System Requirement Specification
simplyatx.com

Current Version:	1.0
Owner(s):	Lanlan Liu, Justin Franz, Borislav Sabotinov
Date Last Updated:	3/5/2020
Last Updated By:	Boris
Author(s):	Lanlan Liu, Justin Franz, Borislav Sabotinov
Date Created:	2/10/2020

Revision History

Version Number	Date Updated	Revision Author	Brief Description of Changes
0.1	2/10/20	Boris	Created
0.2	2/20/20	Lanlan	Added sections 1-3 and populated with preliminary details
0.3	2/22/20	Boris	Updated requirements
0.4	2/24/20	Lanlan	Update section 3 - functions
0.5	3/3/20	Boris	Added basic class diagram, mockup, and updated sections 4 and 5.
0.6	3/4/20	Justin	Updated grammar and structure
0.7	3/5/20	Boris	Added ATDD, updated sections 2 and 3
1.0	3/5/20	Boris	Ready for submission

Table of Contents

Tables and Figures

1 Introduction

1.1 Purpose

Provide guidelines necessary for designing and implementing a software that fulfills the requirements stated in this document.

As in the case outlined below, a movie & television show search website, this document will define all the software requirements as well as their intended design and function.
This document is intended for both clients and developers.

1.2 System Scope

The software product that will be produced is a website called SimplyATX. The URL is https://simplyatx.com/.

SimplyATX will help users find information about movies and TV shows.
Unlike IMDB or Rotten Tomatoes, this site will serve as an aggregator for movie and TV show information. It will take freely available data from available APIs and display reviews for a given title, collected from various sources. Some analytics will be performed and displayed to the users, such as an average score.

SimplyATX will provide an interface for users to find a movie or TV of their choice. Movie and show information will be obtained by scraping data off available public datasets from IMDB and associated providers. A user will be able to perform a search for a title. He or she will be provided a list, displaying relevant results. Each result will display some limited title information, along with relevant links to where the information came from, so a user may navigate to these sites and find more details if needed.

SimplyATX will also provide an interface for users to rate a movie or TV show.This function allows users to submit a custom rating for an existing title. It will use a unique system for evaluating the title, as opposed to a rating system that is already implemented on another site. Emojis will be utilized to symbolize how a movie made the user feel, offering deeper contextual information rather than a relative/ambiguous rating scale.

SimplyATX will provide an interface for users to buy a movie ticket. In order to accomplish this the site will obtain live data from Fandango and other movie theater sites (AMC, Cinemark) by use of their public APIs (this is what Google uses to display showtimes and ticketing information to a user if you search it). The purchase will not be conducted through our site but will redirect to a vendor site to complete the transaction, credit card information will not be processed on or through our site.

SimplyATX will provide an interface for users to create and manage a personal user account. This function allows a user to create an account with the system so that the site may facilitate a users history and favorite movie or show information as well as any ratings or reviews they contribute. The site will create a user account by collecting first name, last name, email address information from the user during the creation process. The system will require a user to pick a username and password. If a user has an account in the

system, the user can log in to their account by entering the associated username and password. An edit feature will be incorporated to allow a user to edit all information except their username. The user space will provide in addition to the basic features listed below a view for tracking their own contributions, milestones and awards.

SimplyATX will aggregate reviews and showtime information from public API's that can be leveraged, as well as get relevant showtime information and display it.

SimplyATX will provide an interface for users to subscribe and receive notifications. At the simplest level, users may subscribe to a newsletter or special alerts from the site. The type and frequency of notifications will be based on the preferences they define in the UI. Some of the newsletter content may include: theater special events, limited time showings of new or previous movies (e.g., Studio Ghibli's Totoro) and /or general notifications from their prefered theaters.

SimplyATX will provide an interface for users to get contribution milestones. Similar to other sites that award the user a certain number of points for reaching certain milestones. If a user contributes a certain number of reviews, for example, they may get a badge or a ribbon on their profile. If they have been active for a certain period of time, added subscriptions or notifications.

SimplyATX will endeavor(as an optional feature) to provide an interface for users to leave feedback about the website. This function allows a user to give feedback of his/her experience with SimplyATX.com. The user can submit textual comments to show his/her satisfaction with the functionality, interface, usability, performance, etc... and give a satisfaction score using the same emoji system.

The ultimate goal of this website is to give users the ability to find movie information, read through reviews and buy tickets for movies.

1.3 Definitions, acronyms, and abbreviations

Term	Definition
SRS	System Requirement Specification, this document will outline all the requirements that the software must fulfil.
UI	User interface
GUI	Graphical user interface
OS	Operating system
Backend	The part of a computer system or application that is not directly accessed by the user, typically responsible for storing and manipulating data.
Frontend	Relating or denoting the part of a computer system or application with which the user interacts directly
Interface	a device or program enabling a user to communicate with a computer.
API	Application programming interface
IMDB	International Movie Database, a website which provides movie and TV show related information

Webkit	An open-source web browser engine that was developed by Apple, Inc.
IE	Internet Explorer, a web browser developed by Microsoft.
Chrome	A web browser developed by Google, based on a fork of Webkit.
WAN	Wide area network. A telecommunications network that extends over a large geographical area for the primary purpose of computer networking.
LAN	Local area network. The SimplyATX application is not accessible via LAN only networks.
Web Safe	Web Safe, or Browser Safe colours consist of 216 colors that display non-dithered, consistent colour on any computer or device capable of displaying 8-bit color.
PII	Refers to any information that can be used to identify an individual, like a name, birth date, or address.
Sensitive PII	Includes information that is not available elsewhere or may harm the individual by being made available. Examples include Social Security Numbers, Tax ID numbers, or unlisted telephone numbers.
Wireframe	Represents a product's structure
Mockup	Shows how the product is going to look. Either a mid or high-fidelity display of design
REST	**RE**presentational **S**tate **T**ransfers. A software architectural style that defines a set of constraints to be used for creating Web services.
SMTP	Simple Mail Transfer Protocol
ATDD	Acceptance Test Driven Development. The acceptance criteria consists of a set of one or more test cases, which have to pass for the requirement to be considered complete.

1.4 References

[1] IEEE Software Engineering Standards Committee, “IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications”, October 20, 1998.

[2] US Code Title 29, Chapter 16, Section 794d (known as Section 508 compliance) https://www.govinfo.gov/content/pkg/USCODE-2011-title29/html/USCODE-2011-title29-chap16-subchapV-sec794d.htm

[3] Meszaros, Gerard, and Janice Aston. (2006) "Adding Usability Testing to an Agile Project."Agile Conference.

[4] Koskela, Lasse. (2007) Test Driven: TDD and Acceptance TDD for Java Developers. Manning Publications

1.5 Overview

Section 2 contains a more in-depth and complete definition of the requirement of the SimplyATX website and will provide a background for those requirements. This section is intended for users who may or may not be familiar with the software engineering developing process. This section contains little to no technical terms or information about the website and is targeted toward non-software engineering subjects.

Section 3 contains all the software requirements for this software, including a full list of functional and non-functional requirements. Those requirements will be at a level if detail sufficient to enable

designers to design the SimplyATX, and testers to test those requirements. This section is intended to be used by the software developers to create the actual website. The majority of the contents in this section would be very technical, and thus is not intended for anyone outside of the software engineering field.

2 Overall Description

2.1 Product Perspective

The SimplyATX is independent and self-contained. A database will be used to store and retrieve related information. Our website will be an aggregator for movie and TV show information.

