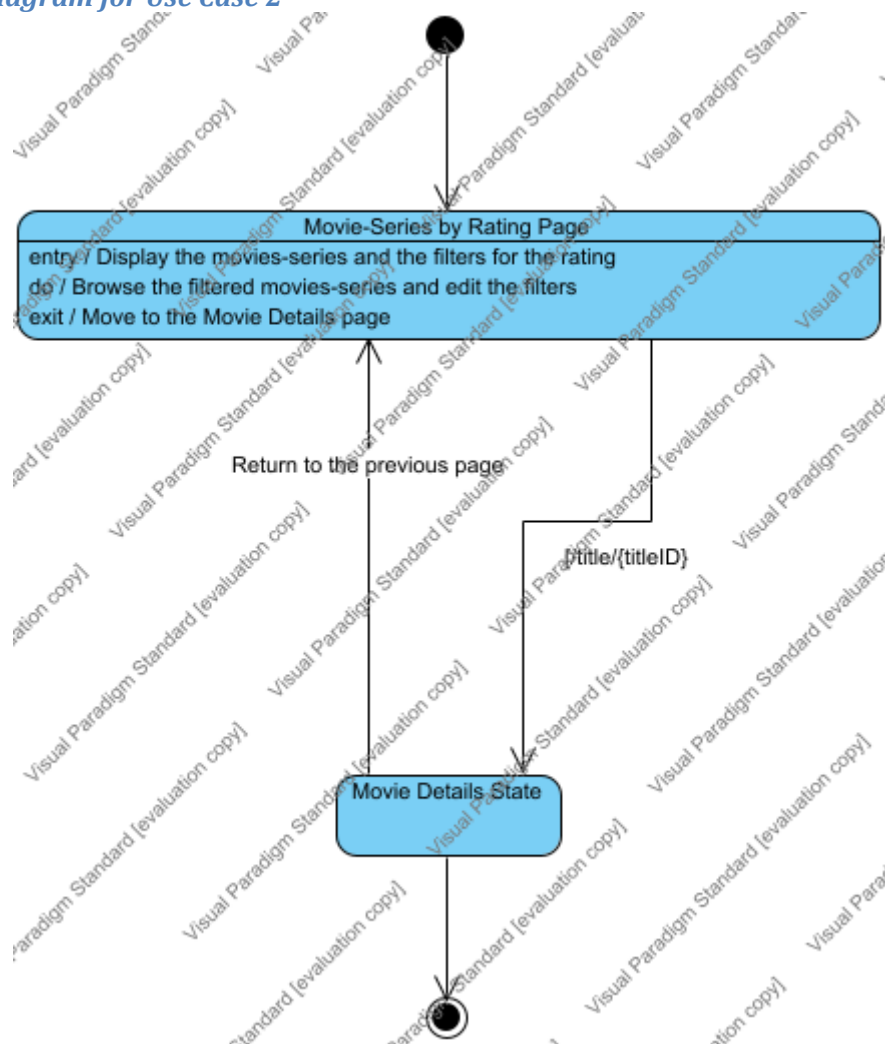
The user opens "NTUAflix"

#### Swimlane5

#### User Choice

The user has the option to go back to the previous page in order to continue browsing the collection.

