CPSC 447 26: Project Description

Read this entire document carefully!

Over the rest of the term, you will work in teams of four towards creating an interactive visualization project using JavaScript & D3, with the following milestones:

- **M0 Teams: Form team** *(Due Jan 21, no feedback)*
- **M1 Abstractions: Data and tasks writeup** *(Due Feb 2, Gradescope feedback)*
- **M2 Design: Visualization design & plan writeup** *(Due Feb 23, Gradescope feedback)*
- **M3 WIP: Work in progress implementation** *(Due Mar 9, in-person feedback)*
- **M4 Final implementation, video demo, & writeup** *(Due April 7, live demo & Q&A by Zoom, then Gradescope feedback later)*

General Project Requirements

Implement a visual analysis app/dashboard or interactive data-driven story (narrative visualization) in D3 based on the following requirements:

Mandatory

- **Data**
  - You must use publicly available dataset(s). You might use a single existing dataset, or create a 'mashup' where you combine multiple datasets together to explore something interesting. You may find one of these resources useful:
    - Data is Plural (or as Google Sheet)
    - Kaggle Datasets
    - Google Dataset Search
  - You may either store the data in your repo, or connect to a source via an API.
  - You may want to pre-process your datasets, for example to eliminate unneeded data, or to derive data, or to harmonize multiple datasets into a 'mashup'.
  - Choose your dataset(s) wisely: your data should be complex and large enough to meet the requirements below, but most of your time should be spent on the visualization itself not on data processing.
- **Tasks**
  - You must target at least three distinct abstract tasks pertaining to at least five distinct data attributes. You may have more than one domain task that maps to the same abstract task.
  - Your project should have sufficient overall complexity that visualizing multiple attributes using multiple views is necessary.

    - All tasks must be plausible and must be addressable by the dataset(s) you are using.
- **Design**
    - At least three distinct views.
    - At least one view should be complex, meaning that it visually encodes at least four attributes at once. These attributes cannot all be the same type. All of these attributes must be encoded for all data simultaneously, so using tooltips or interaction to show additional attributes on demand does not count. Note that many of the standard chart types do not fulfill this requirement (i.e. bar chart, scatterplot, line chart, simple area chart, pie chart, donut chart, choropleth, dot density map, circle-pack (bubble) chart, tile heatmap, radar plot, treemap, Sankey diagram). You could design and implement a unique view that is appropriate to support one or more of your intended tasks.
        - Note: This definition differs from previous years, so past examples might not meet this requirement.
    - At least two views should be bidirectionally linked, and those should be simultaneously visible side by side onscreen.
    - At least one view must support mouse-based interaction (eg. hover, click, drag, tooltip) in addition to the bidirectional linking interaction.
    - At least one UI widget (eg. dropdown, radio button, range slider, calendar).
    - Most projects will be a good fit for dashboard style (where all or nearly all views are visible at once), but you are also allowed to use a scrollytelling style.
    - Layout should use screen space wisely. Layouts that are too sparse force the user to scroll unnecessarily and fall short with supporting bidirectional linking. Layouts that are too dense are cluttered.
    - Interface should include enough headings, legends, axis labels, and any other explanatory text needed to make it self-documenting.
- **Rationale**
    - You must provide detailed rationale about how all aspects of your design address your identified scenario.
- **Implementation**
    - Your web application should be stand-alone and have an `index.html` file as the entry point, similar to the programming exercises. You can either store static datasets (*optionally preprocessed*) in your git repo or access live data through an API.
    - Your code must be readable, reasonably structured, and conform to a consistent style. You should follow the structure and code style of the assignments and tutorials.
    - Your git commits must be reasonably sized and include informative commit messages.

Extra Credit (+1% to +5%)

- High level of polish in final interface.

- Going above and beyond with functionality.
- Outstanding and thoughtful explanation in the write-up.
- Demonstrates true mastery of foundations concepts.
- Overcoming a challenging problem.

Not Allowed

- We do not allow custom backends (Node.js, Python, etc.) and database systems, such as Postgres or MySQL, although they could facilitate more powerful applications. In this course project, you should put your efforts into the front-end development (JavaScript, D3, HTML, CSS).
- You need to implement visualizations in D3. You can use CSS frameworks, such as Bootstrap or Materialize, and include external libraries (jQuery, leaflet.js, moment.js, ...). Layers such as NVD3, Vega-lite, Highcharts, etc. are not allowed. Using external libraries such as React.js for the structure/layout of your system is allowed, but you cannot use these libraries to generate any part of your visualizations. TAs cannot help you with external libraries that are not used in the class.

More Details

Strong design work will make the implementation easier!

It is reasonable and encouraged to change your design as you go, if you learn that your initial ideas would benefit from refining. You should provide rationale for your changes, just as you should provide rationale for all of your other design choices.

Talk to a TA in office hours to discuss your idea if you have any uncertainty, particularly regarding your choice of components.

The goal of the project is to synthesize the material from both the programming exercises and the foundational exercises. We will evaluate projects using the following high-level criteria:

- Visual design: Effectiveness of visualizations and interactions, which should link back to usage scenarios, tasks, and intended audience.
- Level of technical difficulty and quality of implementation.
- Addresses the goals and requirements.
- Quality and clarity of your writing.
- Teamwork.

Within the team, everyone should have roughly equal parts to do for the technical aspects such as programming. You **cannot** have one team member do, for example, all the writeups and none of the programming.

