Test Design Document

To be used by the Testing Team (and Dev Team) in tandem with the Test Matrix to describe the design of tests.

How to Use This Document

Both the Test Matrix and the Test Design Document are living documents and should be updated regularly to reflect the current state of the project.

Template for Automated Tests

Test ID#:
Test Name:
Test Designer:
Test Location: *(e.g. Example.test.tsx)*
Associate Files: *(e.g. test.csv)*
Test Description:

- *What requirement does this test cover?*
- *What is the purpose of this test?*
- *Justification for why we need to have this test.*
- ***Input:***
- ***Output:***
- *Test Outline:*
    - *Step 1*
    - *Step 2*
    - *Etc. (Last step should outline expected behaviour)*
- *How is this test carried out (manual, automated, etc.)?*
- *Etc.*

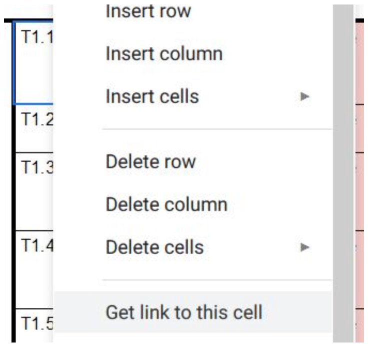
Template for ManualTests

Test Name:
Test ID#: ***(If applicable)***
Performed By:
Test Date:
Test Description:

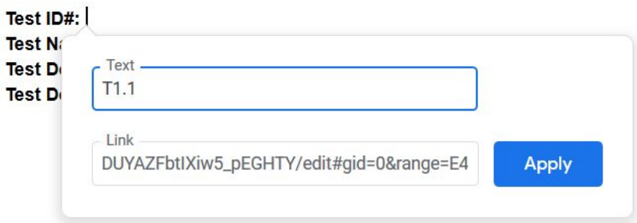
- *What requirement does this test cover?*
- *Purpose of the test*
- *Test Outline:*
    - ***Input***
        - *Break down how test was done into steps so that it can be reproduced by other members of the team*
            - *What OS, Browser, etc was used to perform this test?*
            - *What file was uploaded?*
    - ***Expected Behaviour/Output***
        - *What needs to happen for this test to pass*
    - ***Actual Behaviour/Output***
        - *What did it actually do (did it match the expected behaviour) and did it pass or fail?*
        - *Screenshots are encouraged to document unexpected behaviour*

Linking to Test Matrix

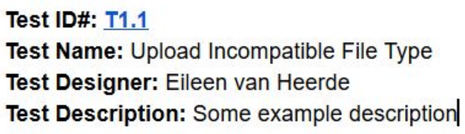
1. Go to Test Matrix.
2. Right Click on corresponding 'Test Case ID#'.



3. Select 'Get link to this cell'.
4. Return to location of test in the Test Design Doc.
5. Right Click beside 'Test ID#' field and select 'Link' (or use Ctrl+K).
6. Put the 'Test Case ID#' in the 'text' field and paste the link from the Test Matrix in the 'link' field. Press the 'Apply' button.



7. … And *Voila!* You have a link to the specific cell in the Test Matrix.



Test ID#: T1.1
Test Name: Upload Incompatible File Type
Test Designer: Eileen van Heerde
Test Description: Some example description

Test Files

Test File Layout

Each Reactjs component will have a corresponding test file (identified by the '.test.tsx' extension) located in the __*tests*__ folder within the *src* directory.

```
--'src'
   |
   |--'__tests__'
   |    |    Column.test.tsx
   |    |    Data.test.tsx
   |    |    Filter.test.tsx
   |    |    ParserComponent.test.tsx
   |    |    ParserInterface.test.tsx
   |    |    TimelineComponent.test.tsx
   |    |    TimelineInterface.test.tsx
   |    |
   |
   |--'components'
   |    |    Column.ts
   |    |    Data.ts
   |    |    Filter.ts
   |    |    ParserComponent.tsx
   |    |    ParserInterface.ts
   |    |    TimelineComponent.tsx
   |    |    TimelineInterface.ts
   |    |
```

Test Organization Within the Test Files

Within each test file, similar tests will be grouped together within `describe(){};` blocks where `it(){};` blocks will contain the actual test implementation.

```
describe('Rendering tests', () => {

        it('renders <ParserComponent /> that accepts CSV files', () => {
                ...
        });

        it('renders <ParserComponent /> that accepts TL files', () => {
                ...
        });

});
```

Automated Tests

R1 Tests

All .csv files mentioned in the following tests are created using the javascript File Object constructor.

Test ID#: T1.1

Test Name: Handling attempted upload of an incorrect file type
Test Designer: Clinton Essien
Test Location: ParserComponent.test.tsx
Test Description:

- This test ascertains that the file type convention is strictly followed
- The ParserComponent is only able to handle two file types (.csv and .tl). Any other file types may be parsed incorrectly or break the timeline and must not be allowed to be uploaded. We want to handle it before it is parsed.
- ***Input:*** file that is not a .csv, (.pdf, .txt, .doc, .js, etc)
- ***Output:*** ParserComponent with empty data value, & error is thrown
- *Test Outline:*
  - Render a ParserComponent that accepts .csv files, as well as another ParserComponent that accepts .tl files.
  - Multiple onChange input events will have to be simulated with different incompatible file types to make sure that they are not accepted by either of the rendered ParserComponents.
  - To pass the test, no file type other than a .csv or .tl must be accepted by the ParserComponents. Prompt the user a message indicating they have attempted to upload a wrong file type.
  - Fail if the user successfully uploads a file type that is not a .csv or .tl.

Test ID#: T1.2

Test Name: Handling a .csv file with incorrect formatting
Test Designer: Amanda Zimolag
Test Location: ParserComponent.test.tsx
Test Description:

- This test focuses on the correct handling of a .csv file with incorrect formatting
- In the future we may consider different methods of handling the .csv file depending on the error in formatting.
- ***Input:*** .csv file with incorrect formatting
  - Row with less data than # of headers
  - Row with more data than # of headers
  - Emojis in the data
- ***Output:*** ParserComponent with empty data value, & error is thrown
- *Test Outline:*
  - Render a ParserComponent that accepts .csv files

    - Simulate an onChange input event that passes in *incorrectFormatTest*.csv
    - To pass the test an error should be thrown and the user should be informed that their .csv file is formatted incorrectly, otherwise, the test should fail.

Test ID#: T1.3

Test Name: Handling .csv file with multiple date formats
Test Designer: Eileen van Heerde
Test Location: ParserComponent.test.tsx
Test Description:

- This test focuses on making sure that we handle the specific case of a user potentially uploading a .csv file with multiple/inconsistent date formats according to design specifications.
- ***Input:*** .csv file with multiple date formats
    - File where dates are sorted
    - File where dates are unsorted
        - Should test sorting by: day, month, year, & time
- ***Output:*** ParserComponent with updated data value
    - Expect unsorted data from csv to be sorted in ParserComponent data value
- *Test Outline*:
    - Render a ParserComponent that accepts .csv files
    - Simulate an onChange input event that passes in *multiDateTest.csv*
    - ParserComponent state should be updated so that the uploaded data is sorted by date and added to the 'data' value.

