

Practical 2 report

Jorge Ibarra Delgado
School of Computer Science
University of St. Andrews

22 April 2015

Contents

1	Introduction	2
1.1	The practical	2
1.2	Purpose and Scope of the document	2
2	Requirements	3
2.1	Functional	3
	Functional requirements priorities	5
2.2	Non-Functional	5
	Availability	5
	Performance	5
	Resilience	5
	Reliability	5
	Security	5
	Maintainability	6
3	Logical View	7
4	Development View	10
5	Process View	11
6	Physical View	12
7	Implementation	13
7.1	Screenshots	14
7.2	results	14

Chapter 1

Introduction

1.1 The practical

The objective of the practice is to put our knowledge of software architecture obtained during these weeks into use by providing a architecture specification description document and the implementation of part of the system according to this document.

1.2 Purpose and Scope of the document

This document will describe the functional and non-functional requirements and the architecture of an e-commerce system that allows to do the basic functionality of a common system of this type: manage products, manage customers and their addresses, place orders and keep track of them, provide seller contact information, etc.

This architecture will be made by using diverse UML diagrams to represent the logical, development, process and physical views of the system, which are part of the 4+1 viewpoint. This is done so different stakeholders can be aware of what the system will be consisting on, in other words, they will know what are the components, how are they connected and the reason behind the choices made.

Chapter 2

Requirements

2.1 Functional

System Administrators have the following attributes: user name, password, e-mail.

System Administrators can perform the following actions:

- Enter to an administrative interfaces allowing.
 - Search, add, update and deletion of Users, which can be Customers or Administrators.
 - Search, add, update and deletion of Products.
 - Search, add, update and deletion of Providers.
 - Search, add, update and deletion of Voucher codes.
 - Update of Store details.
 - Configuration of an e-mail high sales alert that can be triggered by selecting a number of sales of an item in a determined amount of time, which can be set on days or hours.
 - Configuration of an e-mail low stock alert that can be triggered by selecting a minimum number of a certain Product in stock.
 - Configuration of an e-mail information alert on sales of a Product or group of Products. This can be set n-times a day at certain hour.
 - Consult the sales of a Product in a selected period of time.
 - Consult the sales of a group of Products, allowing visualisation of the best sellers and the worst sellers.
 - Consult the stock of Products.
 - Consult the Product delivery status from Providers.
 - Consult Customers order history.

- System Administrators can search, add, update and delete products.
- To enter the administrative interface, the System Administrator must provide a user name/e-mail and password.

Customers will have the following attributes: user name, password, first and last name, e-mail, phone number, a set of Addresses, a set of Payment Methods.

Customers can perform the following actions:

- Create their own profile, providing the data mentioned before as their attributes.
- Search, add, update and deletion of Addresses.
- Search, add, update and deletion of Payment Methods.
- Search for Products and consult Product information.
- Add Products to a Shopping Cart.
- Place an order using the Products in the Shopping Cart and checkout by selecting the desired address from the Customer's Address Book and a Payment Method from its set of Payment Methods.
- Consult their own order history.
- Consult information on how to contact the Store.
- To be capable of enter to their own Profile, add Products to the Shopping Cart and place Orders, Customers must be logged in by using their user name/e-mail and password.
- Cancel orders.

Products will have the following attributes: code, name, a set of Types, a Provider and price.

Voucher Codes attributes are: name and type. The type will be set according on what the Voucher can do:

- Provide a discount in the total of an Order at checkout.
- Provide a discount in only certain Product.
- Provide a discount in a group of Products.

The voucher code must be provided while placing the Order, before checkout.

Providers will have the following attributes: name, e-mail, phone number.

After checking out, the system must send an e-mail to the Customer confirming the order has been placed.

After cancelling an order, the system must send an e-mail to the Customer confirming the order has been cancelled.

Functional requirements priorities

Because the stakeholders need a constant updating on the product, we have prioritised some of the requirements to show them prototypes of the system periodically. These will be the first requirements to work on for the first prototype. Further requirements will be added in later updates of this document.

1. Get the list of products.
2. Create a customer account.
3. Login as a customer.

2.2 Non-Functional

Availability

Because Customers can shop 24/7, the system must be available 100% of the time.

Performance

The system shall be capable to perform correctly even having multiple simultaneous connections.

Resilience

The system shall be capable to work in case of faults

Reliability

The system reliability must increase throughout the software life, errors in software must pass from 1 error per month, to 1 error per year.

Security

Information concerning Customers, specially their payment details, must be encrypted or use a safe third party library/software. Data must be protected as well.

