Project Overview

The project is a **C++ Console Application** designed as an interactive educational tool. It serves as a mini-project to test user knowledge in areas such as geography and programming.

Presentation Analysis

The documentation is structured into eight key sections, providing a clear roadmap of the project's lifecycle.

- – **Visual Structure**: Uses a modular "Table of Contents" with icons to represent different development phases like Code Analysis and Scoring Systems.
  – **Clarity**: High-level summaries define the application type as "Console-Based" with a specific scope of 3 sample questions worth a total of 30 points.
  – **Flow Documentation**: The presentation effectively maps the "Question Processing Cycle," which includes displaying questions, getting input, checking answers, and updating scores.

Technical Code Analysis

The provided C++ code implements a structured approach to a command-line interface (CLI) game. ### Data Architecture The program utilizes a `struct` to handle data encapsulation: - - `struct Question`: Efficiently bundles the question text, a four-element string array for options, the correct character answer, and the integer point value. - **Array Initialization**: The `main()` function initializes an array of three `Question` objects, defining the quiz content directly within the source code.

Logic & Execution

The application logic follows a linear execution path: 1. - **Iterative Loop**: A `for` loop iterates through the question array. 2. **Input Handling**: Uses `cin` to capture user answers and `toupper()` to ensure the input is case-insensitive, improving user experience. 3. - **Conditional Feedback**: Immediate feedback ("Correct!" or "Wrong!") is provided after each submission. 4. **Score Accumulation**: Points are added to a running `score` variable only when `toupper(ans) == quiz[i].answer`.

Scoring System & UI

Feature	Implementation Detail
Point Value	10 points per correct answer.
Max Score	30 points total.
Feedback Logic	Score ≥ 20: "Great job!"; Score < 20: "Better luck next time."
UI Type	Text-based console greeting and results summary.

Future Enhancements

The documentation identifies several critical areas for growth to move beyond a "Mini Project" status:

```
- **Dynamic Gameplay**: Implementing **Random Question Order** to prevent memorisation.
- **Advanced Features**: Adding **Time Limits** per question and **Difficulty Levels** (Easy
- **Persistence**: Using File I/O for **Persistent Score Tracking** to store long-term stati
- **UI Upgrade**: Potential transition from a Console UI to a **GUI Implementation** using t
```