Document Version: 1.0
Last Updated: February 11, 2026
Owner: QA Lead
Status: Draft for Review

Table of Contents

---

1. Executive Summary

Purpose

This document defines the comprehensive test strategy for Alias Web (PartyFlow) MVP, with emphasis on real-time multiplayer edge cases, WebSocket reliability, and cross-device synchronization quality.

Key Focus Areas

- **Real-time multiplayer reliability**: Disconnect/reconnect scenarios, race conditions
- **Data integrity**: Cross-device score synchronization, timer accuracy
- **Cross-platform compatibility**: iOS Safari, Android Chrome, desktop browsers
- **Performance**: WebSocket server load capacity (target: 50 concurrent rooms)
- **PWA functionality**: Installation, offline behavior, notifications

Success Metrics

Metric	Target	Measurement Method
Test Coverage	>90% for critical paths	Code coverage + manual scenario coverage
Defect Detection Rate	>80% bugs found pre-launch	Bugs found in QA / Total bugs
Load Test Pass Rate	100% at 50 concurrent rooms	Artillery.io automated tests
Cross-Device Pass Rate	>95% on primary devices	Manual testing on 7 device matrix
Reconnect Success Rate	>98% within 15sec	Automated disconnect simulation

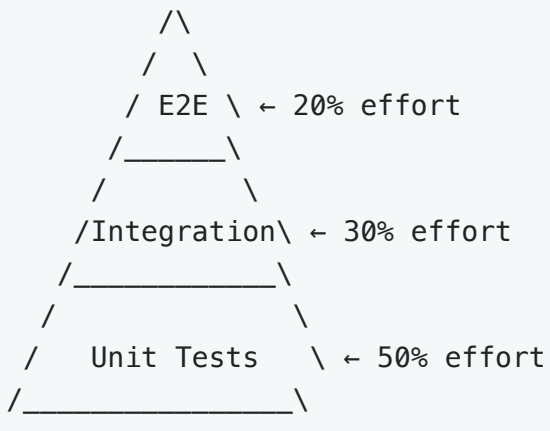
Risk Areas Requiring Extra Scrutiny

1. **WebSocket connection stability** — critical path for entire product
2. **Timer synchronization** — desyncs >2sec destroy game experience
3. **Race conditions on swipe actions** — must prevent double-counting
4. **iOS Safari PWA behavior** — Apple's WebSocket handling differs from spec
5. **Mass disconnect/reconnect** — server must handle Friday night traffic spikes

---

2. Test Strategy Overview

2.1 Testing Pyramid



Distribution:

- **Unit Tests (50%)**: WebSocket handlers, game logic, scoring algorithms

- Integration Tests (30%): Room creation → join → play flow, DB operations
- E2E Tests (20%): Full user journeys across real devices, load tests

2.2 Test Levels

Level	Scope	Owner	Automation
Unit	Individual functions, classes	Dev Team	100% (Jest/Pytest)
Integration	API endpoints, WebSocket messages, DB queries	Dev Team	90% (automated)
System	Full feature flows (create room → play game → see scores)	QA Lead	60% (Playwright E2E)
Acceptance	User scenarios on real devices	QA Lead + PM	40% (manual + automated)
Performance	Load, stress, spike testing	DevOps + QA	100% (Artillery.io)
Exploratory	Unscripted testing, edge case discovery	QA Lead	0% (manual only)

2.3 Test Environments

Environment	Purpose	Data	URL
Local	Dev unit testing	Mocked	localhost:3000
Dev	Integration testing, feature branches	Synthetic	dev.partyflow.game
Staging	Pre-production, load testing	Production-like	staging.partyflow.game
Production	Live monitoring, smoke tests	Real users	partyflow.game

---

3. Acceptance Criteria Framework

3.1 AC Quality Checklist

Every User Story AC must satisfy:

- [ ] Observable outcome — describes what user sees/hears (not internal state)
- [ ] Measurable criteria — includes numbers (time, count, percentage)

