MythBusters Project - Comprehensive Module-Based Test Evaluation Report

This report evaluates the results of the tests conducted on the frontend and backend components of the MythBusters project, in line with the predefined Acceptance Criteria and Definition of Done.

Part 1: Frontend (User Interface) Evaluation

1.1. User Authentication

- **Acceptance Criteria (AC):**
  - When a user registers with valid information, the system must successfully create the account and automatically log the user in.
  - When a user logs in with the correct email and password, they must be redirected to the main page within 3 seconds.
  - The system must display **specific and clear error messages** to the user for invalid operations, such as a duplicate email.
  - The logout process must securely terminate the user session.
- **Definition of Done (DoD):**
  - All ACs have been successfully tested and passed.
  - Relevant backend unit and integration tests (ProfileControllerTest, ProfileServiceUnitTest) have been successfully completed.
  - The code has passed code review and is ready to be merged into the main branch.
- **Evaluation:**
  The core functionalities of this module (registration, login, logout) are stable. However, **Acceptance Criterion 3 is not fully met.** Therefore, the DoD targets for this module have **not been met**.
- **Observations:**
  Testing revealed that the core authentication flow works smoothly. The most significant shortcoming lies in **error handling**. As seen in test REG-002, although the backend correctly returns an error when a user tries to register with an existing email, the frontend displays a generic "Login failed!" message instead of informing the user that "This email is already in use." This negatively impacts the user experience. The axios catch block in the frontend needs to be improved to process the specific error message returned from the API.

1.2. Main Page & Navigation

- **Acceptance Criteria (AC):**
    - When the user clicks on the game cards on the main page, they must be seamlessly redirected to the respective game setup screen.
    - The leaderboard should load by default, showing data for all games.
    - The filter buttons on the leaderboard must refetch the data with the correct gameType parameter and instantly update the list.
- **Definition of Done (DoD):**
    - All ACs have been successfully tested and passed.
    - Necessary backend tests (LeaderboardControllerTest) have been successfully completed.
    - The code has passed code review.
- **Evaluation:**
  All AC and DoD targets for this module have been **fully met**.
- **Observations:**
  The main page and navigation elements work as expected. The game selection cards and leaderboard filtering are stable and responsive. The useEffect hook in LeaderBoard.tsx correctly implements the filtering logic. No critical bugs were found in this module.

1.3. Game Setup & Character Selection

- **Acceptance Criteria (AC):**
    - The user must be able to successfully purchase an avatar if they have sufficient coins.
    - The user's coin balance must be updated instantly after a purchase.
    - In case of insufficient funds, a clear warning like "Insufficient Funds" must be shown to the user.
    - When the user clicks the "Start Game" button, they must be directed to the game with the selected mode and difficulty.
- **Definition of Done (DoD):**
    - All ACs have been successfully tested and passed.
    - The code has passed code review.
- **Evaluation:**
  All AC and DoD targets for this module have been **fully met**.
- **Observations:**
  The character selection and game setup screen function smoothly overall. The avatar purchase process, which uses the Command and Observer design patterns, works as intended. No issues were encountered in the user flow of selecting a game mode, difficulty, and starting the game.

1.4. Car Race Game

- **Acceptance Criteria (AC):**
    - The player's car must move forward and their score must increase when a correct answer is given.
    - When the player wins the game, their score must be automatically saved to the leaderboard.
    - The "Restart" button must reset the game board and scores, starting a new game.
- **Definition of Done (DoD):**
    - All ACs have been successfully tested and passed.
    - Score submission API integration is complete and functional.
- **Evaluation:**
  The core game mechanics are functional. However, as identified in test CAR-002, **Acceptance Criterion 2 is not met**, as the score is not saved to the leaderboard upon winning. Therefore, the DoD targets for this module have **not been met**.
- **Observations:**
  The test confirmed that while the gameplay itself is fine (car movement, scoring locally), the failure to save player achievements is a major shortcoming. Network logs confirmed that the useEffect hook in CarRaceGameScreen.tsx does not trigger the score submission process upon winning. This is a critical bug that directly affects player motivation and must be resolved with high priority before this module can be considered "Done".

1.5. Hangman Game

- **Acceptance Criteria (AC):**
    - Correct/incorrect letter guesses must correctly trigger the game mechanics (revealing letters, drawing the hangman).
    - When the player wins the game by completing all questions, their score must be saved to the leaderboard.
    - The scoring system must be fair and prevent the exploitation of game mechanics like using hints.
- **Definition of Done (DoD):**
    - All ACs have been successfully tested and passed.
    - There are no scoring loopholes in the game.
- **Evaluation:**
  The core mechanics and successful score submission are working. However, due to the scoring loophole identified in test HNG-004, **Acceptance Criterion 3 is not met.** Because of this critical logic flaw, the DoD targets have **not been met**.
- **Observations:**
  The most critical issue is that a player can complete a word just by using the "Reveal Letter" button without guessing any letters and still unfairly receive the word completion bonus. This is an 'exploit' that renders the competitive nature and the scoring system meaningless. It is a logic flaw that must be fixed with the highest priority.

1.6. Balloon Popping Game

- **Acceptance Criteria (AC):**
  - A balloon must be popped and the score increased for a correct answer.
  - A life must be lost for an incorrect answer.
  - The game must process inputs correctly, regardless of the player's response speed.
  - The score must be successfully sent to the leaderboard upon winning the game.
- **Definition of Done (DoD):**
  - All ACs have been successfully tested and passed.
  - There are no timing-related bugs (race conditions) in the game.
- **Evaluation:**
  The score submission and basic answer mechanics are working. However, due to the timing issue identified in test BAL-005, **Acceptance Criterion 3 is not met.** Due to this critical bug, the DoD targets have **not been met**.
- **Observations:**
  The handleAnswer function in BalloonGameScreen.tsx cannot correctly manage consecutive, rapid clicks. When a player clicks the correct answer too quickly, the game can misinterpret it as an error and cause a loss of life. This is a 'race condition' bug that breaks the core gameplay and severely harms the user experience.

Part 2: Backend (Server Architecture) Evaluation

- **Acceptance Criteria (AC):**
  - The application must start successfully without any dependency or configuration errors.
  - All API endpoints must delegate incoming requests to the correct service methods.
  - The service layer must correctly implement the business logic (encryption, data validation, database operations).
  - Invalid or unauthorized requests must be blocked by throwing appropriate exceptions.
- **Definition of Done (DoD):**
  - All unit and integration tests have passed successfully.
  - API endpoints exhibit the data structure and behavior required by the frontend.
  - The code has passed code review and is ready to be merged into the main branch.
- **Evaluation:**
  All backend tests were completed successfully. Therefore, the AC and DoD targets for the backend architecture have been **fully met**.
- **Observations:**
  The backend tests show that the server-side application is **stable, reliable, and works as expected.** The layered architecture (Controller, Service, Repository) is properly established, and each layer fulfills its responsibilities. Test CTX-001 proves that the project's core integration is sound. Security, data processing, and API endpoint behaviors all yielded successful results in testing. The backend is at a production-ready level of maturity.