Test ID#: T1.4

Test Name: Handling .csv files with no temporal field
Test Designer: Eileen van Heerde
Test Location: ParserComponent.test.tsx
Test Description:

- This test checks that we properly handle .csv files uploaded with no Date/Time/Temporal data field.
- ***Input***: csv file with no date columns (only numbers & strings)
- ***Output:*** 'No dates
- *Test Outline*:
    - Render ParserComponent that accepts .csv files
    - Simulate an onChange input event that passes in *noDateTest.csv*
    - Error should be thrown and user should be informed that only temporal data can be uploaded.

Test ID#: T1.5

Test Name:upload csv file that has name with unusual chars in file name
Test Designer: Kevin Zhu

Test Location: ParserComponent.test.tsx
Test Description:

- When the the user upload a file with unusual characters such as \ or emojis, the file should still be accepted and data should be displayed normallynot a
- ***Input***: a csv file with unusual character name
- ***Output***: timeline updates the page and shows the chart of data
- *Test Outline:*
    - Render a ParserComponent that accepts .csv files.
    - Simulate an onChange input event that passed in ususualcharacter.*csv*.
    - Check the ParserComponent's data length to see if file has been successfully parsed
    - Manual test has also been done for this test case.

Test ID#: T1.6

Test Name: Upload .csv file with row of empty data
Test Designer: Eileen van Heerde
Test Location: ParserComponent.test.tsx
Test Description:

- When the the user uploads a .csv file with any empty rows, the parser component should accept the file, and take the empty rows and make a data object out of it, where the date_num is set to -1.
- ***Input***: a .csv file with empty rows
- ***Output:*** ParserComponent state is updated so that the empty row objects are at the front of the data array.
- *Test Outline*:
    - Render a ParserComponent that accepts .csv files
    - Simulate an onChange input event that passes in a csv with empty rows
    - Check the ParserComponent state & make sure that the empty row objects exist with a date_num set to -1.

Test ID#: T1.7

Test Name: Upload .csv file with empty values (not a complete row)
Test Designer: Eileen van Heerde
Test Location: ParserComponent.test.tsx
Test Description:

- When the the user uploads a .csv file with empty values, the parser component should accept the file, and take the rows with empty cells and make a data object out of it
- ***Input:*** a .csv file with empty values
- ***Output:*** parser component takes the rows with empty cells and makes a data object out of it when creating a new timeline component
- *Test Outline*:
    - Render a ParserComponent that accepts .csv files
    - Simulate an onChange input event that passes in emptyVal.csv
    - Check the ParserComponent and confirm that it acce

R2 Tests

Test ID#: T2.1

Test Name: Event Magnitude Data displayed correctly on Timeline using fixed data set
Test Designer: Eileen van Heerde
Test Location: TimelineComponent.test.tsx
Test Description:

- This test ensures that data (event magnitude) is displayed correctly and that it matches the snapshots in TimelineComponent.test.tsx.snap
- ***Input:*** Valid Data Object
- ***Output:*** svg file which matches the snapshots in TimelineComponent.test.tsx.snap
- *Test Outline:*
  - Mount TimelineComponent with Data object (have to set document.body.html() to Timeline Component html() so that d3 can actually generate html for us to test).
  - Call resetTimline() to initialize and draw the timeline.
  - Compare what was rendered to snapshots in TimelineComponent.test.tsx.snap.
  - To pass the test, what is rendered should match the snapshot. Otherwise it will fail.

Test ID#: T2.2

Test Name: Update/Rerender timeline when new data is added
Test Designer: Eileen van Heerde
Test Location: TimelineComponent.test.tsx (will need to import ParserComponent)
Test Description:

- The stakeholder specified that he wanted to be able to add more data to what is already uploaded and displayed on a timeline. This purpose of this test would be to make sure that the timeline is updated to reflect the newly added data.
- ***Input***: valid csv file
- ***Output***: Updated ParserComponent state (data) & correctly rendered Timeline component & d3 html
- *Test Outline:*
  - Render ParserComponent that accepts .csv files that already has data from *generalTest.csv* (we don't need to simulate an onChange event for the initial data).
  - Render a TimelineComponent that displays the data from the ParserComponent data.
  - Simulate an onChange input event to upload *addDataTest.csv*.
  - (Might have to simulate something for the TimelineComponent as well)
  - To pass the test, the ParserComponent's state must include the newly added data and the TimelineComponent must render the timeline again to reflect the changes to the data (will use a snapshot test).

Test ID#: T2.3

Test Name: Update/Rerender timeline when data is removed
Test Designer: Eileen van Heerde
Test Location: TimelineComponent.test.tsx (will need to import ParserComponent)
Test Description:

- Due to the user's ability to upload the data from multiple files, they might (for whatever reason) want to remove one of those files (and as a result the data associated with it). The goal of this test is to ensure that when a data file is removed that the timeline is updated to reflect the newly absent data.
- ***Input***: valid csv file
- ***Output***: Updated ParserComponent state (data) & correctly rendered Timeline component & d3 html
- *Test Outline:*
    - Render ParserComponent that accepts .csv files that already has *generalTest.csv* and *addDataTest.csv* 'uploaded'.
    - Render a TimelineComponent that draws the timeline of the associated data (Possibly compare to a 'before' snapshot).
    - Simulate an onChange event to remove *addDataTest.csv*.
    - To pass the test the TimelineComponent must render the timeline again to reflect the absence of the data caused by the removal of the .csv file (will use a snapshot test).

TEST ID#: T2.4

Test Name: Event Occurrence Data displayed correctly on Timeline using fixed data set
Test Designer: Eileen van Heerde
Test Location: TimelineComponent.test.tsx
Test Description:

- This test ensures that data (event occurrence) is displayed correctly and that it matches the snapshots in TimelineComponent.test.tsx.snap
- ***Input***: Valid Data Object
- ***Output***: svg file which matches the snapshots in TimelineComponent.test.tsx.snap
- *Test Outline:*
    - Mount TimelineComponent with Data object (have to set document.body.html() to Timeline Component html() so that d3 can actually generate html for us to test).
    - Call resetTimline() to initialize and draw the timeline.
    - Compare what was rendered to snapshots in TimelineComponent.test.tsx.snap.
    - To pass the test, what is rendered should match the snapshots. Otherwise it will fail.

TEST ID#: T2.6

Test Name: Interval Magnitude Data displayed correctly on Timeline using fixed data set

Test Designer: Amanda Zimolag
Test Location: TimelineComponent.test.tsx
Test Description

- This test ensures that (interval magnitude) data is displayed correctly and that it matches the snapshots in TimelineComponent.test.tsx.snap
- ***Input:*** Valid Data Object
- ***Output****:* svg file which matches the snapshots in TimelineComponent.test.tsx.snap
- *Test Outline*:
    - Mount TimelineComponent with Data object (have to set document.body.html() to Timeline Component html() so that d3 can actually generate html for us to test).
    - Simulate the user selecting ‘Interval Magnitude’ for the timeline type.
    - Simulate them choosing ‘Order Date’ for the starting range, ‘Ship Date’ for the ending range, ‘Unit Cost’ for the first y column.
    - Call resetTimline() to initialize and draw the timeline.
    - Compare what was rendered to snapshots in TimelineComponent.test.tsx.snap.
    - To pass the test, what is rendered in the svgtarget should match the snapshots. Otherwise it will fail.

