**Team .titanic {float: none;} Consulting Nodeflix Proposal**  
**Sarah Kovar / James Brown / Nicole Forke**  
**Bellevue University**

Date: 27 June 2020

To: Nodeflix

From: Team.titanic{float:none} Consulting

Subject: Redesign of Nodeflix Interface

Team.titanic{float:none} Consulting, is a development company that has talented web developers and designers. We will take the existing development of Nodeflix and redesign it with an updated streaming system to showcase the different levels of access for Nodeflix users. The database will be created within MongoDB, holding collections for video streaming allowing users easy access to content which they can also save in a queue for later viewing. Users will have the ability to create an account within the application where they will be able to change their password, update notification preferences, and adjust billing details as needed. Team.titanic{float:none} Consulting will incorporate a search bar so users can search by title, producer, genre, etc. The updated version of Nodeflix will show users videos they were currently viewing, recommended videos, let users know when new videos are available and which shows are currently trending. Nodeflix has allowed us free reign on design, development and naming conventions. Our goal for Nodeflix is to develop an application using Node.js, Express and Angular Typescript with enhanced security, scalable for future additions with an easy to navigate design.

All of Nodeflix’s existing features will be transformed to utilize modern technologies. As with any engagement, we require an initial requirement gathering workshop to ensure we have not overlooked any desired features within this initial release. As discussed, there are some ‘nice-to-haves' that we will scope in a later engagement with Nodeflix.

Requirements gathering workshop: 8 hrs

1. Confirm all existing functionality is fully understood.
2. Confirm desired future functionality.
3. Define must have/non-negotiable vs nice to have functions and features.
4. Client receives a copy of documentation created during this workshop.

Once we have confirmed all requirements, we will proceed with providing you a proposal of implementation. Based on the information we currently have:

1. Stand up a development/test/production instance of Nodeflix.
2. Create a new MongoDB, migrating a copy of production data.
3. Migrate existing features into a new web application utilizing Node.js, Express and Angular Typescript.
4. Complete user acceptance testing of data and applications, ensuring its integrity.
5. Resolve all defects found in user acceptance testing.
6. Schedule go-live.
7. Create a backup of all existing data.
8. Redirect all web traffic to new infrastructure/application.
9. Smoke testing.
10. Once an agreed upon time period has been reached to ensure all parties are confident of the new system, retire old system(s).

See Appendices for additional related documentation.

Team .titanic {float: none;} Consulting believes in migrating key components of Nodeflix’s system one component at a time, insuring a seamless and world class customer experience. For our initial release, we propose the following requests be delayed in an effort to ensure core functionality is maintained:

1. Ability to create multiple profiles.
2. Ability to manipulate what each profile has access to (block content, example: child filters)
3. Continued support and enhancements of the system.

Our professional services can either be project based or via a subscription service. If you would like continued support for defect and enhancement services, we are happy to provide a quote for our subscription services. We offer a series of hours/month we are able to offer ongoing support and enhancements of your streaming experience. At this time, it is understood you would like to pay for our services on a case-by-case engagement. See Appendices for additional related documentation.

Some risks that we need to consider in our initial rollout include project development exceeding the estimated timeline, payments compliance issues with storing users credit cards, and security issues with having passwords.

We believe that we can complete development of the new platform in the estimated time, however sometimes unforeseen issues may arise. As such, the development time may exceed our initial estimation.

When it comes to handling credit card data, there are strict rules the Payments Card Industry enforces. In order to reduce the amount of security we would need to implement from our end to store financial details, we can opt to use a third-party credit card process to tokenize the card data. This will ease PCI burden.

Another security issue we will need to take into consideration is how we handle user passwords. In order to reduce the chances of creating a non-secure process, we will opt to use a third party for hashing our passwords.

In conclusion, we have added wireframes and ORD’s to showcase the redesign of Nodeflix. Once these wireframes and ORD’s are approved we will start the rollout with updating the ability to stream content, then continue on with creating the account management for users and updating the ability to navigate content easily. Once these are completed, we will test for security issues and proceed with any changes deemed necessary to secure and scale the database for additional video streaming.

