TheatreProfile

Company Dashboard

Run

Official Documentation

August 17, 2017

Tempus Analytics, LLC

Table of Contents

Summary

The Theatre Profile Company Dashboard is the first step in utilizing ticketing and other data to help provide interested parties with informative data about live theatre. Interested parties can be theatre venue owners & operators, theatre company owners, live theatre production companies, etc. The data analysis process is three steps: descriptive analytics, prescriptive analytics, and predictive analytics. The Theatre Profile Company Dashboard is a descriptive analytic solution.

The Dashboard is designed to take ticketing data collected and present it to the user in an informative and intuitive way. The user will then have the ability to understand the data and begin to use it to guide their decision-making when evaluating what actions to take with their business.

Company-wide data is provided first to give the user an overview of their company's performance and to begin to inform them about where changes may be needed. This data also shows what the company's most financially successful shows were.

The user may then review individual shows to determine how successful they were with that show. To achieve this, data from all other companies that ran the show is aggregated and averaged to give the user a clear understanding of how their company's metrics compare to their peers. This peer data can act as a benchmark of sorts for the user, giving them context through which they can review the numbers they are seeing. The user may also review the events of the show to understand what days and times were most popular as well as determine the geographic location from which their greatest sales originated.

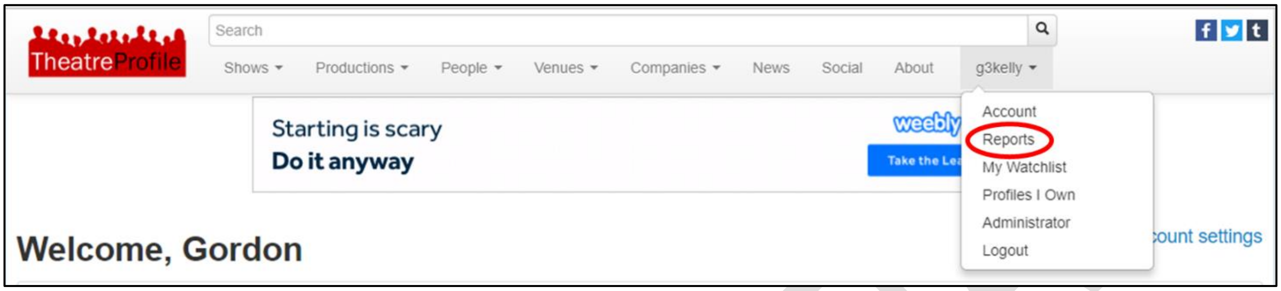
Finally, the user may narrow their results to a certain time period by using the data range function of the Dashboard. This allows the user to evaluate their success month-to-month, year-to-year, season-to-season, etc. NOTE: Peer data is NOT affected by the chosen date range. In order to maximize the number of peer results, peer data is compiled based on all available data, not just data within the specified dates.

Future dashboards may be built based on this framework. Such dashboards can focus on other forms of descriptive analytics, or may begin to delve into prescriptive and predictive analytics.

Dashboard Functionality

1. Accessing the Reports / Dashboard

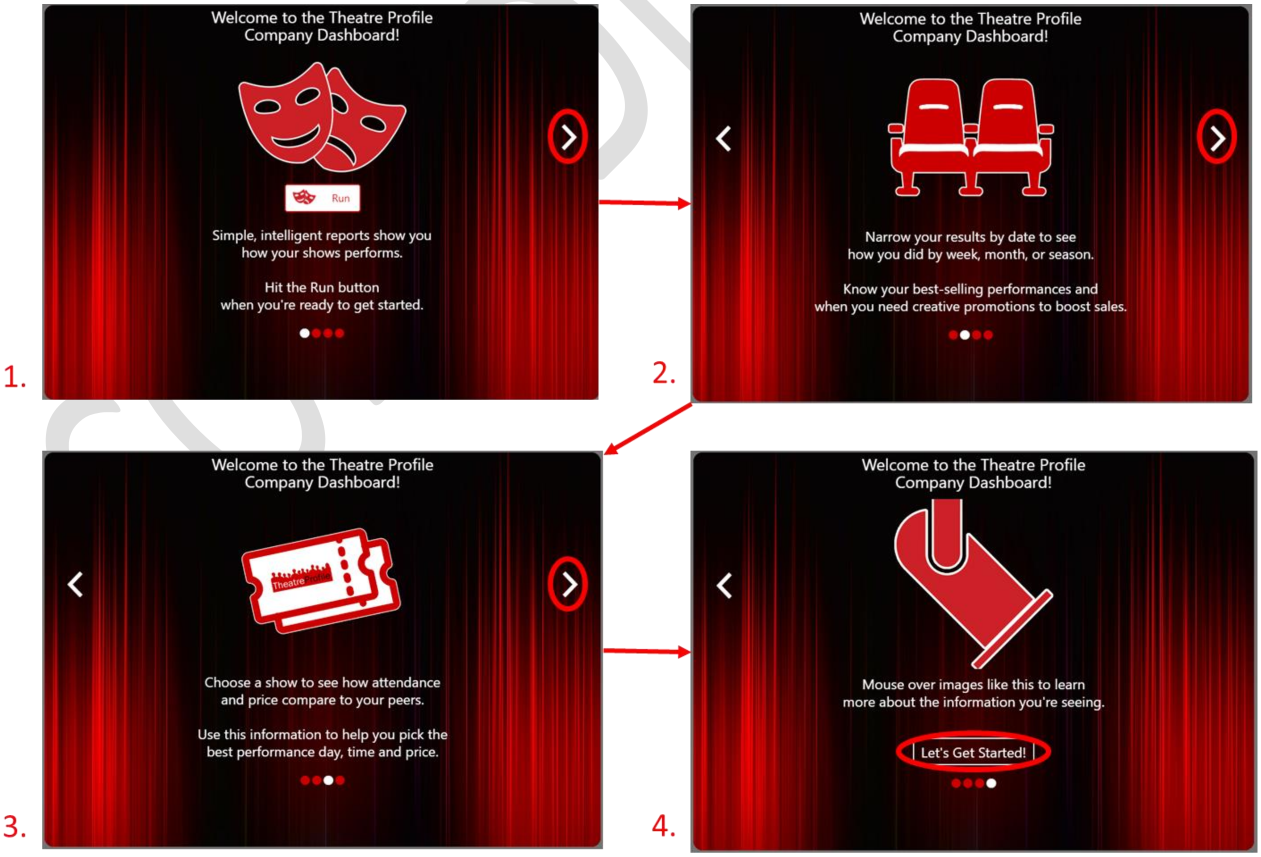
After logging into theatreprofile.com, the user selects their username from the menu bar and clicks on the Reports item. This launches the dashboard.



