**Web Enterprise – Pre-Proposal**

Jean-Philippe Valois, Brad St-John, Tyler Lavergne and Harrison Shirreff.

Submitted to

D. King

In partial requirements

to the completion of

CST8218 - Web Enterprises

Lab section 300

Algonquin College

September 19th 2017

**Table of Content**

Title Page 1

Table of content 2

Introduction 3

Log in 3

Role Architecture 3

User Description 3-4

Client 3

Employee 4

Supervisor 4

Admin 4

Data management 4

Privacy 4

Data Validation 5

Versions 5-6

0.1 5

0.2 5

0.3 6

1.0 6

**Web Enterprise – Proposal**

General idea would be an online store app that will allow client to view two different locations, items in stock and their location, add them to cart and put items on hold for 24hrs. The application would be bilingual from top to bottom (French/English), allow you to pay for your purchase through PayPal or from debit/credit cards and for items to be moved to the closest store based off client’s location using geolocation or simply by having a map with two points and asking/confirming with the client which location would be easier for pick up.

Team is composed of Jean-Philippe Valois, Brad St-John, Tyler Lavergne and Harrison Shirreff.

Client **< Employee < Supervisor < Admin**

**Log in:**

Can be as advanced as everyone having an account with username/email, first name, last name, address, security question, encrypted password and credit card information for easy check out or as simple as only having the employees and above having a log in. Placing an order would then prompt the client for credit card information, name, phone number and pick up location.

**Everyone:**

- Change language.

- Browse items that are store in at least two different location

- View stock

-Place items on hold which would remove it from stock for 24 hrs. This process could require client to be logged in or simply prompt for a name and phone number

Clients**:**

- Manage their cart: allowing them to add, remove and save items for later, place orders which could then prompt them to pay and take items out of inventory.

**Employees:**

-View all currently placed orders which would display the order number, a list of items and how many are ordered, their location to see if they need to be moved to the other location (which wouldn’t change anything as far as the app is concerned).

**Supervisors:**

* Change prices
* Add to stock/order
* Manage orders marking them as complete, make adjustments to already placed orders (cancelling orders/items or adding new items to existing orders)
* View client first name and last name or username and list of order numbers associated with their account as well as allowing them to reset their passwords so clients could create a new one should they have forgotten their security question and not have an email setup.

**Administrators:**

* View all users and their personal information with password and credit card information remaining encrypted
* Change user’s permissions (promoting/demoting)
* Manage which items are displayed (if they’re seasonal)
* Add new items or remove existing ones.

**Data management:**

Placed orders could be stores on the server. After they’ve been placed and picked up they could then be erased. Carts and previous orders made by a certain individual could either be stored on the server or as cookies.

**Privacy:**

Everyone’s password and credit card information would be encrypted.

Users wouldn’t interact with one another and only admins could see user’s information.

Supervisors would only see a list of users and an option to reset their passwords.

**Data validation:**

Seeing as users wouldn’t have to interact with each other. User information would most likely only validate that the necessary fields aren’t are valid for required fields (aren’t entirely composed of empty spaced and/or symbols) and that they have the correct format. For example, credit card numbers having the 16 digits, email addresses having @ followed by a “.” (.ca, .com, etc.)

We could also be annoying and require passwords to be a certain length, have capitals and lower case, numbers and symbols.

**Versions**

**version 0.1**

Front end: Rough menu of all features to implement and the different screen

Middle end: Have prompts for all features even if there isn’t any interaction between the front end and the back end.

Where one works methods/functions that clients, employees and supervisors will use and the other focuses on methods that admins will use and ensuring that there aren’t any issues communicating with the database inquiries

Back end: get the database set up and have all the data populated with at least 2-3 items. Begin looking at ways to improve that architecture to make data as easily accessible as possible

**version 0.2**

Front end: Come up with the CSS for each screen and menu

Middle end: Have one individual finalize any issues connecting with the database and the interactions available based on permissions

Have the other double check that all methods and functions are properly modifying the stock and that users have the methods needed to modify the data

Back end: Populate the database and finalize the architecture

**version 0.3**

Front end: Finalize internationalization for prompts and maybe have the option for items to have pictures.

Middle end: Making purchases through the store, look at saving cart information as cookies or on the server. Geolocation?

Back end: Have encryption set up where it needs to be, make adjustments to database if we decide to store information on database/if it’s feasible based on how much time’s left

**version 1.0.**

Front end: Have a simple yet elegant menu that’s user friendly, bilingual and gets the job done.

Middle end: Have the methods, functions and prompts that displays the information correctly, that validates all manipulation to the data, that gives you access to the right information based upon your permissions. Allows for purchases and orders to be made even if it’s unclear as right now the extend login credentials needed by users to use the app and have it save information or where that will be stored.

Back end: Have a populated database with regular items, items that are on sale, are seasonal and that are temporarily on hold for 24hrs stored in two different stores