# PART 2

1. Introduction
   - Purpose: The objective of this paper is to delineate the prerequisites for the creation of a program for a quiz game that is based on text.

   - Project scope: Involves developing a C# Windows-based Console application that enables interactive quiz sessions.

2. Overall Description
   - Product: The software program will empower users to generate, modify, and engage in quizzes comprising multiple choice, open-ended, and true or false questions.

   - Users: The project primarily targets persons with a penchant for educational or recreational quizzes. The purpose of the game is to enhance general knowledge of the topic of geography.

   - Operational Environment: The code is compatible with all devices and is developed using the .NET Framework, particularly the .NET Core.

3. System Features

   - Description: The system will allow users to create, edit, and delete quiz questions. During quiz sessions, participants will generate questions and submit their answers via text input. Afterwards, the system will evaluate the replies and provide feedback about their appropriateness.

   - Functional Requirements:

- Users retain the ability to add, remove, and amend quiz questions.
- The system displays questions categorised into multiple-choice, open-ended, and true or false formats.
- The system verifies the users' input and generates the appropriate feedback.

4. User Interface Requirements

The user interface will be text-based and presented in a console window. The system will present explicit prompts for user engagement, encompassing choices for the inclusion, modification, and removal of questions, as well as the initiation of quiz sessions.

## 5. Platform Requirements

The game quiz program is implemented using the computer's programming language C#. The Code is compatible with Windows, Mac, and all other operating systems. The code will function on all devices that support the .NET Core framework, which utilises the cross-platform development environment.

## 6. Quality Attributes

- Performance: The system should exhibit immediate responsiveness to user inputs and deliver swift feedback throughout quiz sessions.

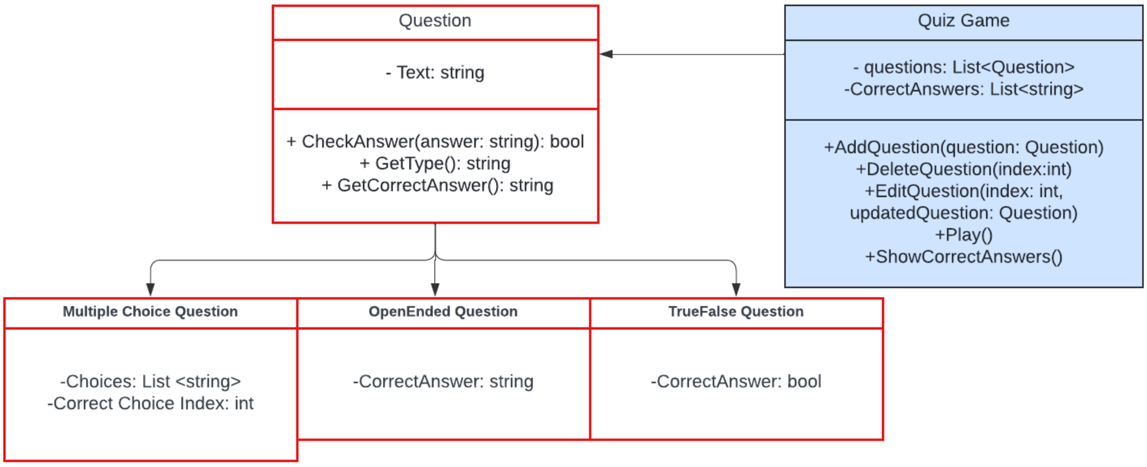
- Security: The quiz game programme maintains an emphasis on user data privacy and security by refraining from persistently preserving any user data. Furthermore, the programme abstains from performing data analytics or disclosing user data to third-party organisations. Furthermore, as all data is saved in volatile memory, it is instantaneously erased when the application is terminated or refreshed. This strategy guarantees the confidentiality of user data, preventing any unauthorised access or exploitation.

- Safety: The programme is intended to ensure the safety of users and their devices by effectively managing user inputs and interactions, thereby avoiding any potential harm. Nevertheless, there is a lack of specific safety measures in place to mitigate possible risks, such as crashes or data loss. Subsequent versions of the programme may prioritise the incorporation of resilient error handling systems, input validation procedures, and secure techniques to minimise risks and provide a secure user experience. Furthermore, implementing comprehensive testing and validation methods can assist in the identification and resolution of safety issues before implementation.

## *PART 3*

### *UML Diagrams*

- UML Class Diagram



Question
- Text: string
+ CheckAnswer(answer: string): bool
+ GetType(): string
+ GetCorrectAnswer(): string
Quiz Game
- questions: List<Question>
-CorrectAnswers: List<string>
+AddQuestion(question: Question)
+DeleteQuestion(index:int)
+EditQuestion(index: int, updatedQuestion: Question)
+Play()
+ShowCorrectAnswers()
Multiple Choice Question
-Choices: List <string>
-Correct Choice Index: int
OpenEnded Question
-CorrectAnswer: string
TrueFalse Question
-CorrectAnswer: bool

- 'Question', 'MultipleChoiceQuestion', 'OpenEndedQuestion', and 'TrueFalseQuestion' are illustrated as classes.