Welcome, Gordon

2. Dashboard Onboarding Process

When the dashboard opens, the user is automatically entered into the onboarding modal. This modal is a simple and quick 4-page introduction to the purpose and functionality of the dashboard. To close the onboarding, the user must click the forward arrow to proceed through each page. On the fourth page, the user will click the "Let's Get Started!" button the close the onboarding modal. The user may also close the onboarding by clicking outside of the onboarding area.



1. Welcome to the Theatre Profile Company Dashboard!

Simple, intelligent reports show you how your shows performs.

Hit the Run button when you're ready to get started.

2. Welcome to the Theatre Profile Company Dashboard!

Narrow your results by date to see how you did by week, month, or season.

Know your best-selling performances and when you need creative promotions to boost sales.

3. Welcome to the Theatre Profile Company Dashboard!

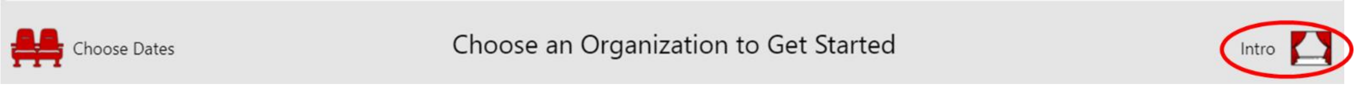
Choose a show to see how attendance and price compare to your peers.

Use this information to help you pick the best performance day, time and price.

4. Welcome to the Theatre Profile Company Dashboard!

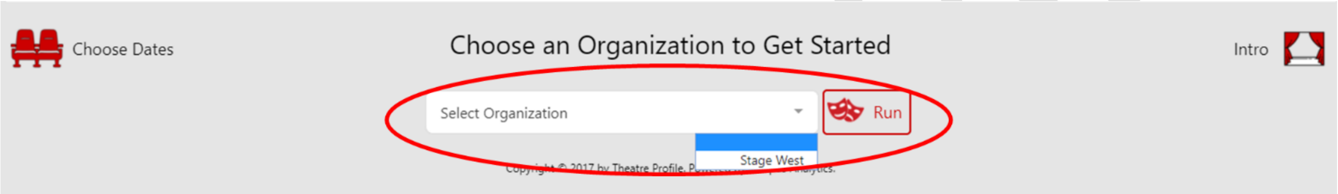
Mouse over images like this to learn more about the information you're seeing.

While the user is advancing through the onboarding, the list of companies that they own is being populated in the background. The user may access this onboarding modal in the future by clicking on the "Intro" button on the dashboard.



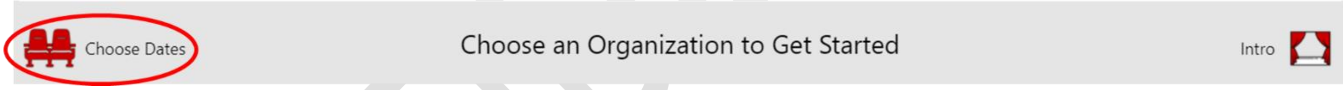
3. Selecting an Organization to View Reports

The user is prompted to "Choose an Organization to Get Started". In the Select Organization dropdown list, the user will have the ability to select from all of the companies that they have claimed ownership of on TheatreProfile.com. Once the user has selected an organization, they must then click Run to populate the first reports.

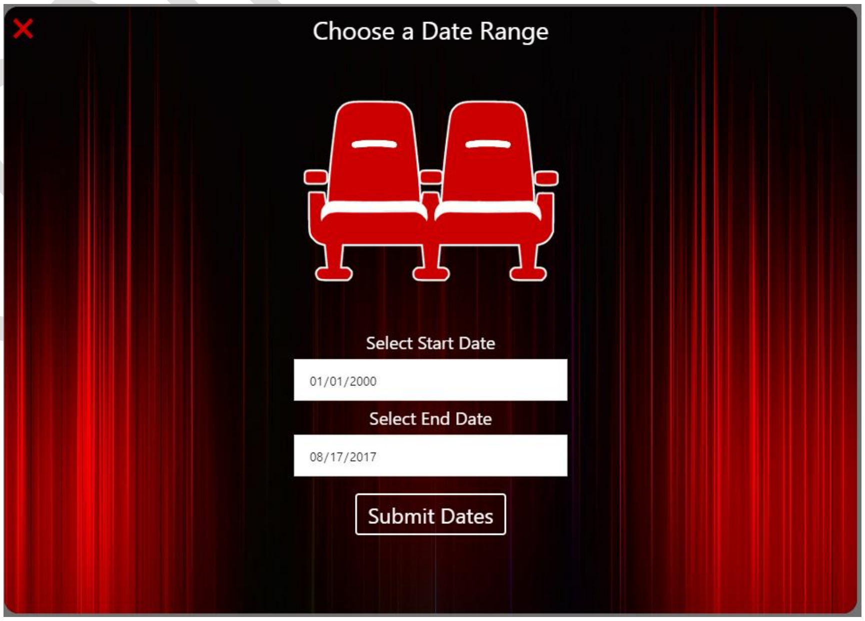

Choose Dates
Choose an Organization to Get Started
Intro
Select Organization
Run
Stage West
Copyright © 2017 by Theatre Profile, P

4. Narrowing Results using a Date Range

The user may choose to narrow the date range of their reports. To do this, they must click on the "Choose Dates" button on the top left of the dashboard.



Clicking the Choose Dates button will open a modal in which the user may input the date range that they wish to view. Default dates are January 1, 2000 (start date) and today's date (end date).