TEST ID#: T2.7

Test Name: Interval Labelled Data displayed correctly on Timeline
Test Designer: Eileen van Heerde
Test Location: TimelineComponent.test.tsx
Test Description:

- This test ensures that (interval occurrence) data is displayed correctly and that it matches the snapshots in TimelineComponent.test.tsx.snap
- ***Input***: Valid Data Object (in this case a smaller version of 10000SalesRecords.csv)
- ***Output***: svg file which matches the snapshots in TimelineComponent.test.tsx.snap
- *Test Outline:*
    - Mount TimelineComponent with Data object (have to set document.body.html() to Timeline Component html() so that d3 can actually generate html for us to test).
    - Simulate the user selecting ‘Interval Labelled’ for the timeline type.
    - Simulate them choosing ‘Order Date’ for the starting range, ‘Ship Date’ for the ending range, ‘Region’ for the first y column, & ‘Order Priority’ for the 2nd y column.
    - Call resetTimline() to initialize and draw the timeline.
    - Compare what was rendered to snapshots in TimelineComponent.test.tsx.snap.
    - To pass the test, what is rendered in the svgtarget should match the snapshots. Otherwise it will fail.

Unit Tests

Unit Tests that verify that pre/post conditions are met should be written for all of the methods listed below. They should primarily be written by developers, but testers are also encouraged to write them. 80% test coverage is required before merging any new code to the Develop branch.

Manual Tests

Test Name: Update/Render Timeline when new data is uploaded to replace (NOT being added to) existing data

Test ID#: T2.5
Performed By: Amanda Zimolag
Test Date: December 4, 2019
Test Description:

- When the user uploads a new .csv to an existing timeline component, the user should be able to replace the existing data with the data belonging to the new .csv
- *Test Outline:*
  - **Input:**
    - Browser: Google Chrome
    - Tested on: https://dev.braunsen.me
    - The provided 1000 Sales Record.csv was uploaded as a base
    - A .csv based on address data was downloaded from this website: https://people.sc.fsu.edu/~jburkardt/data/csv/csv.html
  - **Expected Behaviour/Output**
    - The existing data from 1000 Sales Record.csv should be replaced with the data from addresses.csv
    - The behaviour should be consistent between filters
    - All functionality from the original data set should carry into the new data set
    - The new dataset should be successfully re-replaced by the original 1000 Sales Record.csv upon switching back
  - **Actual Behaviour/Output**
    - The data from the 1000 Sales Record.csv is replaced with the data from addresses.csv
    - Replacing the dataset works across filters
    - All functionality (including bugs) carries into the new data set
    - The new dataset is able to be re-replaced by the original data

Test Name: Event Labelled Timeline resizes correctly when user uses mousewheel to zoom in & out

Test ID#: T3.1
Test Performed By: Eileen van Heerde
Test Date: December 5th, 2019
Test Description:

- One of the ways in which the user is expected to interact with the timeline is using the mouse wheel to zoom in & out of the timeline. This test is intended to confirm whether or not the user is successfully able to execute this interaction.

- *Test Outline:*
    - **Input:**
        - Browser: Google Chrome & Firefox
        - Tests on: https://dev.braunsen.me
        - T3_1.csv (is in ‘Manual Test Csv Files’ in the google drive)
        - Scroll up to zoom in & scroll down to zoom out
    - **Expected Behaviour/Output:**
        - Expect data points to scale accordingly
        - Expect labels on the x axis to move accordingly (widen & shrink)
        - Expect movement to be smooth
    - **Actual Behaviour/Output:**
        - Unfortunately, the file was not parsed correctly on chrome or firefox & data was only put on one data point.
        - Zooming still worked to an extent but we are unable to confirm it for this timeline type.
        - The outcome of this test is a **failure** because timeline didn’t render correctly which prevents the user from using this feature effectively.

Test Name: Event Labelled Timeline resizes correctly when user uses ‘+’ & ‘-’ keys to zoom in & out

Test ID#: T3.2
Test Performed By: Eileen van Heerde
Test Date: December 5th, 2019
Test Description:

- One of the ways in which the user is expected to interact with the timeline is using the ‘+’ & ‘-’ keys to zoom in & out of the timeline. . This test is intended to confirm whether or not the user is successfully able to execute this interaction.
- *Test Outline:*
    - **Input:**
        - Browser: Google Chrome & Firefox
        - Tests on: https://dev.braunsen.me
        - T3_1.csv (is in ‘Manual Test Csv Files’ in the google drive)
        - ‘+’ to zoom in & ’-’ to zoom out
    - **Expected Behaviour/Output:**
        - Expect data points to scale accordingly
        - Expect labels on the x axis to move accordingly (widen & shrink)
        - Expect movement to be smooth
    - **Actual Behaviour/Output:**
        - Unfortunately, the file was not parsed correctly on chrome or firefox & data was only put on one data point.
        - Zooming still worked to an extent but we are unable to confirm it for this timeline type.
        - The outcome of this test is a **failure** because timeline didn’t render correctly which prevents the user from using this feature effectively.

Test Name: Event Labelled Timeline resizes correctly when using mousewheel to zoom in & out

Test ID#: T3.3
Test Performed By: Eileen van Heerde
Test Date: December 5th, 2019
Test Description:

- One of the ways in which the user is expected to interact with the timeline is using the mouse wheel to zoom in & out of the timeline. This test is intended to confirm whether or not the user is successfully able to execute this interaction.
- *Test Outline:*
  - **Input:**
    - Browser: Google Chrome & Firefox
    - Tests on: https://dev.braunsen.me
    - T3_1.csv (is in 'Manual Test Csv Files' in the google drive)
    - Scroll up to zoom in & scroll down to zoom out
  - **Expected Behaviour/Output:**
    - Expect data points to scale accordingly
    - Expect labels on the x axis to move accordingly (widen & shrink)
    - Expect movement to be smooth
  - **Actual Behaviour/Output:**
    - Unfortunately, the file was not parsed correctly on chrome or firefox & data was only put on one data point.
    - Zooming still worked to an extent but we are unable to confirm it for this timeline type.
    - The outcome of this test is a **failure** because timeline didn't render correctly which prevents the user from using this feature effectively.