- [ ] **Pre-conditions** — clear starting state
- [ ] **Post-conditions** — clear ending state
- [ ] **Edge cases** — abnormal paths covered (errors, timeouts, race conditions)
- [ ] **Error handling** — describes what happens when things go wrong

3.2 AC Template — Good vs Bad Examples

❌ BAD EXAMPLE (Not Testable)

```
AC: "Players can join a room"
```

Problems:

- No measurable outcome
- No error cases
- No performance criteria
- No edge cases

✅ GOOD EXAMPLE (Testable)

```
AC: Player Successfully Joins Room

GIVEN:
  - Room code "AB12" exists in DB
  - Room status = "lobby" (not "in_progress")
  - Current players count = 3/10
  - Player has stable internet connection

WHEN:
  - Player enters "AB12" in join input field
  - Player taps "Join Room" button

THEN:
  Success Path:
  - Player sees lobby screen within 500ms
  - Player count updates to 4/10 on ALL devices within 200ms
  - Host sees new player avatar + name appear in lobby list
  - WebSocket connection established (readyState = 1)
  - Player's device plays "join success" sound (optional)

  Error Paths:
  - Room code invalid → error message "Room not found" + suggestion to check
code
  - Room full (10/10) → error message "Room is full, try another"
```

```
  - Room already started → error message "Game in progress, cannot join"
  - Network timeout (>3sec no response) → loading spinner + retry button
appears
  - WebSocket fails to connect → fallback to "Check your connection" message

Edge Cases to Test:
  - Room code with lowercase ("ab12") → auto-converts to "AB12"
  - Room code with spaces (" AB 12 ") → strips spaces
  - Player rejoins same room after disconnect → recognizes returning player
  - Player spams "Join" button 5x in 1sec → only 1 join request processed
  - Host deletes room while player joining → join fails gracefully

Performance Requirements:
  - Join latency P95 < 500ms from button tap to lobby screen
  - Join latency P99 < 1000ms

Non-Functional:
  - Works on iOS Safari 15+, Android Chrome 90+
  - Accessible: Button has aria-label="Join room"
```

3.3 AC Review Process

Step 1: PM writes initial AC in PRD
Step 2: QA Lead reviews using checklist (Section 3.1)
Step 3: QA adds missing edge cases + error paths
Step 4: Dev Team reviews for technical feasibility
Step 5: Final AC locked before development starts

SLA: AC review completed within 48hrs of PRD draft

---

4. Multiplayer Edge Cases Test Suite

4.1 Disconnect Scenarios (DC-xxx)

Test ID	Scenario	Pre-Condition	Trigger Action	Expected Behavior	Recovery Test	Pri
DC-001	Host disconnects during lobby	4 players in lobby, game not started	Host closes browser / loses WiFi	• All players see "Host disconnected" overlay (within 2sec) • Room marked as "orphaned" in	Host reconnects within 60sec → room restored, players can	P0

Test ID	Scenario	Pre-Condition	Trigger Action	Expected Behavior	Recovery Test	Pri
				DB • Players auto-redirected to home screen after 5sec countdown • Room deleted from DB after 60sec (cleanup job)	rejoin using same code	
DC-002	Explainer disconnects mid-round	Round active, Explainer has 30sec left, 3 words explained	Explainer device goes offline (airplane mode)	• Timer pauses immediately on all devices • "Explainer disconnected – waiting for reconnect" overlay shown • Other players see paused game state • 15sec grace period starts • If no reconnect → skip to next player's turn automatically	Explainer reconnects within 15sec → timer resumes from paused value, game continues	P0
DC-003	Mass disconnect (4/5 players)	5 players active, game in round 2	4 devices lose connection simultaneously (simulate router restart)	• Remaining player sees "Multiple players disconnected" • Game state frozen • 30sec grace period for mass reconnect • If <2 players after 30sec → end game, show final scores • If ≥2 players → game resumes	3+ players reconnect → game continues from saved state	P1