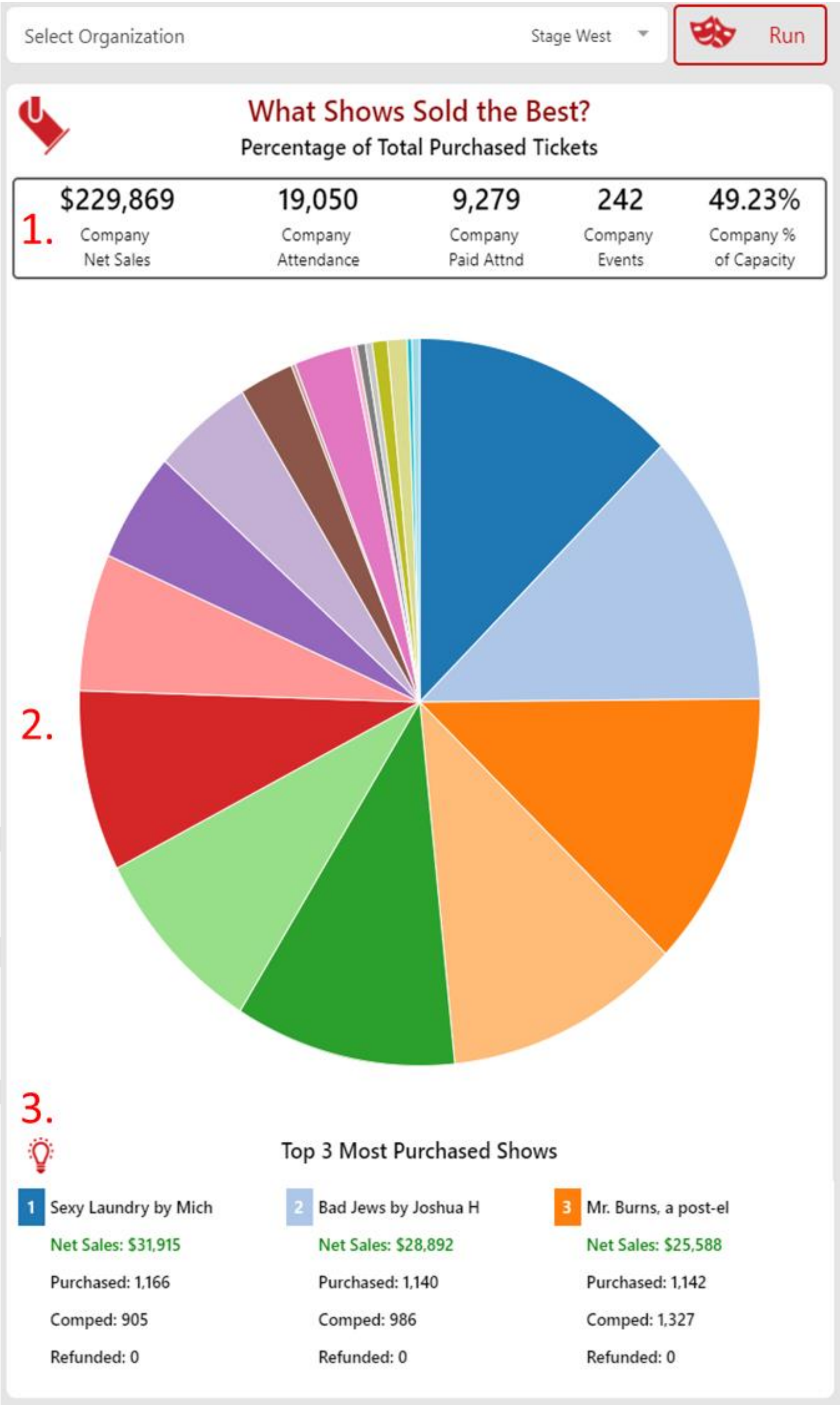


Select Start Date: 01/01/2000

Select End Date: 08/17/2017

5. Reporting for Company-Wide Data

The first data populated is the company-wide data. The data at the top (1) shows some key company stats. The pie chart (2) shows the Percentage of Total Purchased Tickets for the company for the chosen time frame. All shows that ran during that time period are included in the chart. At the bottom of the section (3), the user will see the top 3 most purchased shows during that time period as well as the net sales from those shows.



What Shows Sold the Best?

Percentage of Total Purchased Tickets

$229,869	19,050	9,279	242	49.23%
Company Net Sales	Company Attendance	Company Paid Attnd	Company Events	Company % of Capacity

