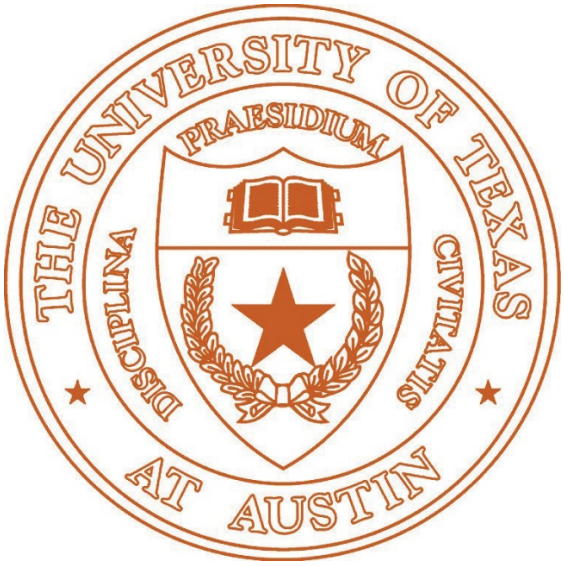
University of Texas at Austin, Cockrell School of Engineering
Software Architecture – EE 382C.7



Assignment # 3
Final Written Report
May 07, 2016

Gabrielson Eapen
EID: EAPENGP

Table of Contents

4. Final Project Report

4.1 Domain

The goal of this project is to create a Web-based travel agency that provides airline ticket price and schedule information across multiple carriers. Travelers can view and filter comparisons between providers based on price, length of travel, and schedule. Travelers can then make reservations directly through the system and then review and modify those reservations that have been made through the system. Additionally, travelers expect to be able to share those reservations with others traveling with them. In some circumstances, such as when a travel administrator or coordinator is making reservations, travel plans may also be made by a planner, acting on behalf of the actual traveler.

Travelers increasingly want to plan, execute, and review travel reservations in one system, which is available from all of their devices. They also expect to be able to manage all aspects of that trip within a user interface with an intuitive format and streamlined processes. The vision for this system is geared toward end users with little to no formal travel booking experience.

Because the application gains revenue through successful travel reservations, the reservation process should be a natural and fluid process, with as little chance for frustration as possible.