You will need to submit peer evaluations of the contributions of all team members (your teammates and yourself) at each of the four milestones. Normally all team members will receive

the same project mark, but if there are major disparities in the levels of contribution we may assign different grades to different group members.

Milestone 0: Team Formation

Form a team by filling out Project Team Formation Qualtrics Survey by Wed Jan 21, 6pm.

- You should form a team of exactly **four** people that you will work with closely for the rest of the term.
- We do not allow teams with more or less than 4 people. If you have a very unusual circumstance with a strong need for a team of 3, we will consider your request, since there may need to be a few such teams depending on final course headcount. Talk to a TA or instructor about it in office hours; if your request is granted, state who approved it in the Additional Comments section of your M0 Qualtrics form.
- You may want to use the pinned Piazza post to find teammates, and/or talk with other students during class. Don't wait until the last minute!
- After the milestone, we will create empty GitHub team repositories for teams. Your team will be responsible for downloading the zip file with skeleton code/files.

Milestone 1: Project Abstractions

Deliverables:

- **Submit a single M1 writeup in PDF format for the whole team via Gradescope by Mon Feb 2 2025 at 6pm**
- **Each team member must submit individual M1 peer evaluation forms via Qualtrics by Wed Feb 4 2025 at 6pm**

Project Abstractions Report Requirements

When submitting your project abstractions as a single PDF, please include the following sections in this order:

1. Basic Info
2. Overview
3. Data
4. Tasks
5. Visualizations (EDA for now)
6. Usage Scenarios (Blank for now)
7. Team Communication Plan
8. Work Breakdown and Schedule
9. Credits (Blank for now)

Your proposal should be somewhere around 5 pages in total, including text, images, and tables. You will be iterating on your writeup in later milestones, adding more in M2 and a lot more in M4. Sometimes you'll be able to re-use text without changes, sometimes you'll be specifically asked to improve sections based on feedback from the TAs, and sometimes you'll decide to change things as your ideas evolve.

Sections

1. Basic Info

- Project title
- A list of all team members, with student numbers and email addresses

2. Overview

- A few sentences that describe what problem your visualization is tackling and how. Your project could provide a platform for data analysis or exploration, or you may choose to

tell an interactive data-driven story. The theme of your visualization can draw from any topic, including current affairs, history, natural disasters, research findings from the sciences and humanities, or anything that you find interesting.

- Your overview should make the goals of your proposed system clear, while still remaining feasible in terms of capabilities that you can implement in one term.
- State the intended audience. Be brief and clear. Be careful with your claims of who will use your tool. It is much easier to create a project that enables curious individuals to investigate something they're interested in than one that someone in a high-up position like a CEO or politician would use.

3. Data

- Provide a link to your data source(s).
- In your report, briefly describe the dataset and the attributes that you will visualize, in both domain-specific and abstract terms. For the abstraction, include the type and cardinality (total number of items) of the entire dataset(s), and for each attribute their types and either the cardinality for categorical/ordered attributes or the range for quantitative attributes. You may find it helpful to include a table, rather than simply relying on text.
- When these numbers are large, you can be approximate with terms like "dozens", "hundreds", "thousands": for example, 200K instead of 196,873; hundreds instead of 376. You should differentiate attributes that come with the dataset from new attributes that you might derive for your visualizations.
- If your dataset has many more than 20 attributes and you plan to visualize them all, then you may provide a high-level descriptor of groups of attributes, for example say the dataset contains *demographic attributes* instead of describing every single variable.
- Discuss any pre-processing or cleanup that you plan to do (or have already done). Do you need to preprocess or clean up the original data? Do you need to join multiple datasets? How do you plan to implement the processing pipeline? (e.g. in a separate tool, such as OpenRefine or Trifacta Data Wrangler; R scripts that are executed once; on the fly in JavaScript).
    - Make sure you explain why you make pre-processing decisions. For example, if you decide to filter some data out, you should have a strong reason both for filtering in the first place and for which data you choose to filter out.

4. Tasks

- Present a list of tasks that your future system is intended to support. You must articulate a minimum of three abstract tasks pertaining to at least five distinct attributes.
    - As a reminder, abstract tasks consist of {action, target, data} triplets.
- All tasks must be plausible and must be addressable by the dataset(s) you are using.
- You must provide both a list of domain-language tasks and a list of abstract tasks. There may not be a one-to-one relationship between these lists: for example, more than one domain-language task might map to the same abstracted task.

    - We recommend writing many domain tasks which map to your three abstract tasks.
    - You must clearly indicate which domain tasks map to which abstract tasks.
    - Make sure that the use case of each domain task is clear, for example by concisely describing what the user's goals or expected takeaways are when conducting the task.
- The tasks should be closely tied to the description of data above, so that it's clear which specific attributes are relevant for each task.
- Your project should have sufficient overall complexity that visualizing multiple attributes using multiple views is necessary.

5. Visualizations (EDA)

- For now, just do some initial exploratory data analysis and explain your results in a few paragraphs. This analysis is meant to give you an initial understanding of your data, it doesn't have to be exhaustive. If you already know that you will filter out or ignore some attributes of your dataset, you do not need to analyze them in this section.
- The type of analysis will depend on the type of data that you have and what your tasks are, but below are some example questions to get you started.
    - What's the spread of your data?
    - Is the data evenly distributed or skewed towards one direction?
    - Do any of your attributes have one value that is significantly more common than other values from the same attribute, or a small number of values that are common in contrast to many others with only a few instances?
    - Are there any errors or anomalies in your data that you will need to fix?
    - Does the data have any notable outliers?
- This section **must** include simple charts to help communicate the results of your EDA. These charts could be as simple as univariate distributions like histograms, column charts, or line charts created in Excel or Google Sheets.

