QUIZICALLY

SYSTEM TESTING TEST CASE

Version 1.0

11/07/2025

VERSION HISTORY

Version #	Implemented By	Revision Date	Approved By	Approval Date	Reason
1.0	Team 10	11/06/2025	Khadija	11/07/2025	

UP Template Version: 12/31/07

TABLE OF CONTENTS

1 INTRODUCTION

1.1 PURPOSE OF THE SYSTEM TEST CASE DOCUMENT

This document outlines the end-to-end system test verifying that all integrated Quizically modules (host, player, backend, and database) function together seamlessly. The goal is to confirm that data, security, and performance components interact correctly in a live simulation.

2 TEST CASE SPECIFICATION

This test case validates the complete interaction flow across Quizically's ecosystem. It ensures that the Host can create and manage quizzes, Players can join and submit answers, and the system can process, score, and store the results without error. The test also confirms that authentication (OAuth 2.0), database updates, and real-time leaderboard synchronization meet the design standards and user expectations.

2.1 DESCRIPTION

This test case verifies that a registered Quizically user can:

- Log in as Host or Player
- Create or join a game session
- Submit answers and receive scores in real time
- View final results and leaderboard positions
- Confirm data is stored in the database correctly

The test participants include:

- **QA Lead-** Executes system test plan and records results
- **Backend Engineer-** Monitors API logs and server responses
- **DevOps Support-** Maintains staging environment and network stability
- **UI/UX Reviewer-** Observers usability and accessibility factors

2.2 RESOURCES

Role	Responsibility
QA Lead	Conducts system testing and verifies integration points
Backend Developer	Checks data consistency and API performance during sessions
DevOps Engineer	Manages server logs and load balancing for testing
UI/UX Reviewer	Evaluates interface responsiveness and user experience.

2.3 PRECONDITIONS

1. Host and Player test accounts exist and are active.
2. Latest staging build deployed on Android and iOS devices.

3. Backend services are operational.
4. Network connection is stable for both devices.

2.4 POST CONDITIONS

1. All test results and logs are recorded for review.
2. Verified database entries confirm session completion and score storage.
3. All critical defects are documented and assigned for resolution.

2.5 FLOW OF EVENTS

Normal Flow:

Steps	Description	Expected Results
1	Host opens Quizically and logs in using OAuth 2.0.	Host is authenticated successfully.
2	Host selects Create Game and confirms a game from their library.	Session code is generated and visible on screen.
3	Player logs into Quizically.	Player reaches the home screen successfully.
4	Player joins the host's session via session code/QR.	Player appears correctly in the host lobby; player count updates.
5	Host starts the quiz.	First question is delivered to the player device.
6	Player submits an answer.	Answer is received and processed successfully by the backend database.
7	System calculates the score and updated leaderboard.	Leaderboard updates for all clients within ≤ 2 seconds.
8	Host and player view final game results at the end of the round/session.	Results display accurately with correct scoring.
9	Backend stores player actions, scores, and game session history in	MongoDB shows successfully written records with no errors.

	MongoDB.	

Alternate Flow A- Player Disconnects

Step	**Description**	**Expected Results**
1	Player disconnects due to temporary network loss.	Player marked as “disconnected” and game continues.
2	Player reconnects.	Player state is restored.
3	Player rejoins the ongoing game.	Player rejoins within ≤ 30 seconds without losing progress.

Alternate Flow B- API Failure

Step	**Description**	**Expected Result**
1	Simulate a failed API response during answer submission.	System retries automatically.
2	Retry fails.	Error is logged; fallback game logic or cached prompt is used.

2.6 INCLUSION/EXCLUSION POINTS

INCLUDED

3 FT-LOGIN-001-Login Functionality
4 FT-GAMECREATE-002- Create game session
5 FT-JOIN-003- Player join game
6 UT-API-004- Game prompt delivery
7 UT-PLAY-005- Player game action submission
8 UT-SCORE-001- Scoring logic
9 RT-LB-004- Leaderboard sync
10 UT-DB-006- Game data storage

LT-REJOIN-001-Rejoin behavior

EXCLUDED

- Social media posting/marketing flows
- Search/browse game library functionalities
- Analytics/event store operations

2.7 SPECIAL REQUIREMENTS

- Two test devices (Host and Player) required on the same network.
- Use Postman and JMeter for API and load testing.
- AWS CloudWatch or Firebase Monitor enabled for log capture.
- Accessibility testing under WCAG 2.1 AA standards.
- All tests performed in staging environment mirroring production.

Appendix A: References

The following table summarizes the documents referenced in this document.

Document Name and Version	Description	Location
Quizically Requirements Definition v1.0	Lists functional and non-functional requirements	Requirements Document

Appendix B: Key Terms

The following table provides definitions for terms relevant to this document.

Term	Definition
System Testing	End-to-end validation of integrated modules under real use conditions.
Session	A game instance where Host and Players interact simultaneously.
Leaderboard	A dynamic ranking display based on user scores in real time.