SE31520 Assignment: Car Insurance System

Edgar Ivanov
edi@aber.ac.uk
Department of Computer Science, Aberystwyth University

December 6, 2013

Contents

Introduction 3
Architecture of the underwriter 3
Architecture of the RESTful broker 5
Test strategy 5
Self-evaluation 5

Introduction

There was a task to implement prototype system which would allow to user request a quotation for a car insurance policy. As part of this system we needed to write two applications. One is underwriter which would represent insurance company and a broker which would collect quote premiums from the different insurance companies. However for this assignment task was simplified and broker needed to communicate just with one insurance underwriter, such approach affected some of my design decisions. For example, unique number which allows user to retrieve previous quotations is stored on the underwriter side, however in real world application it would be generated for the user and stored on the broker side and then linked to all underwriters. Broker had to be developed by web technology of our choice for which I have chosen PHP, underwriter had to be developed in ROR, store data in relational database management system and use ORM support. . This document will describe design of the broker and underwriter systems, testing strategy and includes self evaluation section.

Architecture of the underwriter

Write a section on the architecture of the underwriter application and rationale for decisions made. As part of this, produce a UML diagram(s) that shows the architecture of your application. The design diagram I used for the CSA application discussed in class might be a useful starting point. I drew mine using Powerpoint, but feel free to use another tool or even to draw neatly by hand!

Underwriter application was developed using Ruby on Rails. It is a RESTful web application using exclusively JSON for the representation of the content and data exchange. HTTP is used for the communication and supports all usual HTTP methods like GET, PUT, PATCH, POST, DELETE. At the beginning I tried to implement XML support for the representation exchange but faced some issues which I couldn't overcome. After a further reading about data formats like XML, JSON, YAML I choose to use JSON since it seemed to be lightweight, human-readable, easy to implement on the broker side, as well as CRUD operations with support of JSON baked in. Figure 1 represents database design. Users table holds customer information like name, surname, DOB etc. Vehicles table holds information about the car: registration number, mileage, car value, it is linked to the main users table by user_id field.