6. Usage Scenarios

- You may leave this section blank for now, but include the header.

7. Team Communication Plan

- Include a description of your plan for how you will communicate during the duration of the project.
    - Describe your team meeting strategy in terms of when, how often, and how/where you will meet (whether online or in person).
    - Also describe how you will asynchronously communicate outside of meetings (which platform you will use for messaging) and what the expected response time is for those messages.

    - Finally, explain the steps you will take should any of your team members stop complying with the agreed-on communication plan. We suggest that at minimum if a team member is inactive and not responding to communication attempts for more than 4 days, the other team members should let the course staff know.
- We suggest you schedule at least a regular weekly meeting time that works for your whole team and add on additional meetings as needed.

8. Work Breakdown (To Date)

- All team members must be roughly equally engaged in each part of the project. Everyone should be involved in choosing a dataset, creating the abstractions, and writing the reports.
- Break down the work completed so far into components. One possible breakdown is
    - Choosing dataset
    - Conducting exploratory data analysis
    - Characterizing data
    - Deciding on domain tasks
    - Abstracting the tasks
    - Writing report (possibly split into separate sections)

  You may choose a different breakdown of components; this list is only one possibility.

- Include a table of work completed so far, with each component on a different row. The columns should be which team members worked on it, the date that it was finished, and roughly how many hours it took to do. At the bottom, provide totals for the number of hours per team member, and the number of hours used overall.

9. Credits

- You may leave this section blank for now, but include the header.

Peer Review

You will be asked to assess the ratio of work done by each member of the team (including yourself). The common case is that everybody is doing roughly equal shares of the work. Sometimes some people are doing a little more or a little less. We particularly need to know if some people are doing much more work or much less work. Please provide additional comments in the free-form text field if you indicate any situation other than the work being equally split. Even if you have mutually agreed to a non-equal work breakdown as of an intermediate milestone (for example, a teammate is ill and plans to do more later), you should still report the actual work breakdown and explain the situation in the comments.

The peer review will take place through a Qualtrics form that you fill out personally. At each milestone, report on the cumulative ratio across the entire term, not just the work since the previous milestone.

The link to the M1 peer review form is:
https://ubc.ca1.qualtrics.com/jfe/form/SV_4TNHAW7fVDJ4lE2

Examples from Previous Hall of Fame

These examples from the previous Hall of Fame include a mix of Overview (2), Data (3), and Tasks (4) sections.

Building Bricks

Overview

Building Bricks is an exploratory data analysis visualization project centered on Lego sets, pieces, colours, and themes. Within the expansive Lego universe, consumers frequently encounter challenges in comprehending the extensive product range. Questions such as 'How have set sizes evolved over time?' and 'Which sets exhibit the most overlap?' commonly arise.

The primary objective of our visualizations is to assist Lego consumers, both casual enthusiasts and dedicated fans, in developing a comprehensive understanding of Lego products in an easily digestible manner. Through these visualizations, we seek to illuminate trends in product evolution over the years and unveil associations between various sets and themes.

This project caters to a diverse audience, including casual Lego enthusiasts and more avid collectors, aiming to provide valuable insights and a macroscopic perspective on the Lego product landscape. By addressing these inquiries through visual storytelling, Building Bricks aims to enhance the overall Lego experience, making data on sets, pieces, and themes accessible and engaging for a broad audience.

The project is designed to be optimal on a screen size of 1536 x 725 and is supported on the Chrome browser.

Bored? Games!

Overview

With over 130,000 board games listed on the popular website BoardGameGeek, it isn’t always easy to find the one that will finally replace Catan as the board game night favorite or make a perfect gift for your friend Jane who loves cats but hates complicated rules. To help board game lovers and newbies alike explore and map the board game landscape, we propose to create a

visualization that focuses on the stories games tell (their themes) and on the similarities between games. Users will be able to filter by theme and other attributes that influence whether a game is a good fit for them, and to search for games to discover similar hidden gems.

Data

We propose to use data scraped from BoardGameGeek in January 2022 (source: Kaggle). The data includes 21,432 board games, but we will use a subset: the 1000 highest ranked games. Each board game has 19 attributes that we plan to use. Some attributes will be encoded visually, some will only be used for filtering, and others will be used in tooltips or annotations.

Attribute	Type			Cardinality / range	Notes
	Categorical	Ordinal	Quantitative		
Basic game information: 5 attributes					
id, name, description	x			1000 levels each	
designer	x			703 levels	a game has 1-8 designers
year		x		-2200 to 2021	publication year (or creation: e.g. Go in 2200 BC)
Game play parameters: 4 attributes					
minplayers			x	1 to 6	
maxplayers			x	1 to 100	
minage			x	0 to 18	minimum suggested player age
playingtime			x	0 to 1200 minutes	
Detailed game information: 6 attributes					
type	x			6 levels	derived from typed rank columns in original dataset each game has 1-3 types
category	x			79 levels	each game has 0-9 categories
mechanic	x			174 levels	each game has 1-20 mechanics
family	x			1117 levels	each game has 0-28 families
theme	x			39 levels organized into 6 clusters	derived (extracted) from 'category' and 'family' and manually grouped into thematic clusters each game has 0-7 themes in 0-5 clusters
complexity			x	1.0251 to 4.734	a measure of game difficulty
Rating and rank information: 3 attributes					
usersrated			x	1285 to 109,006	our proxy for popularity
rating			x	6.67 to 8.84	average of user ratings
rank		x		1 to 1001	each game has a unique rank

Visualizing Gun Violence in America

Overview

In recent years, gun-related crimes have skyrocketed across North America. While many are seeking to find a common ground between safety and gun ownership, having greater awareness of factors contributing to gun violence can lead to better discourse surrounding injury

prevention. To address this issue, we propose a visualization that will allow policymakers and the general public to explore the characteristics surrounding a range of gun-related incidents in the US from 2013 to 2018. Users will be able to use a variety of interactions and filters to explore the geographical distribution of gun-related crimes, incident characteristics, and trace incidents back to individuals involved to gain a holistic picture of potential factors contributing to gun violence.