2.2 Product Functions

- Create a user account - Users would be able to create an account specific for this SimplyATX website or they can log in through their google account. SimplyATX will send information to google to get authentication and therefore will not store user's google account in our database.

- Manage user account - If a user has an account in the system, the user can log in to his/her account by entering the username and password in order to edit all information except the user name. They can also change subscription/notifications preferences and view their contribution milestones/awards.

- Add a movie or TV show - This function allows a user to add a movie or TV show only if it does not already exist. The user enters the name, year (yyyy), and other information associated with movies - cast and crew, directors and producers, etc. Some movies and shows may be pre-populated by scraping data off available public datasets from IMDB, Netflix, and similar sites.

- Rate a movie or TV show -This function allows a user to submit a custom rating for an existing title. It will use some unique system for evaluating the title, as opposed to the straightforward IMDB star rating. Perhaps emojis to symbolize how a movie made the user feel, offering deeper contextual information rather than a relative/ambiguous rating scale.

- Buy a ticket - Get real data from Fandango and other movie theater sites (AMC, Cinemark) using their public APIs (this is what Google uses to display showtimes and ticketing information to a user if you search it). The purchase won't be from our site but will simply link to the vendor site, as we will not collect or handle credit card information.

- Aggregate reviews - The site will aggregate reviews from a number of relevant databases:
    - IMDB
    - RottenTomatoes
    - Netflix

It will display these reviews to the user, along with other relevant metrics. For example, what is the average score for a movie across the combined datasets we are able to access.

- Aggregate showtime information - If a movie is currently on offer in theaters, display the relevant results to the user given their zip code. For companies providing a public API that can be leveraged, get relevant showtime information and display it.
    - Fandango
    - Cinemark
    - AMC

  Suppose a user wants to see My Neighbor Totoro (1988). It is an older title but at times shown in theaters. Display showtime availability in Austin to the user and prompt them if they want to subscribe to notifications.

- Subscribe and receive notifications - At the simplest level, let users subscribe to a newsletter or special alerts from the website. They will enter an email, which will be preserved. The type and frequency of notifications will be based on the preferences they define in the UI. Also, certain theaters have special events. Limited time showings of new or previous movies (e.g., Studio Ghibli's Totoro). Allow the user to subscribe to receive notifications about these events.

- Get contribution milestones - Similar to other sites that award the user a certain number of points for reaching certain milestones. If a user contributes a certain number of reviews, for example, they may get a badge or a ribbon on their profile. If they have been active for a certain period of time, added subscriptions or notifications.

- Leave feedback about the website - This function allows a user to give feedback of his/her experience with the hotel or car rented. The user can submit textual comments to show his/her satisfaction with the hotel or car, using the reservation number, and also give a goodness score of 1-10 stars (1 is the lowest) for hotel rooms. The hotel stars, which are shown when booking, along with the comments, is the average of the hotel's previous bookings. Cars do not use the star scores, but do show the previous comments.

- Aggregate and display title metadata - Each movie or TV show has much data associated with it. Trailers, previews, cast and crew, notable quotes. We target a subset of this data, collect it from multiple sources, and display it to the user in one place. Suppose a user wants to watch Castle in the Sky (1986) - we obtain and display the following:
    - Trailer from YouTube, as embedded HTML
    - Reviews from IMDB, Rotten Tomatoes, and/or Metacritic
        - Data availability depends on what the public APIs provide
    - Summary or link to Wikipedia entry

The following image shows the different functions described above and their relationships.

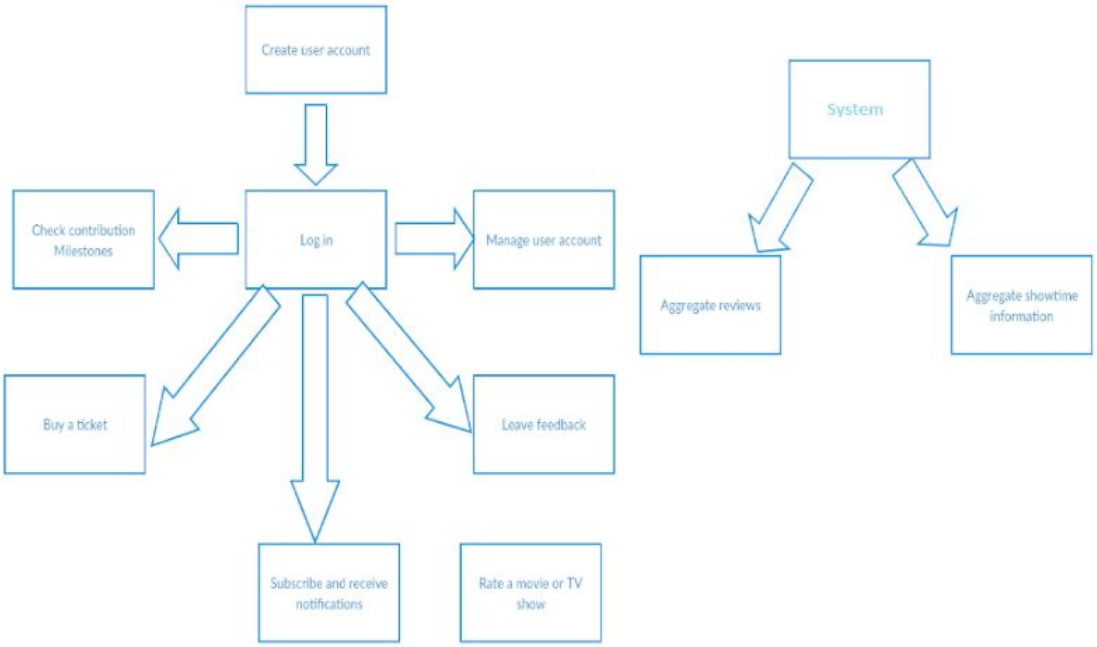


Figure 1 : Functional Relationships Diagram

2.3 User Characteristics

- User Age:
- Educational : All users need to be able to read.
- Technical expertise : All users need to have minimal technical expertise that they need to know how to use a mouse and keyboard.

2.4 Design and Implementation Constraints

- Platform: A browser of choice, preferably a webkit browser (e.g., Google Chrome, Safari, etc.). Browsers that do not natively support webkit, such as IE, will not be supported.
- Operating System : Any
- Hardware: Any
- Network: User must be connected to a WAN network.
- Viewport: The user must have some form of device with the ability to render modern browser graphics. The display size should be no smaller than **360 x 640** pixels.

2.5 Assumption and dependencies

- Regulatory policies;
- Hardware limitations (e.g., signal timing requirements);
- Interfaces to other applications;
- Parallel operation;

- Audit functions;
- Control functions;
- Higher-order language requirements;
- Signal handshake protocols (e.g., XON-XOFF, ACK-NACK);
- Reliability requirements;
- Criticality of the application;
- Safety and security considerations.

2.6 Basic Class Diagram

Our back-end will be a REST API, serving to facilitate the procurement of information from other APIs and feeding it to the client service. As such, it will be limited in scope and complexity. Much of the business logic will be performed on the user's client (browser, via ECMAScript and associated frameworks). We will use **Spring Boot**, which bootstraps the application and handles much of the configuration for us. Primarily, two controllers will be used: @Controller for class *ViewsController*, to serve html views. And @RestController for class *MovieController*, to serve data in a JSON format, for consumption by the client. The *MoviesApplication* class is the entry point, which starts the service and uses the @SpringBootApplication annotation.