## State Machine Diagram for Use Case 2



 Movie Details State

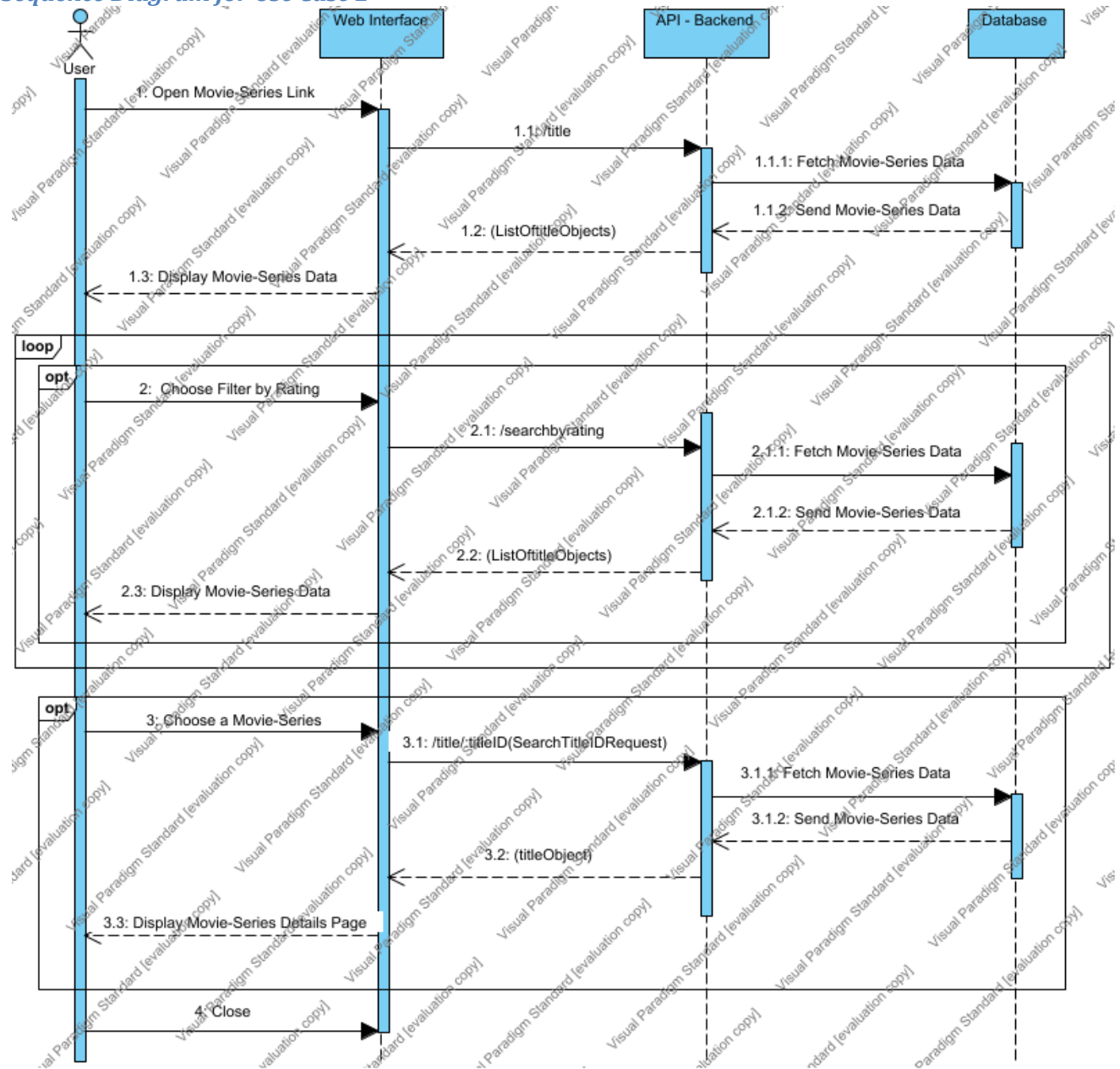
 Movie-Series by Rating Page

In this page, the user can browse the collection of movies and series in NTUAflix. Also, they can filter the results based on the rating.

● unnamed

● unnamed

## Sequence Diagram for Use Case 2



API - Backend

Database

opt







































unnamed

unnamed

User

Web Interface

Messages

From	No.	Name	Type	Action Type	To	Async.
 User	1	Open Movie-Series Link	→	Unspecified	 Web Interface	
 Web Interface	1.1	/title	→	Unspecified	 API - Backend	
 API - Backend	1.1.1	Fetch Movie-Series Data	→	Unspecified	 Database	
 Database	1.1.2	Send Movie-Series Data	→	Reply	 API - Backend	
 API - Backend	1.2	unnamed	→	Reply	 Web Interface	
 Web Interface	1.3	Display Movie-Series Data	→	Reply	 User	
 User	2	Choose Filter by Rating	→	Unspecified	 Web Interface	
 Web Interface	2.1	/searchbyrating	→	Unspecified	 API - Backend	
 API - Backend	2.1.1	Fetch Movie-Series Data	→	Unspecified	 Database	
 Database	2.1.2	Send Movie-Series Data	→	Reply	 API - Backend	
 API - Backend	2.2	unnamed	→	Reply	 Web Interface	
 Web Interface	2.3	Display Movie-Series Data	→	Reply	 User	
 User	3	Choose a Movie-Series	→	Unspecified	 Web Interface	
 Web Interface	3.1	/title/:titleID	→	Unspecified	 API - Backend	
 API - Backend	3.1.1	Fetch Movie-Series Data	→	Unspecified	 Database	
 Database	3.1.2	Send Movie-Series Data	→	Reply	 API - Backend	
 API - Backend	3.2	unnamed	→	Reply	 Web Interface	
 Web Interface	3.3	Display Movie-Series Details Page	→	Reply	 User	
 User	4	Close	→	Unspecified	 Web Interface	

### 2.1.3. Use Case 3: Search people

This use case corresponds to the third and final function of the browser which is people catalogue filtering via name.

#### 2.1.3.1. Users-Roles

The roles involved in this particular use case are the user and the information archive from which the database extracts information and is updated.

#### 2.1.3.2. Execution Requirements

The only requirements are that the user inserts a valid string that contains either a valid name or a valid namepart. In any other case the result will be “No results found”(204)

#### 2.1.3.3. Execution Environment

- Frontend Server
- API Server/Backend
- MySQL Database

#### 2.1.3.4. Input

The user’s written name/namepart string

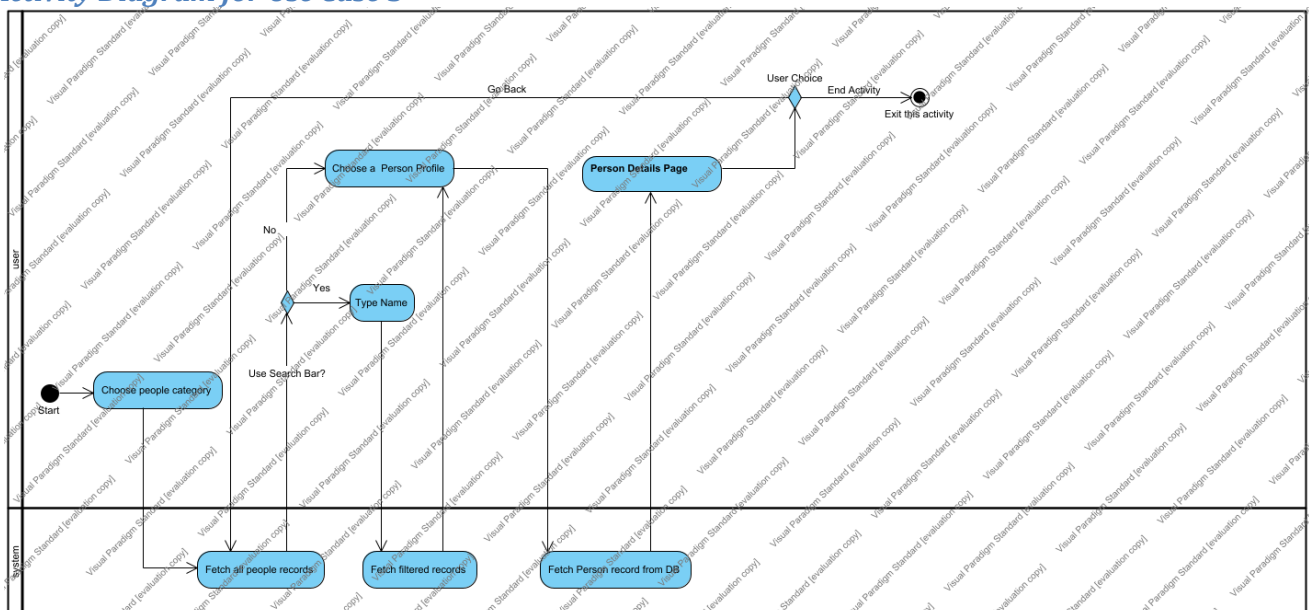
#### 2.1.3.5. Chain of Events

- The user visits the website which lands on the welcome page
- The user clicks the “Search People” button
- The user enters a string in the “Name” field
- The user clicks on “Filter” button
- The results are now filtered and the user can only browse people whose name contains the string submitted

#### 2.1.3.6. Output

The output is the list of people whose name contains the string submitted

#### Activity Diagram for Use Case 3



#### Person Details Page

The user browses the Person Details Page



#### Choose a Person Profile

The user clicks on the person they want to learn more details for.