Data Description

We will be visualizing a dataset that initially has 239677 shooting incidents in America between 2013-2018. We will only be using seven of the given attributes and generating one additional attribute. The attributes describe the logistical details of each shooting incident (date [categorical, 1725 values], state [categorical, 51 values], latitude [quantitative, 19.1114 - 71.3368], longitude [quantitative, -171.429 - 97.4331]) as well as details describing the actual situation of the incident (incident_characteristics [categorical, 671 values], participant_status [categorical, 4 values], participant_age [ordinal, 0 - 311]). We will be generating a new attribute (age_group, [categorical, 11 values]) based on the participant_age attribute. Note that there are 51 states in this dataset as the “District of Columbia” was considered an additional state by the author. Also, for categorical attributes, empty cell values did not constitute an additional unique value. An important thing to note is that while incident_characteristics has 671 unique values, we will only use the top 5 most frequent values as filtering options across our visualizations which is a safe choice since 222073 rows have at least one of the top 5 values for this attribute. The remainder will be represented by the “Other” filtering category.

Milestone 2: Design Proposal

Deliverables:

- **Submit a single M2 writeup in PDF format for the whole team via Gradescope, by Mon Feb 23 2025 at 6pm**
- **Each team member must submit individual M2 peer evaluation forms via Qualtrics, due by Wed Feb 25 2025 at 6pm**

Project Design Report Requirements

When submitting your project proposal as a single PDF, please include the following sections in this order:

1. Basic Info
2. Overview
3. Data
4. Tasks
5. Visualizations
6. Usage Scenarios
7. Reflections (blank for now)
8. Work Breakdown and Schedule
9. Credits (blank for now)

Your design proposal should be somewhere between 5-10 pages in total, including text, images, and tables. It should include 1-5 sketches of your application. If you were given feedback from M1 that anything in sections 1-4 needs improvement, you should do so in this writeup. If you were given feedback that these sections were in good shape you may not need to do much updating, but you should still read the sections carefully to ensure they are fully up-to-date with the current version of your project. You will be assessed on the quality and clarity of your writing, the feasibility of what you propose, and the appropriateness of your designs according to the criteria of effectiveness taught in this class.

Sections

1. **Basic Info**
    - This section can be copied from your previous submission, with any updates to fix problems noted in previous feedback.
2. **Overview**
    - This section can be copied from your previous submission, with any updates to fix problems noted in previous feedback or that you think strengthen the work.

3. **Data**
    - This section can be copied from your previous submission, with any updates to fix problems noted in previous feedback or that you think strengthen the work.
4. **Tasks**
    - This section can be copied from your previous submission, with any updates to fix problems noted in previous feedback or that you think strengthen the work. It is very common that teams need to improve the task writeup for this milestone.
5. **Visualizations**
    - Remove the EDA discussion from the previous submission.
    - Begin this section with a high-level description of the visualization interface you will build in terms of which views it will include. Remember to be realistic since you are actually required to implement this interactive visualization. Most projects will be a good fit for dashboard style (where all or nearly all views are visible at once), but you are also allowed to use a scrollytelling style.
    - Include an overview figure of your planned visualization, showing how the different views will be sized and organized. For this figure, it is fine to have a very rough sketch here, but make sure to include all elements of the project (views, titles, widgets, etc.).
    - Justify why you chose your particular views by providing rationale for why these views support your proposed tasks, as opposed to alternative possibilities.
    - You should analyze the visual encoding in detail, in terms of marks and channels, **only for your most complex view** (which is probably your novel view). For the other views, if they have already been analyzed in any of the async lecture slides, you can describe the visual encoding very concisely - just explain how your data abstraction is mapped to the elements of the view.
    - Your sketch can be hand-drawn or mocked up using Powerpoint, a graphics editor, or wireframe tools such as Balsamiq. Most likely, you will need more than one image to show your proposed visual design and interaction sequences, but don't use more than five sketches. The intent of these sketches is a very basic illustration to communicate your ideas to the marker (so that we can provide feedback about scope and realism) and to help guide your team during the implementation of the first prototype. They should not limit what you end up doing as the final project!
    - The mandatory design requirements are
        - At least three distinct views.
        - At least one view should be complex, meaning that it visually encodes at least four attributes at once. These attributes cannot all be the same type. All of these attributes must be encoded for all data simultaneously, so using tooltips or interaction to show additional attributes on demand does not count. Note that many of the standard chart types do not fulfill this requirement (i.e. bar chart, scatterplot, line chart, simple area chart, pie chart, donut chart, choropleth, dot density map, circle-pack (bubble) chart, tile heatmap, radar plot, treemap, Sankey diagram). You could design and