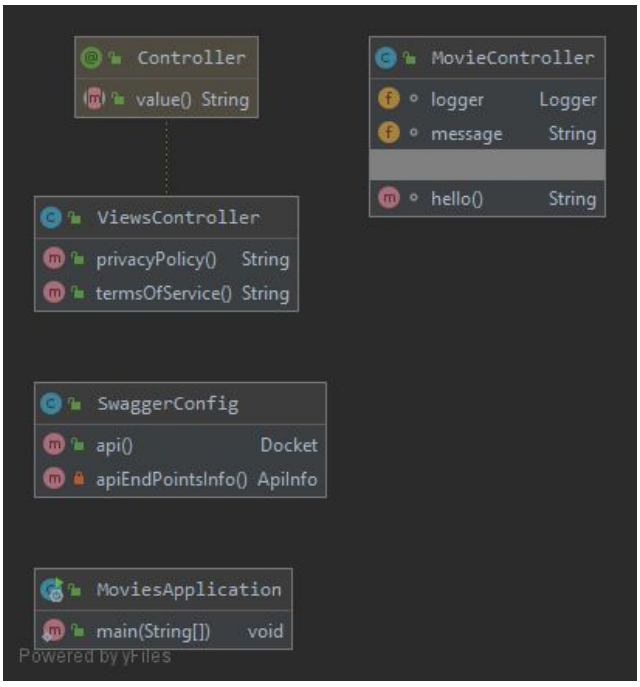


Figure 2: Basic class diagram of REST API

2.7 User interaction with site

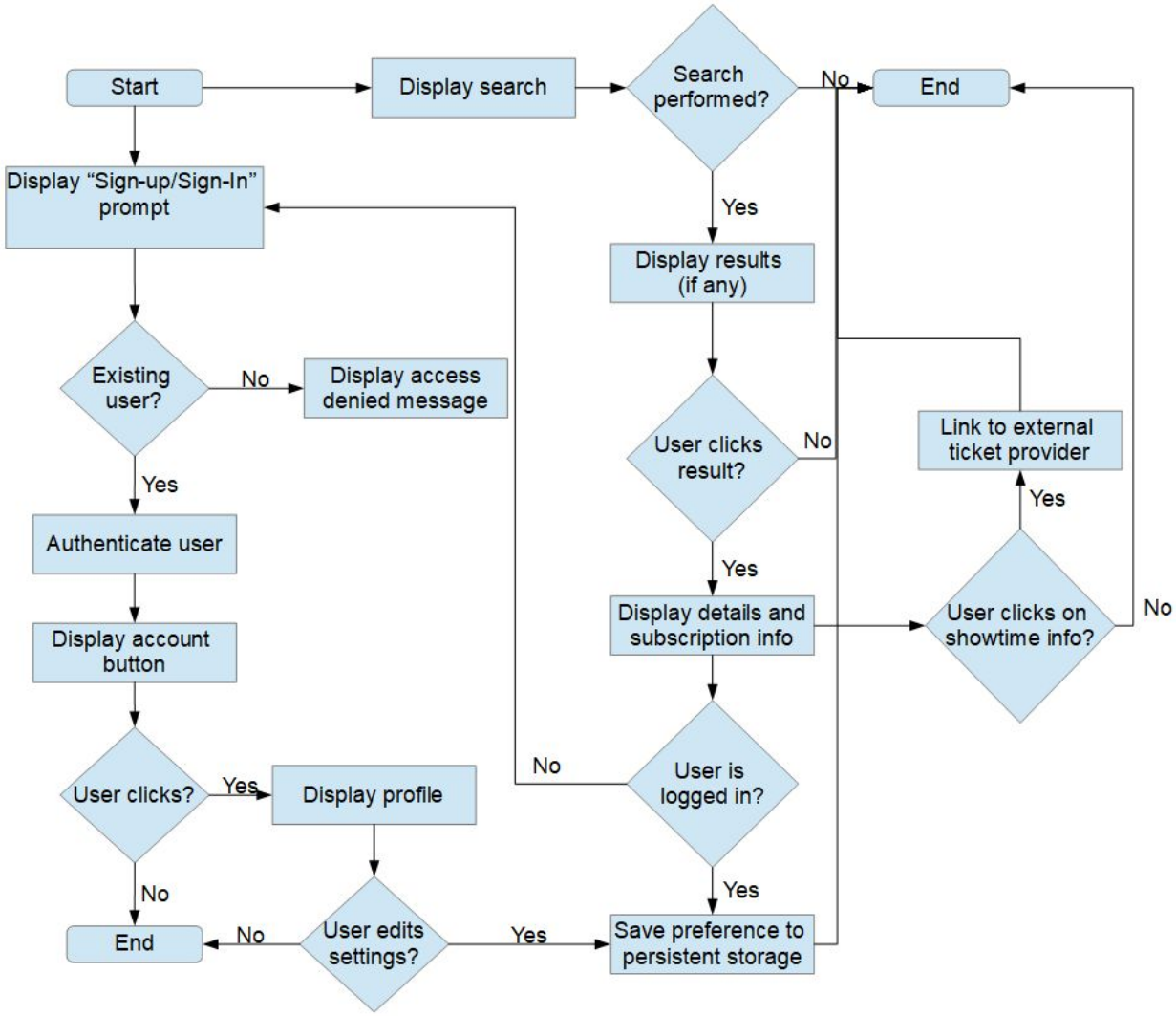


Figure 3: Flowchart depicting the system and its state based on user interaction.

