SWA Project

Overview; you are going to design a software system that allows school students to use their score to receive rewards that their school supplies. There would be admin staff that allow adding schools and their teachers to the system. There would be school teachers that setup their students. There would also be student for each school that redeem their scores to get rewards, being either avatar elements to make their avatar on the system, or to get other rewards.

ToDo; Design and develop a web application using microservices with the following features:

1. The user service contains functionality to maintain users (add user, remove user, update and view user). A user has the following attributes: username, password, role. Role can be one of the three Admin, Teacher, or Student.
2. The school service, contains functionality to maintain schools (add, remove, update, view). A school has the following attributes: name, address, contact (email, phone).
3. The teacher service contains functionality to maintain teachers (add teacher, remove teacher, update teacher, view teachers). A teacher has the following attributes: firstName, lastName, contact (email, phone), school, teachingClass (year, group).
4. The student service contains functionality to maintain students (add, remove, update, view). A student has the following attributes: firstName, lastName, studentNumber, school, class (year, group), score, avatar, rewards. Populate score of each student in the database with the initial value of 1000. Rewards is a list of reward items explained below.
5. The avatar service contains functionality to maintain avatars (add, remove, update, view). An avatar has the following attributes: head, hair, eye, eyebrow, nose, mouth, ears, body, hat, top, top colour, hat colour).
6. The element service contains functionality to maintain element (add, remove, update, view). An element has the following attributes: type (can be one of the 12 types head, hair, eye, eyebrow, nose, mouth, ears, body, hat, top, top colour, hat colour), price. Set the values of prices something between 10 to 50.
7. The reward service contains functionality to maintain element (add, remove, update, view). A reward has the following attributes: name, quantity, type (can be one of the 3 types element, in-school, gift), price. Set the values of prices something between 10 to 50.
8. An admin is able to login using username and password already fixed in a database. Admin is also allowed to administer school using school service, administer user using user service. On any new teacher added by the admin, an email notification should be sent to the teacher.
9. A teacher is able to login with the username and password provided by admin, and administer their students. On any new teacher added by the teacher, an email notification should be sent to the student. Teacher can also administer reward.
10. A student is able to login using username and password provided by teacher, and administer avatar. Student is also able to buy elements using their score as cost to pay to add to their avatar. Before buying can be done, it should be checked if there is enough score for the student to spend to buy that element. Also it should be checked if the student already has one of that type of element or not, as if so, it should remove the previously purchased element from their avatar and add its price back to the student score (like a change of element is possible). Remember student must also be able to remove element, in which case its price would be refunded to the student’s score. In the case of changing an element, it could be checked if the student can pay the difference between the price of the previous element and the new element (if higher) instead of checking if student has score for the

complete price of the new element. This means that you want to do the remove element and refund first and then add new element, so that the score does not go negative.

11. A student is also able to buy rewards, not more than one reward of the same type though.

ToDo; Write standalone client applications using the RestTemplate that executes the following scenarios:

1. User login,
2. School administration by admin,
3. Teacher administration by admin,
4. Element administration by admin,
5. Reward administration by admin, this also includes student reward being redeemed by student, in other words, admin can set that a student uses one of their purchased rewards,
6. Student administration by teacher,
7. Avatar administration by student.

Additional requirements and information:

1. Add any other service if decided necessary, for example, login, email, etc. It is not necessary to hash password, or actually send email, but if done, it will have bonus,
2. It is not necessary to have an attribute imageFilename for elements, but if done, it will have bonus,
3. Try to use CQRS if you decided necessary, it will have bonus,
4. There is no attribute quantity for the element, so it is always available for any student to spend their score and buy element, however, a student cannot have two elements of the same type in their avatar. A student cannot buy two rewards of the same type either.
5. Because the student service is used a lot, run 3 instances of the student service,
6. All microservices use the MongoDB database to store data,
7. Messaging is implemented with Kafka,
8. We need to implement logging and tracing in our architecture,
9. We need to use a central config service,
10. The number of client applications is up to you, as long as they accommodate the scenarios,
11. Implementation of security in your architecture will have bonus.
12. Adding GUI to your project will have bonus.

For this project, use the best practices we learned in the course.

In Sakai you find a daily progress report quiz. You are required to fill in this quiz daily at the end of every day. Do not forget to send this progress report every day. You will lose points if you do not submit your daily progress report.

Optional

If two teams join forces and make one larger team, then that would be for adding the bonus parts as mentioned above (1, 2, 3, 11, 12). In that case the bonus parts will not be bonus anymore, they will be mandatory requirements.

Presentation

Record a presentation with the following parts:

1. Show and explain your design using PowerPoint slides. Explain all patterns, techniques and supporting services you used.
2. Demo your code implementation. Run the client application and show the results. Show that your application works correctly by showing the content in the database.

What to hand in?

The project needs to be submitted on **Thursday, July 20th at 11:00 AM in Sakai**. Projects that are not submitted by 11:00 AM will lose points. You need to submit the following items as separate files (do not zip them together).:

1. Zip file with all code. If your zip file is too large, store it somewhere, for example google drive and submit the link. Make sure I have access rights to download the zip file.
2. PowerPoint slides of your presentation including your architecture design. Make at least a class diagram of all your services. Show all classes in all services.
3. A link to your video recording. Store this link somewhere where I can access it, for example google drive. Make sure I have access rights to watch the video.

How will your grade be determined?

Your grade will be determined on the following aspects:

1. The quality of your architecture

2. Did you implement all requirements?

3. How clear you have explained all architectural and design decisions you have made

4. Your presentation. (Is it clear and professional presentation? Is it easy to follow? Do you demo all required parts?)

5. Did you submit your progress report on time?