implement a unique view that is appropriate to support one or more of your intended tasks.
    - Note: This definition differs from previous years, so past examples might not meet this requirement.
    - You must indicate which view is your complex view.
  - At least two views should be bidirectionally linked, and those should be simultaneously visible side by side onscreen.
  - At least one view must support mouse-based interaction (eg. hover, click, drag, tooltip) in addition to the bidirectional linking interaction.
  - At least one UI widget (eg. dropdown, radio button, range slider, calendar).
6. **Usage Scenario**
    - Provide a usage scenario. Usage scenarios are typically written in a narrative style and include the specific context of usage, tasks associated with that usage context, and a hypothetical walkthrough of how the user would accomplish those tasks with your visualization design. The purpose of the usage scenario is to get you to think about how someone else might use the web application you're going to design, and to think about those needs before you start programming. It should also help you assess the plausibility of your identified tasks and the suitability of your chosen datasets to address those tasks. If you are using a Kaggle dataset, you may use their "Overview (inspiration)" to create your usage scenario, or you may come up with your own inspiration.
    - Make sure to include the user's initial motivations and their takeaways after conducting the task.
7. **Reflections**
    - Remove the team communication discussion from the previous submission.
    - Leave blank for now, but include the section header.
8. **Work Breakdown and Schedule**
    - All team members must be roughly equally engaged in each part of the project. Everyone should be involved in choosing a dataset, creating the abstractions, designing the system, implementing the system, and writing the reports. You **cannot** have one team member do, for example, all the writeups and none of the programming.
    - Break down the planned work into components, so that the work can be planned in terms of modular pieces that can be distributed between team members. Think about these questions: What are must-have components? What are optional components that can be implemented after the third milestone, if there is time? What components have dependencies with other ones? These components should include what you did in this Design milestone and what you will do in the future Implementation/WIP and Final milestones, in addition to the previous components that you discussed in the earlier Abstractions milestone. For example, a component might be data pre-processing, or the creation of a simple static version of one of your views, the linking of two views together, or dedicated time for bug fixing and refinement for a view. You do not have to use these

components exactly, but you should break your work down into similar-sized components.
    - This milestone must have a detailed table of work completed so far **and** work that is yet to be completed. Each component should be a different row in your table. The columns of your table would be Who (which team member(s) did or will work on it), Estimated Hours (how many hours of time you estimate it will take), Estimated Date (the date by which you estimate it will be completed), Actual Hours (how many hours it actually took), and Actual Date (when it actually got done). These last two columns can be blank for future components, but should be filled in for completed work done for Milestones 1 and 2. The point of this project management planning is to gain skills at estimating work times; you will not be penalized for discrepancies between estimates and actual.
    - Your table must include a summary that indicates the totals for each team member **and** also overall, for both estimated **and** actual hours used to date.
    - The scope of the work should be roughly 50 hours per person (200 hours total across the team of four), for the entire project -- across all four milestones.
9. **Credits**
    - Cite any external sources here. Leave blank if none used.

Peer Review

You will be asked to assess the ratio of work done by each member of the team (including yourself). The common case is that everybody is doing roughly equal shares of the work. Sometimes some people are doing a little more or a little less. We particularly need to know if some people are doing much more work or much less work. Please provide additional comments in the free-form text field if you indicate any situation other than the work being equally split. Even if you have mutually agreed to a non-equal work breakdown as of an intermediate milestone (for example, a teammate is ill and plans to do more later), you should still report the actual work breakdown and explain the situation in the comments. We also welcome any comments regarding your experience with the course, overall.

The peer review will take place through a Qualtrics form that you fill out personally. At each milestone, report on the cumulative ratio across the entire term, not just the work since the previous milestone.

The link to the M2 peer review form is:
https://ubc.ca1.qualtrics.com/jfe/form/SV_29S5NdZl33oUvHg

Examples from Previous Hall of Fame

We provide a few example usage scenarios from past Hall of Fame projects

United by Networks

Usage Scenario

Sarah is a seasoned policymaker with a focus on improving education and social outcomes in the United States. She's keen to explore socioeconomic interactions within U.S. colleges to develop data-informed policies and initiatives in hopes of improving educational outcomes. She opens the application and starts her exploration with the dot density map of the USA, where she can see economic connectedness, mutual friendships, and colleges' geographic location. Sarah recognizes that this view is essential for her tasks as it allows her to identify colleges with a high proportion of mutual friends and connections between different SES groups via the size and color of the dots. Next, Sarah looks at the Generational Biases scatterplot where she can discover generational trends in parental SES on college students' socioeconomic interactions, which she does by examining how parental SES and student-based friending bias correlate. To gain insights into controllable factors for increasing SES, Sarah explores the Controllable Factors scatterplot group. Each scatterplot within the group represents a different independent variable on the x-axis and change in SES on the y-axis. She uses the sliders to filter data based on low family SES and the desired SES increase threshold. Sarah recognizes that this view is crucial for analyzing controllable factors that correlate to an increase in SES for individuals from low socioeconomic backgrounds. Intrigued by a specific college's social interactions, Sarah clicks on a college dot in any of the previous views to open the Magnified College View. Here, she can compare the amount of connectedness between students of varying socioeconomic status within a college by examining the college's social connections and understanding the strength of friendships between students of different SES groups. Sarah decides to zoom into a state view to understand the support ratios of colleges within a specific state, which allows her to assess the level of support within a state as this information may be valuable for state-specific policy decisions. As Sarah uses this interactive visualization tool, she gains comprehensive insights into socioeconomic interactions in U.S. colleges. The data-driven exploration assists her in formulating policies and initiatives to enhance social capital and improve educational outcomes across the country.

Bored? Games!