2.8 Main page wireframe

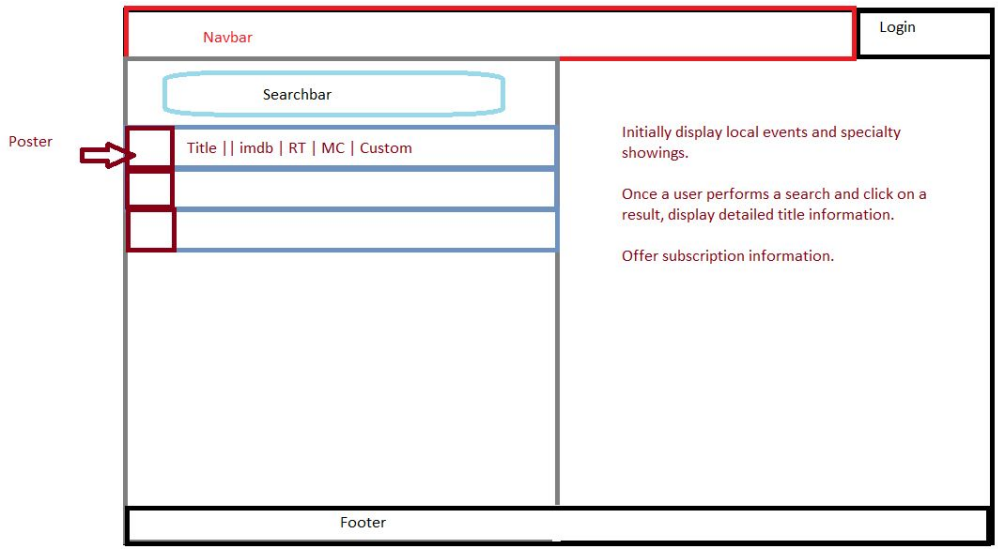


Figure 4: Wireframe of main website screen

2.9 Main Page Mockup

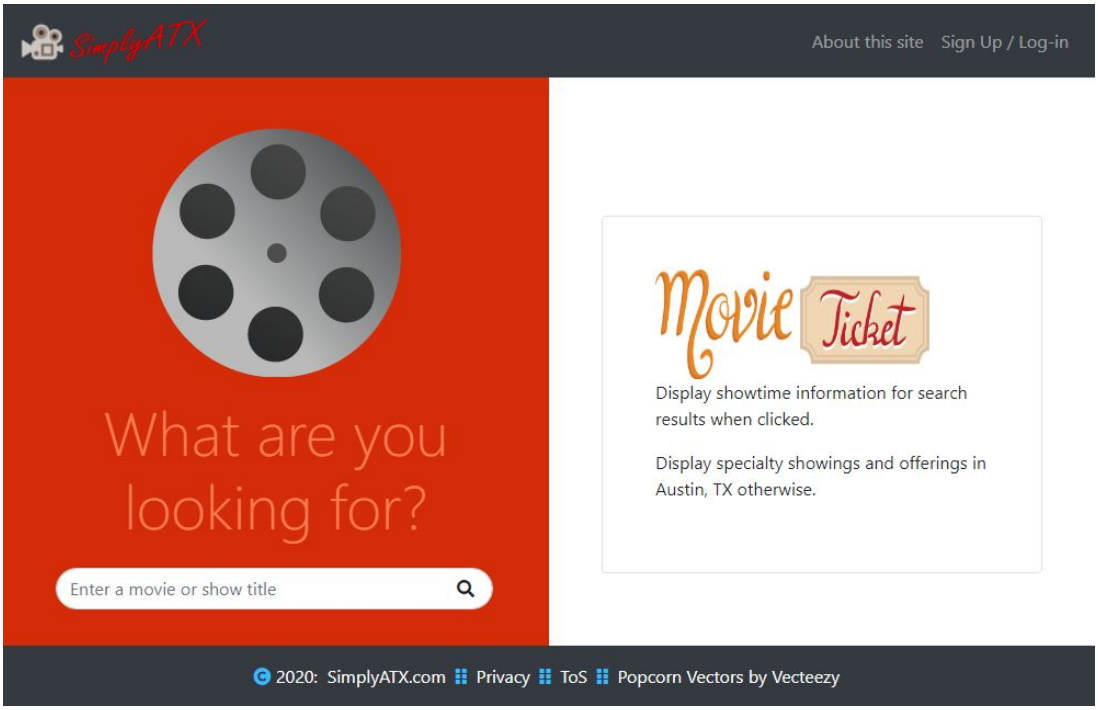


Figure 5: Main page mock-up in HTML5 and CSS3

3 Requirements

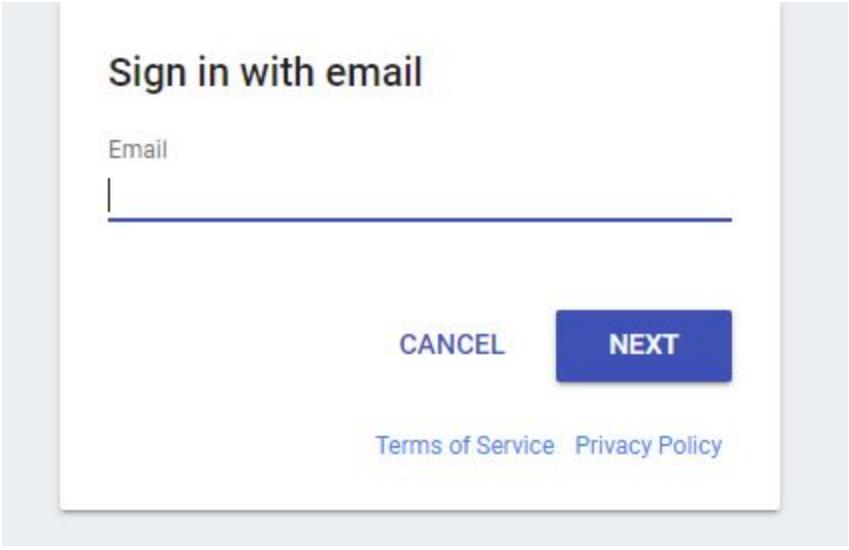
We will use ATDD for each requirement - this requires coming up with a test plan and a set of test-cases ahead of time. The acceptance criteria for each requirement is linked with a set of test cases, which have to pass for the requirement to be said to be complete. For example, in Section 3.1.1 below, requirement #1 is linked with test cases 1, 2, and 3 respectively.

3.1 External interfaces

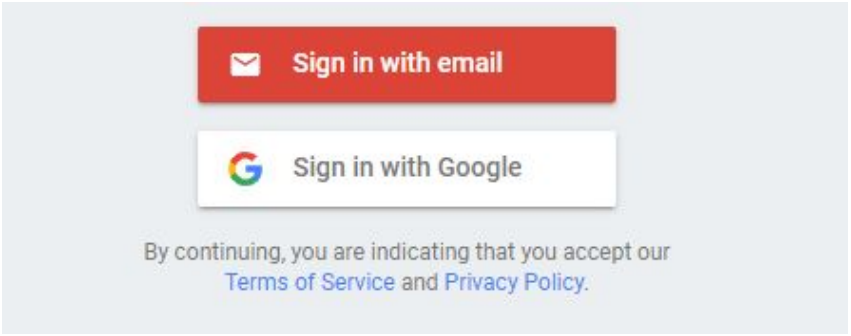
The following subsections will provide the various screens that will represent SimplyATX.
Each of the following subsections will discuss the details of the corresponding subsystems.

3.1.1 User registration subsystem

Requirement ID	1	**Requirement Type**	Business	**Use Case #**	
Description	Onboard a new user via email and password.				
Rationale	This allows a new user to register an account and authenticate via email and password.				
Acceptance/ Fit Criteria	1. Non-existing users are provided with prompt to register. 2. Registration uses a secure, industry standard email and password authentication based scheme. 3. New user is registered and authenticated with the application a. The user should see a “My Account” button or similar, instead of a prompt to Sign-up/Sign-in.				
Global/Local (Select One)	Global				
Priority	Essential				
Non-Functional:	The system should take no longer than 1200ms to respond and display content. The application does not need to be fully functional, but HTML, CSS and some JS content and functionality should display to alert the user that the link is working.				
Pre-conditions:	The application is accessible via a web browser and a provided URL. The user has a valid email address.				
Post-conditions:	An end-user is authenticated to the application, using an email address and a password.				