#### Choose people category

The user clicks on the people page in order to browse the people associated with the movies and series in the database.

#### Exit this activity

This can happen by either exiting the webpage or by simply pressing a movie related to the person.

#### Fetch all people records

The system fetches the data for the people from the database in order to display them to the user.

#### Fetch filtered records

The system fetches the filtered data for the people from the database in order to display them to the user.

#### Fetch Person record from DB

The system fetches the data from the database for the person the user picked in order to display them.

#### Start

The user opens "NTUAflix"

#### Swimlane

#### Type Name

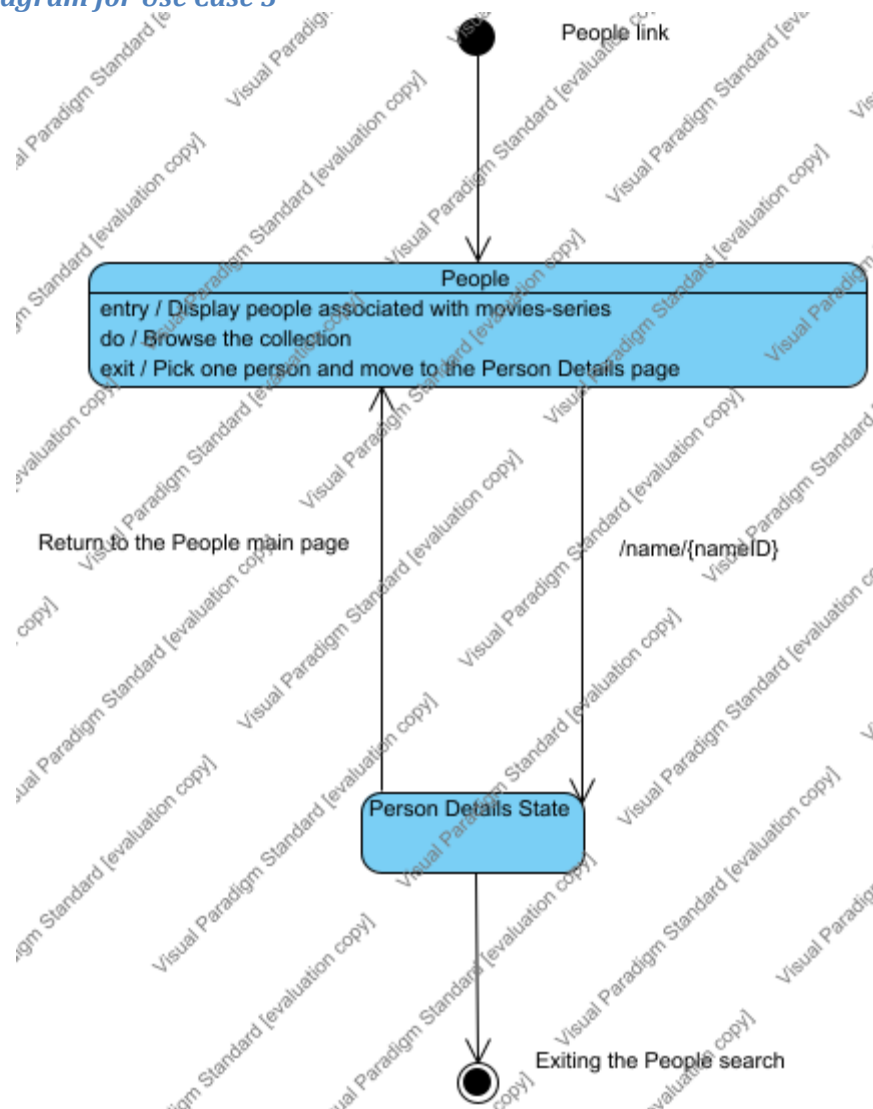
The user types the name they want to filter the data by

#### Use Search Bar?

#### User Choice

The user has the option to go back to the previous page in order to continue browsing the collection.

### State Machine Diagram for Use Case 3



● Exiting the People search

■ People

In this page, the user can browse the collection of people associated with the movies and series in NTUAflix.