**Appendix A – Nodeflix Website Goals**

* Allow users to register and manage accounts (change password, update billing details, etc.)
* Allow users a way of streaming content
* Offer tiered services for users (basic, plus, and premium)
* Grow existing Brand.
* Expand and grow userbase
* 24/7 access to TV shows, movies, documentaries, and prerecorded sporting events through internet-connected devices
* Ability for customers to manage their level of subscription on the fly
* Ability for new customers to sign up without the need of human interaction or approvals, as long as required information is provided and validated via internet (ie, email validation, payment successful).
* Ability to offer suggestions for content based on current viewing history
* Search functionality of all the content you are able to access
* Closed captioning in multiple languages within video content
* Ability to apply language translation/voice over in multiple languages for video content
* Chat support for account and technical questions and issues

**Appendix B – Core Users of Nodeflix**

* People of all ages and backgrounds.
* Existing and new customers of all ages and backgrounds.
* Anyone with an interest in streaming video content either by choice or necessity.
* People who prefer to stream movies, TV shows, sports, etc. Instead of buying cable due to the cost of cable.

**Appendix C – Detailed User Personas**

**User 1**:

  
<https://www.fool.com/investing/2016/06/20/3-reasons-youll-spend-more-at-40-than-at-30.aspx>

* **Name:** Fred
* **Age:** 40
* **Current behavior:**Fred enjoys streaming services with his family in the evenings.
* **Pain points:**Right now, Fred uses several different streaming services. His biggest pain point is that he doesn’t have enough flexibility over adjusting viewing restriction settings.
* **Unique Characteristics:**Color blind. Didn’t grow up with technology, so he sometimes struggles with technical concepts.
* **Goals:**His kids are 6 and 8, so it is important for him to be able to adjust viewing settings. He needs a streaming service that has navigation that takes his disability into consideration.

**User 2**:

   
<https://www.today.com/style/beauty-secrets-80-year-old-goddess-style-1B5521665>

* **Name:** Shyan
* **Age:** 80
* **Current behavior:** Shyan is retired, so she streams content the majority of her day at home. She enjoys documentaries, movies, and pre-recorded events.
* **Pain points:**Text is often not placed in areas that make the most sense, so navigation is often hard for Shyan to figure out. Sometimes closed caption is also not offered for all shows or isn’t working, so she isn’t able to stream certain programs.
* **Unique Characteristics:**Shyan is deaf, so she uses closed caption when streaming content.  Shyan also did not grow up with technology, so the user interface must be easy to use.
* **Goals:**Shyan’s main goals are to be able to use a streaming service that is reliable, user friendly, offers closed caption.

**User 3:**

 <https://depositphotos.com/stock-photos/family.html>

* **Name:** Martin Family
* **Age:** Bob 48, Melissa 44, Brody 6, and Amy 10
* **Current behavior:** The Martin family has 2 children Brody age 6 and Amy age 10.  The children enjoy streaming their favorite shows after school and on the weekends during their free time. Both are very comfortable with searching for the shows they want to watch.
* **Pain points:**The parents Bob and Melissa would like better parental controls, so they are reassured their children are safe when streaming.
* **Unique Characteristics:**Melissa, currently struggles with the user interface. The children are young and need parental controls in order to avoid inappropriate shows.
* **Goals:**Melissa would like it to be more user friendly and offer better parental controls, so the children are only allowed to view certain movies, shows, etc.

**User 4:**   
<https://www.lookfabulousforever.com/us/blog/should-older-women-wear-less-makeup>

* **Name:** Roberta
* **Age:** 75
* **Current behavior:**Lives on a fixed income.  The cost of cable and satellite have risen beyond her means.  Watches over the air (OTA) television; however, has a desire to watch more than what is available OTA.
* **Pain points:**Has a very loose understanding of the concept of streaming services.  She needs clear explanation of all aspects of the packages offered and how to use the service on different platforms (smart tv vs browser vs smartphone vs tablet).
* **Unique Characteristics:**Not particularly technically savvy.  Has a computer, tablet and smartphone; however, no experience with streaming services or the concept of active streaming connections, DVR, or even smart tvs.  Has internet; however also lacks the understanding of how a smart tv may connect; or how she could purchase a streaming device to connect to her existing tv.
* **Goals:**Have more options than OTA without the cost of satellite or cable tv.

**User 5:**

  
<https://depositphotos.com/71022229/stock-photo-young-college-couple-with-laptop.html>

* **Name:** Meghan and Jordan
* **Age:** 22/25
* **Current behavior:** College students that travel frequently.
* **Pain points:**Their existing streaming service is glitchy with regard to recognizing when an active stream has ended on a device, despite closing out of the application/browser.  They both have a laptop, tablet, smartphone, and would like the ability to carry a streaming device that can plugin to a TV in a hotel room.  They do not have one dedicated device they watch movies/tv on.  What they are using to stream depends on which mobile device they are using at the moment, whether in line at the air port (smartphone), waiting on a plane (laptop), or at the hotel.  They would like a seamless, glitch-free experience.
* **Unique Characteristics:**Traveling they don’t always have internet connectivity.  They have a need to be able to DVR/download shows they want to watch.
* **Goals:**Ability to easily switch between different modes of utilizing the service while it recognizes when one active stream ends and another begins.  Supports two simultaneous streams, minimally.  Ability to download content.