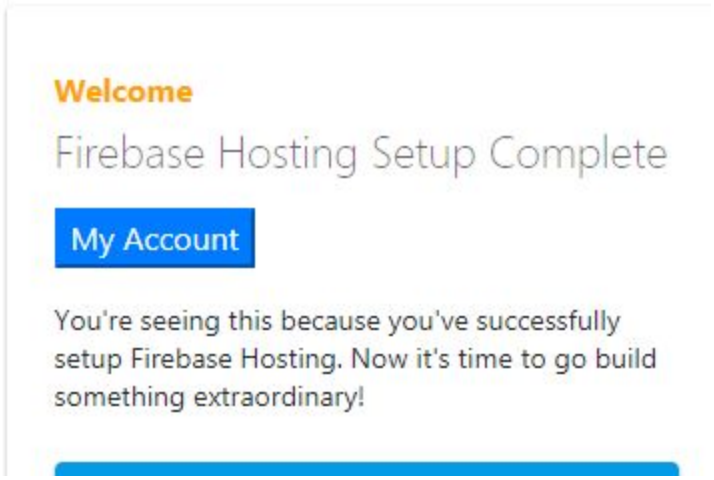
Image / Mockup:	

Requirement ID	2	**Requirement Type**	Business	**Use Case #**	
Description	Onboard a new user via Google sign-in.				
Rationale	This allows a new user to register an account and authenticate via their existing Google account.				
Acceptance/ Fit Criteria	4. Non-existing users are provided with prompt to register. 5. Registration uses Google's authentication scheme. 6. New user is registered and authenticated with the application a. The user should see a "My Account" button or similar, instead of a prompt to Sign-up/Sign-in.				
Global/Local (Select One)	Global				
Priority	Essential				
Non-Functional:	The system should take no longer than 1200ms to respond and display content. The application does not need to be fully functional, but HTML, CSS and some JS content and functionality should display to alert the user that the link is working.				
Pre-conditions:	The application is accessible via a web browser and a provided URL. The user has a Google account.				
Post-conditions:	An end-user is authenticated to the application, using their Google account. Profile information is obtained as necessary for the user's account.				

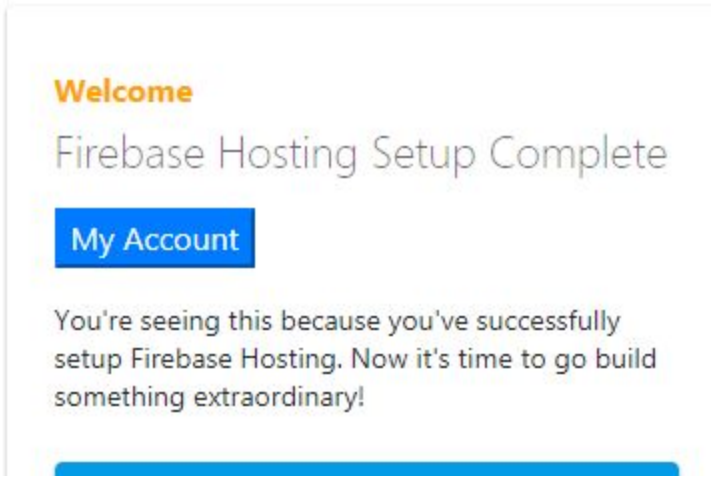
Image / Mockup:	

3.1.2 Log in subsystem

Requirement ID	3	**Requirement Type**	Business	**Use Case #**	
Description	Onboard an existing user via email and password.				
Rationale	This allows an existing user to authenticate via email and password.				
Acceptance/ Fit Criteria	7. Existing users are provided with prompt to sign-in. 8. Upon entering their username and password, if their account is found, they are authenticated. If it is not, access is denied. 9. Existing user is authenticated with the application a. The user should see a “My Account” button or similar, instead of a prompt to Sign-up/Sign-in.				
Global/Local (Select One)	Global				
Priority	Essential				
Non-Functional:	The system should take no longer than 1200ms to respond and display content. The application does not need to be fully functional, but HTML, CSS and some JS content and functionality should display to alert the user that the link is working.				
Pre-conditions:	The application is accessible via a web browser and a provided URL. The user has a valid email address.				
Post-conditions:	An end-user is authenticated to the application, using an email address and a password.				

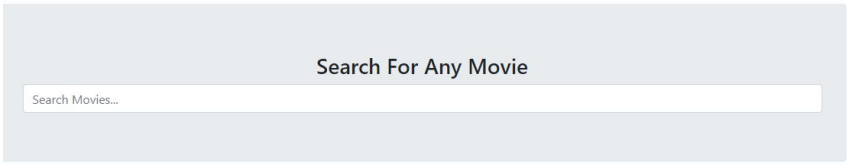
Image / Mockup:	

Requirement ID	4	**Requirement Type**	Business	**Use Case #**	
Description	Onboard an existing user via Google sign-in.				
Rationale	This allows an existing user to authenticate via their existing Google account.				
Acceptance/ Fit Criteria	10. Existing users are provided with prompt to sign-in. 11. Upon entering their username and password, if their account is found, they are authenticated. If it is not, access is denied. 12. Existing user is authenticated with the application a. The user should see a “My Account” button or similar, instead of a prompt to Sign-up/Sign-in.				
Global/Local (Select One)	Global				
Priority	Essential				
Non-Functional:	The system should take no longer than 1200ms to respond and display content. The application does not need to be fully functional, but HTML, CSS and some JS content and functionality should display to alert the user that the link is working.				
Pre-conditions:	The application is accessible via a web browser and a provided URL. The user has a Google account.				
Post-conditions:	An end-user is authenticated to the application, using their Google account. Profile information is obtained as necessary for the user’s account.				

Image / Mockup:	

3.1.3 Movie Search Subsystem

Requirement ID	5	**Requirement Type**	Business	**Use Case #**	
Description	Allow user to search for a film or show title				
Rationale	This allows the user to type in the name of a title of their interest and receive relevant results, so they can then proceed and take other actions depending on their interest.				
Acceptance/ Fit Criteria	13. Display search bar to the user 14. Upon entering their search parameters and submitting, the user is presented with results. 15. If no results are present, notify the user that their search criteria does not have any results.				
Global/Local (Select One)	Global				
Priority	Essential				
Non-Functional:	As multiple data sources will be accessed, procuring the results may take some time. Display a loading icon to the user, to indicate the search is still in progress. This way, they will know something is happening in the background. Disable other elements on the page, as the asynchronous nature of web applications means the search has to complete before the user can perform other actions.				
Pre-conditions:	The application is accessible via a web browser and a provided URL. The user does not have to be authenticated.				
Post-conditions:	A search bar is displayed to the user, prompting them for their query. Upon submitting, results are collected and displayed.				

Image / Mockup:	

3.1.3.1 Aggregated Review Scores Subsystem

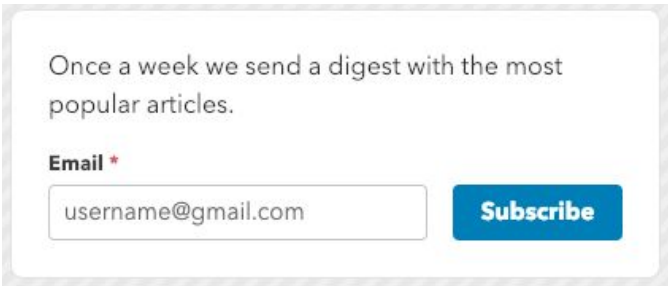
Requirement ID	6	**Requirement Type**	Business	**Use Case #**	
Description	Display title results, with reviews aggregated from multiple sources				
Rationale	Once a user performs a search, they are provided a list of results. Each relevant entry contains aggregated reviews from different providers (e.g., IMDB, Rotten Tomatoes). The scores are displayed as a hyperlink, which the user can click to visit the site in question and find more details about the title.				
Acceptance/ Fit Criteria	16. Valid search returns one or more results 17. Search results are displayed in a table format, with each entry showing the user aggregated review scores 18. Clicking the scoring organization takes the user to the external website in a new tab				
Global/Local (Select One)	Global				
Priority	Essential				
Non-Functional:	Minimize the number of clicks required for the user to find basic title information, such as runtime, review scores, and trailer.				
Pre-conditions:	The user has performed a valid search, for which results are available. The user does not need to be logged in to the application.				
Post-conditions:	A set of results are displayed to the user, providing a quick at-a-glance summary of different review scores. These results are clickable and lead the user to the website.				
Image / Mockup:					