Test Name: Event Magnitude Timeline resizes correctly when user uses '+' key to zoom in & '-' to zoom out

Test ID# T3.4
Test Performed By: Amanda Zimolag
Test Date: December 5th, 2019
Test Description:

- One of the ways in which the user is expected to interact with the timeline is using the keyboard zoom in & out of the timeline. This test is intended to confirm whether or not the user is successfully able to execute this interaction on an event magnitude timeline
- *Test Outline*
  - **Input**:
    - Browser: Google Chrome
    - Tested on: https://dev.braunsen.me
    - 1000 Sales.csv, addresses.csv, insurance.csv
    - Press '+' to zoom in; press '-' to zoom out
  - **Expected Behaviour/Output:**

        - Data points should scale appropriately in response to user handling
        - Expect labels on the x axis to move accordingly (widen & shrink)
        - Expect fluid transitions between states
        - Consistent behaviour between different filters for range
    - **Actual Behaviour/Output:**
        - Zooming into the data using the ‘+’ key only worked when simultaneously holding down the SHIFT key
        - Zooming out of the data using the ‘-’ key was successful
        - When scaling the data, the transitions were fluid and consistent
        - For certain filters, the behaviour was consistent; when using other filters, no data was displayed
        - This test **passed** in spite of conflicts with certain filters, as the feature was able to be tested effectively with other working filters

Test Name: Interval Labelled Timeline resizes correctly when user uses mousewheel to zoom in & out

Test ID# 3.5
Test Date: December 5th 2019
Test Performed By: Amanda Zimolag
Test Description:

- One of the ways which the user is expected to interact with the timeline is using the mouse wheel to zoom in & out of the timeline. This test is intended to confirm whether or not the user is successfully able to execute this interaction on an interval labelled timeline.
- *Test Outline*
    - **Input**:
        - Browser: Google Chrome
        - Tested on: https://dev.braunsen.me
        - 1000 Sales.csv, addresses.csv, insurance.csv
        - Scroll up to zoom in; scroll down to zoom out
    - **Expected Behaviour/Output**:
        - Data points should scale appropriately in response to user handling
        - Labels on x axis should move accordingly (widen and shrink)
        - Fluid transitions between states
        - Consistent behaviour between different filters defining range
    - **Actual Behaviour/Output**:
        - File was not able to correctly parse via Interval Magnitude or Interval Occurrence
        - Unable to confirm functionality based on Timeline type
        - This test is a **failure** because the Timeline was unable to render correctly which prevents this feature from being tested effectively

Interval Labelled Timeline resizes correctly when user uses '+' key to zoom & '-' to zoom out

Test ID#: T3.6
Test Performed By: Amanda Zimolag & Kevin
Test Date: December 5th, 2019
Test Description:

- One of the ways in which the user is expected to interact with the timeline is using the keyboard to zoom in and out of the timeline. This test is intended to confirm whether or not the user is successfully able to execute this action on an interval labelled timeline.
- *Test Outline*:
    - **Input**:
        - Browser: Google Chrome & Firefox
        - Tested on: https://dev.braunsen.me
        - 1000 Sales.csv, addresses.csv, insurance.csv,10000 Sales.csv
        - Use ‘+’ to zoom in; use ‘-’ to zoom out
    - **Expected Behaviour/Output**:
        - Data points should scale appropriately in response to user handling
        - Labels on x axis should move accordingly (widen and shrink)
        - Fluid transitions between states
        - Consistent behaviour between different filters defining range
    - Actual Behaviour/Output
        - File was not able to correctly parse via Interval Magnitude or Interval Occurrence
        - After the Zoom event, if panning the graph, the circles will disappear
        - Unable to confirm functionality based on Timeline type
        - This test is a **failure** because the Timeline was unable to render correctly which prevents this feature from being tested effectively

Test Name: Interval Magnitude Timeline resizes correctly when using mousewheel to zoom in & out

Test ID#: T3.7
Test Performed By: Eileen van Heerde
Test Date: December 5th, 2019
Test Description:

- One of the ways in which the user is expected to interact with the timeline is using the mouse wheel to zoom in & out of the timeline. This test is intended to confirm whether or not the user is successfully able to execute this interaction.
- *Test Outline:*
    - **Input:**
        - Browser: Google Chrome & Firefox
        - Tests on: https://dev.braunsen.me

        - T3_7.csv (is in ‘Manual Test Csv Files’ in the google drive)
        - Scroll up to zoom in & scroll down to zoom out
    - **Expected Behaviour/Output:**
        - Expect intervals scale accordingly
        - Expect labels on the x axis to move accordingly (widen & shrink)
        - Expect movement to be smooth
    - **Actual Behaviour/Output:**
        - Unfortunately, the file was not parsed correctly on chrome or firefox & data was only put on one data point.
        - Was not able to determine whether zooming was working from this file...
        - The outcome of this test is a **failure** because timeline didn’t render correctly which prevents the user from using this feature effectively.

Test Name: Interval Magnitude Timeline resizes correctly when user uses ‘+’ & ‘-’ keys to zoom in & out

Test ID#: T3.8
Test Performed By: Eileen van Heerde
Test Date: December 5th, 2019
Test Description:

- One of the ways in which the user is expected to interact with the timeline is using the ‘+’ &’-’ keys to zoom in & out of the timeline. This test is intended to confirm whether or not the user is successfully able to execute this interaction.
- *Test Outline:*
    - **Input:**
        - Browser: Google Chrome & Firefox
        - Tests on: https://dev.braunsen.me
        - T3_7.csv & 10000SalesRecords.csv (is in ‘Manual Test Csv Files’ in the google drive)
        - ‘+’ to zoom in & ’-’ to zoom out
    - **Expected Behaviour/Output:**
        - Expect data points to scale accordingly
        - Expect labels on the x axis to move accordingly (widen & shrink)
        - Expect movement to be smooth
        - Should zoom in & out at the middle of the timeline
    - **Actual Behaviour/Output:**
        - Unfortunately, the T3_7.csv was not parsed correctly on chrome or firefox & data was only put on one data point.
        - 10000SalesRecords.csv was parsed correctly but zooming was incredibly choppy & the timeline ‘panned’ while it was zooming, which is not ideal.
        - The outcome of this test is a **failure** because timeline didn’t render correctly which prevents the user from using this feature effectively.

Test Name: Event Labelled Timeline pans in response to click and drag event

Test ID#: T5.1

Test Performed By: Amanda Zimolag
Test Date: December 5th, 2019
Test Description:

- One of the ways in which the user is expected to interact with the timeline is using their mouse to click-and-drag and pan across the timeline. This test is intended to confirm whether or not the user is successfully able to execute this interaction on an event labelled timeline.
- *Test Outline*:
    - **Input**:
        - Browser: Google Chrome
        - Tests on: https://dev.braunsen.me
        - 1000 Sales Records.csv, addresses.csv, insurance.csv
        - Mouse Event
    - **Expected Behaviour/Input**:
        - Clicking and dragging with the mouse should scale across the x axis in either direction
        - Fluid transitions across Timeline
        - When panning across the Timeline, no visual artifacts (i.e. data summary) should show up on the user's screen
        - Behaviour should be consistent between filters
    - **Actual Behaviour/Output:**
        - User is able to scale across the x axis in either direction when clicking and dragging with their mouse
        - Transitions are fluid in response to handling
        - When panning across the timeline, no summary points appear but there are red highlights visible
        - When altering the filters for the behaviour is typically consistent, however certain filters cause the Timeline to malfunction
        - This test **passes** because the user is able to be tested effectively for the most part