4.2 Prioritization of Stakeholder Qualities/Constraints along with Quality Categories

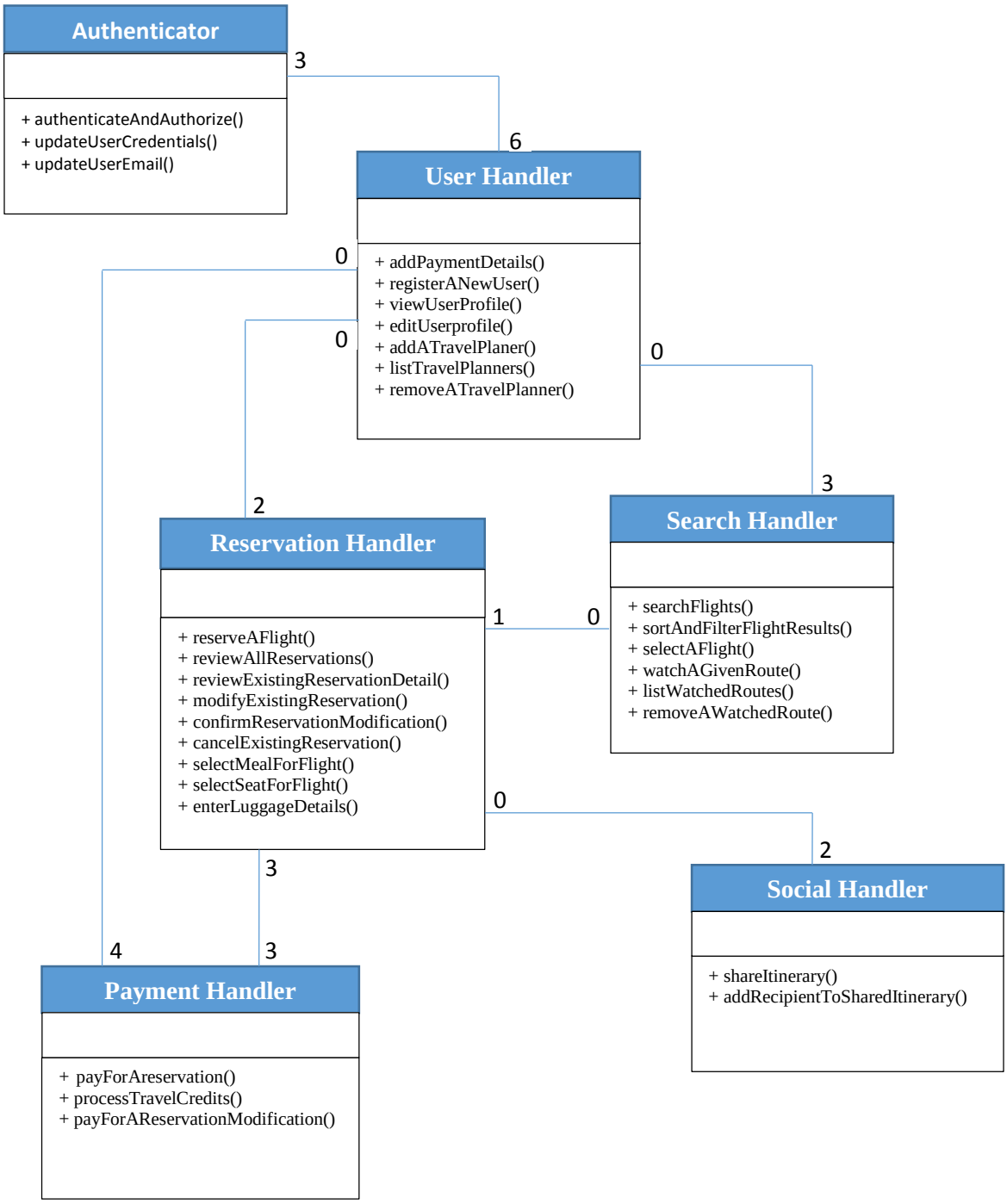
The following is a prioritized table of Stakeholder Qualities and Constraints presented in the problem domain.

Priority	Need/Quality	Classification	Priority Justification
1	The site (application) should be available to end-users for the entire duration of their access (visit). Specifically, the system needs to be available 24x7 with a four 9's (99.99%) uptime goal.	Availability	If travelers are unable to access the site (application) when they need it, they are unable to schedule and plan trips. Besides inconveniencing the traveler, it adversely affects the reputation of the site.
2	The application user interface (UI) should be simplistic and intuitive to use so that end-user needs minimal to no training in order to navigate the site.	Usability	If travelers find it difficult to use or navigate the site, they are less likely to use it. Ease of use leads to higher end-user acceptance and satisfaction.
3	The user should not have to wait for extended periods of time at any point when using the system.	Performance	Since the application is available globally 24x7, end-user usage patterns can vary. But regardless of application load, all application functions must perform consistently and complete in the specified amount of time.
4	Authenticated access to the application should be secure	Security	The application must comply with industry standards in storage and

	and user credentials should not be compromised. SSL can be used to secure the sensitive information. Sensitive personal or payment information should be encrypted when stored in the database.		transmission of sensitive personal information to avoid any legal liability. Specifically collection, transmission, and storage of payment (CREDIT CARD) information or cardholder data must comply with the Payment Card Industry Data Security Standard (PCI DSS).
5	The application should be able to handle large numbers of users and have ability to increase capacity on demand in response to growth or during peak travel seasons.	Scalability	To ensure consistent performance, when user load or activity increased, the application infrastructure needs to be elastic and scale to meet the higher load. This helps satisfy priorities 1 and 3 as well.
6	The application should be easily extensible	Extensibility	As our company desires to agile and aggressive in the market place, it needs to be able to add new features or capabilities with minimal design effort. So tight coupling between modules should be avoided.
7	There should be regularly scheduled backups of critical user and system data to allow for recovery whenever needed.	Data backup and recovery	Hardware failures happen and it would be embarrassing to inform customers that we have lost their data. If possible, continuous or almost real-time backups should also exist to allow restores to a specific point-in-time.
8	The application should be created with proper coding principles and design patterns with controlled releases from development to test to production.	Maintainability	Similar to priority 6, incremental updates and/or bug fixes need to have controlled rollouts (or rollbacks) without availability disruption.
9	Delivery of the application must occur within the agreed upon time frames.	Project Schedule	Our company desires to be agile and aggressive in the market place. However, this is a saturated market and meeting the above goals is more important than schedule.
10	The application should be created within the specified budget without cost overrun.	Project Cost	Adhering to the stipulated budget goes a long way in inspiring confidence in our company's long term viability. But this is the least important as we are a start-up and will take projected costs whatever they may be to the investors.