**Appendix D – Top Five User Requirements**

1. As a user, I need to be able to stream content, so that I can keep myself entertained.
2. As a user, I need to be able to manage my account, so that I can change my password, update notification preferences, or adjust billing details.
3. As a user, I need to be able to navigate the content easily, so that I don’t spend a lot of time searching.
4. *As a user, I need to be able to monitor my children, so I can block certain content and know they are only watching approved shows, movies, etc.* \*\****Out of Scope for this Engagement\*\****
5. As a user, I would like to be able to sort and search content by various filters (ex: genre, title, actor).

**Appendix E – Top Five Interface Designer/Developer Requirements**

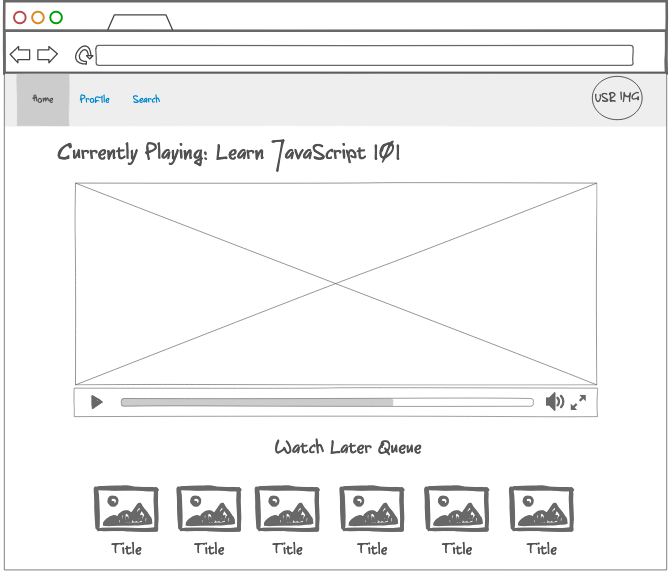
1. As a user interface developer, I need a NoSQL database, so that I can easily grow and change documents in a collection.
2. As a user interface designer, I need a generic category list of the content that will be getting displayed, as this will help me design an interface that is easy to use.
3. As a user interface developer, I need to know the modes in which the application will be accessed (mobile browser, desktop browser, mobile application, streaming device, etc.) in order to develop different applications and make the web interface responsive.
4. As a user interface developer, I need to be able to troubleshoot errors effectively when they arise and be able to fix them efficiently.
5. As a user interface developer, I need to provide security for existing user accounts and new user accounts.

**Appendix F – Proposed Wireframes, Business Rules, ORD, Document Model and JSON Structure**

**User Requirement 1 Story:**

As a user, I need to be able to stream content, so that I can keep myself entertained.

**User 1 Requirement Wireframe:**



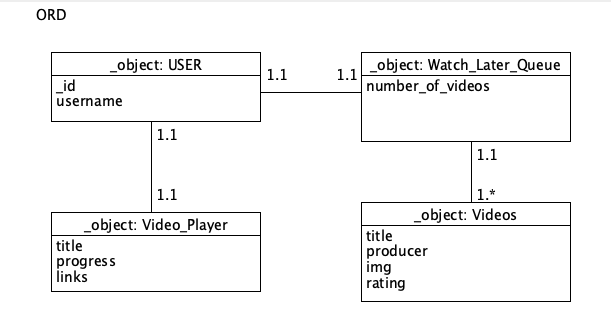
**User 1 Requirement Business Rules:**

A USER can stream one VIDEO at a time

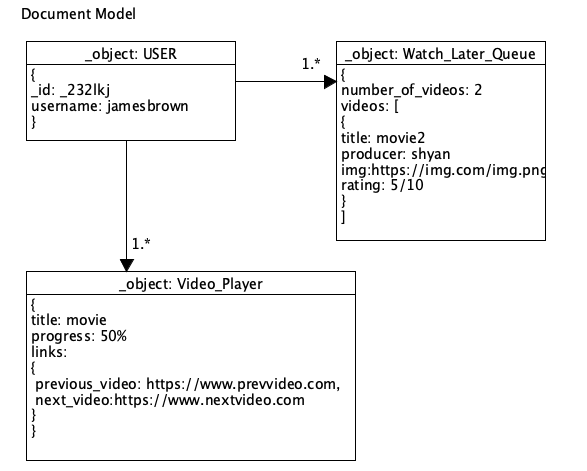
A USER can save many Videos in their Watch Later Queue