Maintainability

Clean code principles shall be encouraged to be applied in the code, so the software can evolve without premature ageing.

Chapter 3

Logical View

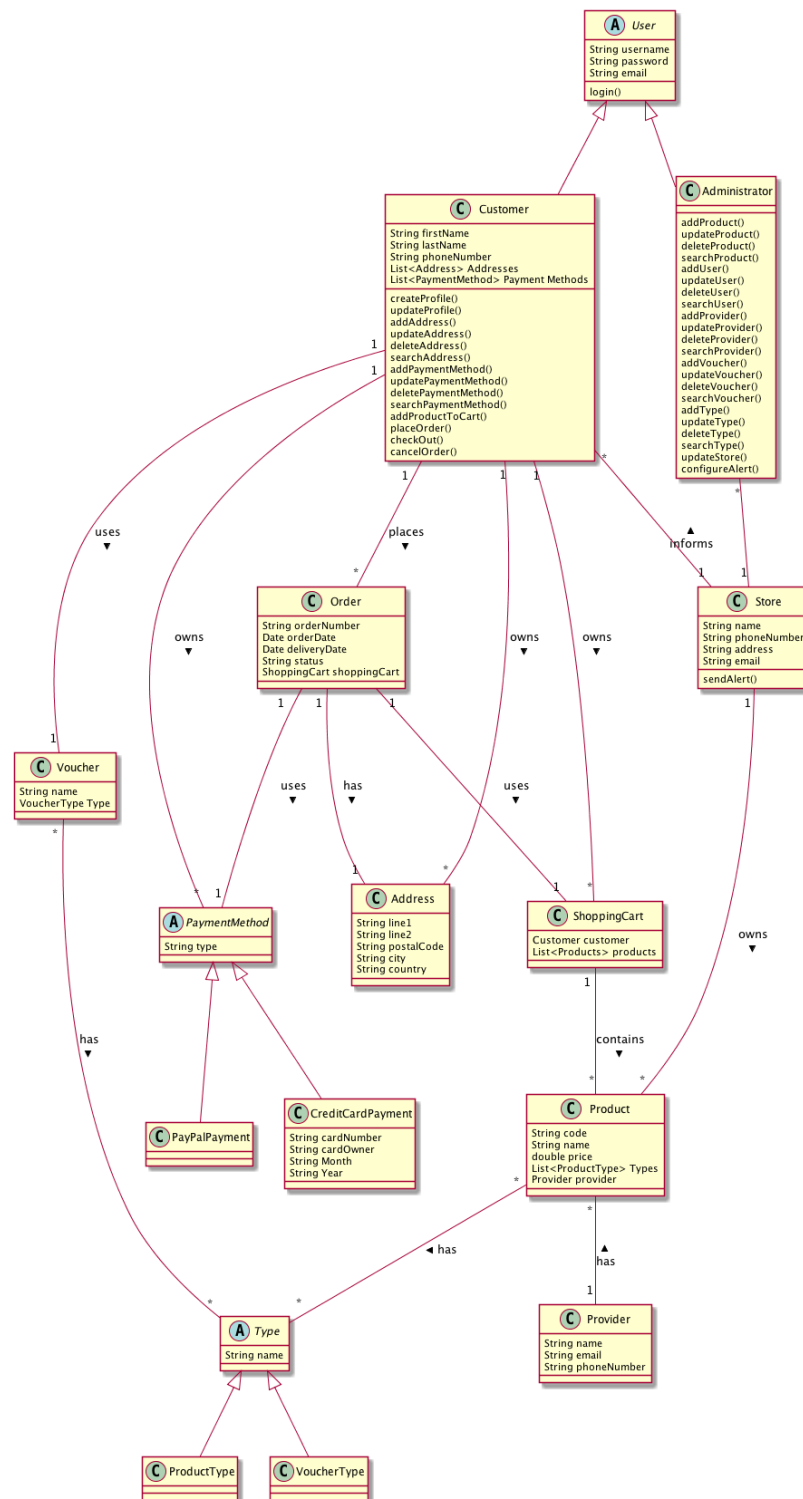


Figure 3.1: Class diagram

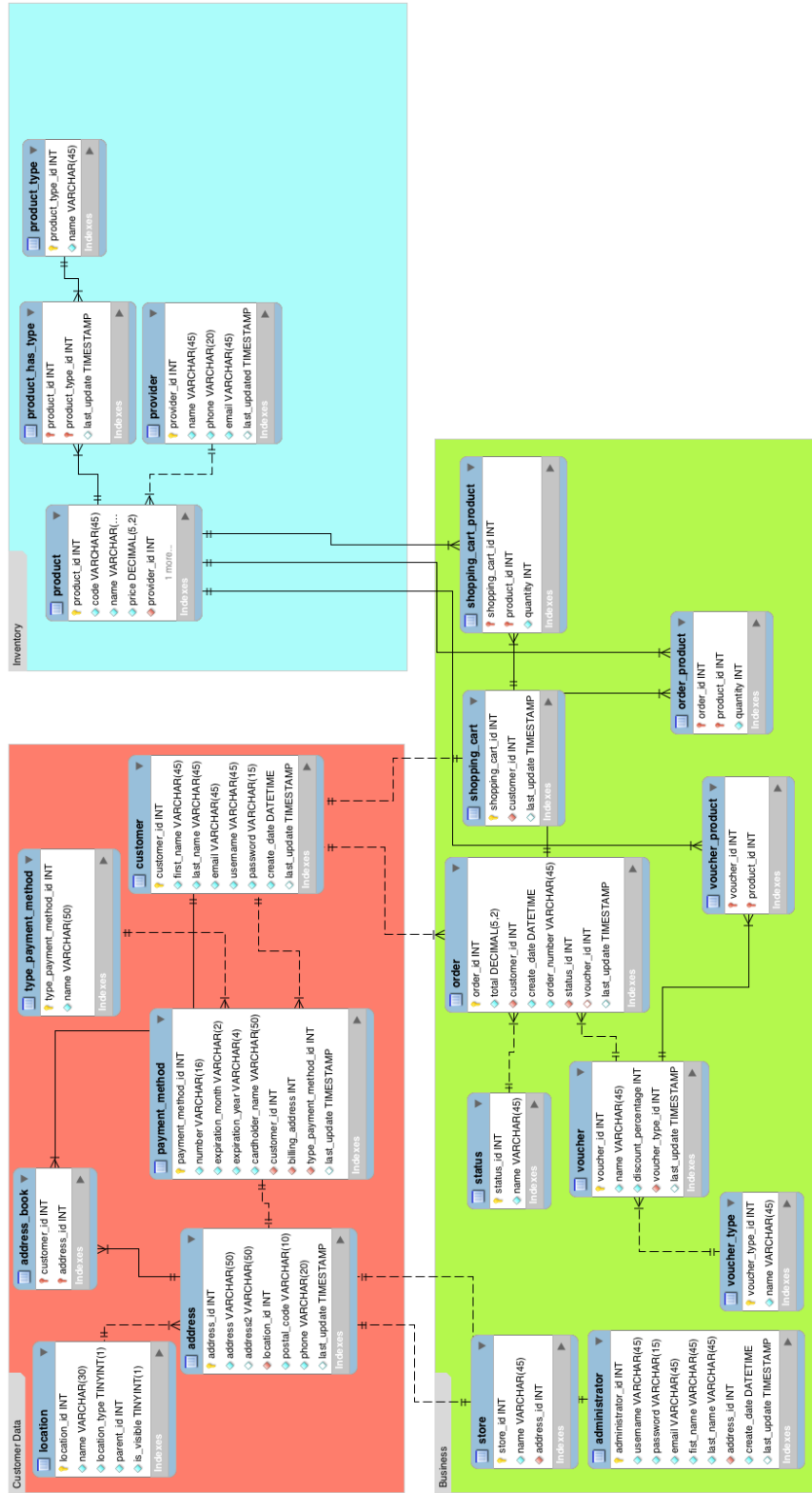


Figure 3.2: Entity-Relation diagram

Chapter 4

Development View

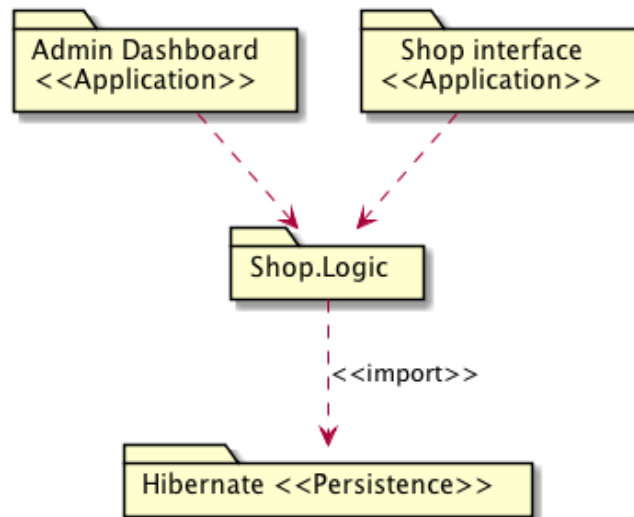


Figure 4.1: Package diagram

Chapter 5

Process View

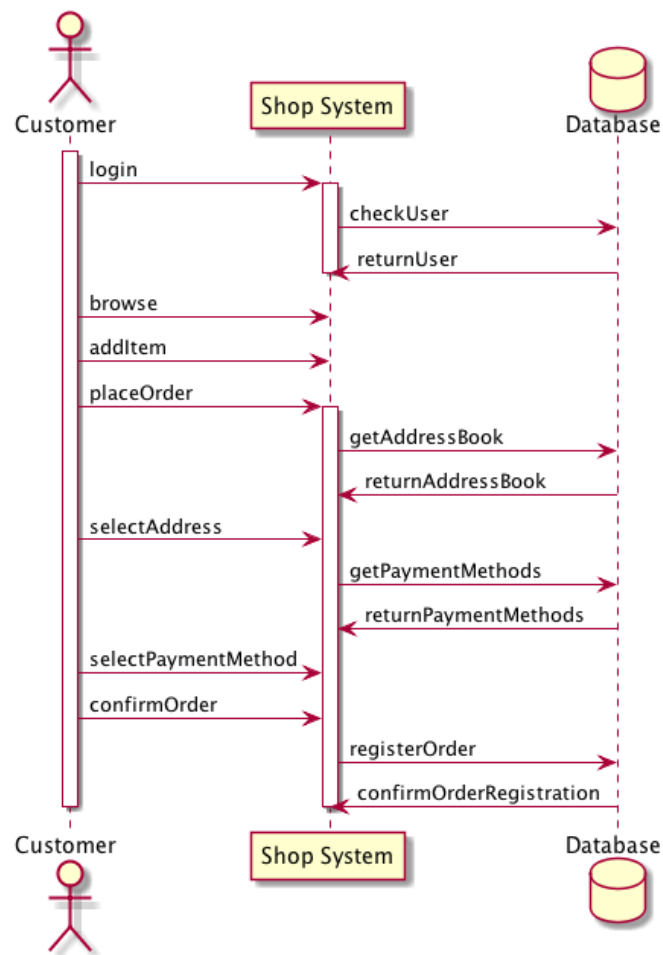


Figure 5.1: Sequence diagram of a customer placing an order

Chapter 6

Physical View

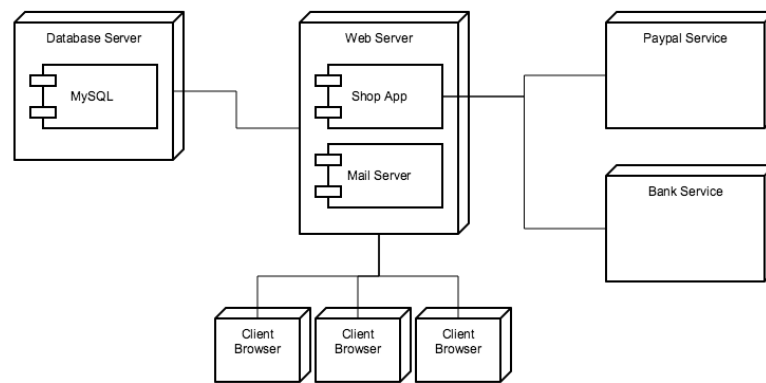


Figure 6.1: Deployment diagram

Chapter 7

Implementation

The application will be made using a client - server architectural style to comply with the requirements, this can be seen by looking at figure 6.1. It will be made using the following tools:

- Javascript, jQuery and Bootstrap on the client side. Bootstrap was chosen because of its ease of use and the possibility to create beautiful responsive layouts
- Java on server side
- We have selected Apache TomCat as application server because of previous and most recent experience with it
- Hibernate as an interface between the database and the application, because of that, we will use Spring as a framework of the application. Also, from previous experiences, and for the short time to implement the system, we think it's a better option than using EJB
- MySQL database
- IntelliJ will be used as an IDE, this decision was made after having bad experiences with NetBeans and finding that Eclipse was not so user friendly because of its big capacity of being configurable. IntelliJ so far has proven to be a lightweight version of Eclipse with many of its benefits already out of the box

For the first iteration of the system implementation a list of products will be provided to the database, so it would be possible to perform testing and demo the system to stakeholders.