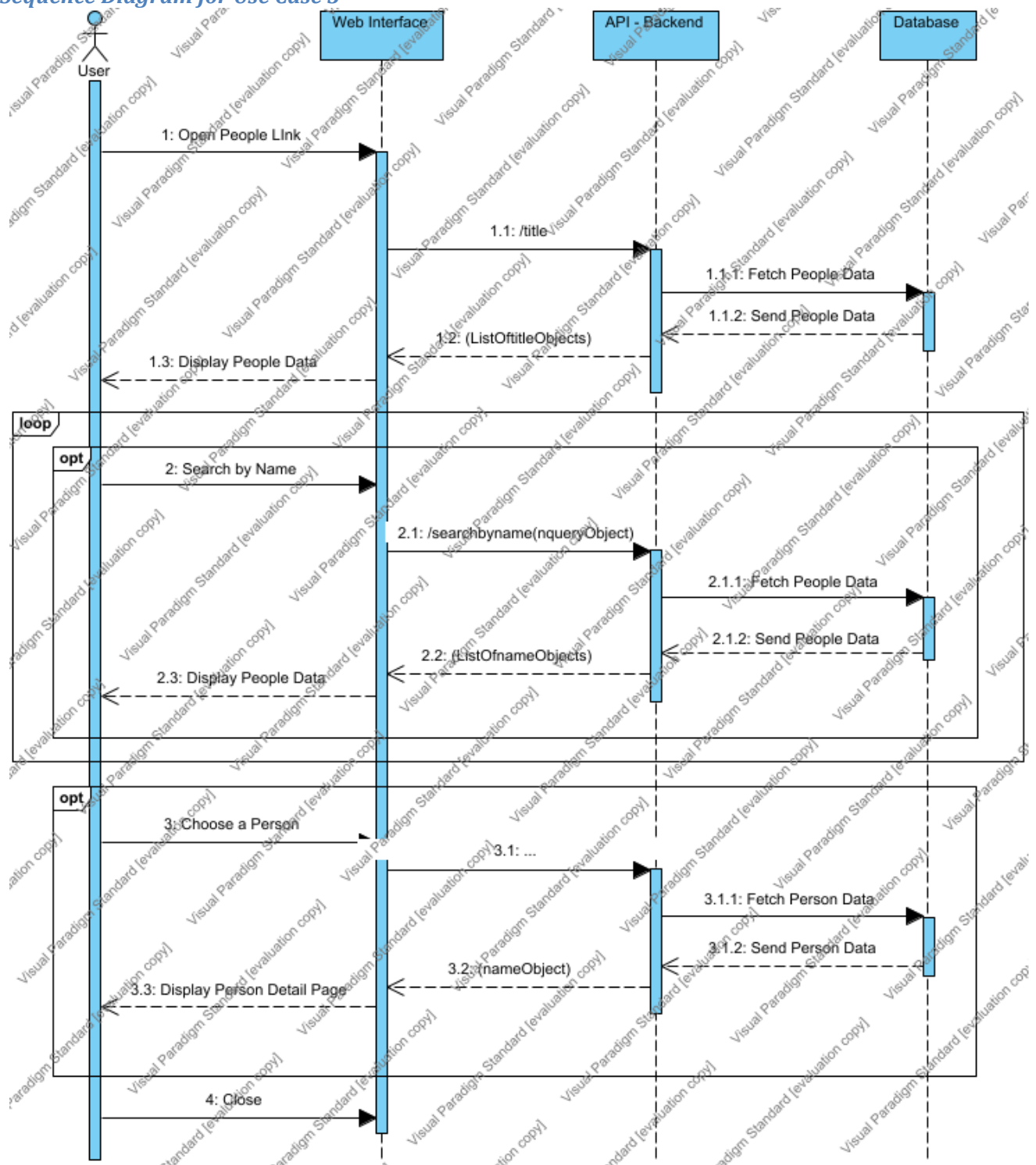
● People link

If the user clicks "People" button from navigation bar he enters this State Diagram

■ Person Details State

In this state all the details of the person that was picked in the previous "People" state are displayed to the user. This State has its own subdiagram.