A USER can only have one Watch Queue

**User 1 Requirement ORD:**



**User 1 Document Model:**

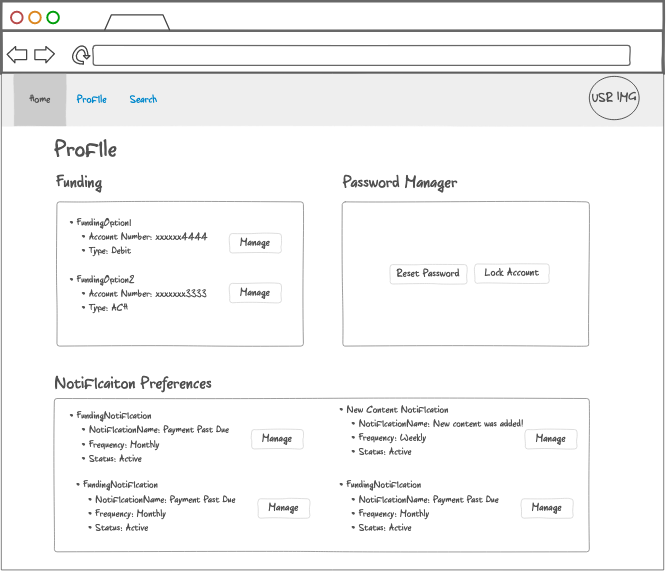


**User 1 Requirement JSON:**



**User 2 Requirement Story:** As a user, I need to be able to manage my account, so that I can change my password, update notification preferences, or adjust billing details.

**User 2 Requirement Wireframe:**



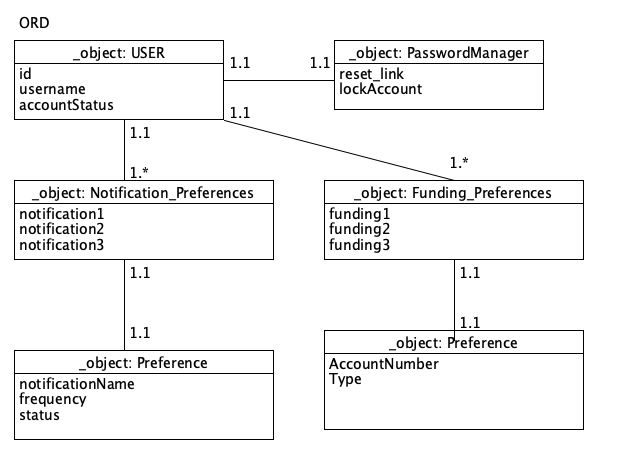
**User 2 Requirement Business Rules:**

A USER can have one password

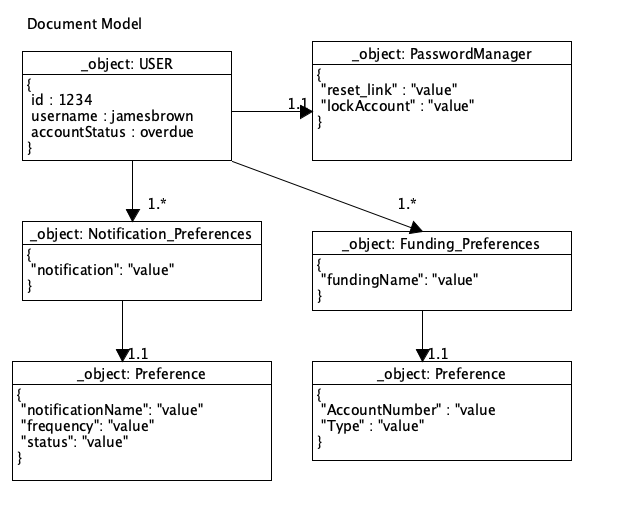
A User can have many notification preferences

