Department of Electrical and Computer Engineering
University of Puerto Rico
Mayagüez Campus

ICOM 5016 – Introduction to Database Systems
Fall 2013

Term Project – Mobile App for an eBay-style Site
Phase I – Conceptual and UI Design
Due Date: October 2, 2013, 11:59 PM

Objectives

1. Understand the design, implementation and use of a mobile app .
2. Understand the use of the E-R model for database application design.
3. Gain experience by implementing applications using layers of increasing complexity and fairly complex data structures.
4. Gain further experience with mobile app programming, and REST APIs .

Overview

You will design, implement and test a mobile app that implements an auction site in the same style as eBay. This site will be implemented using database, web, and mobile apps technologies. The application will have a client-side and a backend side (server side). The client side is the mobile app per se, which you can implement in either: a) iOS, b) Android, c) jQuery Mobile, d) Titanium, e) kendoui, or f) pure HTML 5 (e.g., bootstrap) (**not recommended**). The backend consists of a web/application server and a database server. The web/application server can be implemented with either: a) Node.js, b) Java Play, or c) Apache and Django. The database server can be: a) MySQL, b) PostgreSQL, c) MariaDB, or d) MS SQL Server. The communication between client and backend (server) must happen through a REST API. You can use helper libraries such as AngularJS, Backbone, or Knockout. Communication with the database server must be done through raw SQL statements. **NO ORM can be used**.

Note: No application will be accepted in a language or toolkit other than those indicated before.

The app site will enable the user to perform operations such as:

- Browse categories for goods (i.e., electronics, books, clothing, sports).
- Search based on product name
- Buy products instantly
- Place a bid to a product
- Sell a product

- Accept a bid
- Add products to a shopping cart and then purchase them.
- Allow users to view their account, add shipping information, add credit card or bank information.
- If the person is bidding, see a list of items on which bids have been placed.
- If the person has sold items they can see their history of sold items.
- Rank reputation of seller
    - **** - Excellent
    - *** - Good
    - ** - Regular
    - * - Deficient
    - No start – Dishonest

  You don't need to create a summary for the rank, other than indicating which % of users give the seller **** star. You do have to provide a list of the rankings given by the users posting on the seller (e.g.,) :
        - David123 - ****
        - JoeLi - **
        - Ned - ***

It is a good idea to play with the eBay app to explore ideas for the UI. You are free to implement that UI organization of your choice, e.g., Tab Bars, Drawers (aka "Facebook style menu"), Navigation Group, etc. You might also decide to focus on a specific device: tablet or smartphone. Be creative! But keep things simple!

Your solution MUST follow the Model-View-Controller Design Pattern. In this scheme your solution will be organized as follows:

1) View – the mobile app that runs on the device. It **MUST** NOT interact directly with the database. The mobile app will communicate with the controller code by means of a REST API.

2) Controller – the controllers will be implemented by means of Node.js, Play, or Django. Each controller will get a request, create a business service object to handle the request, collect the results from the methods in this business service object and forward the results as JSON data and with the proper HTTP status code.

3) Model – a set of business service objects that implement all tasks and access to the database system. These will be implemented by means of Node.js, Play, or Django.

You are required to use the facilities of the ECE Computer Labs at UPRM, or your personal laptops. **No project can be implemented to run on desktops at your homes/apartments. I will not grade them.**

<u>**Operations to be supported**</u>

Your app will have two kinds of users:

1) Regular users – Persons that can browse for merchandise and place purchase orders, or sell items
2) Administrators – IT personnel in charge of supporting and maintaining the database. These can add new categories, erase categories, or modify the information of the users (recover password, remove user, etc), view reports.

<u>Regular users operations</u>
A regular user will be able to perform the following tasks:

1) Create an account. The information for the user MUST be included: customer name, account number, mailing address, billing address, credit card information. Note: A customer without an account can only browse and view items, and place them in the shopping cart. However, at the moment of placing an order, the customer **MUST** be logged on to the account.

2) Browse categories of products available. The following categories must be supported:
   a. Books
      i. Children
      ii. Fiction
      iii. Technology
      iv. Business
   b. Electronics
      i. TV
      ii. Audio
      iii. Phones
      iv. Cameras
      v. Video
   c. Computers
      i. Laptops
      ii. Desktops
      iii. Tablets
      iv. Printers
   d. Clothing
      i. Children
      ii. Men
         1. Shirts
         2. Pants
         3. Socks
      iii. Women
         1. Shirts
         2. Pants
         3. Dresses
   e. Shoes
      i. Children
      ii. Women
      iii. Men

    f. Sports
        i. Bicycles
            1. Frames
            2. Wheels
            3. Helmet
            4. Parts
        ii. Fishing
        iii. Baseball
        iv. Golf
        v. Basketball

3) Browse products available within the categories. Products will appear unordered. However, there MUST be an option to allow the products to appear in increasing order of price, or sorted by brand, or sorted by product name.
4) View the details of a product. These **MUST** include: Product id, product name, instant price, bid price, description, model, photo of product, brand, and dimensions.
5) If the product is for bid, need top keep a list of offers (can only be seen by seller), and next acceptable bid must be computed based on this information.
6) For a user, see items for which he/she is bidding
7) For a user, see items being sold
8) Place an item in a shopping cart.
9) View and update shopping cart.
10) Place an order for items in the shopping cart.
11) Get an invoice of an order that has been placed. This MUST be shown once the order has been placed, but MUST also be kept in the records for future reference.
12) Accept a bid once the bidding time has ended.
13) Inform the winner of bids
14) Login, view, and modify account settings (see item 1 for settings).

Administrator Operations

An administrator of the auction app will be able to:

1) Create, view, and modify customer and administrator accounts.
2) Report indicating total sales by day, week and month (all)
3) Report indicating total sales grouped by product
    a. Ask for one of day, week or month
4) Report indicating total revenue grouped by product
    a. Ask for one of day, week, or month

Deliverables for Phase I

In phase I you will need to provide a git repository containing the following:

1) Report containing task analysis, E-R model and description of screen shots: The task analysis consist of a description of each of the operations that will be supported in the site, and a screen shot of the app screen that enables users to perform the given task. The E-R model must illustrate the data to be stored in the site. You must show the whole diagram, and then explain each entity and relationship one by one.

2) Git repository (hosted at github.com) with the working code for app. In this phase, you will provide a working app. However, this systems is not the full fledged system, but rather an initial version with the following capabilities:

   a. Each task can be performed by UI elements in the mobile app.
   b. The actions in the forms, button, and links will trigger a REST call to the appropriate controller.
   c. The controller will respond with JSON results "hard-wired" (i.e., fixed results).

Thus, in this phase there is no interaction with the database server. Also, there is no dynamic interaction in the sense that the actions always return the same value, no matter what options you use (e.g., sort by price).

PROJECT DUE DATE: **11:59 PM – October 2, 2013**.