QA Test - Bruno César Martins

Test Case 1: User Registration via Metamask

Steps:

1. Open the frontend
2. Click "Register with Metamask"
3. Sign the transaction in Metamask
4. Verify the backend stores the wallet address in MongoDB

Expected:

- User is registered
- Wallet address appears in the users collection

Test Case 2: User Login via Metamask

Steps:

1. Click "Login with Metamask
2. Sign the message in Metamask
3. Check if the frontend receives a the token

Expected:

- User is logged in
- Token is stored in frontend (localStorage/cookies)

Test Case 3: Game Level Selection

Steps:

1. Log in
2. Select "Easy"
3. Verify the game board renders correctly

Expected:

- Game board matches the selected difficulty

Test Case 4: Card Matching Logic

Steps:

1. Start a game
2. Click two matching cards
3. Verify they stay flipped
4. Click two non-matching cards
5. Verify they flip back

Expected:

- Correct matches remain visible
- Incorrect matches flip back after a delay

Test Case 5: Game Completion & Score Saving

Steps:

1. Complete a game level
2. Check if the score is displayed
3. Verify the backend stores the result in MongoDB

Expected:

- Score appears on-screen.
- Game result is saved in the results collection

Test Case 6: Session Persistence on Refresh

Steps:

1. Log in and start a game
2. Refresh the page
3. Verify the session is retained

Expected:

- User remains logged in

Test Case 7: Error Handling (Invalid Login)

Steps:

1. Attempt login without Metamask
2. Try login with an unsigned message

Expected:

- Proper error messages appear
- No unauthorized access to the game

Test Case 8: Database Integrity Check

Steps:

1. Register a new user
2. Play some games and save results
3. Manually check MongoDB for correct data

Expected:

- No duplicate or corrupted entries
- Data relationships (user ↔ results) are maintained

Practices for Maintaining Quality

- Automated Testing: unit/integration tests, API testing and user flows simulation
- CI/CD Pipeline
  - Run tests on every push (GitHub Actions).
  - Deploy to a staging environment before production
- Security
  - Validate tokens on every API call
  - Sanitize MongoDB queries to prevent NoSQL injection
- Performance Optimization
  - Simulate 100+ users
  - Optimize API response times