A User can have many funding sources

**User 2 Requirement ORD:**



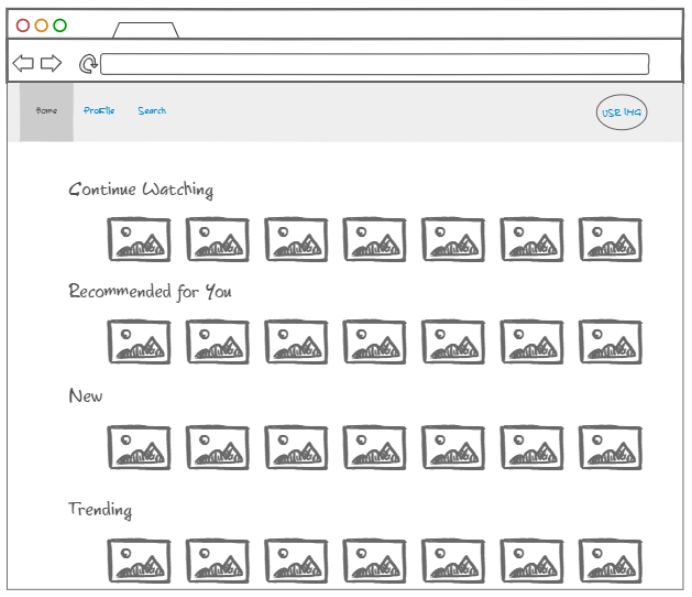
**User 2 Requirement Document Model:**



**User 2 Requirement JSON:**



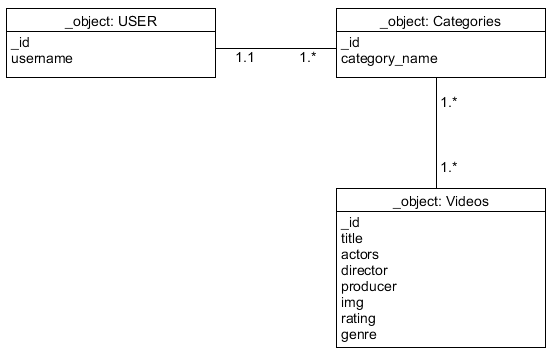
**User 3 Requirement Story:** As a user, I need to be able to navigate the content easily, so that I don’t spend a lot of time searching.

