Assignment 2: Distributed Programming

Home Assignment

Assignment Guidelines

Read the following instructions carefully:

- The assignment coversheet should be the first sheet in your assignment. Moreover, the coversheet should be fully completed with all the necessary details.
- You are required to use .NET technologies for this assignment, and you may use any library or framework for developing the event-driven architecture and front-end Web application.
- All text/code *must be properly referenced*. In the absence of proper referencing, the assignment will be regarded as plagiarised.
- **<u>Copying is strictly prohibited and will be penalized</u>** in line with the College's disciplinary procedures.
- The deadline to submit all deliverables is **<u>24/05/2024</u>**
- You are required to submit all your assignment deliverables (as explained in the assignment brief) via the links on VLE.
- You are required to commit all project files to a Git repository with frequent commits.
- You are required to host your microservices and Web application on free-hosting services and make sure that they are running and accessible.
- You are required to record a (video) demonstration of your project and ensure that you set the correct permissions by sharing it only with the lecturer.

Video Streaming System

A Video Streaming System handles Movies and TV shows orders and playback of video content over the internet to users' devices, ensuring seamless and high-quality viewing experiences. In this assignment, you will be developing a Video Streaming System that provides the functionality as outlined in this brief. The Video Streaming System should be designed using the Microservices architecture and the Event-driven architecture.

You are required to develop microservices (including a Gateway API), event-driven services and a Web application. When the user selects any of the features (outlined in this brief) from the Web application it will send the request to the microservices and the microservices will send back the response in JSON. The microservices should therefore have endpoints for each feature that will send back the response represented in JSON format. The Web application will parse the JSON formatted response and present the results accordingly.

The Video Streaming System will consist of the following microservices:

- Customer microservice – this handles customer details including user account details for logging into the system and notifications about purchases and upcoming Movies and TV shows
- Payment microservice – this handles customer payments and audit trails customer transactions
- Order microservice – this handles customer video orders and logs past orders
- Video Catalogue microservice – this handles video catalogues by retrieving movie or TV Show details from external third-party APIs such as https://rapidapi.com/SAdrian/api/moviesdatabase/ (no need to store the movie or TV show details on the database, they can be retrieved in real time)
- Watchlist microservice – this handles the customer's Movies and TV shows watchlist

Users must first register for a new account within the system by providing generic details such as first name, surname, email (which will be used as a username too) and password. Once a user logs in, the user will be able to select between Movies or TV Shows and view a list of videos genre. Once the user selects a genre and either Movies or TV Shows, the user will then be able to see a list of movies or TV shows titles (related to the selected option and genre) extracted from external third-party APIs via the video catalogue microservice. The system should provide users the ability to order and pay the video through the order and payment microservices. When a user confirms an order by confirming a payment, this triggers an event through an event-driven architecture that will call the respective microservices to create the order, log the payment, record the requested video and notify the customer that the order has been completed. The system's base currency will be in Euro.

Users should also have a feature where the user can view the completed orders. When selecting any past order, the user should be able to view order details such as the video(s) ordered with their individual price and the total ordered cost. The users can also add at any moment a title to their watchlist without paying for it and should be able to retrieve or remove any video in the watchlist.

The system should also provide administrators to log in and add new upcoming videos (TV Shows or Movies). If the upcoming video has the same genre of the user watchlist or past orders, the user should receive a notification through an event-driven architecture.

Moreover, the system should provide an inbox feature where users can view any notifications received by the system. Users should receive notifications when an upcoming video has been released by the Administrator or when an order has been completed.

All data within the system should be logged and stored in a cloud-based database such as Google Cloud SQL, Cloud Firestore, MongoDB or as preferred.

You should therefore implement the following features and requirements (tasks):

- **[Task 1]** A Customer microservice that allows users to manage their account. This microservice should:
    - Allow a user to register for a new user account with the Video Streaming System (through the microservice) - details should be stored in a cloud-based database.
    - Allows a user to login into their user account (through the microservice).
    - Provide user account details when requested from the Web application - i.e. the Web application should have a user account details page containing user account details and preferences.
    - Allow users to receive and view system notifications (i.e. inbox).

[3 marks]

- **[Task 2]** A Video Catalogue microservice that retrieves video titles and details from external third-party APIs. This microservice should:
    - Retrieve a list of videos for a specific genre (through external third-party APIs).
    - Retrieve title details for a specific video (through external third-party APIs).

[3.5 marks]

- **[Task 3]** An Order microservice that allows users to manage orders. This microservice should:
    - Allow users to create video orders – orders are created through an event (see task 6) that is triggered once the user confirms the order payment from the Web application.
    - Allow users to view a list of video orders.

    - Allow users to retrieve order details for a specific order.

[3.5 marks]

- **[Task 4]** A Payment microservice that handles payments and audit trails payments. This microservice should:
    - Allows users to pay for video orders – payments are logged through an event (see task 6) that is triggered once the user confirms the order payment from the Web application.
    - Retrieve order payment details.

