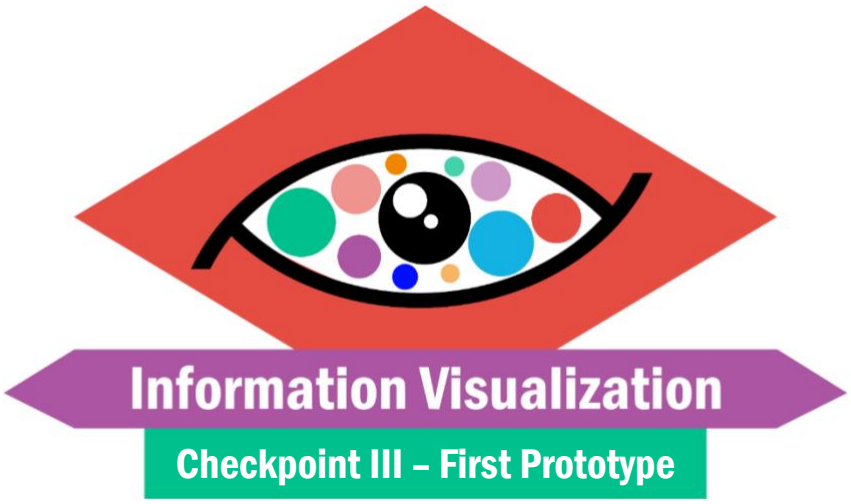

Objective

At the end of this checkpoint, you have a functional version of your visualization mainly consisting of creating the layout and implementing at least one of the idioms you have selected on Checkpoint II.

Requirements

Create your functional visualization prototype based on the feedback you received regarding your sketch (Checkpoint II). In this functional prototype, ***focus on the structure of the visualization interface***: make sure you organize the visual space into different areas according to each idiom you have previously selected. We also want to know the code structure, e.g., modules, you use to implement the prototype. ***No code in the report***! A simple flow chart showcasing how the data transfers from module to module in the prototype is enough.

Also, ***you must implement at least one of the idioms***, even if their look and feel still do not *perfectly* match the final aesthetics you have planned for your visualization. You must use the actual data you have gathered (Checkpoint I).

Make sure that your idioms are interactive. They need to provide the means for some interaction (e.g., a chord diagram must highlight relationships for items selected, a bar chart must show some behavior when moving over a bar, etc.). Static images are not going to get a good grade.

Keep in mind that **your views must be linked in meaningful ways**, so think about how the views share the data and how each view can inform the remaining interactions taking place (e.g., when you select an item in view X, how will you let views W and Z know so that they can choose the same item?). We want to see the **integration structure/function/package** that makes everything work together. You must implement an extensible prototype (i.e., when you have five idioms and not two, the mechanisms you implement here should still work to link all of them). If you only implemented one algorithm, you must use abstraction and proper programming skills to have a modular implementation to quickly receive the new visual idioms in the following checkpoints. Describe how your prototype integrates the visualizations. Again, ***no code***!

Also, ***the idioms need to look coherent*** (e.g., similar text fonts or colors). We want a cohesive system, not a patchwork quilt of visualization techniques. In addition, remember that even if you implement only one idiom, you must leave space for the others. So, where they will ultimately appear, **include a sketch image (you can take these from CPII) of the missing idioms**. Use the expected size/aspect ratio. In short, we expect to see the final prototype's layout.

Deliverables

Create a **3-page document using the provided template** and submit it online, **inside a zip file including both the document and the code, in Moodle,** until two days before your class (ex: classes on Monday must submit until Friday end of day).

The **document** should describe and illustrate:

- The **code architecture** you have created to load and (pre)process the data, to make the charts (and their interactivity), and to handle the integration.
- The **layout of your visualization**. Include at least one image of the complete interface (yes, even the idioms you have not implemented, in which case use screenshots from Checkpoint II).
- The **data (pre)processing** you must do before the data is ready to be loaded to the charts. We are not talking about the data processing from CPI but all the remaining process you have to do for each visual idiom.
- The idioms you have implemented (**chart visualization**) and their functionality (**chart interactivity**). You must include a description with images and explain the interactivity supported by those idioms.
- The **chart integration** techniques used and why adding more charts is easy. How are the views linked? How does that mechanism work/will it work even when you have more idioms to link?

If there are any changes from Checkpoints I and II, describe them and justify why you did those changes using the visualization principles learned in class.

The **code** of your prototype must **include the dataset**. You should deliver in a plug-and-play fashion: we run a local server, and the prototype is ready to be tested.

Name your deliverable as **'CPIII-<group number>.zip'.** For instance, 'CPIII-02.zip'. The zip should contain the report and your code. Make sure your PDF file follows the same name nomenclature.

Penalties

- Documents over 3 pages long: **1 grade point penalty per extra page**.
- PDF and ZIP names are different: **0.5 grade points penalty.**
- Incorrect file type or name: **0.5 grade points penalty per file or name**.
- Zip with document and code uploaded after the deadline: **0.5 grade points per hour of delay.**
- Document template altered (wider margins, smaller font, etc.): **1 grade point penalty**.
- No prototype: **10 grade points penalty**.

Tasks to perform during the lab

Show your working prototype to the professor. The professor will provide feedback. The grade will be made known one week later (see below). If you are not receiving feedback, then you must be peer assessing your colleagues' submissions.

Grading

Your work will be graded according to the following parameters:

- **Prototype architecture (15%)**: describe and illustrate the code architecture to produce the prototype.
- **Dashboard layout (15%)**: present and depict the work done so far according to your Checkpoint II.
- **Additional data processing (5%)**: describe your additional data processing techniques to achieve the result, e.g., `groupby` or `count` in runtime. If you do not have additional data processing, explain why.

- **Chart visualization (20%)**: show the actual visualization plotting real data, with a polished look-and-feel.
- **Chart interaction (20%)**: present the code modules used to achieve chart interaction.
- **Chart integration (20%)**: describe the integration architecture between charts.
- **Peer assessment (5%)**: grade your peers' submissions in the class.

An important note on grading for this and all other Checkpoints: always ***justify*** your choices, based on the basic principles you have learned in this course (adequacy of channels to data types, human perception, etc.). Don't just describe what it is, but especially ***why*** it is as it is.

Additional Notes

After you deliver your document, your work will be graded. HOWEVER, this grade **can be improved by up to two grade points** if you correct any faults pointed out by the professor and submit a revised version of the document HIGHLIGHTING THOSE CHANGES up the beginning of the class taking place 7 days after you receive feedback in class. ***Only highlighted changes will be considered.***