**User 3 Requirement Wireframe:**   


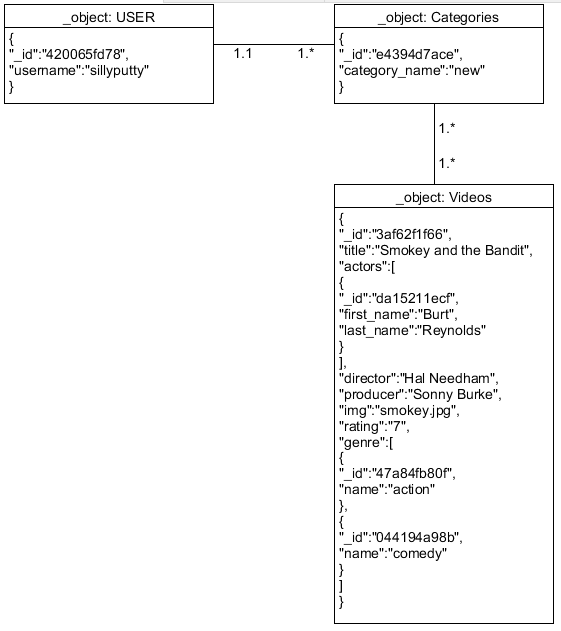
**User 3 Requirement Business Rules:**   
   
A USER can see many CATEGORIES

A CATEGORY can contain many VIDEOS

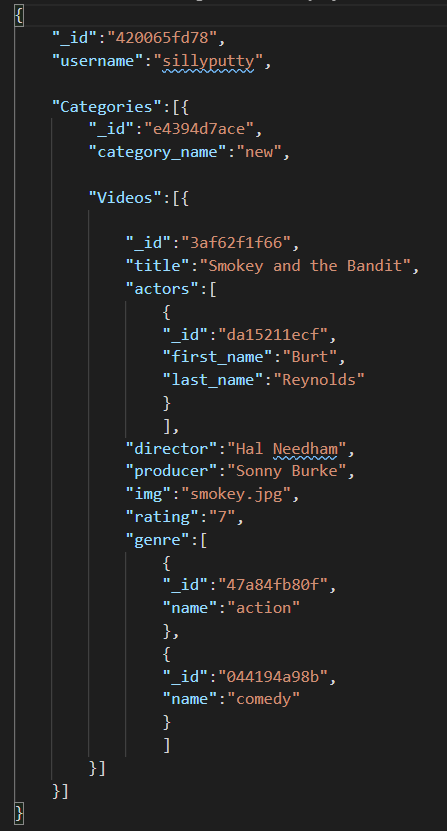
**User 3 Requirement ORD:**



**User 3 Requirement Document Model:**

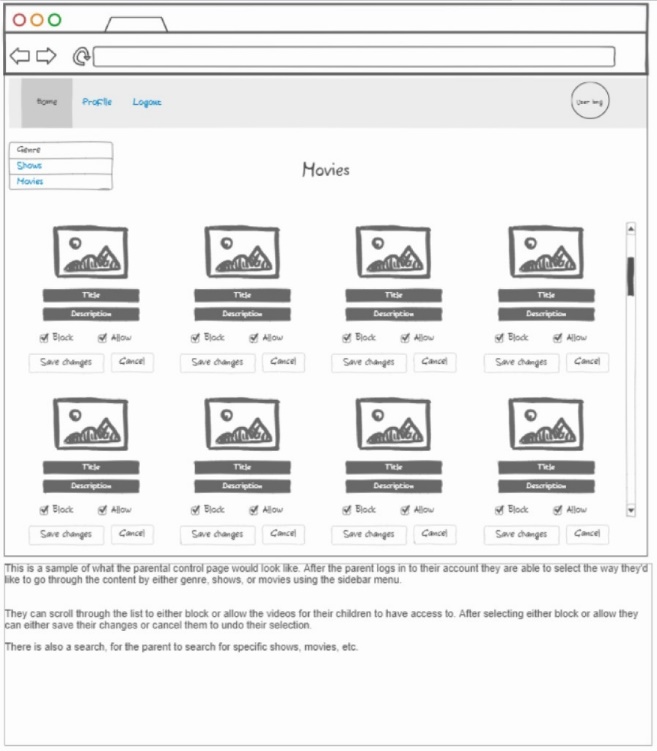


**User 3 Requirement JSON:**



**User 4 Requirement Story:** As a user, I need to be able to monitor my children, so I can block certain content and know they are only watching approved shows, movies, etc. \*\****Out of Scope for this Engagement\*\****

**User 4 Requirement Wireframe:**



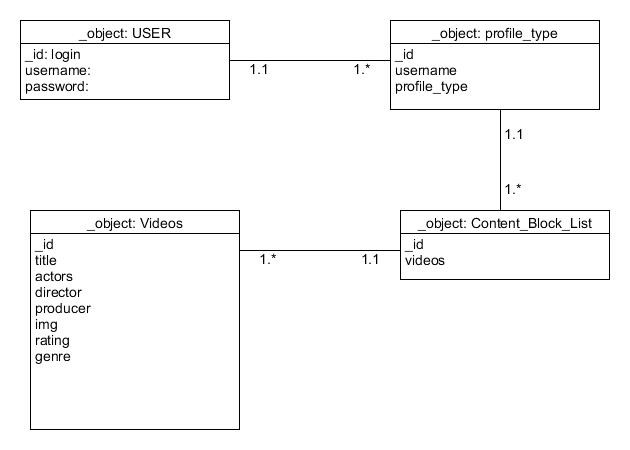
**User 4 Requirement Business Rules:**

A USER can login to parental controls.

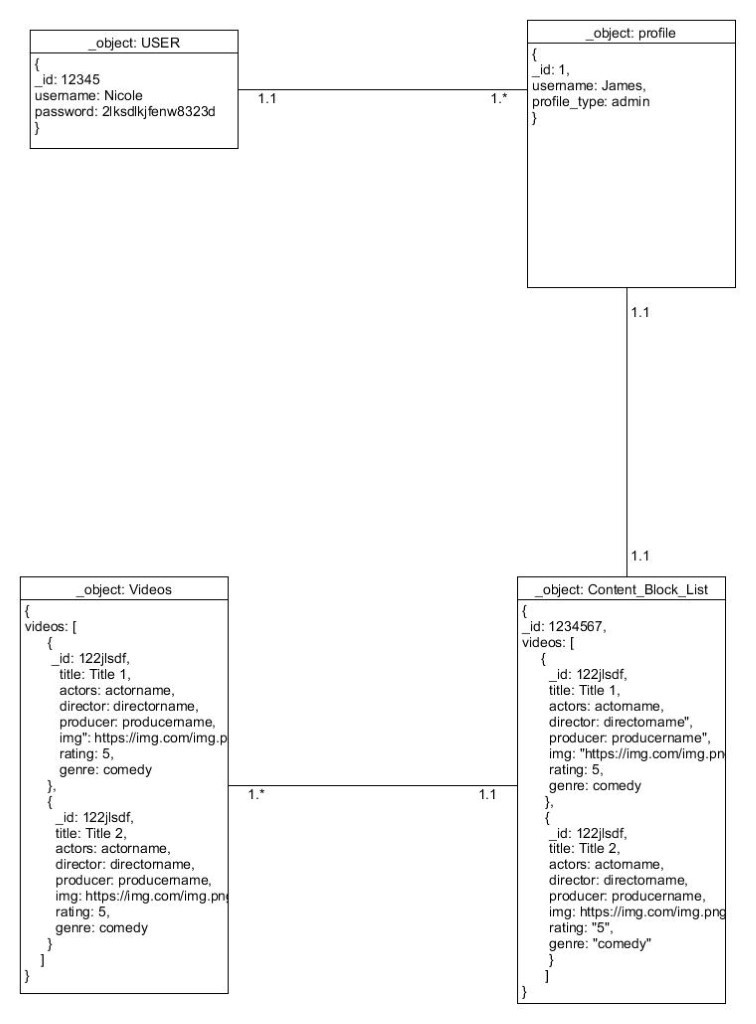
A USER can block content from children.