4.3 Business Blueprint Derivation

4.3.1 Graphical Representation of Components and their Relations



4.3.2 Bootstrap – Focus on Search and Users

Availability is our most prioritized goal and of all the components in the original bootstrap, the search component needs to be the most nimble and have the highest availability. In the original blueprint ancillary functions that are not core to search are present in this component along with an associated data item. The ancillary functions pertain to watching routes and we feel that we could optimize the original blue print by moving the highlighted three functions from Search to User along with the Watched Routes data item

Figure 1: Components in the Original Bootstarp

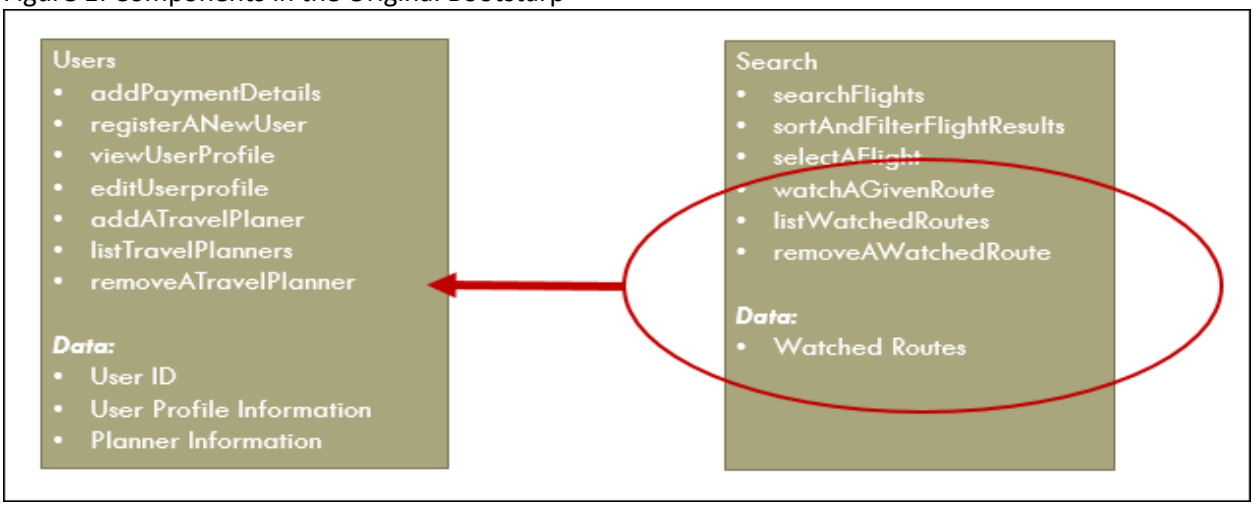
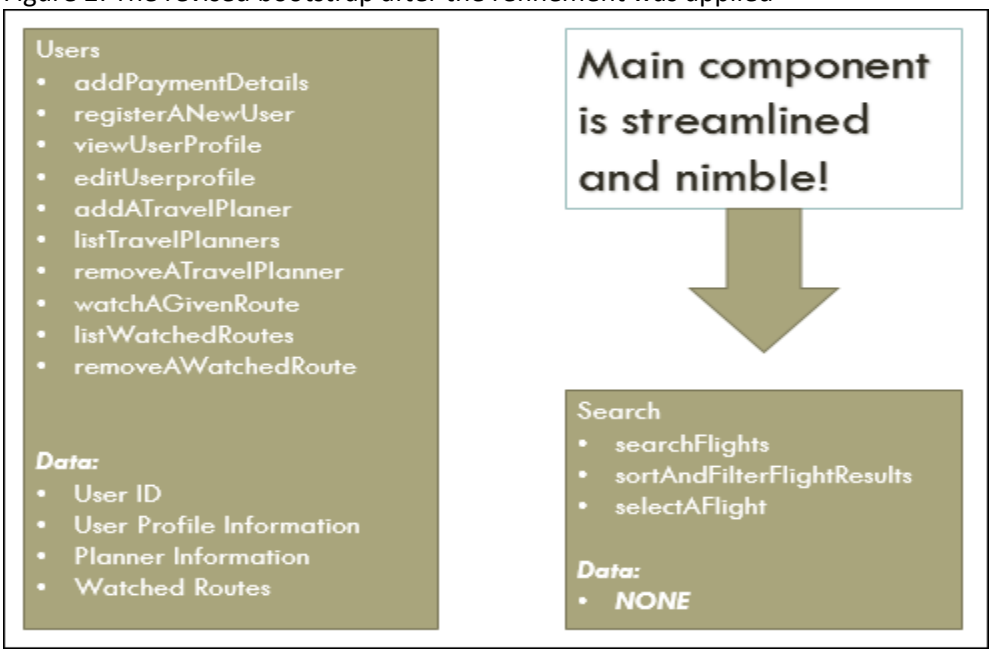


