Prepared by: John J. Lee, Tylor Louis, Yicheng Yang, Sean Zarringhalam, Howard Lee

Version 1.0  
October 24, 2013

# **Software Requirements Specification**

**For**

TReW  
(Ticketmaster Resale Widget)

Table of Contents

[1. INTRODUCTION 2](#_Toc370392095)

[1.1 PURPOSE 2](#_Toc370392096)

[1.2 Document Conventions 2](#_Toc370392097)

[1.3 Intended Audience and Reading Suggestions 2](#_Toc370392098)

[1.4 Project Scope 2](#_Toc370392099)

[1.5 References 2](#_Toc370392100)

[2. Overall Description 2](#_Toc370392101)

[2.1 Project Perspective 2](#_Toc370392102)

[2.2 Product Features 2](#_Toc370392103)

[2.3 User Classes and Characteristics 3](#_Toc370392104)

[2.4 Operating Environment 3](#_Toc370392105)

[2.5 Design and Implementation Constraints 3](#_Toc370392106)

[2.6 User Documentation 3](#_Toc370392107)

[2.7 Assumptions and Dependencies 3](#_Toc370392108)

[3. System Features 3](#_Toc370392109)

[3.1 Price Recommendation 3](#_Toc370392110)

[3.2 Price Trend 4](#_Toc370392111)

[4. External Interface Requirements 4](#_Toc370392112)

[4.1 User Interfaces 4](#_Toc370392113)

[4.2 Hardware Interfaces 4](#_Toc370392114)

[4.3 Software Interfaces 4](#_Toc370392115)

[4.4 Communications Interfaces 4](#_Toc370392116)

[5. Other Nonfunctional Requirements 4](#_Toc370392117)

[5.1 Performance Requirements 4](#_Toc370392118)

[5.2 Safety Requirements 4](#_Toc370392119)

[5.3 Security Requirements 4](#_Toc370392120)

[5.4 Software Quality Attributes 4](#_Toc370392121)

[6. Appendix 5](#_Toc370392122)

[6.1 Glossary – to be defined later 5](#_Toc370392123)

1. INTRODUCTION
   1. PURPOSE

The purpose of this Software Requirements Specification (SRS) document is to provide a description of the functionalities of the Ticketmaster Resale Widget (TReW) system. This document shall cover the basic requirements of the Ticketmaster Resale Widget system, uses cases and their intentions. This document shall show early mockup of the User Interface (UI) as well as a prototype of the system integrated with the Ticketmaster website. This SRS document shall also cover the hardware, software, and other technical dependences for the system.

* 1. Document Conventions

This document may contain terminology which readers may be unfamiliar with. See Appendix A (Glossary) for a list of these terms and their definitions. This document shall refer to the Ticketmaster Resale Widget as TReW from here on out.

* 1. Intended Audience and Reading Suggestions

This document is intended for developers, clients and all individuals participating in any manner with the TReW project. Readers interested in a brief overview of the product can read Parts 1 and 2 of the document, which give an overview description of the product’s functionality. Readers interested in the detailed requirements and technical aspects should read Parts 3 and 4. Part 5 provides information of the non-technical aspects and external requirements such as quality attributes and security.

* 1. Project Scope

The TReW system shall provide a platform for users to resell their tickets that they bought from Ticketmaster. It should help users determine a fair market price for their tickets. The system should encourage users to buy tickets for events longer in advance, which in turn would bring more business to Ticketmaster.

* 1. References

1. Overall Description
   1. Project Perspective

The TReW system is an add-on widget in the Ticketmaster event page that visually displays to the ticket holder how much their ticket should be sold for. The resale prediction shall be provided from other subsystems: The Recommender and Service modules. TReW system shall communicate with only the Service module, which in turn shall determine the recommendation from the Recommender module.

* 1. Product Features

The following list shall offer a brief explanation of the primary features and their functionalities of the TReW system.

1. Overview of Assets
   1. Show overview of the event that the user has ticket(s) for.
   2. Show user history for past sales.
   3. Show current ticket resale trends for event.
2. Recommendation
   1. Recommend resale price at the present time.
   2. Give prediction of price trends with statistics and/or diagrams.
   3. Advise the user on whether to sell the ticket(s) or not.
   4. User Classes and Characteristics

The users of the TReW system are the Ticketmaster users who need to resell their event tickets. The TReW system is meant to provide the user with information to price their event tickets for maximum value while increasing chances for the ticket to sell. The system should be simple and effective.

While there are many different types of sellers, there are two main types we will focus on. Users who have a long-term intention of selling their tickets for maximum profit, and users who unexpectedly need to sell their tickets as soon as possible. Our widget is intended to be helpful for both of these user classes.

* 1. Operating Environment

The TReW system shall be a javascript add-on to the Ticketmaster “My Account” page. The system shall be written for browser-based applications to be portable across platforms with browser capabilities. The view should also be responsive to different display screen sizes, such as desktop, tablet, mobile, etc.

* 1. Design and Implementation Constraints

Our team shall not have access to Ticketmaster’s current framework or system.

As an add-on, our product has limited display size.

The UI should be clean and easy to use, while providing enough information to help users make selling decisions.

* 1. User Documentation

There shall be an icon, which will display a simple help message to describe what each value means when it is clicked.

* 1. Assumptions and Dependencies

The system depends on the service module to provide information to the user. The system also depends on the main event webpage itself to collect information from the user to provide to the service module.

In terms of external libraries, the system will depend on jQuery v2.0.3 for its convenient Javascript functions. It will also depend on Twitter’s CSS Library, Bootstrap v3.0.2. To process its graphs, the widget will make use of the javascript library d3 v3. We will also be using Node.js v0.10.22 with the Express v3 framework to set up the web server.

