CMPE 281 - Team Hackathon

Submit Assignment

Due May 5 by 11:59pm **Points** 50 **Submitting** a file upload **File Types** pdf
Available Mar 18 at 9am - May 5 at 11:59pm about 2 months

Objectives:

In this project, your team of 3-4 members will be building a multi-cloud Starbucks Drink Ordering portal based on the Restbucks CRUD REST API design discussed in class. Each team member will be building a component of the solution in their own AWS account. In addition, the team should also build a PaaS Based Portal deployed either in AWS Elastic Beanstalk or Heroku PaaS.

Resources:

- **RESTBucks CRUD API.pdf**
- **RESTBucks Design.pdf**
- **REST APIs & Microservices - 2017.pdf**
- **https://github.com/paulnguyen/cmpe281/tree/master/restlet/starbucks_v3 (https://github.com/paulnguyen/cmpe281/tree/master/restlet/starbucks_v3)**
- **https://www.starbucks.com/menu (https://www.starbucks.com/menu)**

Requirements:

- The components of your team's solution should include:
  - ***Portal:*** *Heroku/Beanstalk Based Web Application from which Starbucks Orders and be placed and viewed. Orders should be multi-tenant such that each tenant is a different Starbucks Store. The Portal development should be shared among the team members.*
    - ***Implementation Technology*** *can be selected by the team. (i.e. Node.js, JavaScript (React or Angular), Java, Go, Grails, Etc...)*
    - *Portal must make all API requests to the* ***API Gateway****. (hint: Add a tenant/store code in the URL to facilitate routing)*
  - ***API Gateway***:
    - A Kong API Gateway that is deployed on AWS with a **3-Node Cassandra DB Cluste**r.
    - The API Gateway will route all REST API calls.
  - ***Tenant API Back-Ends***:
    - Two or more REST API back-ends that is implemented in different languages/technologies.
    - Each Tenant API back-end is implemented by one team member and deployed to separate AWS VPCs.
    - Implementation of the API should be based on CRUD Restbucks Design. Teams are free to extend this API.
    - Implementation of APIs should be backed by a 3-Node NoSQL database cluster. Each team member can select their choice of NoSQL DB. Team member can chose the same NoSQL DB Technology.
    - Examples of Languages/Technologies are:
      - Java Restlet (can use example from Instructor, but must be modified to persist data in a NoSQL Cluster)
      - Grails REST
      - Node.js
      - Go REST
      - Others acceptable
    - Examples of NoSQL DBs are:
      - Amazon Dynamo
      - Apache Cassandra
      - MongoDB
      - Apache HBase
      - Apache CouchDB
      - Redis, Riak, Neo4J
      - Others acceptable

Team Work Distribution:

- **Portal** - All team members should contribute to this project.

- **API Gateway** - One team member should work on this service.
- **REST API Back-Ends** - One team member for each tenant/back-end.

Grading:

- **50 Points** - GitHub Link from each team member will be individually graded based on regular activity/commits of source code, notes, diagrams, designs, journal entries in wiki form, etc...
- **10 Points (Extra Credit)** - For Top 10 Teams selected during Hackathon Demo Day
- **10 Points (Extra Credit)** - For Winning Alliance from Top 10 Teams during Hackathon Demo Day

Submission:

- **PDF Document with the following links:**
  - Link to Portal (to be Demo'ed during Hackathon Day)
  - Link to API Gateway used by Portal
  - Link to GitHub Project shared by the team for Portal code
  - Links to GitHub Project Source Code from each team member
  - Links to REST API End Point from each team member