3.1.5 Custom Review Subsystem

Requirement ID	7	**Requirement Type**	Business	**Use Case #**	
Description	Allow the user to leave a custom review for the title they interacted with.				
Rationale	This allows the user to leave a custom review on the website. Emojis can be used to allow the user to signal how the film or show made them feel, as opposed to a relative numeric rating.				
Acceptance/ Fit Criteria	19. "Leave custom review" option is displayed to the user 20. Clicking it presents the user with the emoji rating options 21. Upon submitting their review: a. If emoji is not present, add it and set count to 1 b. If emoji is present, increment count to one more than current 22. A different user accessing the site in a new session will see the newly submitted review				
Global/Local (Select One)	Global				
Priority	Essential				
Non-Functional:	Data should persist and be synchronized, so other users visiting the site after a review is added will be able to see it.				
Pre-conditions:	User executed a valid search which returned results. User clicked on one of the results' "leave custom review" and submitted their entry.				
Post-conditions:	UI displays the user's review appropriately as outlined in acceptance criteria.				
Image / Mockup:					

3.1.6 Subscription and Notification Subsystem

Requirement ID	8	**Requirement Type**	Business	**Use Case #**	
Description	Allow the user to subscribe to alerts and notifications for a given title, or special offerings.				
Rationale	This allows the user to subscribe to notifications if a film of their interest is playing. The website will also display a short list of specialty offerings (if available) from a limited list. The user may subscribe to email notifications, so they know when these films are playing in theaters.				

Acceptance/ Fit Criteria	23. If the user has not performed a search or clicked on a search result - display a list of locally available specialty showings and allow the user to subscribe to future notifications for this list. 24. If the user has performed a valid search and clicked on a result, display option to subscribe to notifications for that particular title
Global/Local (Select One)	Global
Priority	Essential
Non-Functional:	
Pre-conditions:	The user has accessed the application OR the user has executed a valid search and clicked on a result.
Post-conditions:	The user is subscribed to notifications. Their preference is saved and persists until changed.
Image / Mockup:	

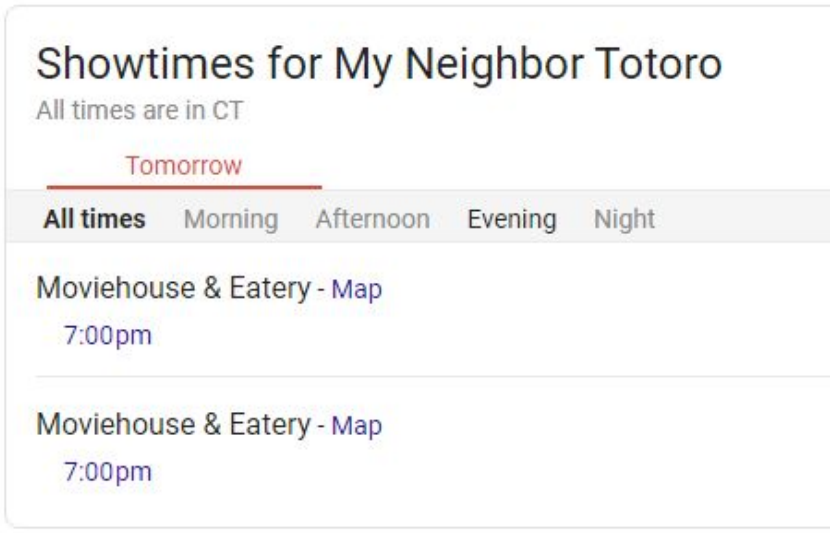
3.1.7 Results Subsystem

Requirement ID	9	**Requirement Type**	Business	**Use Case #**	
Description	List search results in a tabular format, with most relevant title details condensed for easy consumption.				
Rationale	This allows the user to view their search results. .				
Acceptance/ Fit Criteria	25. Given a valid search, table contains a list of relevant results 26. Each result listing contains all available relevant information a user may want to consume right away: a. Genre, runtime, trailer, description, year, poster				
Global/Local (Select One)	Global				
Priority	Essential				
Non-Functional:					
Pre-conditions:	The user has executed a valid search with available results.				
Post-conditions:	A list of results is displayed to the user.				

Image / Mockup:	 Fences Trailer *In 1950s Pittsburgh, a frustrated African-American father struggles with the constraints of poverty, racism, and his own inner demons as he tries to raise a family.* Genre: Drama Actor: Denzel Washington, Jovan Adepo, Stephen McKinley Henderson, Viola Davis Director: Denzel Washington Country: Canada, USA Duration: 139 min Quality: HD 720 Release: 2016 IMDb: 7.2

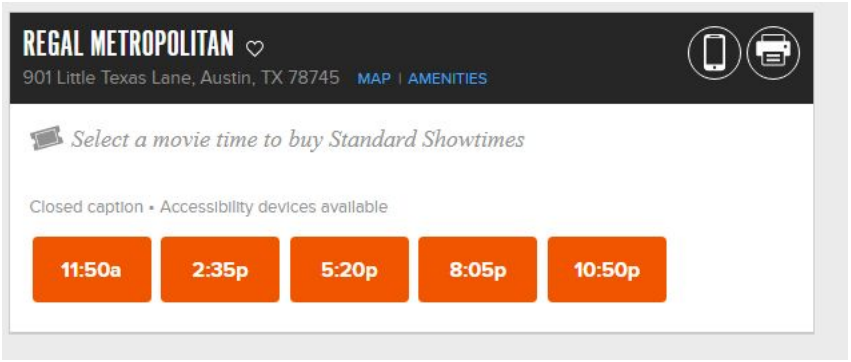
3.1.7.1 Movies Showtime Subsystem

Requirement ID	10	**Requirement Type**	Business	**Use Case #**	
Description	Display movie showtime details for a given title, if available.				
Rationale	This allows the user to type in the name of a title of their interest and receive relevant results, so they can then proceed and take other actions depending on their interest.				
Acceptance/ Fit Criteria	27. Showtime information, if available, is presented to the user on the right side of the screen upon performing a valid search. 28. The results are dynamic and obtained from a valid vendor (e.g., Fandango, Cinemark, and/or AMC)				
Global/Local (Select One)	Global				
Priority	Essential				
Non-Functional:	The system should take no longer than 1200ms to respond and display content. The application does not need to be fully functional, but HTML, CSS and some JS content and functionality should display to alert the user that the link is working.				
Pre-conditions:	The application is accessible via a web browser and a provided URL. The user does not need to be authenticated. The user has performed a valid search.				
Post-conditions:	Movie showtime information is presented to the user on the right side of the screen, with clickable links leading to a vendor site.				

