CS 6314 – Web Programming Languages
Fall 2019
Dr. Mithun Balakrishna
Course Project

A. Project Steps and Deadlines:

- **Project Group Formation**:
  - Due by **Tuesday, October 15th 2019, 11:59pm**
  - A maximum of three (3) students per project group
  - The group should decide on an appropriate group name
  - One group member should submit a document containing the group name and the group member information i.e. Group name and Group member names, via eLearning
    - Please name the document following the convention "ProjectGroupInfo-GROUPNAME.pdf", where GROUPNAME is your project group's name.
    - Submit the document to the "Group Information Submission" assignment inside the "Final Project" folder listed in the course home page on eLearning.
    - Students that want to work on the project individually should also submit this document
  - Students that need help to form a group should meet the Instructor on **Tuesday, 10/15/2019** at **9:45pm** in the class room (GR 2.302)
    - Students that want to work on the project individually do NOT need to do this
- **Project Demo**:
  - Due date: **TBA**
  - Demo sign-up details: **TBA**
  - Submit your project source code and report via eLearning before your group's allocated demo session:
    - One group member should submit a single zip file containing the following via eLearning:
      - Project source code/script file(s)
      - A ReadMe file with instructions on how to access the project demo
      - Project report in PDF or MS Word document format.
    - Please name the zip archive document following the convention "ProjectFinalSubmission-GROUPNAME.zip", where GROUPNAME is your project group's name.
    - Submit the document to the "Project Final Submission" assignment inside the "Final Project" folder listed in the course home page on eLearning.

- Please hand over a hard copy of the project report before the start of your group's demo session with the TA

B. Project Description:

Please design and implement a **responsive web site** and **scalable web application** based on the **Service Oriented Architecture** (SOA) [1]. This project will require the use of **Web Services** [2] for implementing the SOA.

Mandatory Requirements

A. **HTML/CSS/JavaScript**: You are **required** to build your web site's client side Graphical User Interface (GUI) using HTML/CSS/JavaScript. You are **required** to use responsive HTML/CSS/JavaScript templates such as Bootstrap (http://getbootstrap.com) and Foundation 3 (http://foundation.zurb.com), etc.

B. **Server-side Programming**: You can use any programming language for your web site's server-side implementation and your web application's Web Services implementation.

C. **Web Application Domain and Functionalities**: The students will implement the following 2 web applications:

- A real-time stock brokerage Web Application and Website for end users to sell/purchase stocks:

    1. Your brokerage will allow users to buy and sell stocks for a total of 100 companies on weekdays only from 8am CDT to 5pm CDT.
    2. Users will be able to see the current selling price for any company's stock in real-time.
    3. Users will be able to see the selling price history for any company's stock in the following time periods:
        i. Current day or last business day if current day is Saturday or Sunday
        ii. Current week
        iii. Past week
        iv. Month-to-date
        v. Year-to-date
        vi. Past 5 years

4. Users can buy/sell stocks:
    i. One-time
    ii. Recurring based on a set schedule
5. The stocks are bought/sold based on the current price of that company's stock at the time of purchase/selling.

The web application should support the following functionalities via a website:

1. New regular user registration
    a. Username
    b. Password
    c. Physical Address
    d. Email Address
2. Existing user login and logout
3. Regular User
    i. Login
    ii. Logout
    iii. User profile information display and editing
    iv. Forgot password functionality
    v. Add one or more bank accounts
        1. Bank Routing Number
        2. Bank Account Number
    vi. Ability to transfer money from/to bank accounts
    vii. Search and find a stock of interest
        1. Display current Stock price
        2. Display selected Stock price history
    viii. Ability to buy/sell stocks:
        1. Schedule:
            a. One-time
            b. Recurring based on a set schedule
        2. Select one or more stocks and quantity to sell/buy

a. The page should contain a table allowing addition/deletion/modification of rows

ix. Ability to modify/delete a recurring stock buy/sell schedule

x. Page listing the summary of all stocks owned by user

1. The page should contain a table with atleast 4 columns that are sortable

4. Accessible any unavailable page should retrieve a pretty and generic 404 page

- A real-time stock exchange Web Application that return the price for a particular stock
  a. Current price
  b. Price history for given date range

  Note: Students can implement any technique (random, custom function, etc.) to change price of stocks in a real-time manner

D. **Database:** It is mandatory that your project use a database to store all data. There is no restriction on what type of database to use. Any NoSQL database or RDBMS is fine.

The database SQL or ORM request and response information should be available in the Web-Service web/app server logs for the TA to review the implementation of this feature. In addition, the TA might inspect the database's content getting updated via a database SQL console.