A USER can allow content for children.

**User 4 ORD:**



**User 4 Requirement Document Model:**



**User 4 JSON:**   


**User 5 Requirement Story:** As a user, I would like to be able to sort and search content by various filters (ex: genre, title, actor).

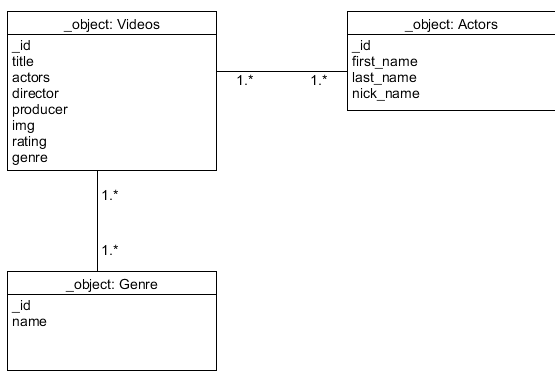
**User 5 Requirement Wireframe:**   

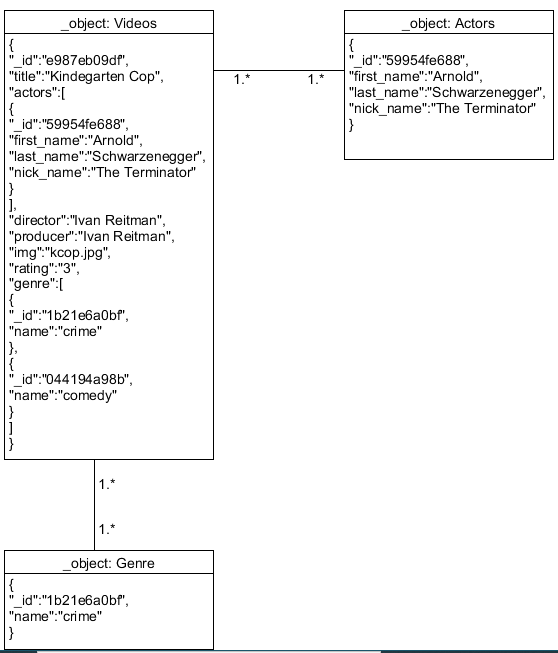

**User 5 Requirement Business Rules:**

A VIDEO can have multiple TAGS (title, actor, genre)

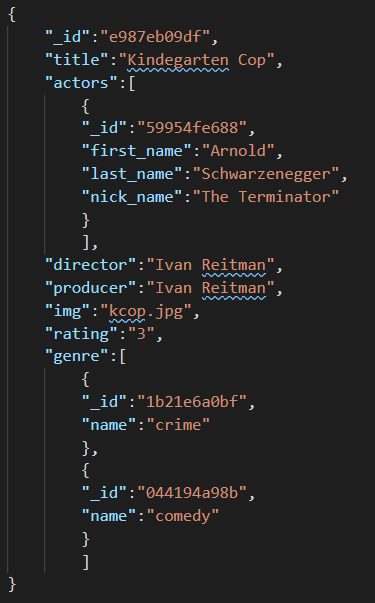
A TAG can be present in multiple VIDEOS

A FILTER can contain multiple TAGS

**User 5 Requirement ORD:**   


**User 5 Document Model:**   


**User 5 Requirement JSON:**



**Appendix G – Proposed MongoDB Commands to Create Database**

Use nodeFlix

db.createCollection("**user**");

db.user.insert({"username": "jamesbrown",”accountStatus”:”overdue”});

db.createCollection("**watch\_later\_queue**");

db. watch\_later\_queue.insert({"number\_of\_videos": "2"});