Test ID	Scenario	Pre-Condition	Trigger Action	Expected Behavior	Recovery Test	Pri
				from last checkpoint		
DC-004	Brief WiFi dropout (2-3sec)	Game active, stable 50ms latency	WiFi drops for 2sec then auto-recovers	• Client-side buffering of swipe actions in memory • Automatic reconnect (exponential backoff: 1s, 2s, 4s) • Buffered actions synced to server on reconnect • No user-visible disruption if <3sec • Optional "Reconnecting..." toast only if >3sec	N/A — transparent recovery	P0
DC-005	Host disconnects during active round	Round 2/3 active, Explainer mid-turn	Host closes app permanently	• Current round continues to completion • After round ends → "Host left the game" message shown • Game ends immediately, scores frozen at current values • No new rounds started • Players redirected to home after 5sec	Host cannot recover — must create new room	P1
DC-006	Player disconnects during scoring animation	End of round, +3 points animating on screen	Player's device goes offline during score update	• Score update persists on server (canonical source) • Offline player's	Player reconnects → sees accurate leaderboard	P2

Test ID	Scenario	Pre-Condition	Trigger Action	Expected Behavior	Recovery Test	Pri
				score still counts in leaderboard • Animation completes for online players • Reconnected player sees correct updated total score	with their updated score	

Test Execution Method:

- Manual testing: Use airplane mode, kill app, router restart
- Automated testing: WebSocket close() method with controlled timing
- Chaos engineering: Randomly disconnect 10-30% of connections in load test

---

4.2 Race Conditions (RC-xxx)

Test ID	Scenario	Setup	Conflict Trigger	Expected Resolution	Test Met
RC-001	Simultaneous swipe on same word	2+ players viewing same word card (e.g., "ELEPHANT")	Both players swipe "correct" within 50ms window	• Server receives both messages with timestamps • First swipe (by server timestamp) wins • Second swipe ignored by server • Second player's client shows "already swiped" haptic feedback • Only one +1 point awarded to team • Server broadcasts winning player's action to all clients	Automat WebSock clients se swipe message with <10r delta, ver only 1 poi added
RC-002	Host starts game during	Host in lobby with 3	Host taps "Start Game"	• Server processes messages in	Automat WebSock

Test ID	Scenario	Setup	Conflict Trigger	Expected Resolution	Test Meth
	late player join	players, 4th player's join request in-flight over network	while Player 4's JOIN message is being transmitted (simulated 200ms network delay)	received order • **If JOIN arrives before START:** Player 4 joins successfully, game starts with 4 players • **If JOIN arrives after START:** Player 4 gets "Game already started" error • No partial join state allowed (either fully in or fully out) • Database transaction ensures atomicity	message order test controlled network d injection
RC-003	Swipe at exact timer expiry (T=0)	Timer showing "00:01" on all devices, player preparing swipe	Player swipes exactly when timer hits 00:00 (within ±100ms)	• Server compares: swipe_timestamp < timer_end_timestamp • **If swipe_ts < timer_end:** Action counts, point awarded • **If swipe_ts ≥ timer_end:** Action rejected, no point • Client shows visual feedback (green ✓ vs red ✕) • Edge case: if within 50ms of timer_end → server decision is final (no appeal)	**Automat** Send swi message exactly at server tim expiry usi NTP-sync clock
RC-004	Host spam-clicks "Next Turn" button	End of round, "Next Turn" button visible to host	Host rapidly clicks button 5 times in 1sec (frustrated user or UI lag)	• Server processes only first click (idempotency via round_id key) • Subsequent clicks return 409 Conflict or are silently ignored • Only one new round created in DB • Client-side	**Automat** Send 5 identical NEXT_RC messages within 1se verify onl round cre