1. System Features
   1. Price Recommendation

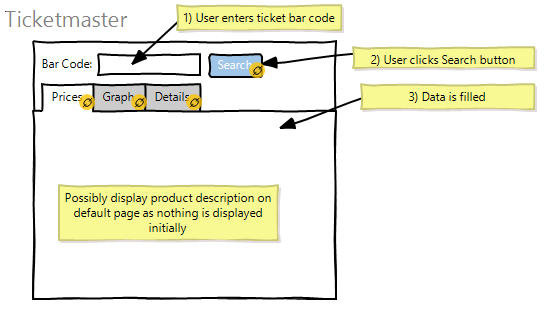
User can select his/her event from a list of events associated with his/her account. Some overview of the selected event comes up, like event time, location and a seat map. Then the user can select the seat of his/her ticket. The widget will then give a recommend resell price based on the seat and time until event, etc.

* 1. Price Trend

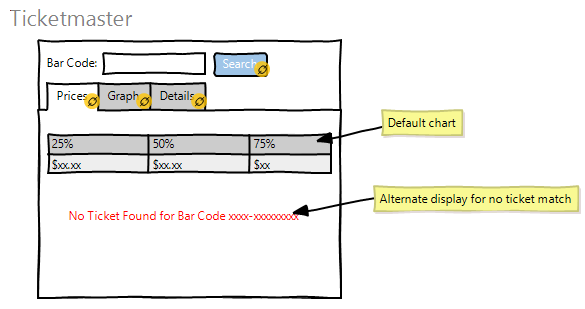
User can select event to see the predicted price trend of resale, so he/she can decide to sell his/her ticket(s) now or wait until a higher price is offered.

1. External Interface Requirements
   1. User Interfaces

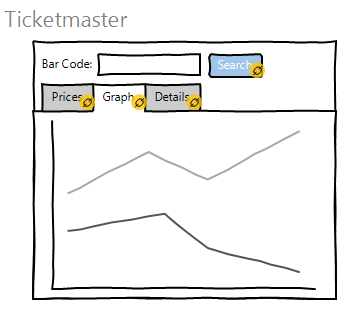
The UI should be easy to use. The UI should only require the user to input a ticket barcode. The UI shall match the style in accordance with the Ticketmaster website. To accommodate both basic and advances users, the widget will be split into tabs. Basic price information will be included in the first tab, advanced graphical information in the second tab, and further details in the third tab. The mockup below demonstrates the basic layout of the widget.



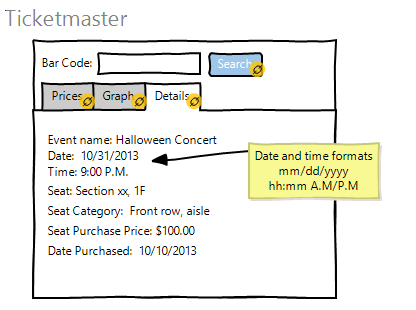
The prices tab should closely follow the design below, which displays the ticket price information in terms of quartiles of likelihood of selling.



The graph tab will be used by more advanced users, because it will convey statistical information for the user to interpret. The x-axis will be in units of time until event, whereas the y-axis will be in dollars of ticket price. For every incremental unit on the x-axis, the graph will display a vertical box-plot of the ticket’s likelihood of selling at a given price. These vertical box plots will align sequentially next to each other, and a line intersecting their midpoints will look like the graph below. This line represents the ticket price for a 50% likelihood of selling, plotted against time until event.



The third tab is displayed below, and will provide additional information on the event the ticket is for, such as the date, seat number, etc.

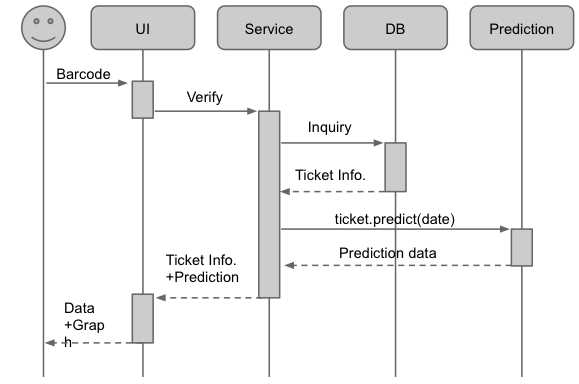


* 1. Software Interfaces

The TReW system shall send user’s query to the service module and the service module shall provide needed data for the UI.

* 1. Communications Interfaces

Using AJAX requests, the widget will send a barcode to the services team. The services team will send back a JSON object containing the event details, and an array of prediction objects. Each prediction object will be composed of a day of sale, an array of prices, and an array of those prices’ likelihoods of selling. The following demonstrates the chain through which this communication takes place.



1. Other Nonfunctional Requirements
   1. Performance Requirements

The whole resale system shall be able to handle the highest peak traffic Ticketmaster receives to date plus 10%. The prediction result and trend diagram shall be displayed within 10% of Ticketmaster’s current load time for the given event page.

* 1. Safety Requirements

The display of the widget will be simple and visually non-intrusive to ensure that users will not feel sick from using the widget.

* 1. Security Requirements

The user authentication shall be done by the Ticketmaster website. Use of user information shall comply with Ticketmaster’s policies. The widget will not be dealing with the actual transactions, it will only provide basic ticket information and analysis.

* 1. Software Quality Attributes

The TReW system shall be designed to be reusable on future mobile device applications. The widget’s priority is ease-of-use, since it is presented as an addition to a pre-existing page for the user’s convenience. It should also be supported in the most widely used browsers. The browsers that will be focused on are Google Chrome, Mozilla Firefox, Internet Explorer, and Apple Safari.

1. Appendix
   1. Glossary

Software Requirements Specification (SRS)

User Interface (UI)