1 Basic Info

Project Title Curly Squeegee goes to Hollywood

Names Brian Kimmig, Jimmy Moore

e-mail brian.kimmig@utah.edu, jimmy@cs.utah.edu

uID u0560080, u1012009

Project Github curly-squeegee-6630

2 Background & Motivation

Discuss your motivations and reasons for choosing this project, especially any background or research interests that may have influenced your decision.

Both of us watch a fair amount of movies and find ourselves looking at sites like IMDB and wanting to see the scope of an actors career through more than text, and more than one parameter at a time.

We are choosing to look at actors careers through visualization because movies and their casts are relate-able to large groups of people with minimal explanation. This semi-dynamic dataset has a large appeal because users have the freedom to learn about the actor of their choice and movies through visualization.

For our project, we want to visually represent the career arc of actors as measured by number of movies, ratings, earnings, spouses as gathered from IMDB and other sites (mentioned below).

We want to see the textual data with high dimensionality in an easy to digest manner.

3 Project Objectives

Provide the primary questions you are trying to answer with your visualization.

- *What would you like to learn and accomplish?* We want to look for trends in actors careers over time with respect to film rating, number of movies a year, estimated salary. Also who they have worked with and how that has affected their careers.

- *List benefits of your project* A single place that aggregates data about actors careers from multiple sites and allows a user to visualize it in multiple ways. We don't know if anything will pop out but it invites people to look and ask questions about our favorite celebrities.

4 Data

1. Where did you get the data? We will be pulling data from a few APIs on the Internet. We have set up web framework using node.js and Meteor that uses a RESTful archetecture to gather data from APIs on the web via GET requests. This data is stored in a MongoDB database.

2. how did you get the data? Our data will be dynamic, so based on the actor the person wants to learn about we will query an API to gather the info.

3. links to data-sources:

   - My API Films
   - OMdb

5 Data Processing

Formatting Do you expect to do substantial data clean-up?

Not too much clean-up as the data comes in JSON format from the API. The data returned from an actor is fairly detailed, so we plan to cull and aggregate the data based on what we plan to visualize and also to make it easier to pass to our views. We will create the following tables:

- **Actors Table:** This will store all data on our selected actor, including the movies they have acted in.
- **Movies Table:** will contain all of the information for each movie we wish to plot or visualize. We will also add genre counts for each film for our force directed genre diagram.

Dimensions What quantities do you plan to derive from your data?

We will want to derive means, medians, standard deviations of the quantitative data, and maintain genre counts of the various films in the actor's filmography.

Method How will data processing be implemented?

Using built in math functions and basic statistical analysis and aggregate counts of parameters in JS.

6 Visualization Design

Presentation How will you display your data? Provide some general ideas that you have for the visualization design. Develop three alternative prototype designs for your visualization. Create one final design that incorporates the best of your three designs. Describe your designs and justify your choices of visual encodings. We recommend you use the Five Design Sheet Methodology.

Below, we discuss our visual designs and the rationale behind our choices. We find The following three visualizations we present to be the most effective method of showcasing an actor's career.

7 Must-Have Features

List the features without which you would consider your project to be a failure.

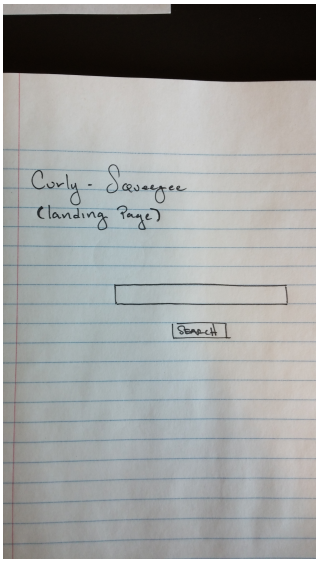
See below.

8 Optional Features

See below.

9 Desigh Criteria

9.1 Landing Page



This is the first screen the user sees. Our goal is to provide a familiar google-esque page layout with text field, search button, and Project name "curly squeegee" prominently displayed

Mandatory Features

- Minimal welcome text and instruction
- Text box
- search button

Optional Features

- graphic designed logo
- Random actor button
- list of currently popular actors as scraped from current releases

9.2 Loading Screen

After the user enters a text input, the screen state should change (perhaps darken or fade?) and display a dynamic loading icon.

