Documentation and Requirements Gathering

Project Overview

This project involves creating a .NET 8 web application for managing application forms for an employer. The application will allow employers to create and manage different types of questions for their application forms, and candidates to submit their responses. The data will be stored in Azure Cosmos DB for NoSQL.

Project Goals and Objectives

- Develop a web application using .NET 8.
- Store and manage different types of application form questions.
- Provide endpoints for CRUD operations on questions.
- Allow candidates to submit their responses.
- Store data in Azure Cosmos DB.
- Implement unit tests using xUnit (if possible).

Stakeholders

- Employers: Users who create and manage application forms.
- Candidates: Users who submit responses to the application forms.
- Front-End Team: Developers who will use the provided endpoints to render forms and submit data.

Functional Requirements

Screen 1: Employer Creating a Program and Application Form

POST Method:
Endpoint to create different types of questions (Paragraph, YesNo, Dropdown, MultipleChoice, Date, Number).

Example payload for creating a question:

```
{
  "questionType": "Paragraph",
  "questionText": "Describe your experience.",
  "options": null
}
```

PUT Method:
Endpoint to edit a question after it has been created.
Example payload for editing a question:

```
{
  "questionId": "12345",
  "questionText": "Update your experience.",
  "options": null
}
```

Screen 2: Candidate Applying for the Program

GET Endpoint:

Endpoint to retrieve the questions based on question type.
Example response:

```
[
  {
    "questionId": "12345",
    "questionType": "Paragraph",
    "questionText": "Describe your experience."
  },
  ...
]
```

POST Endpoint:
Endpoint for candidates to submit their application responses.
Example payload for submitting responses:

```
{
  "candidateId": "67890",
  "responses": [
    {
      "questionId": "12345",
      "response": "I have 5 years of experience in software development."
    },
    ...
  ]
}
```

Non-Functional Requirements

- Performance: The application should handle multiple requests concurrently.
- Scalability: The application should be able to scale with the increase in users.
- Security: Secure data storage and transmission.
- Usability: User-friendly interfaces for employers and candidates.
- Maintainability: Clean and well-documented codebase.

Technical Requirements

- Framework: .NET 8
- Database: Azure Cosmos DB for NoSQL
- Dependency Injection: Utilize built-in .NET DI
- Unit Testing: xUnit (if possible)
- Version Control: Git (hosted on GitHub)
- Development Tools: Visual Studio 2022

Documentation

API Documentation: Detailed documentation of all endpoints, request/response formats.
Technical Documentation: Architecture diagrams, database schema, and setup instructions.
User Documentation: Instructions for employers and candidates on how to use the application.

Next Steps

Set Up GitHub Repository:
Create a public repo and do the initial commit.

Begin Development:
Follow the project plan and milestones.

Submit the GitHub Link:
Once completed, submit the link to Google Forms.