Audit Trail

Manhattan Associates

Kathryn Atkins, Logan Gunnin, Brittney Ochoa, Nil Vallabh

IS Capstone

Group 4

Table of Contents

[1 - Executive Summary 3](#_rzjk8i66wkzz)

[1.1 - Project Overview 3](#_9ohdtzrk0b9z)

[1.2 - Objective 3](#_ppq97h3iba5b)

[1.3 - Key Features 3](#_8c40wx3dctdo)

[1.5 - Conclusion 3](#_m3ops0oy80y7)

[2 - Requirements (Milestone 1) 4](#_v8s84agom9co)

[2.1 - Overview 4](#_ag3rhmhi2o4r)

[2.2 - Features 4](#_aflgo0wqwkyl)

[2.5 - Functionality 5](#_9ww4pxny75cc)

[2.6 - Records 5](#_232dy9u9u4y1)

[3 - Technical Design (Milestone 2) 6](#_erko8amr37he)

[3.1 - Overview 6](#_xzzp95f7hay1)

[3.2 - Class Diagram 6](#_ec3gw8x264ux)

[3.3 - Flow Diagram 7](#_fbpvfh70eqg3)

[3.4 - Filter & Sort 7](#_gho6tf6zcap)

[3.4.1 - Filter 7](#_wmo7m6k8r3jh)

[3.4.2 - Sort 7](#_cutx0m33y8j0)

[3.5 - User Interface General Screen 8](#_55xuznhu8as)

[4 - Presentation (Milestone 3) 8](#_9k5kilg1a6hc)

[5 - Test Script (Milestone 4) 9](#_dyuja4wd9mge)

[5.1 - Test Questions/Statements 9](#_rwem4yf4h2k)

[5.2 - Test Script Summary 9](#_v7nsb49beg6r)

[6 - Test Results (Milestone 5) 9](#_l79a82dbir81)

[7 - Conclusion 10](#_hjohrcff9wwr)

# 1 - Executive Summary

## 1.1 - Project Overview

This project, Audit Trail, is for our client Manhattan Associates. This project is focused on developing UI screens that display an entity audit trail with details within the Manhattan Active Omni (MAO) system. MAO models domain-related objects into data objects known as Entities, which are both transactional and config data.

## 1.2 - Objective

The objective of this project is to provide audit trail information through the parsing of JSON data and present it in a user-friendly UI. It will only include 10 rows per page and the ability to filter the data as liked. The UI is the Audit Trail Viewer (ATV), which shows the parsed data from the JSONS and makes them into a user-friendly table.

## 1.3 - Key Features

* User-Friendly Format: UI screens presenting the JSON data in an accessible, easy-to-read format using Angular
* JSON Parsing: REST requests fetch JSON responses from the specified endpoint and then parse these responses to their necessary columns, providing the necessary information in their correct column
* Filtering, Sorting, Pagination:
  + Filtering: Allows users to limit records based on selected filters
  + Sorting: Enable sorting based on columns
  + Pagination: Display a limit number (10) per page
* Export Data to CSV, Excel, Copy, or PDF:
  + Users can download the data by CSV, Excel, or PDF or they can simply copy the information and paste it where they desire

1.4 - Team

Our team number for this project is group 4. The following members are in this team:

1. Kathryn Atkins (Team Lead)
2. Logan Gunnin
3. Brittney Ochoa
4. Nil Vallabh

## 1.5 - Conclusion

For our client Manhattan Associates, the project Audit Trail parses JSON data into correct columns and is displayed through an Audit Trail Viewer (ATV) UI, which is user-friendly and easy to read. Only 10 rows are provided per page and filtering and sorting are provided, as well as the ability to export the data through copy, excel, csv, or pdf. The impact that this audit trail will provide to MA is it will significantly enhance the efficiency of MA’s workflow, as well as make the ability to view JSON data more efficiently and effectively.