## Sequence Diagram for Use Case 3



API - Backend

CombinedFragment4

CombinedFragment5







































CombinedFragment6

Database

User

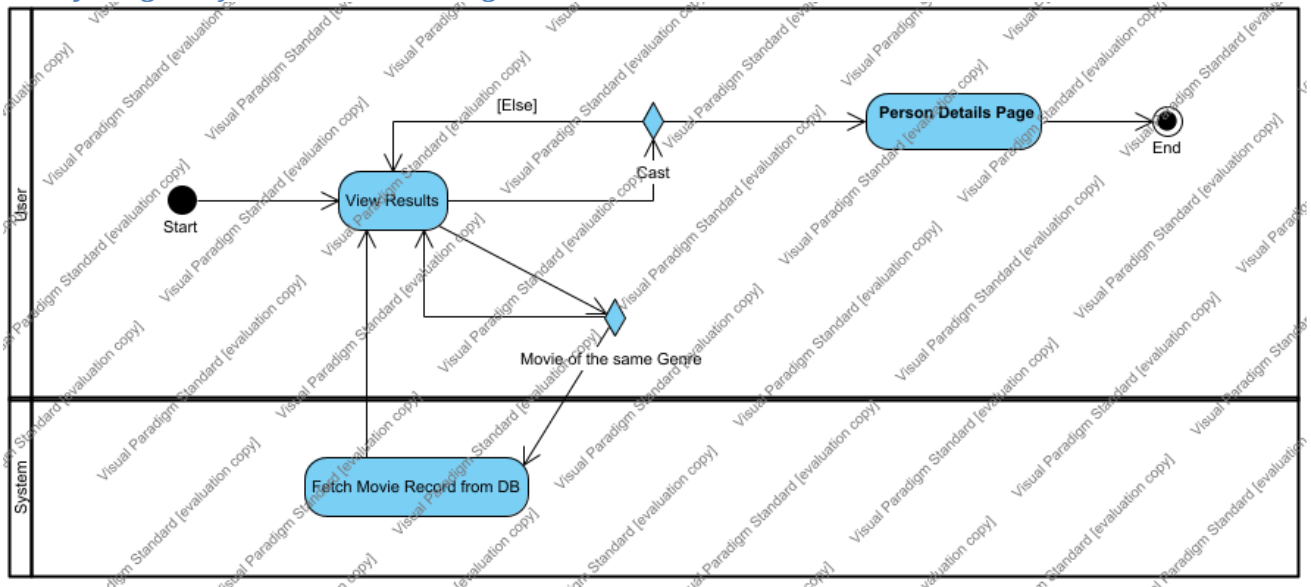
## Web Interface

### Messages

From	No.	Name	Type	Action Type	To	Async.
 User	1	Open People Link	→	Unspecified	 Web Interface	
 Web Interface	1.1	/title	→	Unspecified	 API - Backend	
 API - Backend	1.1.1	Fetch People Data	→	Unspecified	 Database	
 Database	1.1.2	Send People Data	→	Reply	 API - Backend	
 API - Backend	1.2	unnamed	→	Reply	 Web Interface	
 Web Interface	1.3	Display People Data	→	Reply	 User	
 User	2	Search by Name	→	Unspecified	 Web Interface	
 Web Interface	2.1	/searchbyname	→	Unspecified	 API - Backend	
 API - Backend	2.1.1	Fetch People Data	→	Unspecified	 Database	
 Database	2.1.2	Send People Data	→	Reply	 API - Backend	
 API - Backend	2.2	unnamed	→	Reply	 Web Interface	
 Web Interface	2.3	Display People Data	→	Reply	 User	
 User	3	Choose a Person	→	Unspecified	 Web Interface	
 Web Interface	3.1	/name/:nameID	→	Unspecified	 API - Backend	
 API - Backend	3.1.1	Fetch Person Data	→	Unspecified	 Database	
 Database	3.1.2	Send Person Data	→	Reply	 API - Backend	
 API - Backend	3.2	unnamed	→	Reply	 Web Interface	
 Web Interface	3.3	Display Person Detail Page	→	Reply	 User	
 User	4	Close	→	Unspecified	 Web Interface	

## 2.1.4. Supplementary Diagrams which are used in all Use Cases

### Activity Diagram for Movie Details Page



#### Cast

The user has the option to pick one of the people that are associated with the movie-series they picked

#### End

#### Fetch Movie Record from DB

The system fetches the data from the database for the movie or the series the user picked in order to display them to the user.

#### Movie of the same Genre

The user has the option to pick a movie-series from a list of movies-series in the same genre as the one they picked.

#### Person Details Page

The user is moved to the Person Details Page

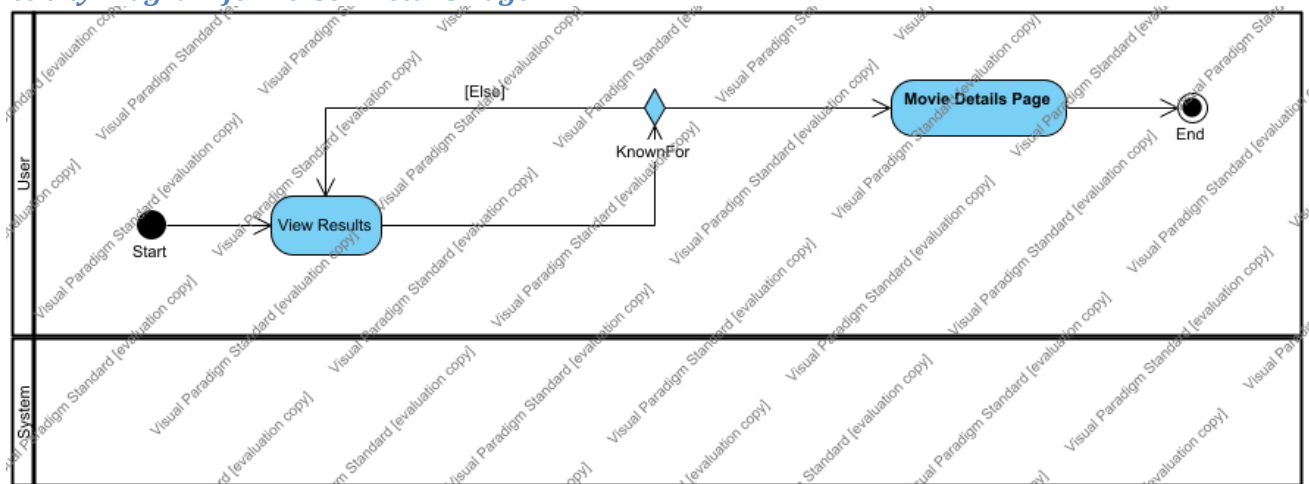
#### Start

#### Swimlane

#### View Results

The user views the details of the movie-series they picked.

## Activity Diagram for Person Details Page



● End

↔ KnownFor

The user has the option to pick a movie the person in this page appears in.