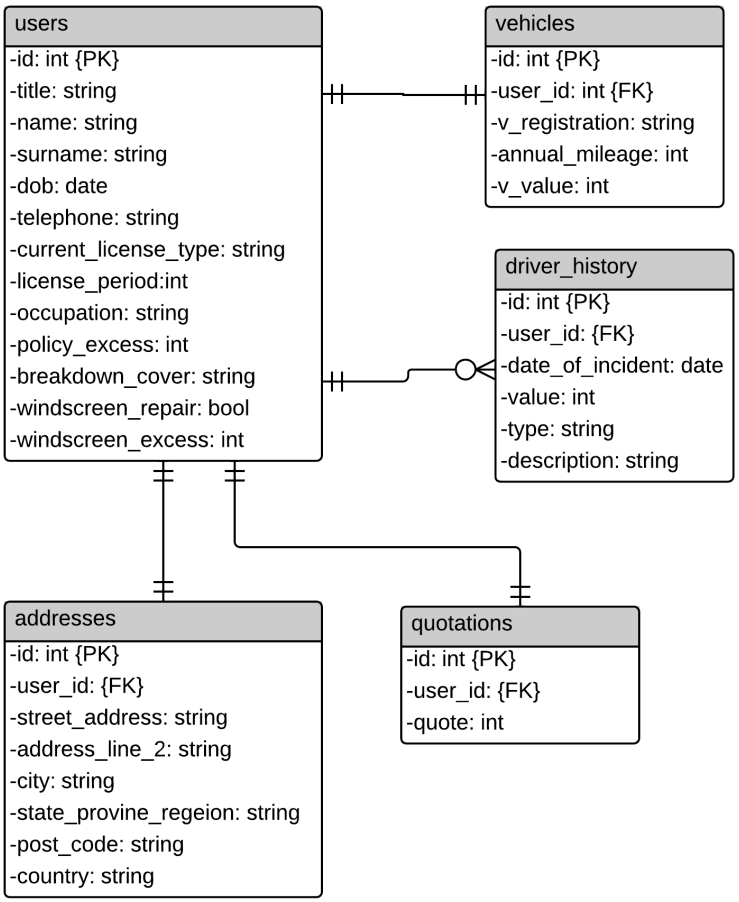

Figure 1: Database Design

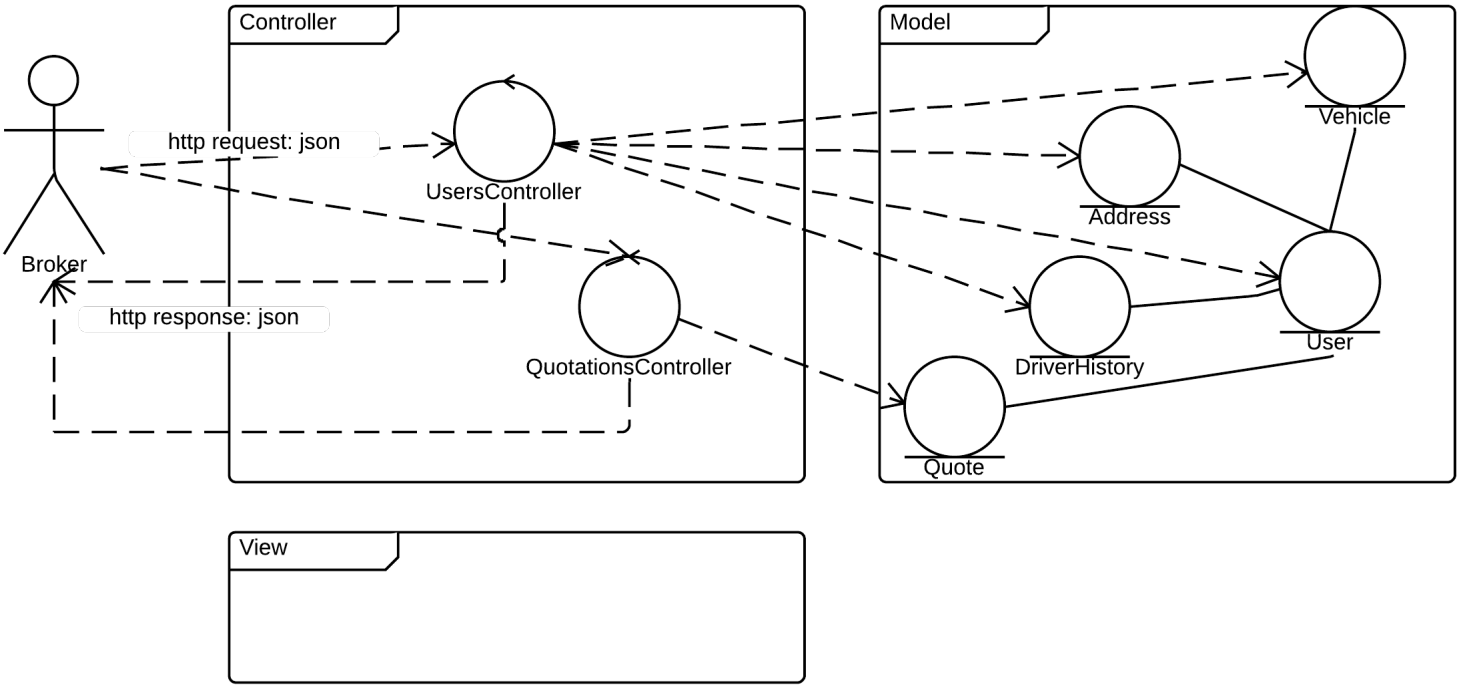

Figure 2: Class Design

Architecture of the RESTful broker

Write a section on the architecture of the RESTful broker client, providing rationale for decisions made.

Test strategy

Write a section on your test strategy. IMPORTANT: Provide a screencast of your underwriter application and RESTful broker client working (some free screen-casting tools can be found online). This must focus on the broker to underwriter interworking and be no longer than five minutes long.

Self-evaluation

Write a self-evaluation section. Say what mark you should be awarded and why. Say what you found hard or easy, and what was omitted and why. Provide an analysis of your design and the appropriateness, or otherwise, of the technologies used.