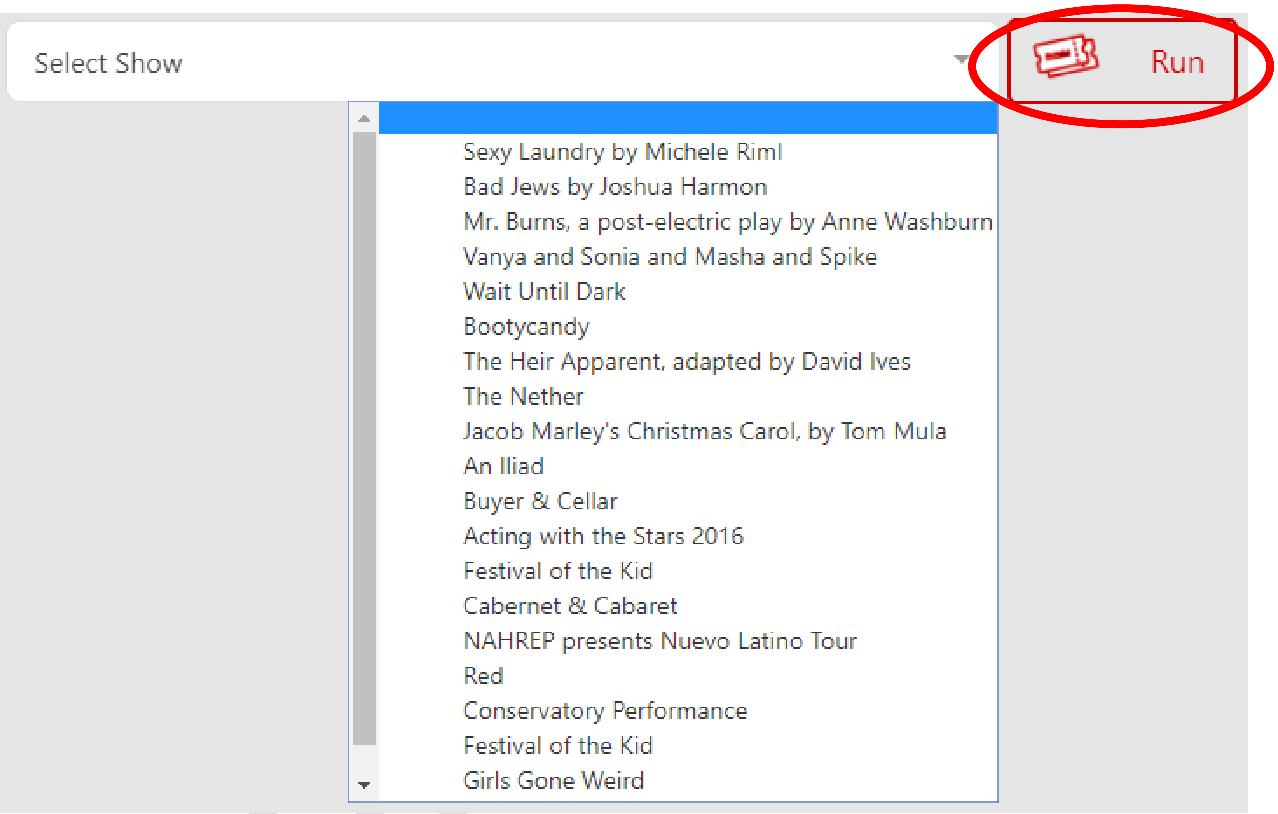
Top 3 Most Purchased Shows

1 Sexy Laundry by Mich	2 Bad Jews by Joshua H	3 Mr. Burns, a post-el
Net Sales: $31,915	Net Sales: $28,892	Net Sales: $25,588
Purchased: 1,166	Purchased: 1,140	Purchased: 1,142
Comped: 905	Comped: 986	Comped: 1,327
Refunded: 0	Refunded: 0	Refunded: 0

At any time, the user may mouse over (or click on if using mobile) the stage light or light bulb icon for more information about what they are seeing.

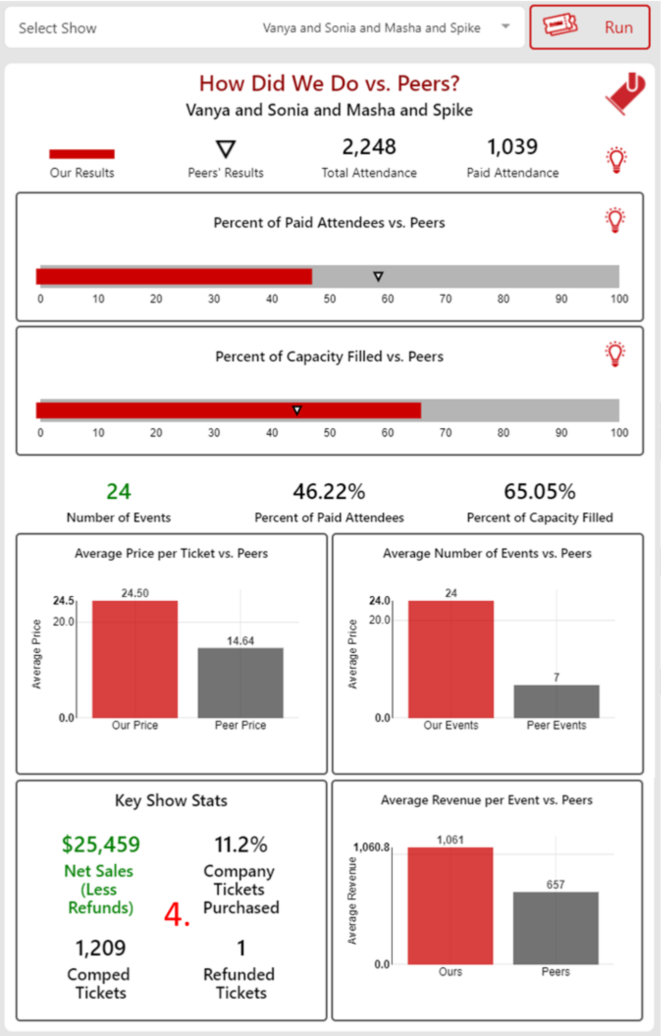
6. Reporting for Specific Shows

The user may then choose a show to view data specific to that show. After selecting a show name, the user must click Run to populate the report.


Select Show
Run
Sexy Laundry by Michele Riml
Bad Jews by Joshua Harmon
Mr. Burns, a post-electric play by Anne Washburn
Vanya and Sonia and Masha and Spike
Wait Until Dark
Bootycandy
The Heir Apparent, adapted by David Ives
The Nether
Jacob Marley's Christmas Carol, by Tom Mula
An Iliad
Buyer & Cellar
Acting with the Stars 2016
Festival of the Kid
Cabernet & Cabaret
NAHREP presents Nuevo Latino Tour
Red
Conservatory Performance
Festival of the Kid
Girls Gone Weird

The show reports display the following data:

1. The show's total attendance vs. paid attendance
2. Bullet charts comparing the user's company's run of the show to peers (other companies that had run the show)
3. Additional bar charts comparing the user's company's run of the show to peers.
4. Key stats for the show (the same stats that the user sees for the top 3 shows in the first section)
5. Historical bar chart showing the show's attendance by date and time during the chosen time period
6. Average attendance for the show by day of the week, time of day, and zip codes.



Select Show: Vanya and Sonia and Masha and Spike

Run

How Did We Do vs. Peers?

Vanya and Sonia and Masha and Spike

1.

🟥	▽	2,248	1,039
Our Results	Peers' Results	Total Attendance	Paid Attendance

2.