▢ Movie Details Page

The user is moved to the Movie Details Page

● Start

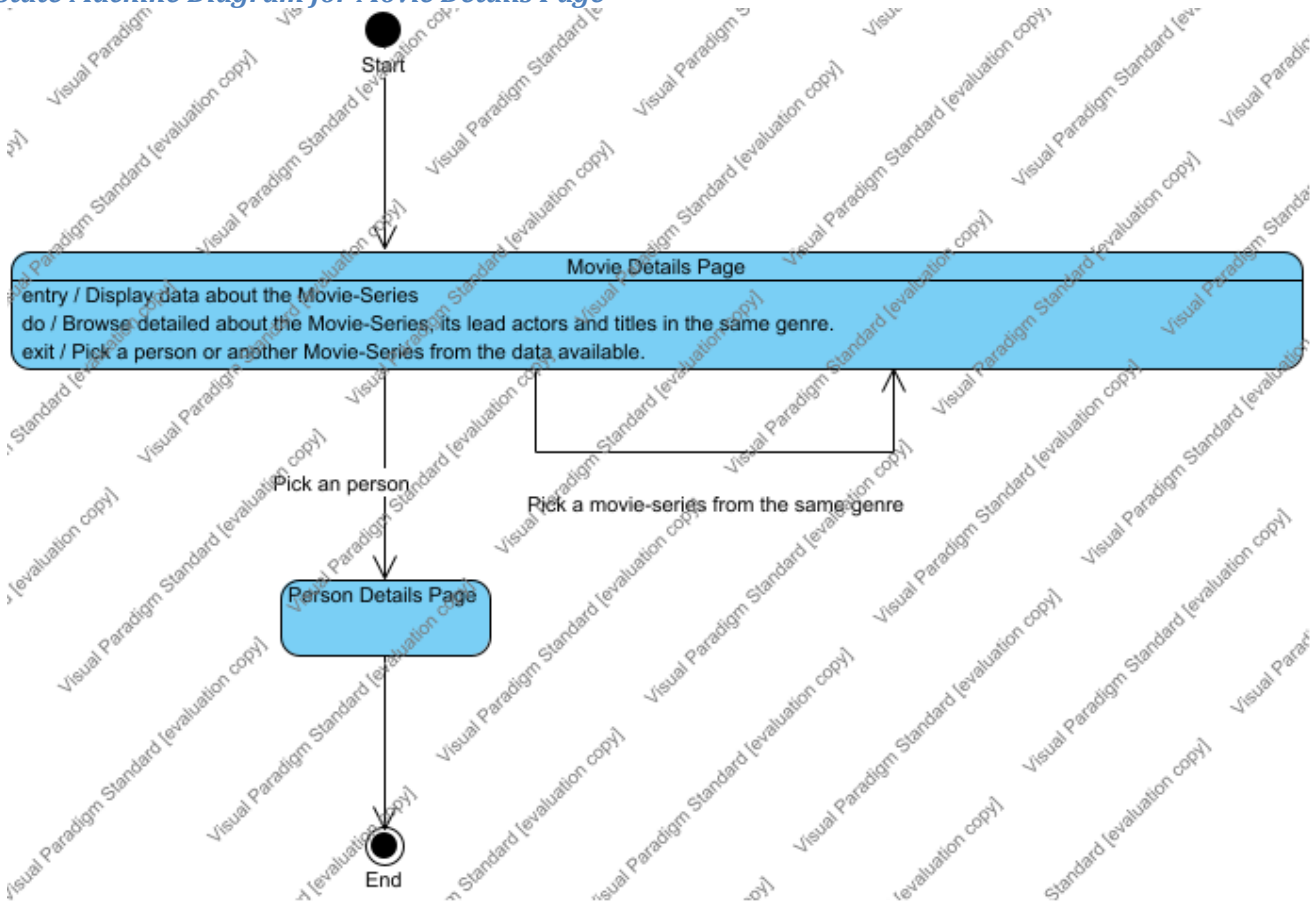
The user enters the Person Details Page

▢ Swimlane6

▢ View Results

The user views the details of the person they picked

## State Machine Diagram for Movie Details Page



● End

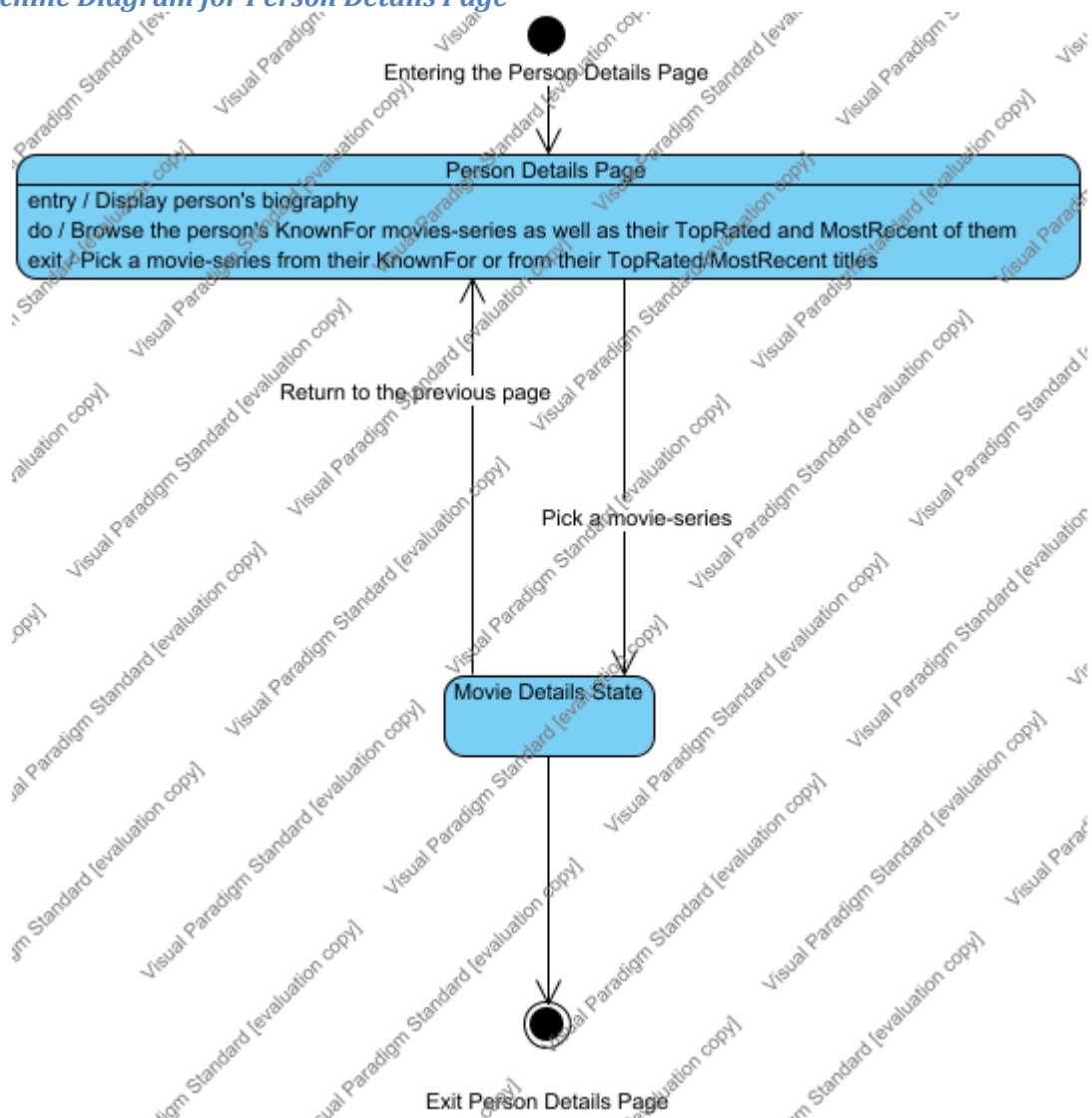
▢ Movie Details Page

In this page, the user can see information about the title they clicked on. Also, they can see the lead actors and also a list of movies-series in the same genre.