Test Name: Event Labelled Timeline pans in response to arrow keys

Test ID# T5.2
Test Performed By: Amanda Zimolag
Test Date: December 5th, 2019
Test Description:

- One of the ways in which the user is expected to interact with the timeline is using their arrow keys to pan across the timeline. This test is intended to confirm whether or not the user is successfully able to execute this interaction on an event labelled timeline.
- *Test Outline*:
    - **Input**:
        - Browser: Google Chrome
        - Tests on: https://dev.braunsen.me
        - 1000 Sales Records.csv, addresses.csv, insurance.csv
        - Left arrow key to pan to the left; right arrow key to pan to the right
    - **Expected Behaviour/Input**:

        - Selecting and holding the left arrow key should cause the Timeline to pan to the left; selecting and holding the right arrow key should cause the Timeline to pan to the right
        - Fluid transitions across Timeline
        - When panning across the Timeline, no visual artifacts (i.e. data summary) should show up on the user's screen
        - Behaviour should be consistent between filters
    - **Actual Behaviour/Output**
        - User is able to scale across the x axis to the left when selecting/holding down the left arrow key
        - User is able to scale across the x axis to the right when selecting/holding down the right arrow key
        - Transitions are fluid in response to handling
        - When panning across the timeline, no summary points appear but there are red highlights visible
        - When altering the filters the behaviour causes the Timeline to malfunction
        - This test **fails** because the user cannot test it effectively within different settings

Test Name: Event Labelled Timeline pans correctly when zoomed in

Test ID# T5.3
Test Performed By: Amanda Zimolag
Test Date: December 5th, 2019
Test Description:

- One of the ways in which the user is expected to interact with the timeline is panning across the timeline when zoomed in. This test is intended to confirm whether or not the user is successfully able to execute this interaction on an event labelled timeline.
- **Test Outline**:
    - **Input**:
        - Browser: Google Chrome
        - Tests on: https://dev.braunsen.me
        - 1000 Sales Records.csv, addresses.csv, insurance.csv
        - Zoom in, then use both the arrow keys and mouse to pan left and right.
    - **Expected Behaviour/Input**:
        - Timeline stays zoomed in when panning.
        - Timeline moves in the correct direction depending on drag direction
        - X axis labels should move in the correct direction
        - Data should be rendered correctly when panning left and right
        - Behaviour should be consistent between filters
    - **Actual Behaviour/Output**
        - When zoomed in, both the arrow keys and clicking & dragging with the mouse work in both directions.
        - When switching between filters, the Timeline malfunctions
        - This test **fails** because the feature cannot be effectively tested under different conditions

Test Name: Event MagnitudeTimeline pans in response to click and drag event

Test ID#: T5.4
Test Performed By:
Test Date: December 5th, 2019
Test Description:

- One of the ways in which the user is expected to interact with the timeline is using their mouse to click-and-drag and pan across the timeline. This test is intended to confirm whether or not the user is successfully able to execute this interaction on an event magnitude timeline.
- *Test Outline*:
    - **Input**:
        - Browser: Google Chrome
        - Tested on: https://dev.braunsen.me
        - 1000 Sales Records.csv, addresses.csv, insurance.csv
        - Mouse Event
    - **Expected Behaviour/Input**
        - Clicking and dragging with the mouse should scale across the x axis in either direction
        - Fluid transitions across Timeline
        - When panning across the Timeline, no visual artifacts (i.e. data summary) should show up on the user's screen
        - Behaviour should be consistent between filters
    - **Actual Behaviour/Output**
        - User is able to scale across the x axis in either direction when clicking and dragging with their mouse
        - Transitions are fluid in response to handling
        - When panning across the timeline, no summary points appear but there are red highlights visible
        - When altering the filters for the behaviour is typically consistent, however certain filters cause the Timeline to malfunction
        - This test **passes** because the user is able to be tested effectively for the most part

Test Name: Event Magnitude Timeline pans in response to arrow keys

Test ID#: T5.5
Test Performed By:
Test Date: December 5th, 2019
Test Description:

- One of the ways in which the user is expected to interact with the timeline is using their keyboard to pan across the timeline. This test is intended to confirm whether or not the user is successfully able to execute this interaction on an event magnitude timeline.
- *Test Outline:*
    - **Input**

        - Browser: Google Chrome
        - Tests on: https://dev.braunsen.me
        - 1000 Sales Records.csv, addresses.csv, insurance.csv
        - Left arrow key to pan to the left; right arrow key to pan to the right
    - **Expected Behaviour/Input**
        - Selecting and holding the left arrow key should cause the Timeline to pan to the left; selecting and holding the right arrow key should cause the Timeline to pan to the right
        - Fluid transitions across Timeline
        - When panning across the Timeline, no visual artifacts (i.e. data summary) should show up on the user's screen
        - Behaviour should be consistent between filters
    - **Actual Behaviour/Output**
        - User is able to scale across the x axis to the left when selecting/holding down the left arrow key
        - User is able to scale across the x axis to the right when selecting/holding down the right arrow key
        - Transitions are fluid in response to handling
        - When panning across the timeline, no summary points appear but there are red highlights visible
        - When altering the filters the behaviour causes the Timeline to malfunction
        - This test **fails** because the user cannot test it effectively within different settings

Test Name: Event Magnitude Timeline pans correctly when zoomed in

Test ID#: T5.6
Test Performed By:
Test Date: December 5th, 2019
Test Description:

- One of the ways in which the user is expected to interact with the timeline is panning across the timeline when zoomed in. This test is intended to confirm whether or not the user is successfully able to execute this interaction on an event magnitude timeline.
- *Test Outline:*
    - **Input:**
        - Browser: Google Chrome
        - Tests on: https://dev.braunsen.me
        - 1000 Sales Records.csv, addresses.csv, insurance.csv
        - Zoom in, then use both the arrow keys and mouse to pan left and right.
    - **Expected Behaviour/Input**
        - Timeline stays zoomed in when panning.
        - Timeline moves in the correct direction depending on drag direction
        - X axis labels should move in the correct direction
        - Data should be rendered correctly when panning left and right
        - Behaviour should be consistent between filters
    - **Actual Behaviour/Output**

        - When zoomed in, both the arrow keys and clicking & dragging with the mouse work in both directions.
        - When switching between filters, when clicking and dragging the behaviour is consistent
        - When switching between filters, using the arrow keys is a hit or miss and may or may not cause the Timeline to malfunction
        - This test **passes** because the functionality is confirmed and can be effectively tested under (most) conditions

Test Name: Interval Labelled Timeline pans in response to click and drag event

Test ID#: T5.7
Test Performed By: Eileen van Heered
Test Date: December 5th, 2019
Test Description:

- One of the ways in which the user is expected to interact with the timeline is using their mouse to click-and-drag and pan across the timeline. This test is intended to confirm whether or not the user is successfully able to execute this interaction on an interval labelled timeline.
- *Test Outline:*
    - **Input:**
        - Browser: Google Chrome
        - Tested on: https://dev.braunsen.me
        - 1000 Sales.csv
        - Click and drag left to pan right, click and drag right to pan left
    - **Expected Behaviour/Input**
        - Timeline moves in the correct direction depending on drag direction
        - X axis labels should move in the correct direction
        - Data should be rendered correctly when panning left and right
    - **Actual Behaviour/Output**
        - When panning left data is not rendered quickly.
        - Panning to the right works as expected.
        - Although panning left is a little choppy, this test still **passes** because it is still functional.