db.createCollection("**video\_player**");

db. video\_player.insert({"title": "movie",“progress”: “50%”, “links”:{”previous\_video”: https://www.prevvideo.com”,”next\_video":”https://nextvideo.com”}});

db.createCollection("**passwordManager**");

db.passwordManager.insert({"reset\_link":"https://www.resetpassword.com",”lockAccount”:”https://www.lockmyaccount.com”});

db.createCollection("**funding\_preferences**");

db.funding\_preferences.insert({"fundingOption1”:{“AccountNumber”:”xxxxxxxx444”,”type”:”Debit”}});

db.createCollection("**notification\_preferences**");

db.notification\_preferences.insert({"fundingNotification”:{“notificationName”:”Payment Past Due”,”frequency”:”Monthly”,”status”:”Active”}});

db.createCollection("**categories**");

db.categories.insert({"category\_name”:”new”});

db.createCollection("**videos**");

db.videos.insert({"title”:”Smokey and the Bandit”,”actors”:{“first\_name”:”Burt”,”last\_name”:”Reynolds”},”director”:”Hal Needham”,”producer”:”Sonny Burke”,”img”:”smokey.jpg”,”rating”:”7”,”genre”:”action”});

db.createCollection("**genre**");

db.genre.insert({"name”:”crime”});

db.createCollection("**actors**");

db.actors.insert({"first\_name”:”Dwayne”,”last\_name”:”Johnson”,”nick\_name”:”The Rock”});

**Appendix H – Proposed Top Five Queries**

1. The below query retrieves a list of all users whose accounts are in an overdue status. This would be helpful when creating a campaign to target overdue users.

db.users.find({"accountStatus" : "overdue"});

2. This query finds a specific user by filtering for the id.

db.users.findOne({"\_id" : "\_232lkj"});

3. The find if there are any videos in watch later queue using $where.

db.watch\_later\_queue.find({"$where" : function(){

for (var current in this){

if ("number\_of\_videos" > 0){

return true;

}

}

return false;

}})

4. This query updates a funding source for a user.

db.videos.update({"\_id" : "\_232lks"")},

... {"$set" : {"funding\_Option1" : "4111-1111-1111-1111"}});

5. The below query finds all categories with specific returned keys.

db.categories.find({}, {"\_id" : 1, "category\_name" : 1});

**Appendix I – Terms Glossary**

o Name: \_ID

Description: document id assigned by mongodb.

Type: string

o Name: username

Description: user name of the user account

Type: string

o Name: video\_player

Description: object that houses video player functions

Type: object

o Name: title

Description: title of movie

Type: string

o Name: progress

Description: Holds current progress of video.

Type: string

o Name: links

Description: links for navigating between videos

Type: string

o Name: watch\_later\_queue

Description: object that houses number of videos the user has saved to watch later

Type: object

o Name: number\_of\_videos

Description: holds the number of videos that are in queue

Type: integer

o Name: videos

Description: an array that houses videos available for streaming

Type: Array

o Name: title

Description: Title of movie

Type: String

o Name: producer

Description: name of the movie producer

Type: string

o Name: img

Description: the image that is displayed to the user, which represents the video

Type: String

o Name: rating

Description: rating of movie

Type: String

o Name: actors

Description: array of actors in the video

Type: Array

o Name: director

Description: director of movie

Type: String

o Name: genre

Description: an array of genres

Type: Array

o Name: accountStatus

Description: status of account. Possible values are overdue or current.

Type: String

o Name: notification\_preferences

Description: An array of preferences related to notifications.

Type: Array

o Name: FundingNotification

Description: object that houses preferences

Type: object

o Name: notificationName

Description: name of the notification

Type: String

o Name: frequency

Description: How often the notification is delivered

Type: String

o Name: NewContentNotification

Description: object that houses notification preferences for new notifications

Type: Object

o Name: passwordManager

Description: houses links for resetting and locking account.

Type: Object

o Name: functionOption1

Description: object that houses funding details

Type: Object

o Name: AccountNumber

Description: field for account number

Type: String

o Name: Type

Description: type of funding instrument

Type: String

o Name: Categories

Description: array that houses various video categories

Type: Array

o Name: category\_name

Description: name of category

Type: String

o Name: Videos

Description: array of videos

Type: Array

o Name: password

Description: password for the user

Type: String

o Name: profile\_type

Description: describes the profile type

Type: String

o Name: first\_name

Description: first name of the user

Type: String

o Name: last\_name

Description: last name of the user

Type: String

o Name: nick\_name

Description: nickname given by the user

Type: String