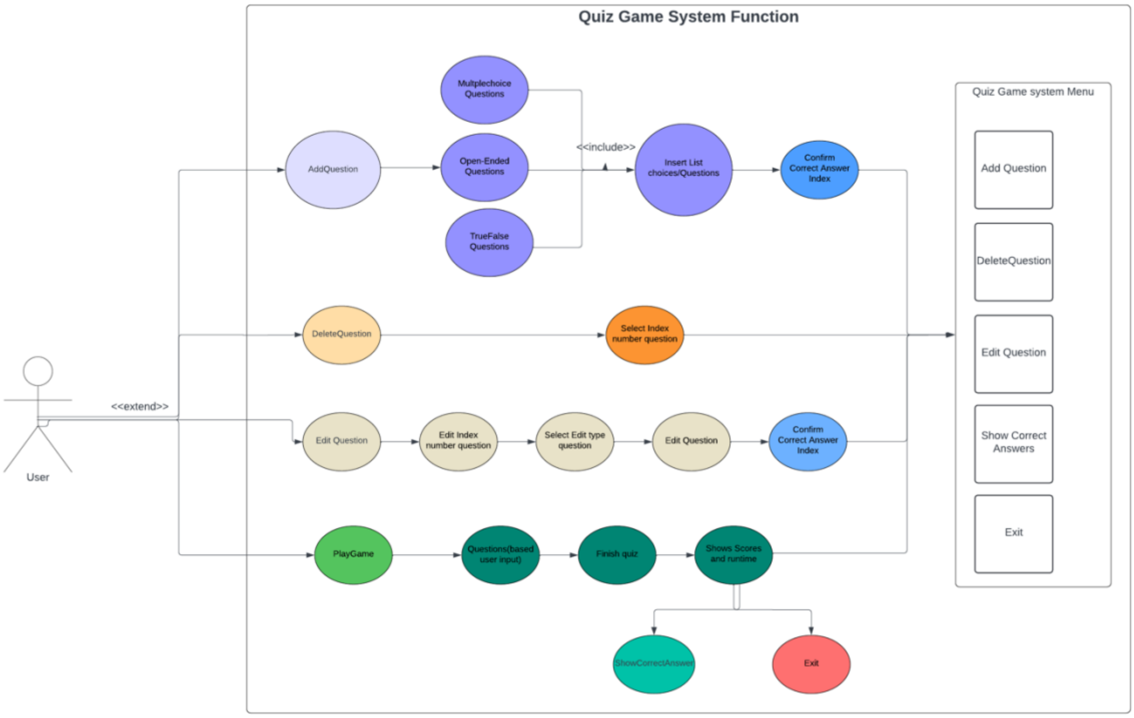
- Attributes such as 'Text', 'Choices', 'CorrectChoiceIndex' , and 'CorrectAnswer' are listed within each class box.

- Methods (functionality) such as 'CheckAnswer', 'GetType', and 'GetCorrectAnswer' are also listed within each class box.

- The 'QuizGame' class is depicted separately, with its attributes questions and 'correctAnswers', along with its methods 'AddQuestion', 'DeleteQuestion', 'EditQuestion', 'Play', and 'ShowCorrectAnswers'.

- Arrows depict the relationships between classes, showing associations between 'QuizGame' and 'Question', as well as inheritance relationships between 'MultipleChoiceQuestion', 'OpenEndedQuestion', and 'TrueFalseQuestion'.

- UML Use Case Diagram



1. AddQuestion: The user requests to add a new question to the quiz game. This entails supplying the textual content of the inquiry, together with the question's category (multiple choice, open-ended, or true/false) and its related responses.
2. DeleteQuestion: The user chooses a question from the available pool of questions and asks for its removal from the quiz game. This operation eliminates the chosen question from the quiz game's catalogue of questions.
3. EditQuestion: The user selects a query to alter and delivers revised data for the chosen inquiry. The procedure entails modifying the content of the inquiry, its format, and its responses.
4. PlayGame: The user initiates participation in the quiz game, wherein they are required to respond to several questions supplied by the user. The user's replies are assessed, and a score is generated depending on the accuracy of their answers.
5. ShowCorrectAnswer: Upon finishing the quiz game, the user expresses a desire to get accurate solutions to the questions they have responded to. This action presents accurate responses with the questions to review.
6. Exit: The user chooses to terminate the quiz game. This action concludes the quiz game session and shuts off the programme.