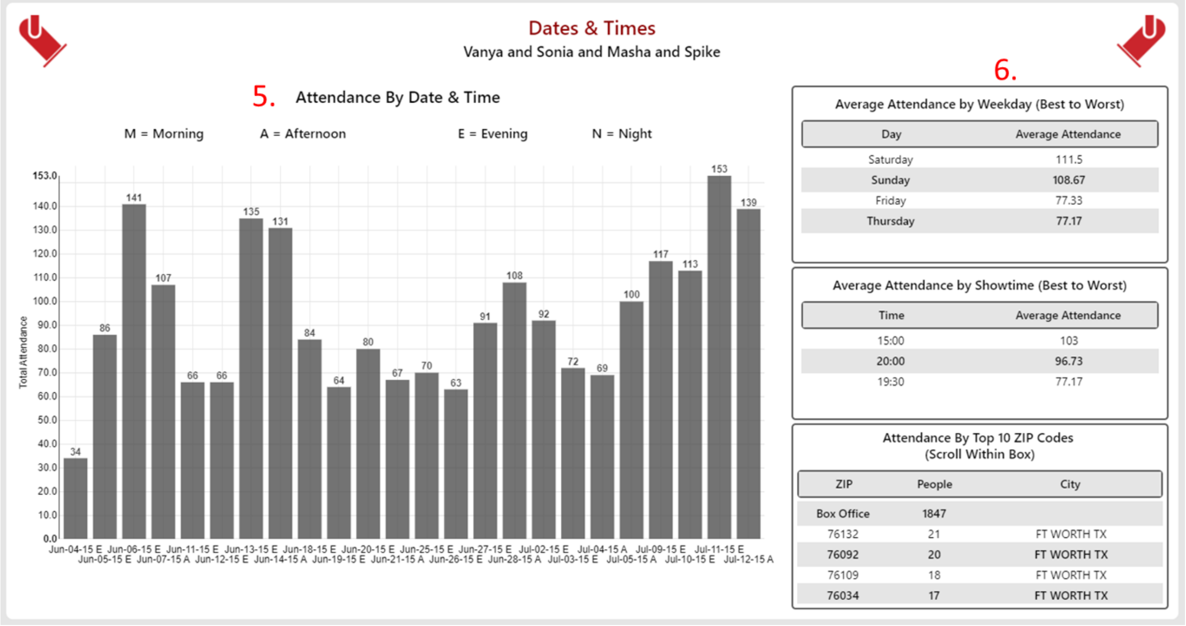
24	46.22%	65.05%
Number of Events	Percent of Paid Attendees	Percent of Capacity Filled

3.

Key Show Stats

4.

$25,459 Net Sales (Less Refunds)	11.2% Company Tickets Purchased
1,209 Comped Tickets	1 Refunded Tickets



Dates & Times

Vanya and Sonia and Masha and Spike

5. Attendance By Date & Time

M = Morning A = Afternoon E = Evening N = Night

6.

Average Attendance by Weekday (Best to Worst)

Day	Average Attendance
Saturday	111.5
Sunday	108.67
Friday	77.33
Thursday	77.17

Average Attendance by Showtime (Best to Worst)

Time	Average Attendance
15:00	103
20:00	96.73
19:30	77.17

Attendance By Top 10 ZIP Codes (Scroll Within Box)

ZIP	People	City
Box Office	1847	
76132	21	FT WORTH TX
76092	20	FT WORTH TX
76109	18	FT WORTH TX
76034	17	FT WORTH TX

Frontend Architecture

A. Ionic Framework (https://ionicframework.com/) version 1.2.4 – Free, open source mobile SDK for developing progressive web apps. "A Progressive Web App uses modern web capabilities to deliver an app-like user experience." (https://developers.google.com/web/progressive-web-apps?hl=en)

B. Angular.js (https://angularjs.org/) version 1.4.3 – "AngularJS lets you extend HTML vocabulary for your application. AngularJS is a toolset for building the framework most suited to your application development. It is fully extensible and works well with other libraries. Every feature can be modified or replaced to suit your unique development workflow and feature needs."

C. Lodash (https://lodash.com/) version 3.10.1 – "Lodash makes JavaScript easier by taking the hassle out of working with arrays, numbers, objects, strings, etc." Used to create models.

D. D3 Charting (https://d3js.org/) version 3.5.17 – Javascript library. Basis for NVD3 charting (see next).

E. NVD3 Charting (http://nvd3.org/) version 1.8.1 – Javascript and CSS library used to create all charts in the app.

F. Angular NVD3 Charting (http://krispo.github.io/angular-nvd3/#/) version 1.0.9 – "An AngularJS directive for NVD3 re-usable charting library (based on D3)." Chart types used: Pie Chart; Bullet Chart; DiscreteBar Chart; and HistoricalBar Chart.

G. Angular Tooltips (http://intellipharm.github.io/angular-tooltips/) version 1.0.2 – "Tooltips for any element using any html in AngularJS with no requirements." Used for mouse-over notes on stage lights and light bulbs.

Backend Architecture

A. RStudio (server-based IDE for R programming language https://www.r-project.org/about.html) version 3.4.1 - R is a language and environment for statistical computing and graphics.

B. OpenCPU (https://www.opencpu.org/) version 2.0 - OpenCPU is a system for embedded scientific computing and reproducible research. The OpenCPU server provides a reliable and interoperable HTTP API for data analysis based on R.