Test Name: Interval Labelled Timeline pans in response to arrow keys

Test ID#: T5.8
Test Performed By:
Test Date: December 5th, 2019
Test Description:

- One of the ways in which the user is expected to interact with the timeline is using their keyboard to pan across the timeline. This test is intended to confirm whether or not the user is successfully able to execute this interaction on an interval magnitude timeline.
- *Test Outline:*
    - **Input:**

        - Browser: Google Chrome
        - Tested on: https://dev.braunsen.me
        - 1000 Sales.csv
        - Press right arrow key to pan right, press left arrow key to pan left
    - **Expected Behaviour/Input**
        - Timeline moves in the correct direction depending on which arrow key is pressed
        - X axis labels should move in the correct direction
        - Data should be rendered correctly when panning left and right
    - **Actual Behaviour/Output**
        - Both directions pan properly, except when a drop down is selected, in which case the timeline glitches out (either pans infinitely or twitches from left to right). Because of the latter, this test **fails.**

Test Name: Interval Labelled Timeline pans when zoomed in

[Test ID#: T5.9]()
Test Performed By: Eileen van Heerde
Test Date: December 5th, 2019
Test Description:

- One of the ways in which the user is expected to interact with the timeline is using their keyboard and mouse to pan across the timeline. This test is intended to confirm whether or not the user is successfully able to execute this interaction on an interval magnitude timeline.
- *Test Outline:*
    - **Input:**
        - Browser: Google Chrome
        - Tested on: https://dev.braunsen.me
        - 1000 Sales.csv
        - Zoom in, then use both the arrow keys and mouse to pan left and right.
    - **Expected Behaviour/Input**
        - Timeline stays zoomed in when panning.
        - Timeline moves in the correct direction depending on input
        - X axis labels should move in the correct direction
        - Data should be rendered correctly when panning left and right
    - **Actual Behaviour/Output**
        - When zoomed in, both the arrow keys and clicking & dragging with the mouse work in both directions. However, panning left is still a little choppy compared to panning right. Again, when a drop down is selected and the arrow keys are pressed, the timeline pans infinitely and the application becomes unresponsive and must be refreshed. It also zooms the timeline back out, this means that the test **fails.**

Test Name: Interval Magnitude Timeline pans in response to click and drag event

Test ID#: T5.10
Test Performed By: Eileen van Heerde
Test Date: December 5th, 2019
Test Description:

- One of the ways in which the user is expected to interact with the timeline is using their keyboard and mouse to pan across the timeline. This test is intended to confirm whether or not the user is successfully able to execute this interaction on an interval magnitude timeline.
- *Test Outline:*
  - **Input:**
    - Browser: Google Chrome
    - Tested on: https://dev.braunsen.me
    - 1000 Sales.csv
    - Click and drag left to pan right, click and drag right to pan left
  - **Expected Behaviour/Input**
    - Timeline moves in the correct direction depending on drag direction
    - X axis labels should move in the correct direction
    - Data should be rendered correctly when panning left and right
  - **Actual Behaviour/Output**
    - Clicking and dragging on the Interval Magnitude timeline successfully pans the timeline in both directions. However, panning left is still a little choppy compared to panning right. This test **passes** because the feature functions without breaking the application & works sufficiently.

Test Name: Interval Magnitude Timeline pans in response to arrow keys

Test ID#: T5.11
Test Performed By: Eileen van Heerde
Test Date: December 5th, 2019
Test Description:

- One of the ways in which the user is expected to interact with the timeline is using their keyboard (arrow keys) to pan across the timeline. This test is intended to confirm whether or not the user is successfully able to execute this interaction on an interval magnitude timeline.
- *Test Outline:*
  - **Input:**
    - Browser: Google Chrome
    - Tested on: https://dev.braunsen.me
    - 1000 Sales.csv
    - Press right arrow key to pan right, press left arrow key to pan left
  - **Expected Behaviour/Input**
    - Timeline moves in the correct direction depending on which arrow key is pressed

        - X axis labels should move in the correct direction
        - Data should be rendered correctly when panning left and right
    - **Actual Behaviour/Output**
        - Both directions pan properly, except when a drop down is selected, in which case the timeline glitches out (either pans infinitely or twitches from left to right). Because of the latter, this test **fails.**

Test Name: Interval Magnitude Timeline pans when zoomed in

Test ID#: T5.12
Test Performed By:Kevin Zhu
Test Date: December 5th, 2019
Test Description:

- One of the ways in which the user is expected to interact with the timeline is using their keyboard and mouse to pan across the timeline when zoomed in. This test is intended to confirm whether or not the user is successfully able to execute this interaction on an interval magnitude timeline.
- *Test Outline:*
    - **Input:**
        - Browser: Firefox
        - Tested on: https://dev.braunsen.me
        - 10000 Sales.csv, Event8.csv
        - Press right arrow key to pan right, press left arrow key to pan left
    - **Expected Behaviour/Input**
        - Timeline moves in the correct direction depending on which arrow key is pressed and mouse drag event
        - X axis labels should move in the correct direction
        - Data should be rendered correctly when panning left and right
    - **Actual Behaviour/Output**
        - When panning with mouse after keyboard zoom in, the graph's scale has been resetted

Test Name: Hovering over data point on occurrence timeline gives a summary of the point

Test ID#: T17.1
Test Performed By: Kevin Zhu
Test Date: November 30, 2019
Test Description:

- The user should be able to hover over data points on an occurrence timeline and retrieve a summary for the necessary information.

- *Test Outline:*
  - **Input:**
    - Browser used: Google Chrome
    - Test performed on https://test.braunson.me/
    - The provided 1000 Sales Record.csv was used, and can be substituted with any compatible .csv
    - The steps taken were: uploading the .csv file and hovering over the individual points
  - **Expected Behaviour/Input:**
    - When hovering over an individual data point in an event occurrence timeline, a summary should appear
    - The summary should correspond to the data point in question
    - When no longer hovering over the data point no visual artifact should be visible
    - The behaviour should be consistent between different filters
  - **Actual Behaviour/Output**
    - When hovering over a data point in an event occurrence timeline, a summary is displayed
    - No visible data points exist in an interval occurence timeline
    - The summarized data in an event occurrence timeline corresponds to the correct data point
    - When cursor no longer hovers over a data point, no visual artifact is visible
    - The behaviour does not appear to be consistent between filters; when changing the filters, no data points are displayed

Test Name: Hovering over data representation on interval timeline gives summary of that interval

Test ID#: T17.2
Test Performed By: Amanda Zimolag
Test Date: November 30, 2019
Test Description:

- The user should be able to hover over data representation on an interval timeline and retrieve a summary for the necessary information.
- *Test Outline:*
  - **Input*:***
    - Browser used: Google Chrome
    - Test performed on https://test.braunson.me/
    - The provided 1000 Sales Record.csv was used, and can be substituted with any compatible .csv
    - The steps taken were: uploading the .csv file and hovering over the represented data on an interval based timeline
  - **Expected Behaviour/Output:**