Image / Mockup:	

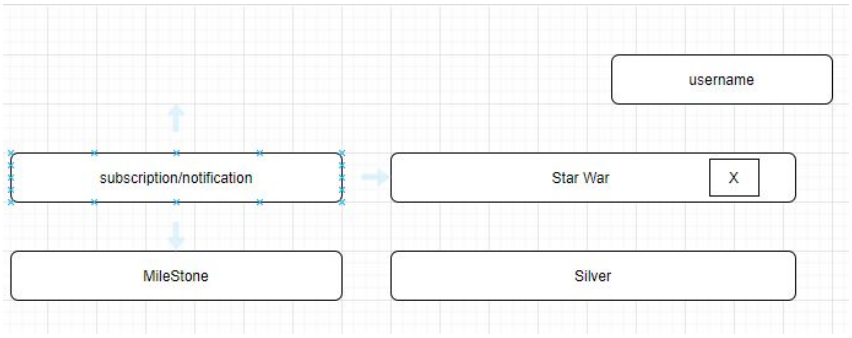
3.1.7.2 Movie Ticket Purchasing Subsystem

Requirement ID	11	**Requirement Type**	Business	**Use Case #**	
Description	Allow users to purchase tickets for movies through a third party website				
Rationale	This allows the users to purchase tickets from third party websites such as Fandango and Atom.				
Acceptance/ Fit Criteria	29. Clicking on a listed showtime will lead to a vendor web page where the user may make a ticket purchase if they wish.				
Global/Local (Select One)	Global				
Priority	Essential				
Non-Functional:					
Pre-conditions:	The application is accessible via a web browser and a provided URL. Showtime information is available and presented to the user.				
Post-conditions:	The user is led to a vendor in a new tab, where they can make a purchase if they wish.				

Image / Mockup:	

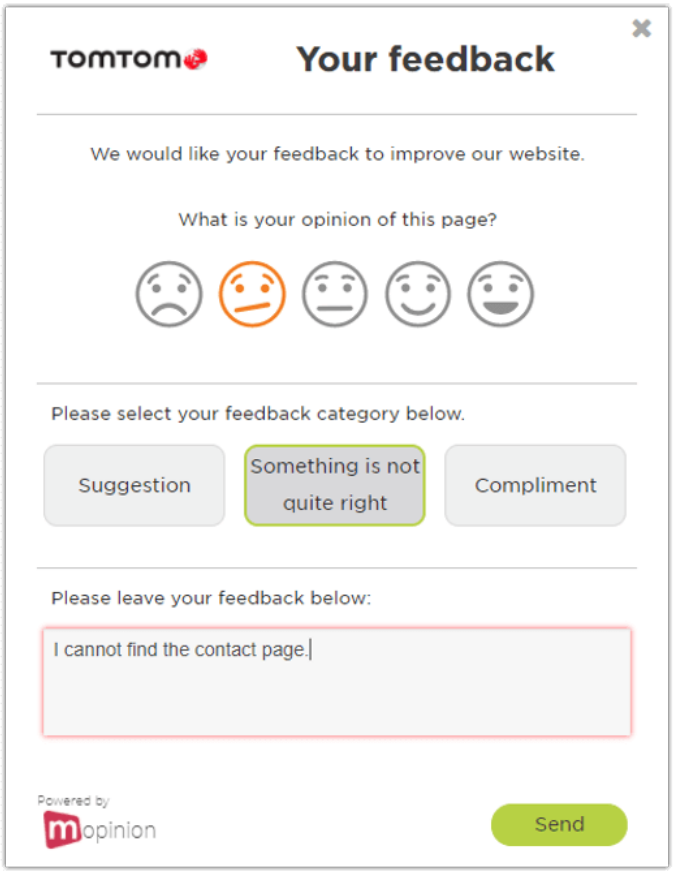
3.1.8 Account Management Subsystem

Requirement ID	12	**Requirement Type**	Business	**Use Case #**	
Description	Allow users to manage their subscription/notification preferences and view their contribution milestones.				
Rationale	This allows the users to modify the subscription/notification and also view their milestone contributions.				
Acceptance/ Fit Criteria	30. User is authenticated properly 31. User clicks on account management button and is presented with their account page 32. User modifies their preferences 33. Upon returning, those new preferences persist				
Global/Local (Select One)	Global				
Priority	Essential				
Non-Functional:					
Pre-conditions:	The application is accessible via a web browser and a provided URL. The user has a Google account or an email address based account.				
Post-conditions:	An end-user is authenticated to the application, using their account. If the user is using a Google account, the profile information is obtained as necessary for the user's account. Having edited their subscriptions and preferences, the changes take place and persist for future visits.				

Image / Mockup:	

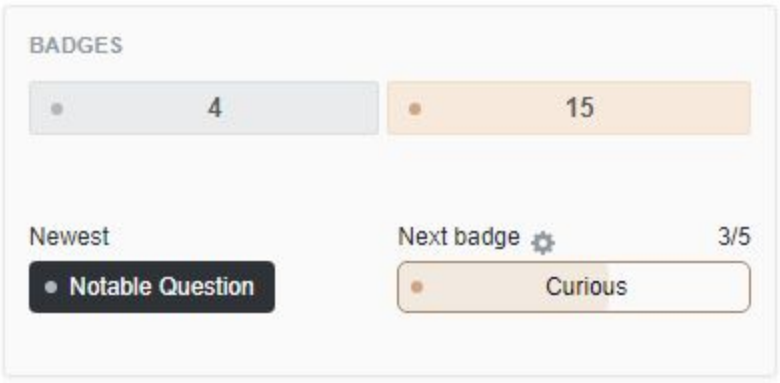
3.1.9 Feedback Subsystem

Requirement ID	13	**Requirement Type**	Business	**Use Case #**	
Description	Allow user to leave feedback for the website				
Rationale	This allows the user to leave feedback to the website. The users can submit textual comments to show his/her satisfaction, and also choose from emojis to show their opinions.				
Acceptance/ Fit Criteria	34. Existing users are provided with prompt to leave feedback. 35. The user submits either a bug-report, a suggestion, or positive feedback 36. The administrator is notified via email of the user's message 37. If the user leaves an email for follow-up, the feedback is not anonymous. An automated "acknowledgement" email is sent to the user as a receipt of the feedback.				
Global/Local (Select One)	Global				
Priority	Essential				
Non-Functional:					
Pre-conditions:	The application is accessible via a web browser and a provided URL. The user need not have an account. The user interacts with the feedback form and leaves feedback.				
Post-conditions:	Website admin is notified via email, triggering a workflow and/or a response to the user as needed.				

Image / Mockup:	

3.1.10 User Milestones Subsystem

Requirement ID	14	**Requirement Type**	Business	**Use Case #**	
Description	If the user reaches a certain milestone, provide them a custom flair or badge as a reward and incentive to continue interacting with the site. Proposed: • **Pioneer**: Leave 1st custom review • **Well-informed**: Subscribe to notifications • **Grrr!**: Leave 5 “angry” reviews • **I’m not crying, you’re crying**: Leave 5 “sad” reviews • **Heh…**: Leave 5 “funny” reviews • **Is the paint dry?**: Leave 5 “boring” reviews • **Much ado about nothin’**: Leave 5 “meh” reviews • **Huh?**: Leave 5 “hmm” reviews • **WTF?!**: Leave 10 “hmm” reviews • **Bruh**: Leave 5 “expressionless” reviews					

Rationale	This rewards the user for performing certain actions on the site and incentivizes them to keep coming back and interacting with the content. It allows the user to further configure their account with flair they've earned. As this is a review site, we want to encourage users to leave feedback for titles.
Acceptance/ Fit Criteria	38. Users are rewarded for taking certain actions on the site with a flair 39. The flair is correctly added to the user's account 40. The new user information persists when they return
Global/Local (Select One)	Global
Priority	Essential
Non-Functional:	
Pre-conditions:	The user is registered and has a valid account. The user has performed the necessary actions to reach the specified milestone and earn the reward.
Post-conditions:	The user is rewarded with a badge and/or custom flair as appropriate, which persists throughout their future visits.
Image / Mockup:	

3.2 Logical Database Requirement

The system needs to store user account information if the user creates an account on the website. All the data shall be stored in text-based flat files. For each user account, the email address, subscription/notification preference, milestone record shall be stored in one file. Each attribute shall be delimited by a semicolon, and all the entries shall be sorted alphabetically by the email.