# 2 - Requirements (Milestone 1)

## 2.1 - Overview

Audits can leave logs, information, and much more in order to recall what happened and what could’ve potentially happened in situations where systems, etc fail. Manhattan Associates, the client, is asking for an entity audit trail with details included. With a UI interface, the audit trail (AT for short) should be able to parse and present the information in a list format.

## 2.2 - Features

The following features **must** be implemented:

● Create and send a REST request to fetch JSON response

● Parse the REST response to read the audit information

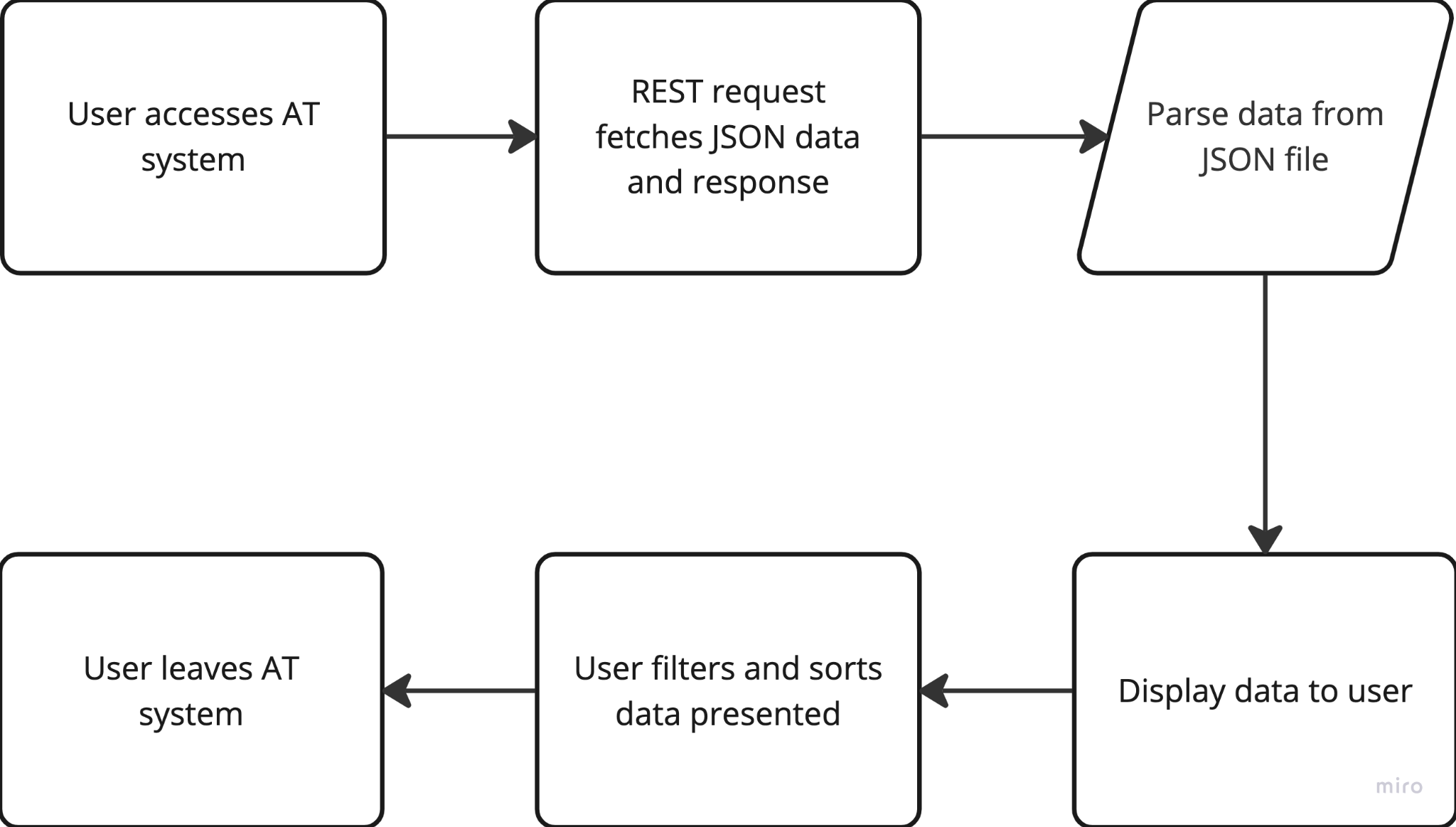
● Display attributes in easy-to-read format

