Cloud Computing Coursework

1. Development of a Cloud Software as a Service

The goal of the coursework is to make you apply the concepts and software development methods and frameworks seen in class in order to develop a Cloud Software as a Service (SaaS). The coursework requires you to install, develop and test a Cloud SaaS in a virtualized environment according to the guidelines described in this document. You should upload your final scripts and a technical report to describe the functionality of your solution following the guidelines of the coursework brief document.

2. Online auctioning system.

Your task is to develop a RESTful SaaS for an auctioning system where users sell items while other users bid for these items. You should install, test and document your developments as described in Section 3. Your software should support the following **Actions**.

Action 1: Authorise users to access the auctioning API using the oAuth v2 protocol.
Action 2: Authorised users could post items for selling in the auctioning API.
Action 3: Authorised users could browse all items for sale in the auctioning API.
Action 4: Authorised users could bid for an item while it is in an auction.
Action 5: Highest bid authorised user wins the item after the end of the auction.
Action 6: Authorised users could browse for bidding history of an item that is sold.

You are advised to use the Django RESTful API framework to develop your software. You are also advised to use code samples from the lab tutorials and online sources by providing the appropriate references (in text and in code). You are encouraged to improvise providing clear descriptions of your functionality and your implementation decisions.

3. Coursework phases

Phase A: Install and deploy software in virtualized environments [10 marks]

- Install all the necessary packages in your virtual machine.
- Deploy your backend code in the virtual machine (code scripts of Phase B and C).
- Your REST API endpoints should be available under your virtual machine IP address based on examples and guidelines seen in Lecture 3.
- Provide a short description of your setup in the report. Briefly discuss about your installation and the structure of your folders. Do not include a list of the Linux commands in the report.

Phase B: Development of authorisation and authentication services [20 marks]

- Create an authorisation server. The server will authorise the auctioning RESTful API to store data in your database, on behalf of authorised users.
- Authenticate users each time you perform any Action point of Section 2, e.g. selling items, biding an item, browsing a list of items etc.
- Your authorisation service should follow the oAuth v2 protocol as shown in Class 4.
- You should provide a brief description of the functionality of the oAuth v2 protocol and how you integrate it in your software to perform authorisation in the report.

Phase C: Development of Auctioning RESTful APIs [35 marks]

- Your API should allow the basic functionalities as provided in Action points of Section 2.
- Each item should include the following data:
  - An item identifier
  - An item title
  - A timestamp of the item registration in the system
  - The condition of the item, this could be “New” or “Used”
  - The item description
  - The auction expiration time
  - Information about the item owner
  - Any other information you might need to store essential for your project
- Each auction should include the following data:
  - An auction identifier
  - An auction bidding price
  - The user that bids for the item
  - An action status e.g. open for offers or completed
  - The time left to complete
  - An auction winner
  - Any other information you might want to store essential for your project
- You are encouraged to develop your own auctioning database tables and application logic and expand Phase C as required.
- You should provide a list of the RESTful API endpoints in your report with a simple example of how to use your API.

You should always consider the following.

- Only authorised users should access the API.

- The API should allow any registered user to post items.
- Users that post items cannot bid for their own items.
- You are encouraged to set any other constraints to improve functionality of the software.

Phase D: Development of a testing application [20 marks]

Develop a testing application using Python to automate test cases of users bidding for an item. This application should be developed outside of your virtual machine e.g. in your desktop or laptop. The application should connect to the API endpoints and perform the necessary HTTP calls to manipulate data as required. The test cases should demonstrate Action points as described in Section 2. The test cases will demonstrate your system functionality.

Let us assume a use case scenario of three users such as Olga, Nick and Mary that access your API. Provide the following test cases (TCs).

TC 1. Olga, Nick and Mary register in the application and are ready to access the API.

TC 2. Olga, Nick and Mary will use the oAuth v2 authorisation service to get their tokens.

TC 3. Olga makes a call to the API (any endpoint) without using a token. This call should be unsuccessful as the user is unauthorised.

TC 4. Olga adds an item for auction with an expiration time using her token.

TC 5. Nick adds an item for auction with an expiration time using his token.

TC 6. Mary adds an item for auction with an expiration time using her token.

TC 7. Nick and Olga browse all the available items, there should be three items available.

TC 8. Nick and Olga get the details of Mary's item.

TC 9. Mary bids for her own item. This call should be unsuccessful, an owner cannot bid for own items.

TC 10. Nick and Olga bid for Mary's item in a round robin fashion (one after the other).

TC 11. Nick or Olga wins the item after the end of the auction.

TC 12. Olga browses all the items sold.

TC 13. Mary queries for a list of bids as historical records of bidding actions of her sold item.

Feel free to develop any other test cases to test your implementations. You should provide appropriate screenshots or descriptions of each test case in the report.

Phase E: Report your solution in a technical annex [15 marks]

Provide details on your implementations, your database design, descriptions of your services, API resources and any other information required. You do not need to include a list of the Linux commands in the report at any Phase of the development. Provide references to any online resources, including source code used from online tutorials and code from online repositories. You are advised to keep the report short and clear in terms of the development phases.

4. General coursework guidelines

Consider the following when developing your software.

- You are recommended to provide your solutions using Python programming and the Django REST and oAuth frameworks.
- You are encouraged to use the lab tutorials to deploy and develop software as seen in class.
- Follow the instructions of the coursework brief and coursework description on the Moodle page for time and mode of submission of your software.
- Provide comments in your code to explain key functionalities and implementations.
- Whenever necessary, provide screenshots of your API calls to demonstrate functionality.
- Provide a clear description and example of your API endpoints in the report.
- There is no report limit, provide clear explanations on the test cases and on future work.
- The coursework requires you to create database tables to save your data. You are recommended to follow the Django object relational mapping as seen in class. There could be multiple ways to implement tables to manage the data of your developments.
- Extra marks will be awarded for clear and well-designed code and well defined REST APIs.
- Best coursework(s) will win a special prize.