Test ID	Scenario	Setup	Conflict Trigger	Expected Resolution	Test Meth
				debouncing (300ms) prevents multiple requests • Button disabled after first click until next state	
RC-005	Concurrent word pack purchase	2 players in same room viewing shop screen	Both tap "Buy Pack A" ($2.99) simultaneously	• First transaction acquires DB lock on pack_id for room_id • Second transaction sees "Already purchased by [Player1]" message • Only one payment processed (Stripe idempotency key) • Pack unlocked for entire room (all players get access) • Second player sees "Pack already unlocked!" confirmation	Automat API reque to /purch with same pack_id + room_id, verify onl charge

Race Condition Testing Tools:

- Custom WebSocket load test client (Node.js + ws library)
- Artillery.io for concurrent user simulation
- Database transaction isolation testing (SQL `BEGIN; ... COMMIT;`)

4.3 Data Integrity & Desync Detection (DI-xxx)

Test ID	Check Description	Validation Frequency	Failure Threshold	Auto-Recovery Mechanism	Alert Trigger
DI-001	Score desync between devices	After every point scored (every	Any difference >0 points between: • Host device	• Server broadcasts FORCE_SYNC message to all clients • Clients overwrite local score with	Log error to Sentry if desync detected >2 times

Test ID	Check Description	Validation Frequency	Failure Threshold	Auto-Recovery Mechanism	Alert Trigger
		~5sec during active round)	score • sum(all player scores) • Server canonical score	server value • Animation re-plays to show corrected score • Canonical source = server DB (always wins)	in single game
DI-002	Timer drift cross-device	Every 5sec during active round	Drift >1.5sec between any client timer and server timer	• Client sends heartbeat with local timer value • Server compares: `abs(client_time - server_time) > threshold` • Server responds with authoritative timer value • Client resets timer to `server_time + (latency / 2)` for compensation • If drift >3sec → force reload game state	Alert DevOps if >10% of clients experience >3sec drift
DI-003	Word card order mismatch	At start of each new round	Hash mismatch on any client's word deck	• Server generates word deck for round: `[word1, word2, ..., word20]` • Server sends `SHA256(word_ids)` hash to all clients • Each client computes local hash and compares • If mismatch → client requests full deck re-send • Server re-broadcasts authoritative word array • Clients rebuild UI with correct order	Log + notify dev team (possible cache corruption)

Test ID	Check Description	Validation Frequency	Failure Threshold	Auto-Recovery Mechanism	Alert Trigger
DI-004	Player list desync	After every join/leave event	Player count mismatch: `host_count ≠ server_count`	• Periodic sync (every 10sec): server broadcasts full player list • Clients compare: `local_player_ids` vs `server_player_ids` • If mismatch → client re-renders player list from server data • Optimistic UI updates allowed but verified within 500ms	Alert if desync persists after >2 sync attempts
DI-005	Game state inconsistency	Before critical state transitions (start round, end game, next turn)	Any client in wrong state: • Client thinks "lobby" but server says "round_active" • Client thinks "round_2" but server says "round_3"	• Client sends GET `/room/{code}/state` to fetch current state • Server responds with canonical state machine position • Client rehydrates entire local state from server response • UI force re-renders to correct screen	Force client reload if desync occurs >3 times in single game session

Desync Testing Approach:

1. **Simulate clock skew:** Offset client device time by +5sec, verify auto-correction
2. **Inject bad data:** Modify client-side score in localStorage, verify server overrides
3. **Network partition:** Block messages between client and server for 10sec, verify state recovery
4. **Stress test:** 10 concurrent games, monitor for any desync occurrences