Person Details Page

Start

### State Machine Diagram for Person Details Page



Entering the Person Details Page

Entering either by clicking on a certain person on the movie-series details page or by selecting one person on a general people search(use case 3)

Exit Person Details Page

Movie Details State

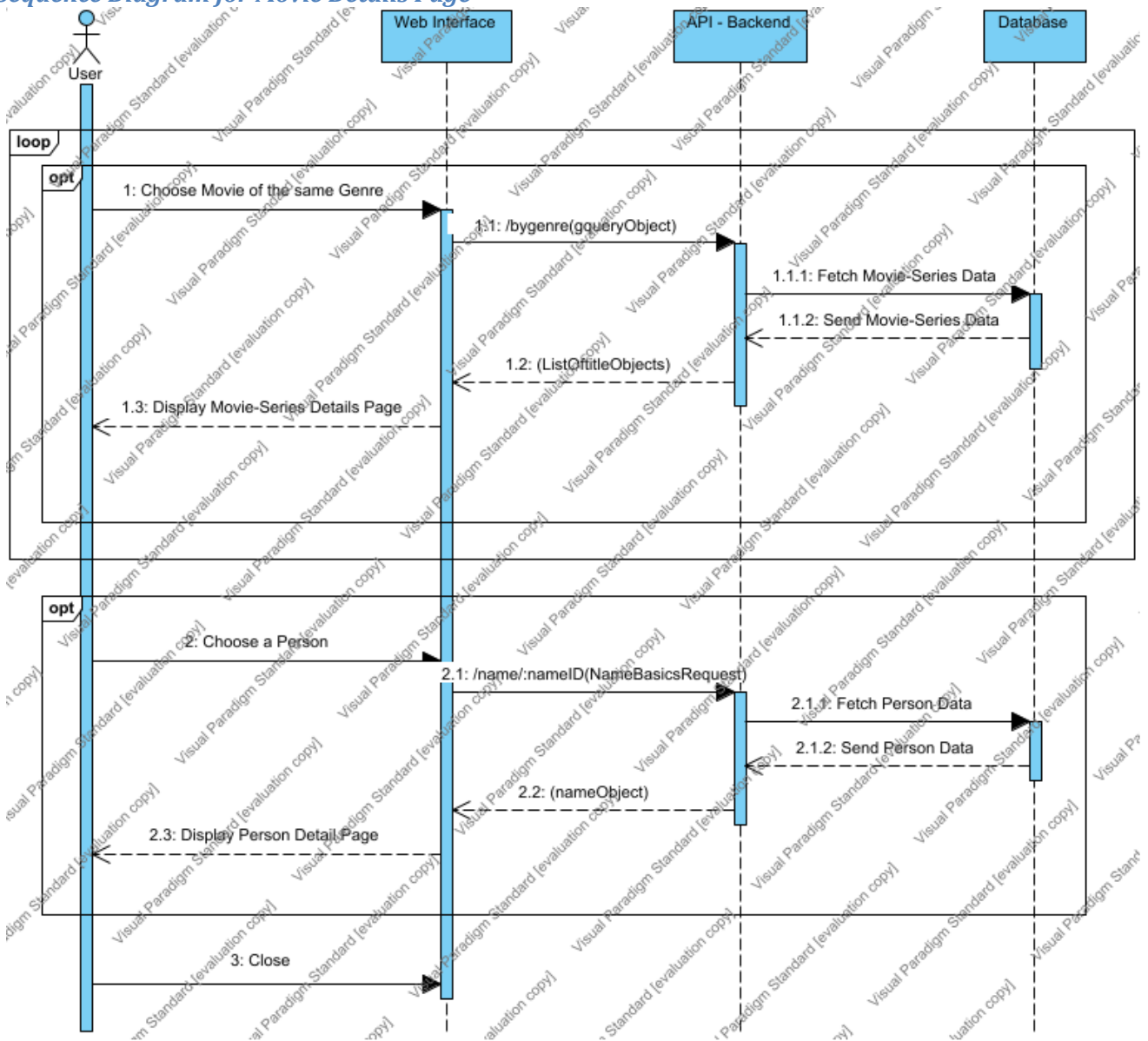
Including: title,

Person Details Page

In this page, the user can see information about the Person they clicked on. They can see the movies-series they are known for as well as their most recent and top rated titles.