Figure 2: The revised bootstrap after the refinement was applied



After better understanding our application and its needs, the proposed refinement is a more optimal one. Search and its functions are core to the application and does not even require a user to be logged in. On the other hand, watching a given route is tied to a user's desire to monitor that route for a period

of time. That activity along with the list of watched routes intuitively now feels more like a user profile (or preference) persisted setting that needs to be kept with all other user settings. Furthermore when the external process runs to perform a search on a watched route, it is a background process that does not need to be as nimble as a real-time search being performed by a user.

4.3.3 Quantitative Analysis

*4.3.3.*1 Number of Inputs/Outputs between components

Component	# Data/Events In	# Data/Events Out	Total
Authenticator	3	6	9
User Handler	9	7	13
Payment Handler	7	3	10
Reservation Handler	6	5	11
Search Handler	0	1	1
Social Handler	2	0	2

*4.3.3.*2 Number of dependencies between components

Component	# Components to which this component sends or from which this component receives data/events
Authenticator	1
User Handler	5
Payment Handler	2
Reservation Handler	4
Search Handler	2
Social Handler	1

*4.3.3.*3 Degree of Cohesion

Component	# Functions	# Functions that receive all inputs and send all outputs within component (not counting external)	% Functions that receive all inputs and send all outputs within component (not counting external)
Authenticator	3	0	0%
User Handler	10	3	**30%**
Payment Handler	3	0	0%
Reservation Handler	9	6	66%
Search Handler	3	1	**66%**
Social Handler	2	0	0%

4.3.4 New Size and Complexity Metrics

4.3.4.1 New Number of functions allocated to a component

Component	# Functions Allocated
Authenticator	3
User Handler	10
Payment Handler	3
Reservation Handler	9
Search Handler	3
Social Handler	2

4.3.4.2 Number of data elements allocated to a component

Component	# Data Elements Allocated
Authenticator	2
User Handler	1
Payment Handler	2
Reservation Handler	4
Search Handler	1
Social Handler	2

4.3.4.3 Number of components in the blueprint

Components in Blueprint
6

4.3.4.4 New Component Complexity

Component	# Functions	# Data Elements	# Inputs and Outputs (across all functions)	Complexity
Authenticator	3	2	14	19
User Handler	10	2	32	40
Payment Handler	3	2	16	19
Reservation Handler	9	4	27	40
Search Handler	3	0	14	9
Social Handler	2	2	5	9

4.3.5 Qualitative Analysis

- By reducing coupling, we reduce potential dependency failure
- By limiting the size of the component, it is easier to scale independently, cluster, provide redundancy.
- By reducing complexity, we can avoid disruptive misconfigurations, resource over-utilization, and software errors.

4.3.5 Trade Offs

- User Handler:
  - Complexity goes up by 37%
- Potentially harder to provide Maintainability for this component.
- However, this is mitigated by the increased cohesion and decreased coupling for the component.
- Regardless, the User Handler does not affect the basic ability of a user to search for a flight and to make a reservation, which are the two primary functions on the site.