        - When hovering over represented data in an interval based timeline, a summary representing the necessary information should appear
        - The summary should correspond to the correct interval
        - When no longer hovering over an interval no visual artifact should be visible
        - The behaviour should be consistent between different filters
    - Actual Behaviour/Output
        - No visible data exists for an interval magnitude timeline
        - No visible data for an interval occurence timeline
        - Unable to handle different filters

Test Name: Error handling allows user feedback

Test ID#: TI.1
Performed By: Braunson Mazoka
Test Date: November 26, 2019
Test Description:

- In the event of an unrecoverable error, the user should be presented the error screen and the option to write feedback. If you do not have an error to test with, you can create one simply by creating any bug in the render() method of any of the components!
- *Test Outline:*
    - **Input:**
        - By creating an error in the ParserComponent, navigate to the homepage.
    - **Expected Behaviour/Output:**
        - The homepage should show the error page, with an error description and the button to provide feedback.
    - **Actual Behaviour/Output**
        - Correct expected behaviour.

Test Name: Telemetry is prompted when no cookies exists

Test ID#: TI.2
Performed By: Braunson Mazoka
Test Date: November 26, 2019
Test Description:

- In the event that the user’s browser does not have any cookies set, a prompt appears on the homepage to opt-out to telemetry data if the user chooses to.
- *Test Outline:*
    - **Input:**
        - Ensure you have cleared your cookies for the webpage
        - Navigate to the homepage
    - **Expected Behaviour/Output:**

    - The telemetry prompt should appear, selecting ‘OK’ or the ‘Close’ button should cause the prompt to disappear and reloading the page no longer shows the prompt.
    - Or: The telemetry prompt should appear, selecting ‘Disable Telemetry’ button should cause the prompt to disappear and reloading the page no longer shows the prompt.
    - Check that the cookie ‘Telemetry’ was added with the value ‘True’ if you closed the prompt or clicked ‘OK’ or ‘False’ if you clicked ‘Disable Telemetry’
  - **Actual Behaviour/Output**
    - Correct expected behaviour.

Deprecated Manual Tests

Test Name: Use mousewheel to resize timeline (zoom in)

Test ID#: T3.1
Performed By: Amanda Zimolag
Test Date: November 12, 2019
Test Description:

- One of the ways in which the user is expected to be able to interact with the timeline is by using the mouse wheel. When handling the mouse wheel the user should be able to zoom in on the data, and allow greater visual space between occurrences across the entire timeline. The purpose of this test is to confirm whether or not the user is able to successfully carry out this interaction.
- *Test Outline:*
  - **Input:**
    - Browser: Google Chrome
    - Tested on: https://dev.braunsen.me
    - The provided 10000 Sales Record .csv was used, which may be substituted with any other compatible .csvs
    - The exact steps which were taken were uploading the .csv file and using the wheel to zoom in on the data and see whether or not it executes the required output
  -
  - **Expected Behaviour/Output**
    - The user should be able to zoom in on the data when scrolling “up” with the mousewheel.
    - The user should be presented with an increase in visual space between occurrences.
    - There should be a max value that limits how far in the user can zoom
    - The behaviour should be consistent between filters
  - **Actual Behaviour/Output**
    - When scrolling “up” the user able to zoom in on the data and perceive an increase in visual space between occurrences across different filters

        - The user is presented with an increase in visual space between occurrences
        - There is a max value that limits how far the
        - Strange behaviour when using the event occurrence filter
        - Strange behaviour when using the interval occurence filter

Test Name: Use mousewheel to resize timeline (zoom out)

TestID#: T3.2
Performed By: Amanda Zimolag
Test Date: November 12, 2019
Test Description:

- One of the ways in which the user is expected to be able to interact with the timeline is by using the mousewheel. When handling the mousewheel the user should be able to zoom out from the data, and revert it to a previous state. The purpose of this test is to confirm whether or not the user is able to successfully carry out this interaction.
- *Test Outline:*
    - **Input:**
        - Browser: Google Chrome
        - Tested on: https://dev.braunsen.me
        - The provided 10000 Sales Record .csv was used, which may be substituted with any other compatible .csvs
        - The exact steps which were taken were uploading the .csv file, zooming in data using a working feature, and then promptly zooming out of the data to see whether or not it meets the criteria for a passing test
    - **Expected Behaviour/Output**
        - The user should be able to zoom out of the timeline when scrolling “down” with the mousewheel.
        - The user should be able to revert to a previous state and perceive more occurences and less visual space.
        - There should be a minimum value which limits how far out a user can zoom
        - The behaviour should be consistent between filters
    - **Actual Behaviour/Output**
        - When scrolling “down” the user is able to successfully zoom out of the timeline and perceive less of a visual distance between occurrences.
        - There is a fixed limit as to how far out the user is able to zoom.

Test Name: Use keyboard to resize timeline (zoom in)

Test ID#: T3.3
Performed By: Amanda Zimolag

Test Date: November 12, 2019
Test Description:

- One of the ways in which the user is expected to be able to interact with the timeline is by using the keyboard. When handling the '+' key, the user should be able to zoom in on the data, and allow greater visual space between occurrences across the entire timeline. The purpose of this test is to confirm whether or not the user is able to successfully carry out this interaction
- *Test Outline:*
    - **Input:**
        - Browser: Google Chrome
        - Tested on: https://dev.braunsen.me
        - The provided 10000 Sales Record .csv was used, which may be substituted with any other compatible .csvs
        - The exact steps which were taken were uploading the .csv file, and using the '+' key to see whether or not it meets the criteria for a passing test
    -
    - **Expected Behaviour/Output**
        - The user should be able to zoom in on the data when pressing the '+' key on their keyboard.
        - The user should be able to zoom in continuously on the data when holding down the '+' key
        - The user should be presented with an increase in visual space between occurrences.
        - There should be a max value that limits how far in the user can zoom
        - The behaviour should be consistent between filters
    - **Actual Behaviour/Output**
        - When holding down the 'SHIFT' key and pressing the '+' key the user can zoom in on the data but not when pressing the '+' exclusively
        - When holding down the 'SHIFT' key and holding down the '+' key, the user is able to continuously zoom in on the data but not when holding down the '+' key exclusively
        - There is a fixed limit as to how far the user is able to zoom in
        - The behaviour is not consistent between filters

Test Name: Use keyboard to resize timeline (zoom out)

Test ID#: T3.4
Performed By: Amanda Zimolag
Test Date: November 12, 2019
Test Description:

- One of the ways in which the user is expected to be able to interact with the timeline is by using the keyboard. When handling the '-' key, the user should be able to zoom out from the data, and revert it to a previous state. The purpose of this test is to confirm whether or not the user is able to successfully carry out this interaction.