Usage Scenario

Bob is a board game enthusiast and loves spending his free time learning about and playing games. He wants an application that will allow him to explore and discover new board games. When he opens the application he wants to see an overview of games available, so he focuses on the graph visualization of all the games in the dataset. To help guide his exploration, he wants to be able to quickly identify which games and themes are the most popular. For example, looking at the bar chart of themes, he might be surprised to see how many games have a Train theme. He decides he wants to further explore that theme, by selecting it as a filter and browsing the remaining games in the graph. As he hovers over a game mark, he's able to see its description, year published, designer, and more. He can also see which 5 other games are most similar to it. While exploring, he is reminded of a game he really likes named Catan. So he searches the dataset in order to locate the game. After finding it, he is able to see where it

falls on the complexity scale, if it is considered a highly reviewed game or not, and what games are similar. Additionally, since Bob plays games with different groups of friends and family, he wants to be able to find games that suit a specific occasion. Therefore, he wants to filter the games in the dataset based on the number of players, minimum age, average playtime, theme, and complexity. For example, Bob has a Halloween party to attend. He sees that Horror is a theme that contains quite a few games so he applies that filter. Since the event is a party, he wants a highly reviewed game that is not too complex: he uses the hexbin plot to discover the distribution of games for these two attributes and discovers a few games that meet his criteria.

Milestone 3: Project Work-in-Progress (WIP)

Deliverables:

- **Tag a release of your project with tag version `v0.1-wip` by Mon Mar 9, 6pm. Tagging a release allows you to continue the development, even before this milestone is graded. See instructions for creating releases on Github. Your release must contain your first prototype implementation.**
- **Each team member must submit individual M3 peer evaluation forms via Qualtrics by Wed Mar 11, 6pm**

Implementation Guidelines

We created a git repository for your team on github.students.cs.ubc.ca. Develop your project by using this repository (don't forget, we'll be considering your commits when grading).

Turn your project proposal into an interactive web-based visualization with D3.

- Keep your usage scenario (from your proposal) in mind when designing your interface. Make sure that your visualization's interface and functionality either (a) allows a user to answer your proposed question or (b) successfully communicates a message or series of messages (narrative visualization).
- Make sure that your code is well-organized and easy to read. Use separate files for visualization components, use functions to promote code reuse. Agree upon a JavaScript style guide (e.g., Airbnb or Google) that every team member will follow. Don't work with a messy repo and then try to clean it up before the final milestone. Comment your code early on!
- We don't allow custom backends (Node.js, Python, etc) and database systems, such as Postgres or MySQL, although they could facilitate more powerful applications. In this course project, you should put your efforts into the front-end development (JavaScript, D3, HTML, CSS). Your web application should be stand-alone and have an `index.html` file as entry point, similar to the programming exercises. You can either store static (*optionally preprocessed*) datasets in your git repo or access live data through an API.
- It can be easy to get sucked into a rabbit hole when trying to implement a stubborn feature. While it is important to build troubleshooting skills, we do not want getting stuck on one feature to prevent you from completing the full first prototype. If you're struggling with a particularly tough problem, some ideas are: save it for later, discuss with your teammates, or ask a TA for help!

WIP Requirements

Implement the general user interface including at least static and unlinked version of each major view (visualization component) that you proposed. You don't need to have all details fully implemented for this milestone, such as interactivity or linkages between the views, but you should be able to see what the data looks like in each of them.

Don't worry about missing tooltips, small usability issues, or fully implementing your bidirectional linking for this milestone. That said, we encourage you to work closely with your teammates while implementing, in particular to carefully design how you will integrate your views together, i.e. through interaction or layout, later on. If each team member implements their assigned views completely independently, it may take significant time and effort to integrate them later.

Although your eventual goal is to create a self-documenting interface, you will not be marked down if you have not yet fully achieved this goal in this milestone. Eventually, your interface should be as self-documenting as possible, with appropriate labels for panes, axes, and widgets, a legend documenting the meaning of visual encodings, and a meaningful title and description for the project.

In-Person Feedback

During class time in Week 10 (Mar 11), one day after this milestone is due, you will meet with your assigned TA as a group for up to 20 minutes per group. You will present your WIP project to the TA and discuss your plans for next steps. The TA will mostly provide formative feedback to you, and you will also be given a summative mark based on whether you sufficiently completed enough of your project, as described above in Requirement 1 (basic version of every view must exist). The marking will be in bins: meets all requirements (100%), meets some requirements (80%), meets few requirements (60%), far below expectations (40%).

You must be in class in person for this session. Each TA will call up groups for discussion at their discretion, so your group may be asked to present earlier or later during the session. During the rest of the class time that you're not talking to the TA, your group should work on your project.

Peer Review

You will be asked to assess the ratio of work done by each member of the team (including yourself). The common case is that everybody is doing roughly equal shares of the work. Sometimes some people are doing a little more or a little less. We particularly need to know if some people are doing much more work or much less work. Please provide additional comments in the free-form text field if you indicate any situation other than the work being equally split. Even if you have mutually agreed to a non-equal work breakdown as of an intermediate milestone (for example, a teammate is ill and plans to do more later), you should

still report the actual work breakdown and explain the situation in the comments. We also welcome any comments regarding your experience with the course, overall.

The peer review will take place through a Qualtrics form that you fill out personally. At each milestone, report on the cumulative ratio across the entire term, not just the work since the previous milestone.

