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

Assignment # 3
Derivation and Evaluation of Solution and Deployment Blueprints

May 07, 2016

Gabrielson Eapen
EID: EAPENGP

Table of Contents

3. Derivation and Evaluation of Deployment and Solution Blueprints

3.1 Deployment Blueprint #1 (DB1)

Part: 1

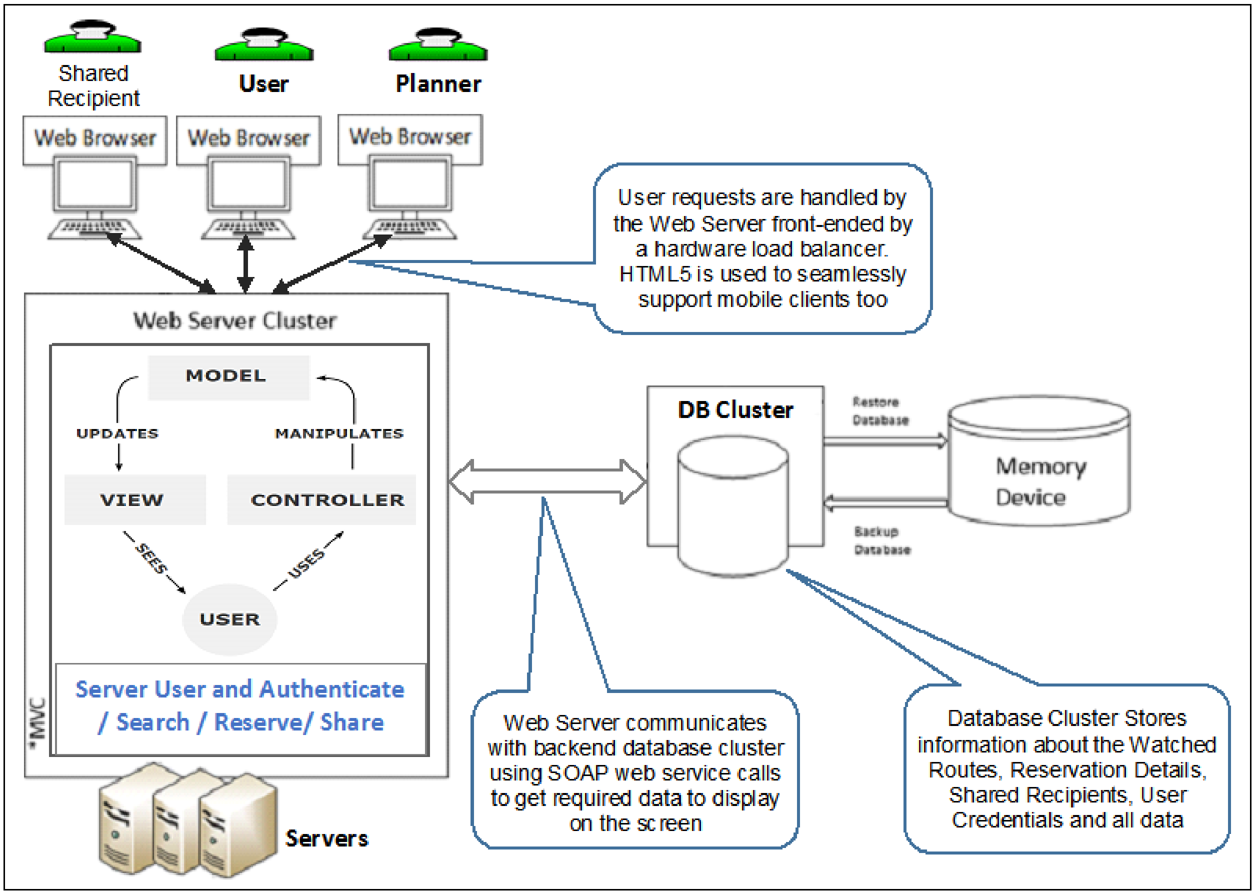
Satisfaction of domain functions by solutions

SB Solution Component	BB Functions and Data satisfied
Server User and Authenticate / Search / Reserve / Share	Functions: • Authenticate and Authorize • Search for a flight • Select a seat for a flight • Enter number and types of luggage • Select meal for a flight • Modify an existing reservation • Confirm reservation modifications • Select a flight • Reserve a flight • Sort and filter flight results • Review existing reservation details • Review all reservations • Cancel an existing reservation • Register a new user • View User Profile • Edit User Profile • Update User’s Email Address • Update User’s Credentials • Add a Travel Planner • List Travel Planner(s) • Remove a Travel Planner • Watch a given route • List watched routes • Remove a watched route • Share Itinerary • Add Recipient to a Shared Itinerary • Pay for a reservation • Process travel credits • Pay for a reservation modification • Add payment details Data: • User Credentials • User Email • Credit Amount • User Reservation List