---

5. Cross-Browser & Device Testing Matrix

5.1 Primary Test Matrix (P0 Priority)

Device	OS Version	Browser	Screen Size	Viewport	Critical Test Scenarios	Pass Criteria
iPhone 13 Pro	iOS 16+	Safari	390×844px	Portrait	• Join room flow (QR code + manual code) • Swipe gestures (up/down recognition) • PWA install (Add to Home Screen) • Reconnect after 30sec background • Timer accuracy (no drift >1sec) • Audio playback (timer beep)	100% pass on all scenarios
Samsung Galaxy S21	Android 12+	Chrome	360×800px	Portrait	• Join room flow • Haptic feedback on swipe • Push notification (turn reminder) • Install banner triggering • WebSocket on mobile data vs WiFi	100% pass
iPhone SE (2020)	iOS 15+	Safari	375×667px	Portrait	• Small screen layout (no text truncation) • Button tap	95% pass (minor UI adjustments OK)

Device	OS Version	Browser	Screen Size	Viewport	Critical Test Scenarios	Pass Criteria
					targets ≥44px (accessibility) • Readability at default font size	
iPad Air	iOS 16+	Safari	820×1180px	Portrait + Landscape	• Host tablet UI (larger lobby grid) • Landscape mode support • Keyboard shortcuts (Space = next word)	100% pass

5.2 Secondary Test Matrix (P1 Priority)

Device	OS	Browser	Purpose	Pass Criteria
MacBook Pro	macOS 13+	Chrome	Host desktop experience	90% pass
Pixel 6	Android 13	Firefox	Browser compatibility edge cases	85% pass
iPad Pro	iOS 17	Safari	Latest iOS WebSocket behavior	100% pass
Windows Laptop	Windows 11	Edge	Corporate network testing	80% pass

5.3 Browser-Specific Known Issues & Tests

iOS Safari Critical Tests

Test Case	Expected Behavior	Known Issue to Verify Fixed
WebSocket survives app backgrounding	App backgrounded 30sec → foregrounded → connection auto-reconnects within 3sec	iOS kills WS after 30sec background (test workaround)

Test Case	Expected Behavior	Known Issue to Verify Fixed
Audio playback without user gesture	Timer beep plays automatically when round ends	Safari blocks auto-play (need user interaction first)
Add to Home Screen flow	PWA manifest detected, install banner shown after 2 visits	Safari's Service Worker implementation differs from spec
Viewport height with address bar	UI doesn't jump when address bar hides/shows (use `dvh` units)	100vh includes address bar, causing layout shift
Swipe gestures don't trigger back nav	Swipe up/down on word cards doesn't navigate back	Need `touch-action: pan-y` CSS

Android Chrome Critical Tests

Test Case	Expected Behavior	Known Issue to Verify Fixed
WebSocket on network switch	Connection survives WiFi → mobile data switch (auto-reconnect within 5sec)	Some Android versions drop WS on network change
Install banner triggering	Banner shows after: 2+ visits, 5min engagement, HTTPS, valid manifest	Criteria must be exact or banner won't show
Fullscreen standalone mode	App opens without browser chrome (status bar only)	`display: "standalone"` in manifest
Vibration API	Haptic feedback works on swipe actions	`navigator.vibrate([50])` support varies by device

Cross-Browser Universal Tests

- [ ] WebSocket reconnect after 3min idle (all browsers)
- [ ] LocalStorage persistence across sessions (room code autofill)
- [ ] ServiceWorker registration and update (PWA cache)
- [ ] WebRTC fallback if WebSocket fails (future feature)
- [ ] HTTPS enforcement (no mixed content warnings)

5.4 Device Testing Logistics

Physical Device Lab:

- iPhone 13 Pro (iOS 16.5) — QA Lead's personal device
- Samsung Galaxy S21 (Android 12) — QA Lead's personal device
- iPad Air (iOS 16) — borrowed from team
- Budget: $0 (use personal devices + BrowserStack)

Cloud Testing (BrowserStack):

- Real device testing for devices not in physical lab
- Automated Playwright tests run on cloud devices
- Cost: $99/month (included in dev tools budget)

Testing Schedule:

- Every feature branch: Quick smoke test on iPhone + Android (15min)
- Pre-merge to main: Full matrix test (2hrs)
- Pre-launch: 2-day full regression on all 7 devices

---

6. Load Testing Strategy

6.1 Load Testing Objectives

Primary Goal: Validate that WebSocket server can handle **50 concurrent rooms** (500 active players) with acceptable performance.