The link to the M3 peer review form is:
https://ubc.ca1.qualtrics.com/jfe/form/SV_8wUcoQoB3l04W90

Milestone 4: Final Project Submission

Deliverables:

- **Submit a single M4 writeup in PDF format for the whole team via Gradescope by Tue Apr 7 2026 at 12pm**
- **Tag a release of your project with tag version `v1.0-final` by the same deadline. Your release must contain the final implementation and a thumbnail screenshot of your vis (`thumbnail.png`).**
- **Each team member must submit individual M4 peer evaluation forms via Qualtrics by Wed Apr 8 2025 at 6pm**

Reminder: Face to Face Grading

Your team will sign up for a face-to-face grading slot with the TA assigned to your group (roughly 5-10 minutes per group), through a scheduled Zoom session. The TA will have already read your report and watched the video version of the demo. You should be prepared to answer any questions they have for clarifying the behavior of your system through a short demo, be ready to share your screen with the submitted version running live. The TAs may ask you to talk through any details of the software architecture or implementation. All members of your team must attend the face to face demo. A signup sheet for time slots will be released before the due date.

Implementation Requirements

Finish developing your visualization project by using your git team repository (don't forget, we'll be considering your commits when grading your project).

Finish turning your project proposal into an interactive web-based visualization with D3 that is ready to be deployed and that can be consumed by your intended audience.

- **Refine the features you started earlier and implement the rest of the features you outlined in your proposal, paying close attention to usability, information density, and having a self-documenting interface.**
- You must have at least three distinct views.
- At least one view should be complex. Your complex view should be cleared by your TA through M2, office hours, or M3. As a reminder, it must visually encode at least four attributes at once. These attributes cannot all be the same type. All of these attributes must be encoded for all data simultaneously, so using tooltips or interaction to show additional attributes on demand does not count. Note that many of the standard chart types do not fulfill this requirement (i.e. bar chart, scatterplot, line chart, simple area

chart, pie chart, donut chart, choropleth, dot density map, circle-pack (bubble) chart, tile heatmap, radar plot, treemap, Sankey diagram). You could design and implement a unique view that is appropriate to support one or more of your intended tasks.
    - Note: This definition differs from previous years, so past examples might not meet this requirement.
- At least two views should be bidirectionally linked, and those should be simultaneously visible side by side on screen.
- At least one view must support mouse-based interaction (eg. hover, click, drag, tooltip) in addition to the bidirectional linking interaction.
- You must have at least one UI widget (eg. dropdown, radio button, range slider, calendar).
- Most projects will be a good fit for dashboard style (where all or nearly all views are visible at once), but you are also allowed to use a scrollytelling style.
- Your layout should use screen space wisely. Layouts that are too sparse force the user to scroll unnecessarily and fall short with supporting bidirectional linking. Layouts that are too dense are cluttered.
- Your interface should be as self-documenting as possible, with appropriate labels for panes, axes, and widgets, a legend documenting the meaning of visual encodings, and a meaningful title and description for the app.
- Apply what you have learned in the *color* foundation lectures and choose appropriate color schemes.
- Your web application should be stand-alone and have an `index.html` file as entry point, similar to the programming exercises. You can either store static (*optionally preprocessed*) datasets in your git repo or access live data through an API.
- Remember, we don't allow custom backends (Node.js, Python, etc) and database systems, such as Postgres or MySQL, although they could facilitate more powerful applications. Your web application should be stand-alone and have an `index.html` file as entry point. You can either store static (*optionally preprocessed*) datasets in your git repo or access live data through an API.
- Your code must be readable, reasonably structured, and conform to a consistent style. You should follow the structure and code style of the assignments and tutorials.
- Your git commits must be reasonably sized and include informative commit messages.
- Review the project description document to ensure you are meeting all requirements.

Thumbnail Image

Create a single representative screenshot of your project (size: **1280 x 720 px**) named **`thumbnail.png`** and add it to the root directory of your repository. We will use it to publicize the top projects.

Project Final Report Requirements

In addition to the code implementation pushed to the provided git repository, you must submit a final report as PDF.

Your final report should be a standalone document that fully describes your project.

When submitting your project proposal as a single PDF, please include **exactly** the following sections in this order:

1. Basic Info
2. Overview
3. Data
4. Tasks
5. Visualizations
6. Usage Scenarios
7. Reflections
8. Work Breakdown and Schedule
9. Credits

Your report should be at least 5 pages of text (~2500 words), in addition to screenshots; there is no hard limit on length because we encourage you to include as many screenshots as you need to illustrate your project clearly and to write as much as you need to explain your design choices.

You can build on your project proposal report, but make sure to carefully update everything. In particular, remove/reword any future tense when discussing any part of what you built, since with the completion of this milestone the project is finished.

1. **Basic Info**
    - Project title
    - A list of all team members and student numbers
2. **Overview**
    - *Teaser image* (screenshot) of your visualization.
    - Refine your overview to be a concise 250 words: a few sentences that describe what problem your visualization is tackling and how. The theme of your visualization can draw from any topic. State the intended audience. Be brief and clear.
    - This section can be copied from your previous writeups, with any updates to fix problems noted in previous feedback or that you think strengthen the work.
3. **Data**
    - This section can be copied from your previous writeups, with any updates to fix problems noted in previous feedback or that you think strengthen the work.
4. **Tasks**
    - This section could be copied from your previous writeups, with any updates to fix problems noted in previous feedback or that you think strengthen the work, but it