## Sequence Diagram for Movie Details Page



API - Backend

CombinedFragment7

CombinedFragment8

CombinedFragment9

Database

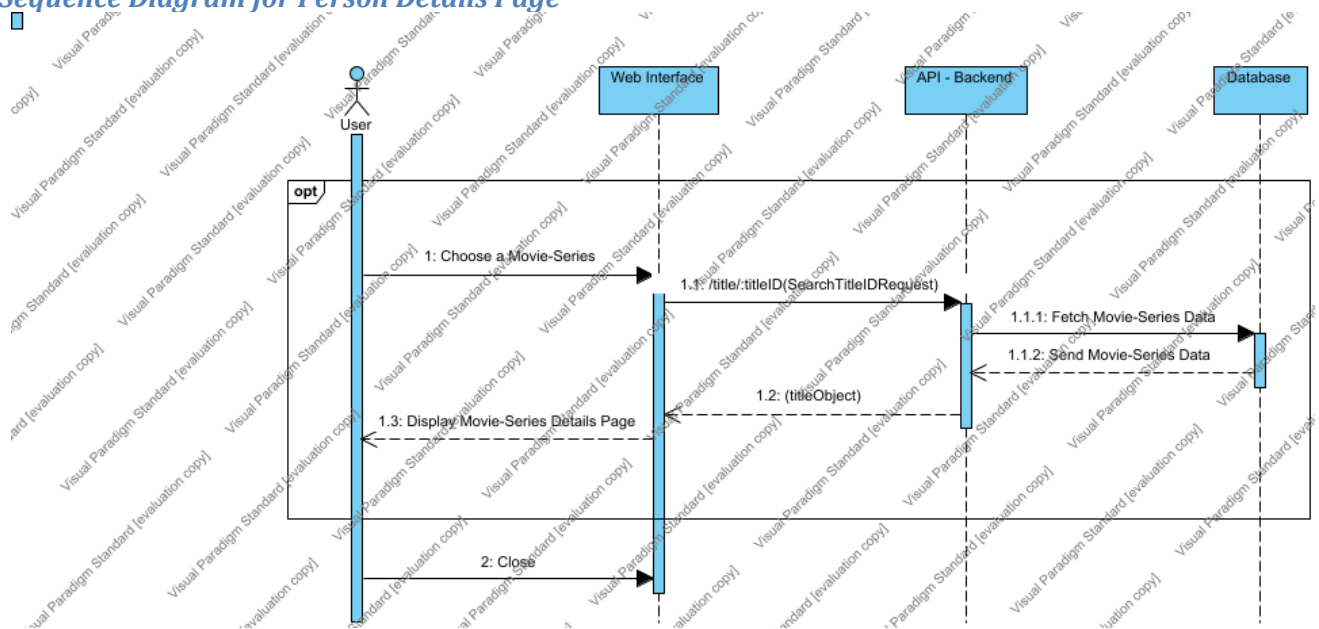
User

Web Interface

Messages

From	No.	Name	Type	Action Type	To	Async.
User	1	Choose Movie of the same Genre	→	Unspecified	Web Interface	
Web Interface	1.1	/bygenre	→	Unspecified	API - Backend	
API - Backend	1.1.1	Fetch Movie-Series Data	→	Unspecified	Database	
Database	1.1.2	Send Movie-Series Data	→	Reply	API - Backend	
API - Backend	1.2	unnamed	→	Reply	Web Interface	
Web Interface	1.3	Display Movie-Series Details Page	→	Reply	User	
User	2	Choose a Person	→	Unspecified	Web Interface	
Web Interface	2.1	/name/:nameID	→	Unspecified	API - Backend	
API - Backend	2.1.1	Fetch Person Data	→	Unspecified	Database	
Database	2.1.2	Send Person Data	→	Reply	API - Backend	
API - Backend	2.2	unnamed	→	Reply	Web Interface	
Web Interface	2.3	Display Person Detail Page	→	Reply	User	
User	3	Close	→	Unspecified	Web Interface	

### Sequence Diagram for Person Details Page



API - Backend

CombinedFragment10

Database

User

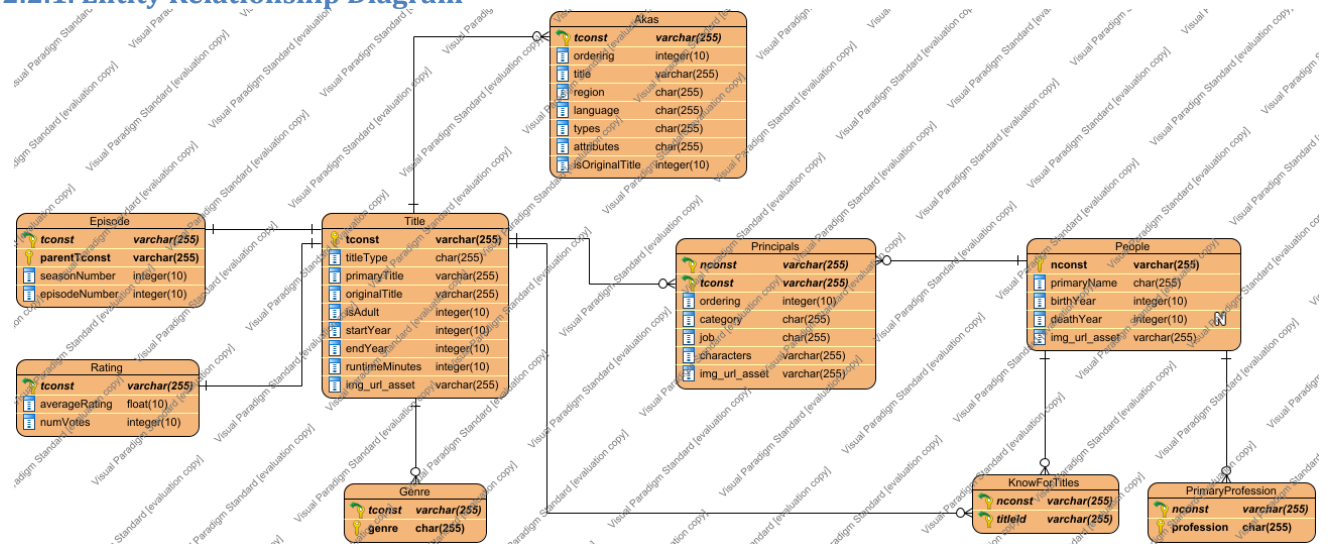
Web Interface

## Messages

From	No.	Name	Type	Action Type	To	Async.
User	1	Choose a Movie-Series	→	Unspecified	Web Interface	
Web Interface	1.1	/title/:titleID	→	Unspecified	API - Backend	
API - Backend	1.1.1	Fetch Movie-Series Data	→	Unspecified	Database	
Database	1.1.2	Send Movie-Series Data	→	Reply	API - Backend	
API - Backend	1.2	unnamed	→	Reply	Web Interface	
Web Interface	1.3	Display Movie-Series Details Page	→	Reply	User	
User	2	Close	→	Unspecified	Web Interface	

## 2.2. Data Organization Requirements

### 2.2.1. Entity Relationship Diagram



Akas

Episode

Genre

 *KnowForTitles*

 *People*

 *PrimaryProfession*

 *Principals*

 *Rating*

 *Title*

## 2.3. Other requirements

### 2.3.1. Software availability requirements

- The system platform will be active throughout the day and for all users, regardless of role or location.
- There will also be compatibility with all available web browsers in their latest version.

### 2.3.2. Security requirements

Database data should be in a secure place where it is not easily accessible by anyone. Sending and receiving data is required to be done using the secure HTTPS protocol.