[3.5 marks]

- **[Task 5]** A watchlist service that handles customers' Movies and TV shows watchlist. This microservice should:
    - Allow to add movies or tv shows to the watchlist without paying for them
    - Retrieve or remove titles in the watchlist
    - In case the customers order any title present in the watchlist, it should be removed from the watchlist

[3.5 marks]

- **[Task 6]** An event-driven function that will create a purchase order once the user confirms the payment of the order from the Web application. This event should create the order, log the payment, record the details and notify the customer that the requested title has been purchased.

[5 marks]

- **[Task 7]** An event-driven architecture that will allow administrators to add new upcoming videos (Movies or TV Shows) through the Web Application. This event should notify the customer about the new upcoming title in case:

  - the upcoming title has the same genre of any video in the watchlist
  - the upcoming title has the same genre of the customers' past orders

  The upcoming video details should not be stored in the database and they should be retrieved in real time from the third-party API

[5 marks]

- **[Task 8]** A Web application that provides a user-friendly interface to allow the user to interact with all the features provided by the microservices.

[4 marks]

- **[Task 9]** The microservices endpoints should provide the data formatted in JSON. The Web application should be able to parse the JSON data and display the results accordingly. The Web application should interact with a Gateway API, which should redirect the requests to the microservices.

[3 marks]

- **[Task 10]** The microservices should interact with a cloud-based database that stores the user's account details, user notifications, order details, payment details and shipping details.

[2.5 marks]

- **[Task 11]** Upload and deploy your microservices and Web application online to any free hosting of your choice such as Google App Engine, Back4App, Free ASP.net Hosting, MyASP.net or Firebase.

[7 marks]

- **[Task 12]** The Web application should be able to communicate with the microservices hosted online.

[2.5 marks]

Furthermore, you are required to record a video demonstration of not longer than 20 minutes. In the video recording you should demonstrate that all features are working through the Web application and also through testing tools such as Swagger or Postman. You are also requested to briefly explain your code demonstrating how you have implemented each feature. Therefore, it is important that in your video recording you:

- **[Task 13]** Explain how each microservice is implemented, deployed, and working through the Web application and testing tools such as Swagger/Postman.

  [5 marks]

- **[Task 14]** Explain how the event-driven architecture is utilised in the Video Streaming System by explaining how each event function is implemented, deployed, and working through the Web application and testing tools such as Swagger or Postman. Moreover, you are required to demonstrate using both customer accounts (for demonstrating video orders, payments and watchlist) and administrator accounts (for demonstrating addition of upcoming titles)**.**

  [5 marks]

- **[Task 15]** Explain how you have deployed the Web application, microservices and database to cloud-based hosting services. Moreover, demonstrate that Web application, microservices and database hosted online on cloud-based services are working and accessible. Furthermore, in your explanation, explain the advantages and disadvantages of using cloud services for hosting Web applications, microservices and databases.

[5 marks]

Deliverables:

1. Upload your code to a Git repository and submit the link to your repository on VLE (make sure that your Git repository is accessible). It is important to commit your code in stages to show your progress whilst working on this assignment. Include clear comments and notes to explain key concepts in your code.
2. Host your Web Application, microservices and database to free hosting sites and provide the links on VLE. When hosting online, make sure not to store any personal information but only test data.
3. Record a video demonstration of your assignment (not more than 20 minutes). In your demonstration, you should go through your application demonstrating each task and also provide an explanation of your code as explained above. Submit the link to your recording on VLE. **Note**: Ensure that you set the correct permissions and share it only with your lecturer.

Marking Scheme

	Assessment Criteria	Marks Awarded	Task Marks
	Development		
SE2.3	Construct a solid and robust Microservices Architecture **[Task 1] – 3 marks** **[Task 3] – 3.5 marks** **[Task 5] – 3.5 marks**		**10**
AA3.3	Use emerging libraries to consume Web Services and to persist data on the Client. **[Task 8] – 4 marks** **[Task 9] – 3 marks**		**7**
SE3.4	Construct a solid and robust Event-driven Architecture. **[Task 6] – 5 marks** **[Task 7] – 5 marks**		**10**
KU4.2	Describe the main services provided by cloud services. **[Task 10] – 2.5 marks** **[Task 12] – 2.5 marks**		**5**
AA4.3	Use Cloud Web Services to create and deploy a Web Server. **[Task 11] – 7 marks**		**7**
AA4.4	Use third party Web Service APIs to consume data. **[Task 2] – 3.5 marks** **[Task 4] – 3.5 marks**		**7**

Demonstration			
KU2.1	Describe the context of a Microservices Architecture. **[Task 13] – 5 marks**		**5**
KU3.1	Describe the context and core principles of an Event-driven Architecture. **[Task 14] – 5 marks**		**5**
KU4.1	Describe the advantages and disadvantages of cloud computing. **[Task 15] – 5 marks**		**5**
Total			**61**

(End of Assignment Brief)