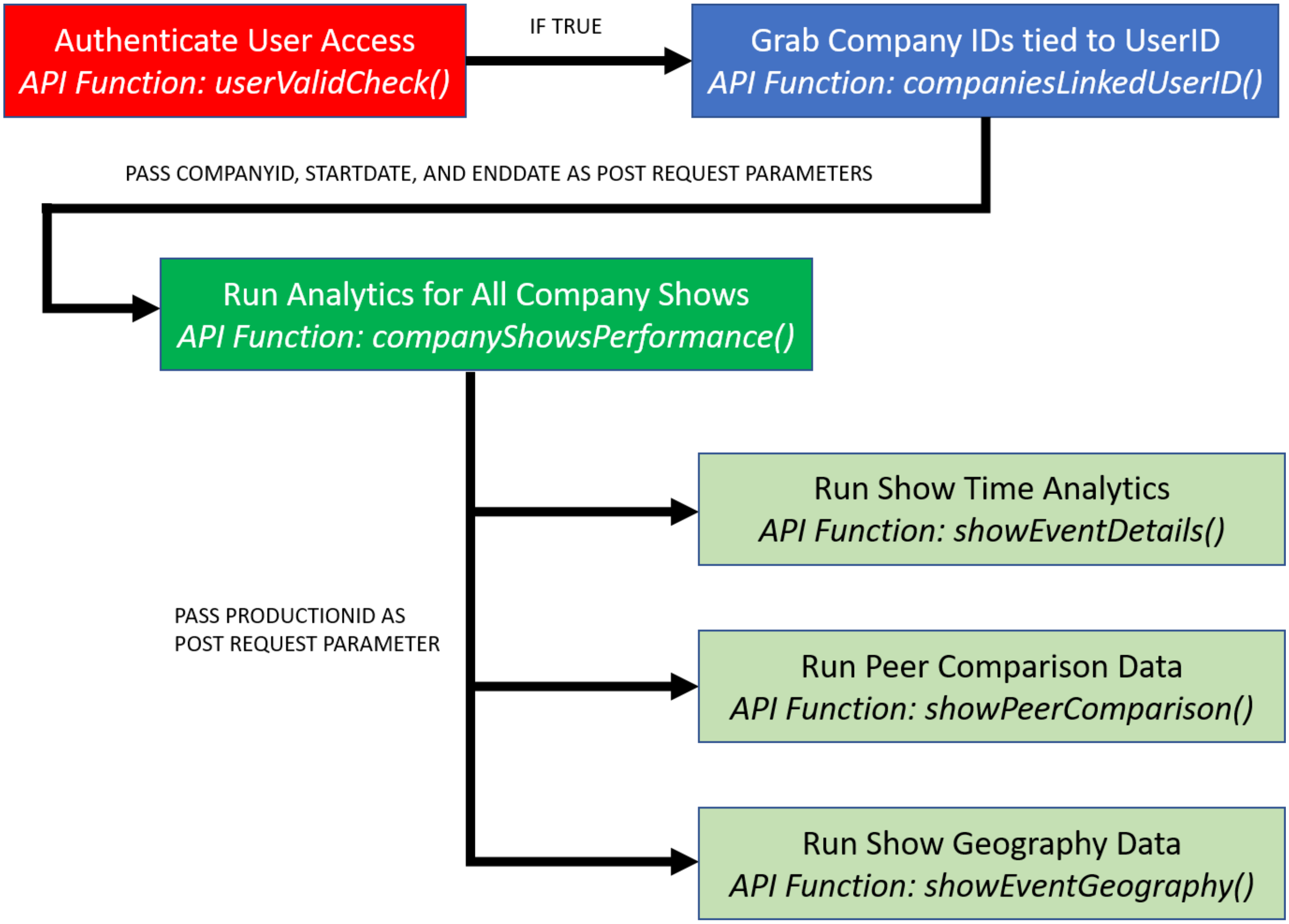
C. Linux Ubuntu Server (hosted on Amazon AWS https://www.ubuntu.com/server) version 16.0.4 LTS - Ubuntu Server brings economic and technical scalability to your datacentre, public or private.

D. MySQL (https://www.mysql.com/about/) version 5.7.1 - MySQL is the world's most popular open source database. With its proven performance, reliability and ease-of-use, MySQL has become the leading database choice for web-based applications.

E. Amazon AWS EC2 Engine (https://aws.amazon.com/ec2/instance-types)

C4.XLarge, 4 vCPUs Intel Xeon E5-2666 v3 (Haswell), 7.5GB RAM
20GB SSD, EBS Optimized
750Mbps Dedicated EBS Bandwidth

REST API Design and Process Flow


Authenticate User Access
API Function: userValidCheck()
IF TRUE
Grab Company IDs tied to UserID
API Function: companiesLinkedUserID()
PASS COMPANYID, STARTDATE, AND ENDDATE AS POST REQUEST PARAMETERS
Run Analytics for All Company Shows
API Function: companyShowsPerformance()
PASS PRODUCTIONID AS POST REQUEST PARAMETER
Run Show Time Analytics
API Function: showEventDetails()
Run Peer Comparison Data
API Function: showPeerComparison()
Run Show Geography Data
API Function: showEventGeography()

Efficient Design for Performance and Speed

The API was designed to be quick and efficient. We created each API to be efficient in its querying process and computation of the analytics it produces. We also built and tested each API to have speed, so that each API that ran would do so in roughly 1-2 seconds. This format allows for scalability as TheatreProfile's datasets continue to grow in size without sacrificing performance to the end-user. By chaining the API in this manner, we can also take advantage of single-threaded performance.

Smart Dependency for Eliminating Redundant Computation and Querying

Each API represents a component in another service call. We did this to avoid doing excessive computations or querying, making the API bigger. By using a UI/UX design on the frontend that displays data to the user in increments (mainly not to overwhelm them), we take advantage of keeping API requests smaller and faster.

REST API Function Descriptions

Function: *userValidCheck*

API Endpoint: /ocpu/user/rstudio/library/TAtheatreprofileMAIN/R/userValidCheck/json

Request Parameters

- admin (string that is username for TempusAnalytics database)
- password (string for TempusAnalytics database)
- host (string for host TempusAnalytics database server)
- table (string for table holding user credentialing data)
- dbname (string for schema within TempusAnalytics database)
- userID (string retrieved from Authorize API call from TheatreProfile)
- ipAddress (string of client’s private IP address for their device)
- activeKey (string generated from TheatreProfile’s website)

Response Parameters

- userValid (string returning “TRUE” or “FALSE”)

Function: *companiesLinkedUserID*

API Endpoint: /ocpu/user/rstudio/library/TAtheatreprofileMAIN/R/companiesLinkedUserID/json

Request Parameters

- admin (string that is username for TheatreProfile database)
- password (string for TheatreProfile database)
- host (string for host TheatreProfile database server)
- companytable (string for table holding TTTC data)
- ownershiptable (string for table holding TP userID connection to TP company data)
- tpcompanytable (string for table holding TP company data)
- dbname (string for schema within TheatreProfile database)
- userID (string retrieved from Authorize API call from TheatreProfile)

Response Parameters (Array with Objects)

- id (number of companyID)
- companyName (string referencing TTTC company name)
- index (number used for dropdown menus)

Function: *companyShowsPerformance*

API Endpoint: /ocpu/user/rstudio/library/TAtheatreprofileMAIN/R/companyShowsPerformance/json

