Luke Breitfeller
Final project

Part 1:

The purpose of this website is as a specialized tool for allowing trained human annotators to quickly organize instances in a text where events are mentioned into a chronological timeline, using an intuitive visual interface. I am currently working on a project to train computer models to read texts and automatically order the events they encounter in that text. A challenge in this task is annotating enough "orders" between events, since the size of the data to be annotated is enormous and most pairs of events in a text do not have an immediately intuitive "order" to human annotators. The aim with this specialized tool is to simplify the problem by making all the information easily accessible, visually intuitive, and reducing the size of the task to a single timeline. The tool is not designed to teach the concept of timeline annotation--the expected users already how to annotate a dataset--but the webpage expects users to be unfamiliar with the interface and includes instructions. The webpage is highly interactive, with an interface that is simple but (hopefully) aesthetically appealing. Even if you're not a trained annotator, it should be fun to drag and drop events while seeing the timeline change. The target audience is, however, trained annotators, who understand the idea of building a timeline out of events in a text and have used simpler annotation tools before.

Note that the tool uses visualization of data to help users through annotating a concept that we expect them to associate with visual imagery. I think that executing this concept for visually-impaired users (I tried to accommodate for colorblindness, though) might require more extensive accommodations than work for ordinary web use.

Part 2:

- Drag events from bucket. Events within the Past Bucket and Future Bucket can be dragged anywhere on the page.
    - Click event (from "Events in The Past" block or "Events in the Future" block only) and drag to see. Browser will change cursor to show event is being dragged.
- Display segments of timeline event should not go. Each event has a property "past" or "future". This property means an event should not, logically, be able to be placed on specific segments of the timeline. When event is clicked or dragged, the user will be able to see which segments are "bad".
    - Click or drag event to see which segments of the timeline are noted as "bad". This will also mark all the events within the opposite bucket (that is necessary for future features not included in this project).
- Drop events on timeline. Events can be dropped onto the area directly above the timeline. Webpage notes which segment of the timeline an event has been dropped on, which existing events that segment falls between, and will reload the timeline with the dropped event placed on the timeline in the order indicated by where it was dropped.

    - Drop event on area directly above black line of timeline to see. Browser will change cursor when event can be dropped from original position onto the timeline.
- Drop events on “bad” segment of timeline. An event can technically be dropped on a segment of the timeline that is labeled as “bad”. This will prompt an error message asking the user to confirm if the event should go in this segment. This allows users to override automated systems that may be wrong, but makes it more difficult for users to accidentally place an event in the wrong spot.
    - Drop event on area above the timeline marked by a red X.
    - Select “no” to see event will not be added to timeline.
    - Select “yes” to see event added to the timeline in that segment.
- Expand bucket. The buckets may contain more than three events, though that is what they will initially show. If you expand the bucket, you can see and select all of the events inside the bucket.
    - Click the “. . .” at the bottom of the bucket to expand.
    - Click the “X” at the top right of the expanded bucket to collapse.
    - With this demo, you will only see extra events if no events from the Future Bucket have been placed on the timeline. Clicking the expanding menu will show four events, one of which was not visible in the collapsed version.
- Automatic event generation. The script automatically generates the list of events in the past and future buckets based on the text within the textbox. Currently the ability to directly modify the text is limited, future iterations will allow users to link to a specific text file.
- Distinct highlights for sorted and unsorted spans. When an event is moved from the buckets to the timeline, the highlights in the text box will change from dark yellow to bright green.
    - Place an event on the timeline to see change in its corresponding highlight value.
- Automatic timeline reformatting. Calculations behind the scenes allow the timeline to consistently be re-generated as a series of increasingly small droppable segments, though to the user the black line of the timeline will appear not to change.
- View instructions. User can view instructions page. All work on timeline is saved when users move to instructions page.
    - Click “View instructions” link to access instructions page.
    - Click “Back to timeline” link to return to timeline work.
- Responsiveness. The tool I am building relies on the visual perception of the timeline in order to make annotation decisions clear and intuitive. As a result, making the tool responsive to every possible change in window size would be difficult to pull off and not benefit the end task of performing the timeline annotation. As a result, I designed the timeline tool to remain mostly static except for one large responsive element:
    - If the screen width is decreased below 750 pixels, the timeline will flip to a vertical configuration and the buckets/text will break into two rows. Every other element of functionality (dragging events, adding to timeline, viewing invalid timeline segments) remains functional in this configuration.

    - If the screen width is increased above 750 pixels, the timeline flips back to its default horizontal configuration.
- CURRENT KNOWN ISSUE: I realized somewhat late in the game that I had not planned for if the user gets to the error message but does not try to resolve the error (ie clicks elsewhere). I've done honestly a lot so far and it's an edge case resulting from the overlap of two complex web functions I did not have time to resolve/test with everything else. I know what the fix should look like and will implement that for my user launch, sorry I could not fix it before submission.

Part 3:

- HTML Drag-and-Drop API
- I used this API because the project requires users to metaphorically "move" elements between unsorted piles to specific spots on a timeline, and aligning that metaphorical task with actually "dragging" the element makes the task easy and intuitive for a user. Given how specific the commands to this interface must be, being able to drag and drop over a large area is also the most straightforward way, computationally, to execute this. Finally, being able to drag and drop reduces the odds of mode error from the user, since relying on clicking for each command type makes it easier for the user to forget a click and execute an unintended command.
- I used it for the task of "grabbing" unsorted events from a bucket and "moving" them onto the timeline. This is the task of my webpage which most relies on a concept of spatial orientation, and the most complicated task for a user to perform since it needs to be able to select any possible segment of the timeline.
- In addition to the above notes about human intuition, computation, and avoiding mode error, the ability to add events to the timeline is the central focus of this entire web project. If we could not add elements to a timeline, the timeline annotation would be worthless. Additionally, though, it is just fun and engaging to be able to drag and drop things on a webpage and see the webpage change.

Part 4:

My original scope for this project was going to include a few more advanced functions, like the ability to merge any two nodes together, add partial information like that one event is before another although their exact placement on the timeline is unknown, and remove events from a timeline if the user changes their mind. Based on feedback from talking with the TA, I chose to reduce the scope of my project, which meant elements like the arrow toolbar and ability to expand nodes distracted rather than added to the project and could thus be removed.

Part 5:

A challenge specific to my website is that I'm building a tool which necessarily requires a certain type of visual interaction to view and manipulate the timeline, so while responsiveness to window size was a requirement of this assignment, finding a visual layout that could capture all

the information needed to perform this kind of function at any possible size seemed impossible. I did come up with two layout which I think work, but truthfully the 1000+ pixel desktop size is best suited for this particular task.

Another challenge I dealt with was properly aligning the droppable timeline segments and the events themselves--because each event should be positioned on the edge between timeline segments, I had to experiment with a few different formulae to calculate the correct positions for timelines of different sizes before I found one that worked for any theoretical number of events on a timeline.