MTAT.03.295 – Agile Software Development
Regular Exam – 17 December 2019
Part 2. Practice

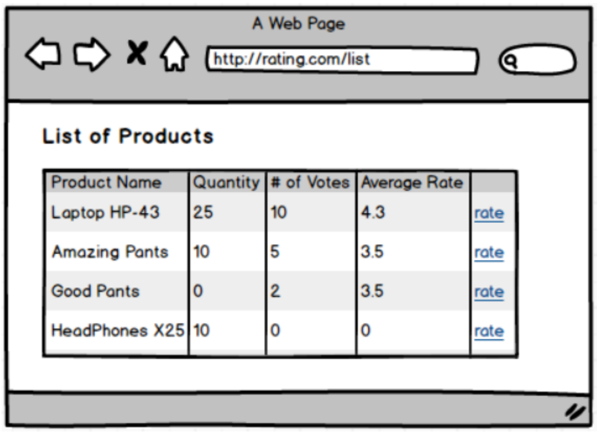
Notes:

- You are allowed to access any resource in the web
- You are not allowed to communicate with anyone during the exam in any way (except with the lecturer).
- You must create a Bitbucket repository and add me there as a collaborator (with admin rights). Push changes on your solution as frequently as possible, so as to demonstrate that you followed the BDD-TDD cycle in your work.
- If you find that some information is inaccurate and that you need to make some additional assumptions, please write them down (you can add them as a README file in your root folder on your Bitbucket repository).

Product's Rating System

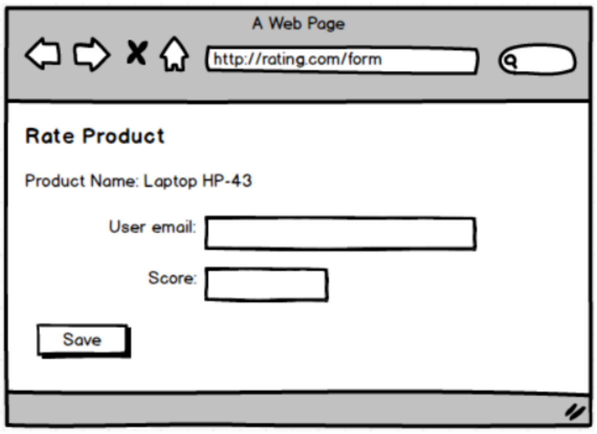
Our team has been hired to implement a system for the users of online store to rate products. For first release of the software, we will implement a Phoenix application with the basic set of features: maintaining a product catalog, rate products and sort products based on ratings.

The product manager has agreed with the customers on the following wireframes that illustrate the underlying process.



List of Products

Product Name	Quantity	# of Votes	Average Rate	
Laptop HP-43	25	10	4.3	rate
Amazing Pants	10	5	3.5	rate
Good Pants	0	2	3.5	rate
HeadPhones X25	10	0	0	rate



Rate Product

Product Name: Laptop HP-43

User email:

Score:

In the first form, the user can see the list of available products sorted by average rating (in decreasing order). If the average ratings are the same, the products are sorted alphabetically by their names. Note that for each product, the following information is displayed: name (must be unique in the system), availability in the store, number of votes, and average rate. For each product, a link is provided such that a user can rate the product using the second form. To rate a product, a user must provide an email and a score (between 0 and 5), then once the user saves the score, the system should redirect him/her to the page with the sorted list of products.

The system must satisfy the following restrictions:

1. The system must allow a user to rate a product, and update the database accordingly. The system must reject the rating of non-existing products, as well as scores which aren't numbers between 0 and 5.
2. A user is allowed to rate a product only once. Thus the system should reject the rating if the user (identified by his email) already rated such product before. Note that a user is allowed to rate several products in the system.
3. The system must calculate the average ratings. The average rating of a product with 0 votes is by default 0.
4. When displaying the list of products, they must be sorted by average ratings (higher rates first), and later alphabetically (if the average rates are equal).

The task breakdown and corresponding marks are as follows:

- (BDD) Specification of a Gherkin user story describing the selection and rating of a product. (**4 pts**)
- (BDD) Implementation of the white_bread steps. (**3 pts**)
- Setup of the application routes. (**2 pts**)
- Setup & implementation of models (via migrations), and seeding the database with some initial data based on your models. (**5 pts**)
- Implementation of controllers. (**10 pts**)
- Implementation of views and templates. (**3 pts**)
- (TDD) Unit tests for either models / controllers checking the restrictions 1, 2, 3 and 4. Note that a single test may include several requirements. (**8 pts**)