E. **Web Services:**
1. The stock brokerage website should make a Web Service call to the stock brokerage web application for any user operation that requires database access (i.e. to retrieve information or add/update information in the database).
2. All calls from the stock brokerage web application to the stock exchange website should be via a Web Service.
3. The stock brokerage website, stock brokerage web application, and stock exchange web application should be all hosted on different web servers or application servers.
4. The stock brokerage website, stock brokerage web application, and stock exchange web application can all reside on the same physical machine.
5. All Web Services should require authentication/authorization to allow only a valid user to access/modify data.

Note: Web Services are defined as platform/programming-language independent, unassociated, loosely coupled units of functionalities that are self-contained and implemented via SOAP/WSDL or RESTful methodologies.

The Web Services request and response information should be available in the both the Website and Web Services web/app server logs for the TA to review the implementation of this feature. The implementation of RESTful Web Services and its authentication/authorization feature can also be shown to the TA via browser-based REST clients such as Postman.

F. **Asynchronous Service:** The "Stock Buy/Sell Processing" Web Service is asynchronous. Its functionality should be supported and implemented using a queue. For example, a "stock sell processing" request to the web service API should result in the request being added to a queue. The relevant "stock selling processor" web service retrieves this request from the queue and processes it.

   Note: The web service performing the actual "sell/buy processing" require authentication/authorization to allow only a valid user to perform this task.

   The asynchronous webservice request, database query, and response information should be available in the Web Service web/app server logs for the TA to review the implementation of this feature.

G. **Other Required Features**: Your web site/application implementation should also include the **two (2) of the following four (4)** features:
   1. High Performance: perform distributed caching. Memcached is a good option for implementing a distributed caching mechanism.
      Cache miss and cache hit information should be available in the web/app server logs for the TA to review the implementation of this feature.
   2. Client-Server Communication Encryption: encrypt the communication channel between the client (i.e. browser), web site server, and Web Services server using TLS/SSL.
      The TA will check the implementation of this feature on the Website web/app server by checking if the URL in the browser address bar contains the HTTPS protocol.
      The TA will check the implementation of this feature on the Web-Services web/app server by:
      - **Examining the web/app server logs for the Web Services request calls being requested and responded to with the HTTPS protocol**

OR

- **Making HTTPS calls to the RESTful WebServices using browser-based REST clients such as Postman**

OR

- **Examining the capture logs of packet analyzers such as Wireshark**

3. Request/Response Compression: perform compression (e.g. gzip) of:
    a. web site server's response to the client
    The TA will check the implementation of this feature by looking at the "Content-Encoding" HTTP response header field either in the browser debug console (a.k.a. inspect element console) or in the Website's web/app server log file
    b. web site server's request to the Web Service server
    Optional: The TA will check the implementation of this feature by looking for the "Content-Encoding" HTTP request header field in the Web-Service's web/app server log file
    c. Web Service server's response to the web site's server
    The TA will check the implementation of this feature by:
        - **looking for the "Content-Encoding" HTTP response header field in the Web-Service's web/app server log file**

        OR

        - **looking for the "Content-Encoding" HTTP response header field in the RESTful WebServices call made using browser-based REST clients such as Postman**
4. Single Sign-On: perform single sign-on using SAML or OpenID/oAuth
    Note: This is tricky given that Web Services require authentication/authorization as well.

C. Project Report

Please write a project report (5 to 10 pages) with the following details:

- An architectural diagram showing how the various components (i.e. client browser, web/application servers, database, cache, etc.) interact with each other in your project
- For each module, a clear description of the various technologies considered and the technology that was finally used in the module development. Also provide a reason why a particular technology was selected
- A clear description of the various functionalities that were available to users on your web site
- A clear description of the Web Services supported by your web application
- A summary of the problems encountered during the project and how these issues were resolved

- Please specify your group name and group member names on the document's cover/start page

D. Project Point Distribution

1. Maximum points available: 100 points
    a. Aesthetics (i.e. look and feel of web application): 5 points
    b. Web site functionality: 30 points
    c. Web Services implementation: 30 points
    d. Asynchronous web service implementation: 10 points
    e. Other required features implementation: 18 points total (9 points per feature)
    f. Group information: 2 points
    g. Project report: 5 points

E. References

[1] "New to SOA and web services" Available at : https://www.ibm.com/developerworks/webservices/newto/

[2] "Understanding Web Services" Available at : https://www.ibm.com/developerworks/websphere/library/techarticles/0307_ryman/ryman.html