	• Trip Reservation Details • User ID • User Profile Information • Planner Information • Payment Information • Watched Routes

Allocation of Solutions to deployment components

DB Component	SB Components Allocated to DB Component
Web Server Cluster	Server User and Authenticate
Web Server Cluster	Server Search
Web Server Cluster	Server Reservation
Web Server Cluster	Server Share
Database	N/A
Web Browser	N/A



User requests are handled by the Web Server front-ended by a hardware load balancer. HTML5 is used to seamlessly support mobile clients too

Web Server communicates with backend database cluster using SOAP web service calls to get required data to display on the screen

Database Cluster Stores information about the Watched Routes, Reservation Details, Shared Recipients, User Credentials and all data

Figure 1: Graphical Depiction of DB1[1]

[1] *MVC: Server User and Authenticate / Search / Reserve / Share modules all use MVC architecture

Rationale

- Satisfaction of Stakeholder qualities (with priorities):
    - Availability: Because of the multiple servers, redundancy exists and there is no single point of failure. An individual server can be down, but the remaining servers keep the application or service online. Even when there is need for planned maintenance, each server can be serviced individually without taking application down.
    - Usability: MVC is a software architectural pattern for implementing user interfaces on computers. Because of separate view, model and controller interconnected components, different views can be created for Shared Recipient, User (Traveller) and Planner with the use of same model and controller and thereby improve usability.
    - Performance: User requests are distributed among servers with the use of a hardware load balancer in web farm. Keeping the ratio of requests per server at lower level, reduces the work each server has to do. Contention is minimized and it does not cause additional slowdown or performance impact.
    - Security: Web service proxy is used to call web service from UI which secures the system. Also, XML encryption is used by SOAP web service to secure user confidential data and payment information. Furthermore, the data is stored on the separate database server, which offers a more control of security than on client machines.
    - Scalability: Load balancer and multiple servers can manage increasing number of user requests. By increasing the scalability will not affect performance of the system. Also if required new server can be added easily and load balancer will start distributing requests on the new server as well to manage scalability of the system.
    - Extensibility: Everything about MVC has been designed with extensibility in mind. Anything in its processing pipeline is replaceable and if needed new services can be injected into the main pipeline. For example, view component can be extended without making changes in model or controller. Similarly new services or features can be added without changing UI or database.
    - Data backup and recovery: Additional memory disk (storage) is used to back up data regularly. In recovery scenarios or if data accidentally deleted we have ability (rollback) to restore our database.
    - Maintainability: User interface requirements tend to change more rapidly than business rules. Users may prefer different colors, fonts, screen layouts, and levels of support for any new devices or mobile clients. Just as we saw with extensibility, because the model does not depend on the views, adding new types of views to the system generally does not affect the model. As a result, the scope of change is confined to only the view component.
    - Project Schedule: MVC divides application into three interconnected components called view, model and controller. So that makes it easy to estimate time for a task and track it to maintain the project schedule.
    - Project Cost: As we are using web server cluster and database cluster, the onetime installation cost may be high. But it is the least prioritized constraint as we are a new player in this industry segment. It can also be argued that maintenance cost, development time will be less with MVC and this can be used to recoup other costs. In addition most of the applications and software used are open source which incurs less cost.

Design inspired by:

- MVC design pattern: Model view controller separates the application logic from the user interface and the control between the user interface and the application logic. Planner, Shared Recipient and User (Traveler) can have different views from same functionality.
- Object Oriented Architecture: Inheritance is used for similar based functionality or related functions such as select a meal for the flight or select a seat for the flight.
- Web server redundancy: Web server requests are distributed across multiple Servers in cluster. It increases availability, redundancy and performance of the system.
- Web Service Proxy Pattern: Proxying web services secures web service call from the User Interface to secure the application.

References:

- MVC design pattern: https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller
- object-oriented architecture: https://www.classes.cs.uchicago.edu/archive/2013/spring/51050-1/lectures/lecture.1/lecture.1.pdf
- Web server redundancy: https://rimuhosting.com/knowledgebase/rimuhosting/load-balancing-and-failover
- Web Service Proxy Pattern: https://docs.mulesoft.com/mule-user-guide/v/3.5/web-service-proxy-pattern

Notes:

- Independent software modules run under the web servers that support Server User and Authenticate / Search / Reserve / Share modules. These modules satisfy domain functions. However, there may be multiple implementation components involved in satisfying a domain function and an "Implementation Task Decomposition" decomposes a domain function into steps.
- There is no way to directly show BB functions in the DB. BB functions are distributed in solution component and showed how it will be fulfilled in the DB