is very common that teams will need to refine their task descriptions after working on the project.
    - You must have at least three distinct tasks (in both domain and abstract language) and together they must pertain to at least five distinct attributes. The tasks must have sufficient overall complexity that visualizing multiple attributes using multiple views is necessary. All tasks must be plausible and must be addressable by the dataset(s) you are using. You must provide both a list of domain-language tasks and a list of abstract tasks. The tasks should be closely tied to the description of data above, so that it's clear which specific attributes are relevant for each task.
5. **Visualizations**
    - Describe the visualization interface that you have built. What views are there and what do they allow users to do?
        - For each view, describe your visual encoding choices and include the rationale for your design choices.
        - You should analyze the visual encoding in detail, in terms of marks and channels, **only for your most complex view**. For the other views, if they have already been analyzed in any of the async lecture slides, you can describe the visual encoding very concisely - just explain how your data abstraction is mapped to the elements of the view.
        - How can users interact with your project within each view, and how are views linked?
    - **Provide rationale for your design choices**, for both visual encoding and interaction. Why are they appropriate, in terms of the principles covered in this course, for your chosen data and tasks?
        - Your rationale should link together your design, tasks, and data and discuss why they are appropriate together.
        - You may find it helpful to include comparisons to other possible design choices and describe how your design choice was more effective than these alternatives.
        - Some examples of questions to consider are:
            - What makes this choice a good idiom for the task, domain, etc?
            - Why is this encoding approach better than other specific alternatives for this scenario?
            - Why did we decide to choose this color scheme over another?
        - Simply describing the marks and channels is not enough, you must justify why a design choice is reasonable for the intended task.
            - Example bad answer: We used a bar chart to represent all the items. [does not connect visual encoding to task]
            - Example bad answer: This view supports the task. [view not actually described, the way it supports the task not described]
            - Example good answer: The task was to compare a set of items, so I used bar charts since they support comparison well.

        - Simply describing the interaction is not enough, you must explain why an interaction helps with a task.
            - Example bad answer: We have a filter so that users can filter for the correct time. [no explanation of why a user would want to filter their data]
            - Example good answer: The filter function allows users to look at a specific range of time, allowing them to compare sentiment over a specific period of time.
    - Screenshots illustrating all aspects of your design are mandatory.
6. **Usage Scenario**
    - Include a usage scenario showing how your current visualization can be used during an interactive session. It should be written in a narrative style and include the specific context of usage and the tasks associated with that usage context, and a walkthrough of how the user can accomplish those tasks with your implemented visualization.
    - At minimum, you must update the scenario from your previous milestone by illustrating with actual screenshots of your system in action, rather than your original sketches. Make sure to provide enough illustrations of what your implemented system actually shows: this final version should typically have more illustrations than your M2 proposal writeup.
    - If you changed your design approach along the way, you should also update the writeup more substantially to a scenario that is different than what you originally envisioned in your proposal.
7. **Reflections**
    - Describe any ways that your project changed along the way, from your initial abstraction and design proposals, through your work in progress milestone, to your final product.
    - How have your visualization goals changed?
    - How have your technical goals changed?
    - In retrospect, how realistic was your original proposal in terms of what you now understand is technically possible in D3?
    - Was there anything you wanted to implement that you ultimately couldn't figure out how to do? If so, then what workarounds did you employ, or did you abandon your original idea?
    - If you were to make the project again from scratch (or any other interactive visualization), what would you do differently?
8. **Work Breakdown and Schedule**
    - Include a detailed table of all work completed for the project, split into components like in previous milestones.
    - This milestone must have a detailed table of all work completed. Each component should be a different row in your table. The columns of your table would be Who (which team member(s) did or will work on it), Estimated Hours (how many hours of time you estimate it will take), Estimated Date (the date by which you estimate it will be completed), Actual Hours (how many hours it

actually took), and Actual Date (when it actually got done). The point of this project management planning is to gain skills at estimating work times; you will not be penalized for discrepancies between estimates and actual.
    - Your table must include a summary that indicates the totals for each team member **and** also overall, for both estimated **and** actual hours used to date.
    - If there are any major discrepancies in the amount of work done between team members, explain the situation in detail.
9. **Credits**
    - Indicate any sources of inspiration, including any specific D3 code blocks that you consulted or built upon. Explain what changes you made and their magnitude (e.g. unchanged vs. minor tweaks vs. major functionality additions) for any code that you built upon.

Video

You must make a video that mirrors what you would show in a live demo. Instructions and tips for making the video are available in video-demo.pdf. Include a link to your video in the report.

Peer Review

You will be asked to assess the ratio of work done by each member of the team (including yourself). The common case is that everybody is doing roughly equal shares of the work. Sometimes some people are doing a little more or a little less. We particularly need to know if some people are doing much more work or much less work. Please provide additional comments in the free-form text field if you indicate any situation other than the work being equally split. Even if you have mutually agreed to a non-equal work breakdown as of an intermediate milestone (for example, a teammate is ill and plans to do more later), you should still report the actual work breakdown and explain the situation in the comments. We also welcome any comments regarding your experience with the course, overall.

The peer review will take place through a Qualtrics form that you fill out personally. At each milestone, report on the cumulative ratio across the entire term, not just the work since the previous milestone.

The link to the M4 peer review form is:
https://ubc.ca1.qualtrics.com/jfe/form/SV_cHLR1SzLYA9qE0S

You will be asked to fill it out after submitting the project.

Submission Checklist (for the Team)

1. Project repository tagged as `v1.0-final`.
2. Final project report uploaded to Gradescope as PDF.
3. Peer evaluations uploaded on Qualtrics.
4. Thumbnail image added to your repository.
5. Link to the demo video.