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.
- **Copying is strictly prohibited and will be penalized** in line with the College's disciplinary procedures.
- The deadline to submit all deliverables is **30/05/2025**
- You are required to submit all your assignment deliverables (as explained in the assignment brief) via the links on VLE/Moodle.
- 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.

Cab Booking Platform

In this assignment, you will be developing a cab booking platform that allows customers to book rides quickly and easily. Key features will include booking for specified dates, locations, and number of passengers, along with other functionalities described below. The cab booking platform should be designed using the Microservices Architecture.

You are required to develop microservices 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 accordingly. Each microservice should expose specific endpoints corresponding to its features, returning responses in JSON format. The Web application will parse the JSON formatted response and present the results accordingly.

The Cab Booking Platform will consist of the following microservices:

- **Customer microservice** – this handles customer details including user account registration and login, as well as inbox notifications.
- **Booking microservice** – this handles ride bookings, including creating new bookings and viewing past rides.
- **Payment microservice** – this manages payments, processes fare confirmations and logs transactions.
- **Fare Estimation microservice** – this retrieves estimated taxi fare information from an external API such as https://rapidapi.com/3b-data-3b-data-default/api/taxi-fare-calculator (no need to store fare data, it can be retrieved in real time).
- **Location microservice** – this manages user-saved pickup locations and retrieves real-time weather forecasts for these locations using external APIs such as https://rapidapi.com/weatherapi/api/weatherapi-com

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 logged in, users will be able to view their current and past bookings, as well as create new cab bookings. When creating a new booking, the estimated ride cost should be retrieved from an external third-party API, such as the following.

- Taxi Fare Calculator: https://rapidapi.com/3b-data-3b-data-default/api/taxi-fare-calculator

Furthermore, users should be able to manage their favourite pickup locations and automatically retrieve the weather forecast for those locations using an external third-party API, such as the ones listed below:

- Weather Forecast: https://rapidapi.com/weatherapi/api/weatherapi-com

Moreover, the system should provide an inbox feature where users can view any notifications received by the system. Users should receive notifications when a ride is ready or when there is a discount.

All data within the system should be logged and stored in a cloud-based database such as Google Cloud SQL, Back4App, Firebase, 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 cab booking platform (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 microservice.
    - Allow users to receive and view system notifications (i.e. inbox).

[5 marks]

- **[Task 2]** A Booking microservice that allows users to create and manage cab bookings. This microservice should:
    - Allow users to create a new cab booking. A booking includes a starting location, an ending location, a date and time, the number of passengers, and a cab type. Cab types can be Economic, Premium, or Executive. Once a booking is confirmed, all details should be stored, including the trip information, cab type, date, and time.
    - Allow users to view current cab bookings (together with the details for each booking)
    - Allow users to view past cab bookings (together with the details for each booking).

[5 marks]

- **[Task 3]** A Payment microservice that handles payments and audit trails payments. This microservice should:
    - Allow users to pay for cab bookings. The total price should be based on the requested ride, and it should be computed as follows:
        - Total price = *cab_fare* × *cab_multiplier* × *daytime_multiplier* × *passengers_multiplier* × *discount*

            Where:

            - *cab_fare* should be retrieved from the external API as specified above
            - *cab_multiplier* is a multiplier based on the requested cab:
                - Economic: 1
                - Premium: 1.2
                - Executive: 1.4
            - *daytime_multiplier* is a multiplier based on the requested daytime:
                - Between 8:00 AM and 11:59 PM: 1
                - Between 12:00 AM and 8:00 AM: 1.2

- *passagers_multiplier* is a multiplier based on the requested number of passengers:
    - 1-4: 1
    - 5-8: 2
    - > 8: not allowed
- *discount* (optional)*:* A discount multiplier available after three successful bookings. This is triggered by an event defined in **Task 6.**

- Allow users to retrieve payment details

[3.5 marks]

- **[Task 4]** A Location microservice that handles users' favourite pickup locations. This microservice should:
    - Allow the user to add, update or remove a favourite pickup location
    - Allow the user to retrieve the favourite pickup locations
    - Allow the user to retrieve weather forecast details for each favourite pickup location with an external API as indicated above

[3.5 marks]

- **[Task 5]** Implement an event-driven function that notifies users when a discount becomes available. This event should meet the following criteria:
    - The event should be triggered only after a user has successfully completed three bookings
    - The discount availability notification must be created only once per user.

[5 marks]

- **[Task 6]** An event-driven function that will notify the user that the cab is ready to pickup the user. This event should be triggered 3 minutes after a booking is made (to simulate the driver search process) and then published as notification that contains the details of the requested ride in the user microservice.

[5 marks]

- **[Task 7]** A Web application that provides a user-friendly interface to allow the user to interact with all the features provided by the microservices.
  The Web application should interact with a Gateway API, which should redirect the requests to the microservices.

[4 marks]

- **[Task 8]** 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.

[3 marks]

- **[Task 9]** The microservices should interact with a cloud-based database that stores the user's account details, user notifications, bookings details, and payment details.

[3.5 marks]

- **[Task 10]** 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.

[3.5 marks]

- **[Task 11]** The Web application should be able to communicate with the microservices hosted online. The Web application should offer a smooth and error-free user experience.

[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. 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 12]** Explain how each microservice is implemented and working through the Web application and testing tools such as Swagger or Postman.

[5 marks]

- **[Task 13]** Explain how the event-driven architecture is utilised in the cab booking platform by explaining how each event function is implemented, deployed, and working through the Web application and testing tools such as Swagger or Postman.

[5 marks]

- **[Task 14]** 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/Moodle (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/Moodle. 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/Moodle. **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] – 5 marks** **[Task 2] – 5 marks**		**10**
AA3.3	Use emerging libraries to consume Web Services and to persist data on the Client. **[Task 7] – 4 marks** **[Task 8] – 3 marks**		**7**
SE3.4	Construct a solid and robust Event-driven Architecture. **[Task 5] – 5 marks** **[Task 6] – 5 marks**		**10**
KU4.2	Describe the main services provided by cloud services. **[Task 11] – 5 marks**		**5**
AA4.3	Use Cloud Web Services to create and deploy a Web Server. **[Task 9] – 3.5 marks** **[Task 10] – 3.5 marks**		**7**
AA4.4	Use third party Web Service APIs to consume data.		**7**

	[Task 3] – 3.5 marks **[Task 4] – 3.5 marks**		

Demonstration			
KU2.1	Describe the context of a Microservices Architecture. **[Task 12] – 5 marks**		**5**
KU3.1	Describe the context and core principles of an Event-driven Architecture. **[Task 13] – 5 marks**		**5**
KU4.1	Describe the advantages and disadvantages of cloud computing. **[Task 14] – 5 marks**		**5**
Total			**61**

(End of Assignment Brief)