Request Parameters

- admin (string that is username for TheatreProfile database)
- password (string for TheatreProfile database)
- host (string for host TheatreProfile database server)
- tickettable (string for table holding TTTC ticket data)
- venuetable (string for table holding TTTC venue data)
- eventtable (string for table holding TTTC event data)
- productiontable (string for table holding TTTC production data)
- dbname (string for schema within TheatreProfile database)
- organizationID (number retrieved from companiesLinkedUserID call)
- startDate (string from user input)
- endDate (string from user input)

Response Parameters (Array with Objects)

- tttcProductionID.id (number of TTTC production)
- tttcProductionID.title (string referencing TTTC production name)
- netSales (number of ticket sales for TTTC production)
- netAttendedTickets (number of tickets purchased and comped minus refunds)
- netPurchasedTickets (number of tickets purchased minus refunds)
- totalCompedProdTkts (number of comped tickets for production)
- totalRefundedProdTkts (number of refunded tickets for production)
- pctCapacity (number is percentage of people attended vs venue capacity)
- numEvents (number of events for production)
- pctTotalPurchasedTkts (number is percentage of tickets sold for production vs all productions)
- pctPaidAttendees (number is percentage of tickets purchased out of total attendance)
- avgTktPrice (number based on aggregate price per ticket for all ticket sales for production)
- index (number used for dropdown menus)

Function: *showEventDetails*

API Endpoint: /ocpu/user/rstudio/library/TAtheatreprofileMAIN/R/showEventDetails/json

Request Parameters

- admin (string that is username for TheatreProfile database)
- password (string for TheatreProfile database)
- host (string for host TheatreProfile database server)
- tickettable (string for table holding TTTC ticket data)
- eventtable (string for table holding TTTC event data)
- dbname (string for schema within TheatreProfile database)

- productionID (number retrieved from companyShowsPerformance call)
- startDate (string from user input)
- endDate (string from user input)

Response Parameters (Array containing Array with Objects)

Array #1

- id (number of TTTC event)
- tttcProductionID (number referencing TTTC production)
- eventDate (string referencing date and part of day for TTTC event)
- numAttended (number of attendants for that particular event)
- wdayNum (number from 1 – 7 based on which day of week event took place)
- weekday (string of weekday name based on wdayNum parameter)
- time (string in military localized time format of the event)

Array #2

- days.i. (string containing day for which average attendance is calculated)
- avgAttendance (number of average attendants for a particular day)

Array #3

- times.i. (string containing time for which average attendance is calculated)
- avgAttendance (number of average attendants for a particular time)

Function: *showEventGeography*

API Endpoint: /ocpu/user/rstudio/library/TAtheatreprofileMAIN/R/showEventGeography/json

Request Parameters

- admin (string that is username for TheatreProfile database)
- password (string for TheatreProfile database)
- host (string for host TheatreProfile database server)
- otheradmin (string that is username for TempusAnalytics database)
- otherpassword (string for TempusAnalytics database)
- otherhost (string for host TempusAnalytics database server)
- tickettable (string for table holding TTTC ticket data)
- usertable (string for table holding TTTC user data)
- eventtable (string for table holding TTTC event data)
- dbname (string for schema within TheatreProfile database)
- geotable (string for table with 3-Digit Zip Code city references)
- tatheatredb (string for schema within TheatreProfile database)
- productionID (number retrieved from companyShowsPerformance call)
- startDate (string from user input)
- endDate (string from user input)

Response Parameters (Array with Objects)

- zipCodesEvents (string of zip code where users purchased tickets from)
- attendance (number of users that purchased tickets from a particular zip code)
- city (string equivalent based on 3-digit zip code, compiled from USPS)

Function: ***showPeerComparison***

API Endpoint: /ocpu/user/rstudio/library/TAtheatreprofileMAIN/R/showPeerComparison/json

Request Parameters

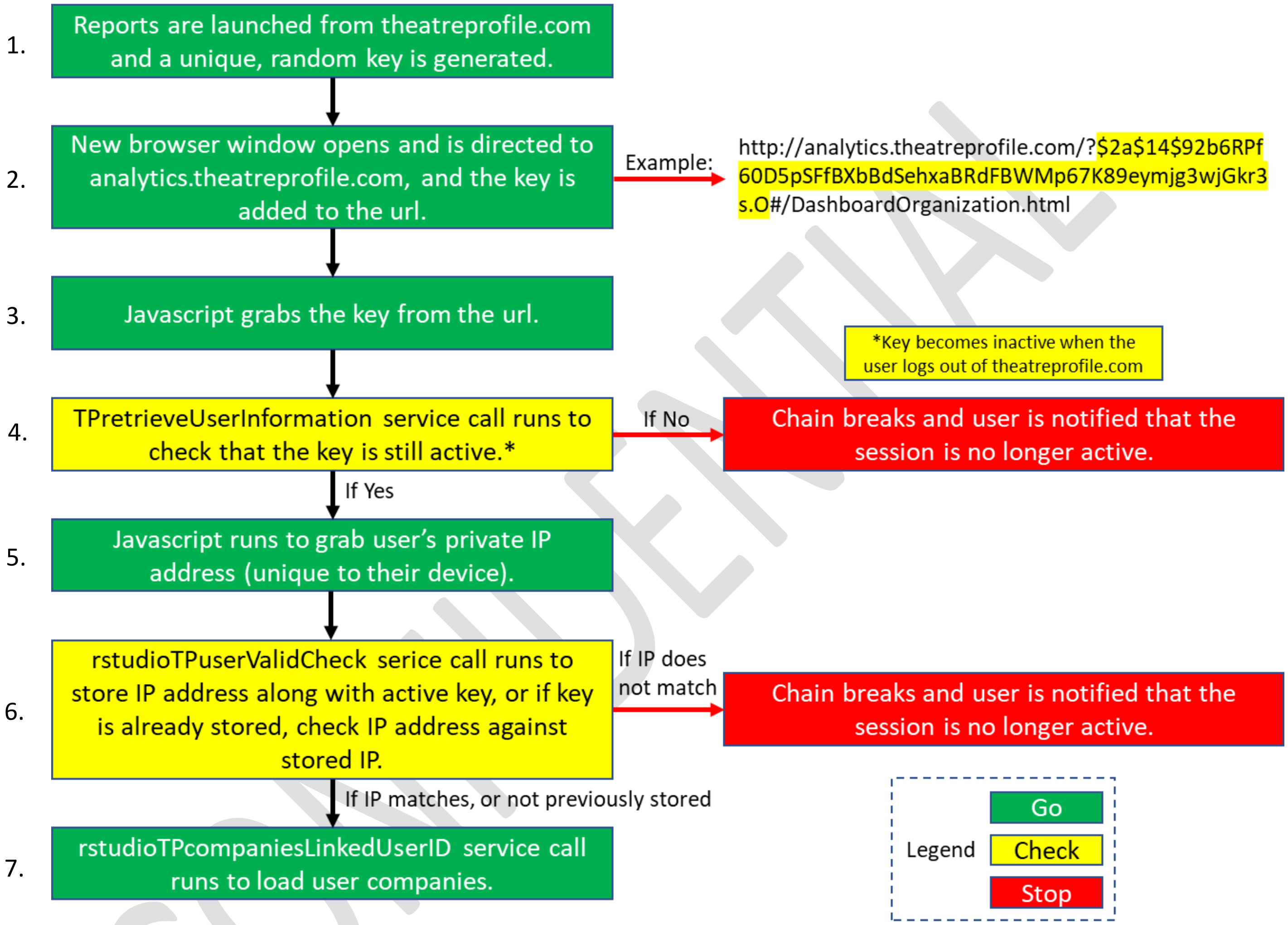
- admin (string that is username for TheatreProfile database)
- password (string for TheatreProfile database)
- host (string for host TheatreProfile database server)
- tickettable (string for table holding TTTC ticket data)
- eventtable (string for table holding TTTC event data)
- venuetable (string for table holding TTTC venue data)
- tpshowtable (string for table holding TP show data)
- showtable (string for table holding TTTC show data)
- productiontable (string for table holding TTTC production data)
- dbname (string for schema within TheatreProfile database)
- productionID (number retrieved from companyShowsPerformance call)

