SUPINFO Academic Dept.

SUPINFO International University

INSTITUTE OF INFORMATION TECHNOLOGY

Group Projects
B.Sc. - Train Commander

Contents

2015-2016

Version 1.0
Last update: 8/12/2015
Use: Students/Staff
Author: Samuel CUELLA

TABLE OF CONTENTS

1. Project Overview

Rail Commander is a new travel company focused on train tickets reservations. The plan to start their business with France as the first target and then expending to other European railroads.

You are the subcontractor for their technical infrastructure.

2. Functional Expression

Rail Commander is a web-based ticket booking service that allow users to travel from one city to another, using railroad stations. Based on a departure/arrival stations, it can find the most efficient (shortest/cheapest, two options) solution, and handle interconnections between railroad networks.

The engine has knowledge of all trains on all networks, including their departure/arrival/stops, along with hours and price of each segment. Using these information and interconnection stations, it can generate any existing itinerary.

2.1. Software development

A common backend engine can be used using two front-ends: Web and mobile client. Both applications provide the same features.

The objective of Rail Commander is to get rid to all clutter and provide a clean and lean interface that only keeps the bare essentials: Booking a train ticket from a departure/arrival within an hour/date range.

2.1.1. Searching

Users can search for tickets from the web interface. They can enter:

- Departure (city/station)
- Arrival (city/station)
- Departure date
- Departure time range (before 10 A.M, between 11 A.M and 2 P.M, etc.)

The engine will then return a list of trains (or set of) matching these criteria, along with their actual details (departure time, total travel time, price, etc).

2.1.2. Ordering

Once the user has selected his ticket, he can order it online using paypal (use Paypal sandbox in your POC). He can create an account if he wants, but that is not a mandatory step.

Once the order validates, the user can print his ticket and gets a confirmation mail with the PDF.

2.1.3. User accounts

Users can create accounts on the site if they want to. They can do so by:

- Creating an account with their email address.
- Use their existing Facebook account

- Use their existing Google account

The personal account will allow users to:

- Review they travel history.
- Do a search using an existing travel to replicate
- Print tickets/receipts of all travels.

2.2. Supporting architecture

The system architecture that powers the application is critical to the business: Downtimes means loss of money. The architecture must be designed to provide a fast and reliable service.

The application will run on two different physical sites, in active/passive mode. This will allow the application to survive a complete disaster on one of the sites. If the main (active) site goes down, the secondary site will take up the load to ensure business continuity. The secondary site will only be used in case of an emergency. It doesn't need to be able to process all the load the main site can.

2.2.1. Virtualization

To allow more flexibility, all machines will be virtualized. This will help migrate data from one site to another and to distribute the load on more physical server as load temporary increases. In your POC, each site will be represented as a cluster of two hypervisors.

2.2.2. Engine cluster

The core application logic should be powered by a high-availability cluster that always make the service available. If one node goes down, the service is still available on another node.

2.2.3. Web client cluster

The web client must also run on an high-availability cluster. This layer runs the actual web-application delivered to users. End users however don't directly interact with it.

2.2.4. Web CDN

The application makes use of a Content Delivery Network which is a set of webservers (www1,www2,etc.) acting as surrogates for the real web client. This CDN also does load balancing by automatically redirecting users from www.domain.tld to www1,www2, etc.

3. Deliverables

Students should include the following elements in their final delivery:

3.1. Software development

- A zip archive with the project source code. The source code must also come with the build system used (Project file, autotools...), if any.
- Project documentation, based on the template.
  - Technical documentation explaining your choices and/or implementation choices/details on the following items (at least):
    - Path finding
    - Database layout optimizations
- Deployment manual

The first document is an academic document. Address the reader as a teacher, not a client. The last one (deployment manual) should address the reader as a user. These documents can be in French or in English, at your option.

3.2. Supporting architecture

- Architecture schema, with all implemented components.
- All server configuration files.
- Anything you find relevant.
- Project documentation, based on the template.
  - Technical documentation explaining your choices and/or implementation choices/details on the following items (at least):
    - Virtualization
    - Clustering
- Step-by-step deployment manual

The first document is an academic document. Address the reader as a teacher, not a client. The last one (deployment manual) should address the reader as a user. These documents can be in French or in English, at your option.

4. Graded Items

The project will be graded as follows, on a 220/200 scale:

- Software development (110 points)
  - Engine core (40 points)
    - The engine knows about trains and stations. (5 points)
    - The engine can find a way from A to B. (20 points)
    - The engine returns a list of trains matching criteria. (5 points)
    - The engine can process a reservation through paypal. (5 points)
    - The engine can generate a PDF ticket and send confirmation mails. (5 points)
  - Accounts (10 points)
    - Users can create an account with their email address. (1 point)
    - Users can use their Facebook to create an account. (2 point)
    - Users can use their Google to create an account. (2 point)
    - Users can review their travels. (2 point)
    - Users can replicate a travel from an previous one. (1 point)
    - Users can print tickets/receipts for all travels. (2 point)
  - Website (30 points)
    - Users have access to all features through the website. (30 points)
  - Mobile application (20 points)
    - Users have access to all features through the mobile application. (20 points)
  - Bonus features (10 points)
    - Bonus features done by the students. (10 points)
- Supporting architecture (110 points)
  - Virtualization (30 points)
    - Each site has a virtualization infrastructure to power the app. (10 points)

- Two sites are cooperating in a active/passive mode. (20 points)

- Engine cluster (20 points)
  - The application core is high available and can survive losing nodes. (20 points)
- Web cluster (20 points)
  - The web client is high available and can survive losing nodes. (20 points)
- Web CDN (30 points)
  - Clients never use the web client directly but through the CDN. (15 points)
  - There is load balancing between the surrogates. (15 points)
- Bonus features (10 points)
  - Bonus features done by the students. (10 points)