The development of the application will be made using an agile process. That implies the content of this document will be constantly changing according to the evolution of the development process. Contact with stakeholders will be constant as well in order to solve questions and receive constant feedback.

7.1 Screenshots

This section contains some screenshots of the system.

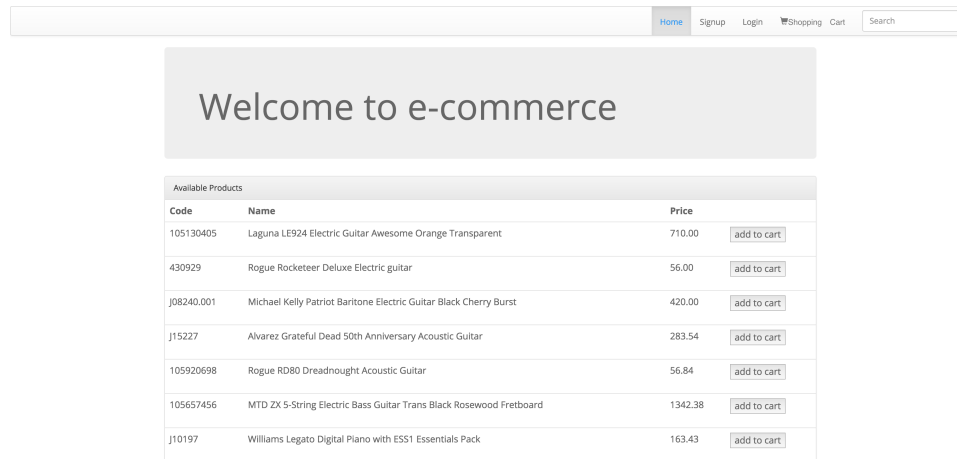


Figure 7.1: Home screen, in here we can see that the products are shown, this decision was taken because it would be nice for the customer to see the things he can buy since the first screen. Details about login in or registering can be done after if the customer is interested in a product

Sign up *Its absolutely quick!*

Customer Signup Form

Email Address

Password

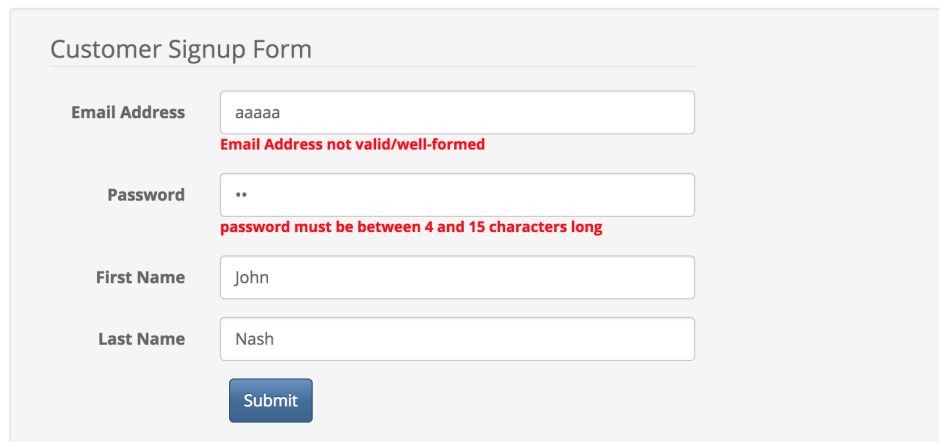
First Name

Last Name

Figure 7.2: Customer registration form. As seen in figure 3.2, there is a username and email address for the customer, we have thought that for the time being, they will have the same value since each email address is unique

7.2 results

As a result of the demanded architecture, we have created a prototype of the e-commerce system that already has a pretty decent (but a bit too generic)



The image shows a web form titled "Customer Signup Form". It contains four input fields: "Email Address" with the value "aaaaa", "Password" with the value "**", "First Name" with the value "John", and "Last Name" with the value "Nash". Below the "Email Address" field, there is a red error message: "Email Address not valid/well-formed". Below the "Password" field, there is a red error message: "password must be between 4 and 15 characters long". At the bottom of the form is a blue "Submit" button.

Figure 7.3: Demonstration of errors in input. The system uses some of the advantages of using the Spring framework, providing some validation to our web app

UI. The application can register customers in the database and can verify their existence by providing username (email) and password. We can also populate a table with product information. This information can be found as part of the MySQL model.

The contents of the project will be provided as well as the MySQL model and a sql script containing the model and the data to populate a location Table, the name of the schema is ecommerce.