Response Parameters (Array with Objects)

- tpShowID.tpShowID (number of TP show ID)
- tpShowID.title (string of TP Show Title)
- netSales (number of ticket sales for peer TTTC production)
- netAttendedTickets (number of peer tickets purchased and comped minus refunds)
- netPurchasedTickets (number of peer tickets purchased minus refunds)
- totalCompedProdTkts (number of peer comped tickets for production)
- totalRefundedProdTkts (number of peer refunded tickets for production)
- pctCapacity (number is percentage of people attended vs venue capacity for all peers)
- numEvents (number of peer events for production)
- avgTktPrice (number based on aggregate price per ticket for all peer ticket sales)
- pctPaidAttendees (number is pct of tickets purchases out of total attendance for all peer events)

Security Process Flow

The Theatre Profile Company Dashboard uses two-factor authentication to ensure that only the authorized user is able to access the sensitive company data that is provided. The two factors are active key authentication and private IP address verification. The authentication process is outlined below.



1. Reports are launched from theatreprofile.com and a unique, random key is generated.

2. New browser window opens and is directed to analytics.theatreprofile.com, and the key is added to the url.

Example: http://analytics.theatreprofile.com/?$2a$14$92b6RPf60D5pSFfBXbBdSehxaBRdFBWMp67K89eymjg3wjGkr3s.O#/DashboardOrganization.html

3. Javascript grabs the key from the url.

*Key becomes inactive when the user logs out of theatreprofile.com

4. TPretrieveUserInformation service call runs to check that the key is still active.*

If No → Chain breaks and user is notified that the session is no longer active.

If Yes

5. Javascript runs to grab user's private IP address (unique to their device).

6. rstudioTPuserValidCheck serice call runs to store IP address along with active key, or if key is already stored, check IP address against stored IP.

If IP does not match → Chain breaks and user is notified that the session is no longer active.

If IP matches, or not previously stored

7. rstudioTPcompaniesLinkedUserID service call runs to load user companies.

Legend
🟩 Go
🟨 Check
🟥 Stop

This process runs in the background while the user is viewing the onboarding. Users are unlikely to notice any delay in information loading.

Steps 3-6 run again each time the user clicks one of the two Run buttons, ensuring that the session is checked each time data is to be loaded.

Recommendations

Preventing the Dashboard Popup from being Blocked

Optimize TheatreProfile.com website to notify user to allow popup windows when report is launched to prevent user from being unaware that the dashboard was blocked. This is particularly important on mobile (any device that uses the Safari web browser) as some tablets do not inform the user that a popup window was blocked. This will affect user experience as the user may not be able to gain access to the dashboard and would be unaware as to why. Possible resolutions include: a notice on the TheatreProfile.com main site that popup blockers should be disabled; testing for different methods of opening new tabs/windows to access the report that would not be impacted by popup blockers; and, as a worst-case scenario, directly navigating to the dashboard rather than opening in a new tab/window.

Creating a Process for Linking TP Companies to TTTC Companies

One of the core issues going forward that can affect dashboard performance is the manual linking of TheatreProfile companies to TTTC companies. Because there is no automated process in place of doing this, and manual data entry is needed, this can affect the user not having access to companies they should be able to gain access to. This also leaves the risk for human error in that the person doing the data entry for manual linking could potentially give a user access to the wrong companies. We feel there is a process that could be created in order to solve this potential issue, involving a web interface that allows the user to properly link their company, and verifying it through email or some other credentialing process. It would also ensure a risk-free exchange of data between TheatreProfile and Tempus Analytics dashboard(s).

Creating a Process for Linking TP Shows to TTTC Productions

Another potential issue going forward that can affect the user experience is the manual linking of TheatreProfile show IDs to TTTC production IDs. Because the Peer Comparison data relies on this link, manual entry can involve errors that could either omit data that should be included in computations of peer analysis, or include additional data that should not be in the computations. Using an automated process to correctly link the TheatreProfile show IDs to the TTTC production IDs would have to involve the use of linguistic analytics to properly link shows together. While this still has some risk, it is one step closer in the direction of automating the process. The other solution would involve a solution between TTTC and TheatreProfile.