- *Test Outline:*
    - **Input:**
        - Browser: Google Chrome
        - Tested on: https://dev.braunsen.me
        - The provided 10000 Sales Record .csv was used, which may be substituted with any other compatible .csvs
        - The exact steps which were taken were uploading the .csv file, zooming in on the data using a working feature, and using the '-' key to see whether or not it meets the criteria for a passing test
    - **Expected Behaviour/Output**
        - The user should be able to zoom out of the timeline when pressing the '-' key on their keyboard.
        - The user should be able to revert to a previous state and perceive more occurences and less visual space.
        - There should be a minimum value which limits how far out a user can zoom
    - **Actual Behaviour/Output**
        - When pressing down on the '-' key the user is able to successfully zoom out of the timeline and perceive less of a visual distance between occurrences, but not when holding down the 'SHIFT' key
        - When holding down the '-' key the user is able to zoom out continuously, but not when holding down the 'SHIFT' key
        - There is a fixed limit as to how far out the user can zoom
        - The behaviour is not consistent between filters

Test Name: Timeline pans in response to mouse click-and-drag events

Test ID#: T5.1
Performed By:Kevin Zhu
Test Date: November 2, 2019
Test Description:

- The user should be able to see no visual artifacts when dragging through the timeline(ghosting of labels and ghosting pop ups when put mouse on a variable on chart)
- The user should be able to drag data freely without terrible lag.
- The user should be able to see zooming when double click the chart
- The user should be able to see updates when buttons and drop down list are clicked
- Input:
- Test Outline:
    - Expected Behaviour/Output:
        - Fluid response when dragging, zooming timeline
        - No visual artifacts
        - Visual Updates correctly when pressed button/ selected drop down list

Actual Behaviour/Output:

        - When dragging and place mouse on data, the pop-up info shows multiple pop-up boxes if dragged quickly.
            - It also shows blinking red boxes when selected
        - Chart sometimes will not update when a filter is switched
        - When big dataset is present with development console open on browser, the performance will be slow.
    - No visual artifacts
    - Consistent behaviour between filters
- **Actual behaviour/output:**
    - Interface shows a fluid and consistent response to trackpad motions in either direction
    - When dragging over data points no visual artifact displaying a summary is presented to the user
    - Response is consistent between filters
    - When starting trackpad motion begins on a data point, the visual artifact continues to be displayed
    - Trackpad panning does not cover the entirety of large data that is represented

Test Name: Timeline pans in response to button events (left, right)

Test ID#: T5.2
Performed By: Amanda Zimolag
Test Date: November 24, 2019
Test Description:

- One of the ways in which the user is expected to interact with the timeline is using the keyboard to pan across the timeline. The user is expected to be able to use the left and right keys to ‘walk along’ the timeline and view earlier or later points in time. This test is intended to confirm whether or not the user is successfully able to execute this interaction.
- *Test Outline:*
    - **Input**:
        - Browser: Google Chrome
        - Tested on: https://dev.braunsen.me
        - The provided 10000 Sales Record .csv was used, which may be substituted with any other compatible .csv
        - The exact steps which were taken were: uploading the .csv file, and using the left and right arrow keys to see whether or not the interaction meets the criteria for a passing test. The motions were conducted under different settings and filters for the purpose of consistency
    - **Expected behaviour/output**:
        - Fluid response to keyboard events
        - Selecting the left key pans to the left incrementally
        - Holding down the left key pans to the left continuously
        - Selecting the right key pans to the right incrementally
        - Holding down the right key pans to the right continuously

        - No visual artifacts
        - Consistent behaviour between filters
    - **Actual behaviour/output**:
        - Incremental panning to the left when pressing the left key
        - Continuous panning to the left when holding down the left key
        - Incremental panning to the right when pressing the right key
        - Continuous panning to the right when holding down the right key
        - Fluid response to all keyboard events
        - Occasional visual artifacts displayed during continuous panning
        - When panning over data points a visual artifact is displayed
        - When altering the filter settings, interface demonstrates an infinite zoom
        - Behaviour is not consistent between filters

Test Name: Timeline pans in response to click-and-drag events when zoomed in

Test ID#: T5.3
Performed By: Amanda Zimolag
Test Date: November 23, 2019
Test Description:

- The user is expected to interact with the timeline from up-close and afar. The user is expected to be able to use click-and-drag to walk along the timeline and view earlier or later points in time. This test was designed with the intention of confirming whether or not the user is able to successfully execute this interaction.
- *Test Outline:*
    - **Input:**
        - Browser: Google Chrome
        - Test performed on: https://dev.braunsen.me
        - The provided 10000 Sales Record .csv was used, which may be substituted with any other compatible .csvs
        - The exact steps which were taken were uploading the .csv file, zooming into the data, and using a click-and-drag event to see whether or not it meets the criteria for a passing test. The motions were conducted under different settings and filters for the purpose of consistency
    - **Expected Behaviour/Output:**
        - Fluid response to click-and-drag events when zoomed in via mouse wheel
        - Fluid response to click-and-drag events when zoomed in via keyboard
        - Clicking and dragging to the left causes display to pan to the right
        - Clicking and dragging to the right causes display to pan to the left
        - No visual artifacts
        - Consistent behaviour between filters
    - **Actual Behaviour/Output:**

            - Interface appears to demonstrate a consistent and somewhat fluid response to click-and-drag motions in response to user handling while zoomed
            - When dragging over data points, although no data summary is displayed, the point in question is highlighted in red
            - When initiating a click-and-drag motion over a highlighted data point, the summary stays present when panning in either direction
            - Response appears to be consistent between filters

Test Name: Timeline pans in response to button events (left, right) when zoomed in

Test ID#: T5.4
Performed By: Amanda Zimolag
Test Date: November 24, 2019
Test Description:

- The user is expected to interact with the timeline from up-close and afar. The user is expected to be able to use the left and right keys to walk along the timeline and view earlier or later points in time. This test was designed with the intention of confirming whether or not the user is able to successfully execute this interaction.
- *Test Outline:*
    - **Input**:
        - Browser: Google Chrome
        - Test performed on: https://dev.braunsen.me
        - The provided 10000 Sales Record .csv was used, which may be substituted with any other compatible .csv
        - The exact steps which were taken were: uploading the .csv file, zooming into the data, and using the left and right arrow keys to see whether or not the interaction meets the criteria for a passing test. The motions were conducted under different settings and filters for the purpose of consistency
    - **Expected behaviour/output**:
        - Fluid response to keyboard events when zoomed in via mouse wheel
        - Fluid response to keyboard events when zoomed in via keyboard
        - Selecting the left key pans to the left incrementally
        - Holding down the left key pans to the left continuously
        - Selecting the right key pans to the right incrementally
        - Holding down the right key pans to the right continuously
        - No visual artifacts
        - Consistent behaviour between filters
    - **Actual behaviour/output**:
        - Incremental panning to the left when pressing the left key
        - Continuous panning to the left when holding down the left key
        - Incremental panning to the right when pressing the right key

- Continuous panning to the right when holding down the right key
- Fluid response to all keyboard events
- Occasional visual artifacts displayed during continuous panning
- When panning over data points a visual artifact is displayed
- When altering the filter settings, interface demonstrates an infinite zoom
- Behaviour is not consistent between filters