3.5 Standard Compliance

Though USC Title 29 - LABOR - Section 508 of the Rehabilitation Act of 1973 - applies only to federal government websites, our intent is to ensure compliance. This will provide an improved customer experience, ensuring the following:

- Use of Web Safe colors throughout
- Availability of “alternate text” when appropriate (e.g., image elements on the page)

4 Software System Attributes

The requirements in this section specify the required reliability and robustness, performance, availability, security, maintainability and portability of the software system.

4.1 Reliability and Robustness

The guarantee placed on the system to provide relative up-to-date information on movie statistics and showtime information is dependent on third party systems not supported in this design but utilized by it. While indeterministic the source points for information maybe the services handling transactions can only be as robust as the infrastructure that supports it.
Supported and Maintained by SimplyATX.com:

1.) Web Site: media interface housed on a hosting platform
2.) Web Services: functional api points integrated as part of the architecture of the site
3.) Web DB content: any database or intermediate storage information used by the main site

Not supported:

4.) Point-to-Point or network interfaces
5.) Any website linked to the main site

Dependent on:

1.) Third party Information Brokers: IMDB, AFI, Rotten Tomatoes, MetaCritic
2.) Third Party Movie Ticket Vendors: Fandango

4.2 Performance

This section provides a detailed specification of the user interaction with the software and measurements placed on the system performance.
The main performance issues the system will encounter are during the following steps:

1) Spooling the database
2) Authentication of a user
3) Populating a users account info and domain specific alerts

4) Processing Search requests
5) Generating results in an easily digestible format that is scrollable
6) Seamless navigation for external links embedded in result sets displayed
7) Contractless design which indemnifies the user when initiating navigation from SimplyATX.com

Performance metrics for the components and services measurable by this design are as follows:

- All searches should return a result that consists of no less than 1 genuine response to a query
- All user reviews tied to a user's account must be retrieved and presented when requested
- Any link in a search list must display the corresponding result infographic and movie statistics in accordance with the current industry metrics
- Any user that logs in to their account should see any/all data displayed preconfigured for their account based on user preferences
- Any/All navigation from SimplyATX.com will correctly navigate the user to the desired movie related website with accurate querystring data required for their chosen action

4.2 Availability

The website and any service provided by SimplyATX.com not reliant on a third party system shall be available 24/7/365. New features will be rolled out without major service interruptions. There will be no maintenance downtime windows at night or during the weekend.
The system will allow the user to restart the application after a crash. The user will be able to load his or her data file after the system has been restarted and continue using the system.

4.3 Security

The system will use the computer's default operating system security. As users will be able to authenticate with the application and have an account profile, industry standard methods for handling their PII will be utilized. The system will provide a Privacy Policy and Terms of Service to the user, to satisfy legal requirements.

4.4 Maintainability

The planned version of this software uses a web-based client so that users can log into the system remotely. This makes it easy to maintain and update the software. New pages can be created with the system in full operation and implemented easily and quickly without service interruption.

4.5 Portability

The architecture is entirely web-based and web-hosted, therefore any and all devices that may access a web port or official IP provider will have access to the site. A device compatibility - mobile first development approach ensures users can visit the site from their preferred device.

5 Appendixes

5.1 Requirements Prioritization

5.1.1 Fibonacci Method of Story Pointing

“The fibonacci sequence is used by Scrum teams for story point estimates – 1, 2, 3, 5, 8, 13, 21, and so on. Teams use this sequence, rather than a linear 1 – 10 as it forces them to provide a relative estimate. Easier to ask ‘is that a 5 or an 8?’ than ‘is that a 6 or a 7?’.”

http://www.velocitycounts.com/2013/05/why-do-high-performing-scrum-teams-tend-to-use-story-point-estimation/

Any nonlinear estimation scheme helps improve decision making. For this project, we will use the Fibonacci sequence.

The following scale will be used: 1, 2, 3, 5, 8, 13

The worst-case score will be used as the final score, and it will be **bold** text highlighted in **yellow**.

5.1.2 Developer Scoring

1. Onboard a new user via email and password.
    - Lanlan: 5
    - Justin: **8** | to account for unknown’s I am adding 3 hours for research and analysis, possibly testing
    - Boris: 5 | Why: some unknowns surrounding Firebase Auth still (form for new users) and callbacks and redirects. But mostly handled by third parties.

2. Onboard a new user via Google sign-in.
    - Lanlan: **5**
    - Justin: 5
    - Boris: 5

3. Onboard an existing user via email and password.
    - Lanlan: **5**
    - Justin: 5
    - Boris: 5

4. Onboard an existing user via Google sign-in.
    - Lanlan: **5**
    - Justin: 5
    - Boris: 5

5. Allow user to search for a film or show title
   - Lanlan: **13**
   - Justin: 13 | I concur, as to say that even though I have flushed out the framework between two separate sites we may want to consider adding a backend backup archive of some kind
   - Boris: 5 | Why: searching and getting the data is mostly straightforward REST calls but multiple APIs and data formats, and risk that some data may not be available.

6. Display title results, with reviews aggregated from multiple sources
   - Lanlan: 5
   - Justin: 5 | less because the results I have worked down to a list of json objects, and if the sources are flushed out from the previous story this should be a easy implementation
   - Boris: **8** | Why: Multiple APIs and edge cases

7. Allow the user to leave a custom review for the title they interacted with.
   - Lanlan: 13
   - Justin: 13, concur
   - Boris: **13** | Why: Persistence and global updates

8. Allow the user to subscribe to alerts and notifications for a given title, or special offerings.
   - Lanlan: 13
   - Justin: **13** | I think this would require a stateful system that self updates and self monitors aside from user presence
   - Boris: 5 | Why: given a user who leaves their email for notifications, simply trigger a job on a timer to send an aggregated report of showtimes to the user. The difficulty is in gathering the report from multiple APIs.

9. List search results in a tabular format, with most relevant title details condensed for easy consumption.
   - Lanlan: **3**
   - Justin: 3
   - Boris: 3

10. Display movie showtime details for a given title, if available.
   - Lanlan: 5
   - Justin: 5
   - Boris: **5** | Basic REST GET call in back-end and update UI. Some risk due to multiple data formats

11. Allow users to purchase tickets for movies through a third party website
   - Lanlan: 5
   - Justin: 5
   - Boris: **5** | Why: REST call to get data but simply update UI with a link to the external site using the correct URL

12. Allow users to manage their subscription/notification preferences and view their contribution milestones.
   - Lanlan: 13
   - Justin: 13, concur
   - Boris: **13** | Why: Maintain user account (persistence) and track contributions

13. Allow user to leave feedback for the website
    - Lanlan: 2
    - Justin: **3** | add extra testing for edge cases and discovery
    - Boris: 2 | Why: basic notification workflow, can be done with SMTP. Edge case: robots/malicious users - may need to add ReCaptcha to filter out.

14. If the user reaches a certain milestone, provide them a custom flair or badge as a reward and incentive to continue interacting with the site.
    - Lanlan: 13
    - Justin: **21** | The reason is the additional design and verification work for usable media that has no copyright, such as icon's or other signets of this system, and a way to store and manage this additional content
    - Boris: 13 | Why: persistence of data, correct tracking of user activity and association with reward. Multiple edge and test cases.