● Filtering capabilities, ability to filter through search

● Ability to sort records through the arrow next to the column name

● Pagination capabilities to display no more than 10 records per page

2.3 - How It Works



2.4 - Assumptions

* Parsing will display the attributes from the JSON file
* Any changes in the JSON file directly will show upon the next load up

## 2.5 - Functionality

* Scenario 1 - Filtering will be primarily done through Search.
  + Instead of providing time limits and multiple different, confusing ways to filter the data, a search bar is simple. Anything that matches what is searched will be displayed, all else will be hidden.
* Scenario 2 - No data will be displayed at all if it can’t be found
  + If no data is found upon filtering, no data will just not show. No errors, etc will be provided and just nothing will show up instead to let the user know that no data has been found for whatever is filtered.

When data is found, the data is shown in the table. Filtering can be done for all columns, for example: UserId, UpdatedBy, etc. Data that is NULL in rows that do have data will show as empty. The user will also be able to sort the data alphabetically A-Z. This is done through jQuery imports.

For a user to filter, they will use the search bar and for the user to sort, they can press the arrow button right next to the column names, like UserID, etc. Sorting works for all columns.

## 2.6 - Records

| Entity | Description |
| --- | --- |
| UserId | User Id. Can be in email or UserId. Holds all information below |
| Updated By | User who updated the account. Can be shown in email or UserId |
| Updated Timestamp | When it was updated. Format: YYYY-MM-DD: HH-MM-SS |
| Created Timestamp | When it was created. Format: YYYY-MM-DD: HH-MM-SS |
| Created By | User who created the account. Can be shown in email or UserId |
| Cp-Trace-Id | A single request or job that triggered in the application. |
| Context Id | Uniquely identifies an SNMP entity that recognizes an instance of a context with a particular context name |
| Password | Password of the User. Hidden |
| Password Expiry | When the password expires. Can be NULL. |
| Old Values | Old Value before any changes happened |
| PK | Primary Key |
| Primary Org Id | Organization primary ID. |
| Image Url | Images URL. Hidden |

# 3 - Technical Design (Milestone 2)

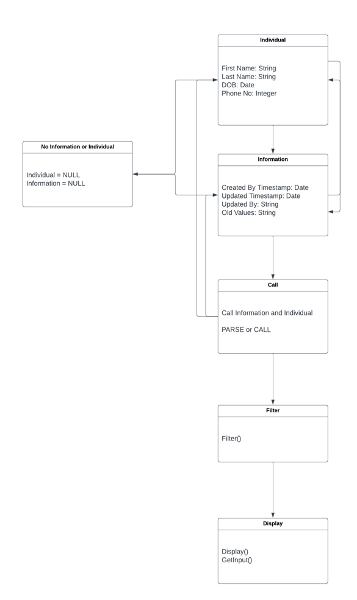
## 3.1 - Overview

Users will be using the program as a way to view audit information. The records are in a JSON file and will be fetched through a REST request. The JSON file is provided by Manhattan Associates. The link is below:

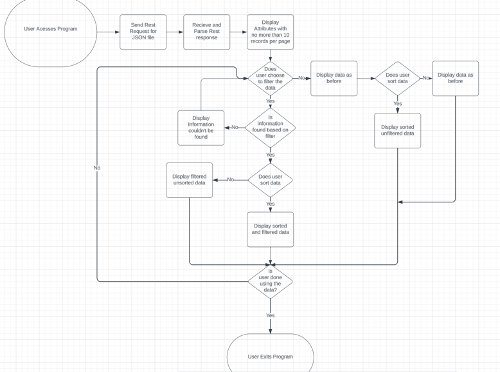
[Manhattan Associates JSON File](https://run.mocky.io/v3/ba12d051-8f60-40c4-a9bd-6b2478d3b6ca)

The REST request will fetch this JSON file/link and its data and display it in a UI user-friendly format.