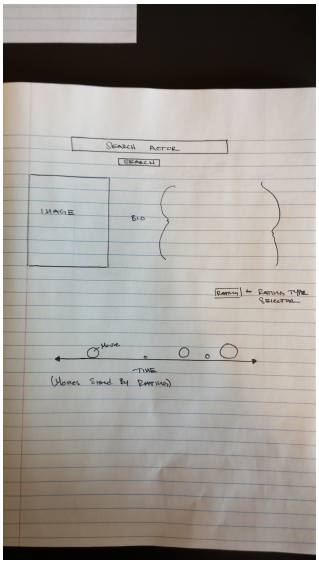
Mandatory Features

- Display "Fetching Actor filmography" or similar static message
- Dynamic loading icon

Optional Features

- Screen darkens or fades
- Live updates on what is currently being queried or fetched
- some type of loading bar

9.3 Main Screen



Once the Actor information has been obtained, parsed, and stored in our database, we will display a brief Bio. Displaying this information servers two purposes. One, It gives context to the data, and two, it is a simple way to let the user know whether they are seeing information for the right actor. For instance, Nic Cage is a separate actor from Nicolas Cage.

Mandatory Features

- Actor photo
- Actor Bio
  - Name
  - Birth Date
  - Years Active
  - Number of Films
  - etc.

Optional Features

- Disambiguation (but don't hold your breath)

9.3.1 Career Timeline

This graph shows the selected actors entire filmography as a timeline, with each film represented as a circle, sized by film rating (rating selection TBD). This timeline represents the best of our previous design iterations (drill-down barchart, pie chart), and we feel conveys the actor's career most naturally. At a glance, we can determine the critical merit of the body of their work, as well as thier acting output over their career.

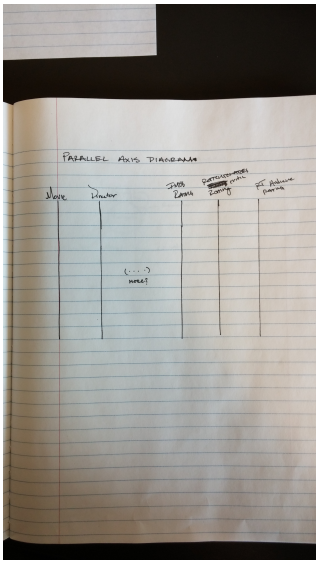
Mandatory Features

- Circle sizing by film rating
- Hover text for each film
  - film title
  - plot summary
  - ratings

Optional Features

- Film poster on hover
- Circle fill color based on genre
- zoomable timeline

9.4 Parallel Axis Diagram



This presents a large collection of filmography data for the user to explore based on their own search criteria. Incorporating a parallel axis diagram reduces the number of individual plots we originally wanted to show. It combines multiple parameter groupings in one view allowing us to explore several variables at once.

Prospective axes include:

- Actors (Top-billed)
- Film Title
- Years
- Director
- Rating
  - IMDB Rating
  - Rotten Tomatoes Critical rating
  - Rotten Tomatoes user rating

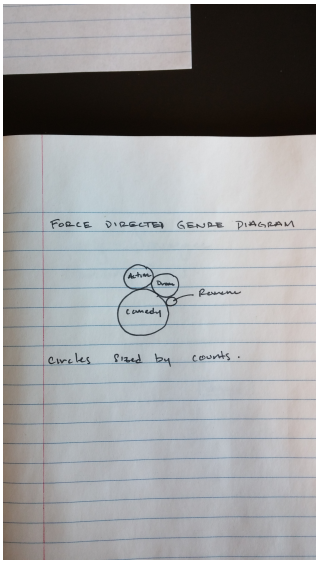
Mandatory Features

- brushing enabled on all axes
- Highlighting corresponding selected films
- hovering on crowded text

Optional Features

- N/A

9.5 Force Directed Genre Diagram



This view shows an aggregate of all genres the selected actor has acted in over their career. This view can quickly illustrate the actor's breadth of work in different areas. Each circle corresponds to a single genre classification ('comedy', 'action', etc), and each genre receives a count for the number of films which use that as a descriptor.

Mandatory Features

- Various genre classifications
- highlight movies in Actor filmography timeline by selected/hovered genre.

Optional Features

- N/A

9.6 Extras

9.6.1 Income

Create a scatter plot of the actors active years and plot annual income as reported through our data sources.

9.6.2 Awards and Nominations

Generate a barchart view (Stacked, based on nominations and awards).

9.6.3 Co-actors

Maintain an aggregated count of common actors which appear on screen with the selected actor. This can be displayed as a Force diagram with each Node sized based on the number of times they appear with the selected actor.

9.6.4 Actor as Other

Change plots to show the Actor filmography in cases other than acting (ex. director, producer).

10 Project Schedule

Plan your work so that you can avoid a big rush right before the final project deadline, and delegate different modules and responsibilities among your team members. Write this in terms of weekly deadlines.

Week	**Goals and Milestones**
October 11 - 17	Brain storm page layout data views
October 18 - 24	Set up web framework
October 25 - 31	Set up API calls/ error catching/ loading screens/ user experience
November 1 - 7	Wrangle data into workable model (start MVC)
November 8 - 14	Begin views
November 15 - 21	Time-line view / testing
November 22 - 28	Graph view / testing
November 29 - December 4	finalize views / testing