## 3.2 - Class Diagram



## 3.3 - Flow Diagram



## 3.4 - Filter & Sort

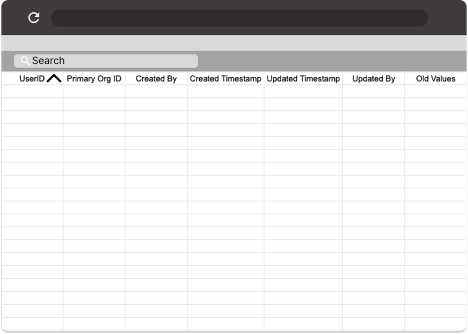
### 3.4.1 - Filter

The columns, or attributes, will filter through a search bar. For example: ahil. If a user puts a, ah, ahi, or ahil in the search bar the userId ahil will appear, as well as other data that has the letter a in it. The search bar is in the upper-right corner saying Search and providing the user an area to search.

### 3.4.2 - Sort

If a user wants to sort by A-Z or Z-A, the user will press the arrow button (Δ) to sort through the data.

## 3.5 - User Interface General Screen



Using the JSON file/link below, the information should parse into the correct column:

[Manhattan Associates JSON File](https://run.mocky.io/v3/ba12d051-8f60-40c4-a9bd-6b2478d3b6ca)

# 4 - Presentation (Milestone 3)

Below is a presentation of the general UI and the code

[Presentation](https://kennesawedu-my.sharepoint.com/:v:/g/personal/katkin27_students_kennesaw_edu/ET8pnsVebKlAlEe0mdn3nB4BRgd1sKgoHcKsr7DTxK3XCw?e=3Fc0Rl)

# 5 - Test Script (Milestone 4)

Project: Audit Trail Viewer (ATV)

Authors: Kathryn Atkins, Logan Gunnin, Brittney Ochoa, Nil Vallabh

Functional Area: Auditing

Extensions: Filtering, Sorting, and Exporting

Test Area: /Users/kathrynatkins/Downloads/CapstoneProject-main/src/index.html

## 5.1 - Test Questions/Statements

1. Filter for a UserId “ahil”
   * Was there any confusion at any point?
2. Sort A-Z on column UpdatedBy
   * What was the first name/UserId/Email that appeared?
3. Export to a PDF
   * Was it easy to locate?

## 5.2 - Test Script Summary

* Filter for UserId “ahil”
* Sort A-Z on column UpdatedBy
* Export to a PDF

# 6 - Test Results (Milestone 5)

1. Filter for UserId “ahil”
   * Was there **any confusion at any point?**
     + **No.** It was generally easy. Use the Search bar to filter. Passed
2. Sort A-Z on column UpdatedBy
   * What was the **first name** that appeared?
     + **“Emal” Passed**
3. Export to a PDF
   * Was it **easy to locate**?
     + **Yes**. It is in the upper-left corner. Press the button and it downloads it. Passed

# 7 - Conclusion

Our client, Manhattan Associates, wants an audit trail of the JSON file they have presented to us. Using Angular, jQuery, TypeScript, HTML, and CSS this project was brought together. Teammates Kathryn Atkins (Team Lead), Logan Gunnin, Brittney Ochoa, and Nil Vallabh created an Audit Trail Viewer (ATV) that parses data from the JSON file into a user-friendly UI. This UI provided the ability to the users of filtering, sorting, and pagination of 10 rows per page.

This ATV will provide Manhattan Associates with a more efficient and effective overall workflow.

Test results concluded the program worked efficiently and had no issues that came up.

User Documentation: [User Documentation](https://docs.google.com/document/d/1sLJttwsfJ2Sva2LQzr6xKwm2VmUvRskvkYTtMdM9vK0/edit?usp=sharing)