Key Metrics to Validate:

Metric	Target	Measurement
Message Latency (P95)	<200ms	Time from client sends swipe → all other clients receive update
Message Latency (P99)	<500ms	99th percentile to account for network spikes
Message Loss Rate	0%	Every swipe action must be received by server
Connection Success Rate	>99%	Room join success rate
Server CPU	<70%	Sustained load shouldn't exceed 70% (need headroom for spikes)
Server Memory	<2GB	Memory usage under load (check for leaks)
Connection Drops	<1%	Unexpected WebSocket disconnections

Metric	Target	Measurement
Database Query Time	<100ms	Room fetch, score update, player join queries

6.2 Load Test Scenarios

Scenario 1: Steady State Load (Baseline)

Objective: Validate normal Friday evening traffic

```
Configuration:
- Concurrent Rooms: 50
- Players per Room: 10 (500 total connections)
- Duration: 10 minutes
- Message Rate: ~30 msg/sec per room
  - 1 swipe every 3-5sec × 10 players = ~2-3 swipes/sec
  - Score updates, timer ticks = ~28 msg/sec overhead
- Total Message Throughput: 1,500 msg/sec across all rooms

Success Criteria:
✓ P95 latency <200ms (swipe action → all clients update)
✓ P99 latency <500ms
✓ 0% message loss (every swipe received by server)
✓ Server CPU <70% sustained
✓ Memory <2GB RAM, no memory leaks
✓ 0 connection drops
✓ Database query P95 <50ms
```

Test Execution:

```
artillery run --config load-test-steady-state.yml
```

Scenario 2: Spike Test (Launch Day Traffic)

Objective: Simulate sudden traffic surge (Product Hunt launch, viral tweet)

```
Configuration:
- Ramp-up: 0 → 100 rooms in 2 minutes (500 → 1000 players)
- Hold: 100 rooms for 5 minutes
- Ramp-down: 100 → 0 in 1 minute

Monitors:
```

```
- Connection establishment time (should be <1sec even during spike)
- WebSocket upgrade success rate (>99%)
- Database connection pool exhaustion (should not happen)
- Error rate (4xx, 5xx responses <1%)

Success Criteria:
✓ All connections established within 3sec during ramp-up
✓ No database connection pool exhaustion
✓ Graceful degradation if capacity exceeded (user sees "Server busy, retry")
✓ System recovers fully after spike (memory returns to baseline)
```

Scenario 3: Reconnect Storm (Network Instability)

Objective: Simulate mass disconnect/reconnect event (WiFi router restart, ISP outage)

```
Configuration:
- Initial: 50 rooms, 500 active connections, stable for 2min
- Event: 30% of players (150 connections) force disconnect simultaneously
- All 150 attempt reconnect within 10sec window
- Continue game after reconnect

Success Criteria:
✓ All 150 reconnects successful within 15sec
✓ No ghost connections in database (orphaned records cleaned up)
✓ Game states correctly restored (scores, timer, round number)
✓ Server doesn't crash or become unresponsive
✓ Memory doesn't spike >3GB during reconnect storm
✓ Players resume gameplay seamlessly
```

Test Execution:

```
// Custom Artillery processor
module.exports = {
  simulateReconnectStorm: async (context, events, done) => {
    // Wait for stable state
    await sleep(120000); // 2min

    // Disconnect 30% of connections
    const connections = context.connections;
    const toDisconnect = connections.slice(0, Math.floor(connections.length
* 0.3));

    toDisconnect.forEach(ws => ws.close());
```

```
    // Attempt reconnect after 1sec
    await sleep(1000);
    toDisconnect.forEach(ws => ws.reconnect());

    done();
  }
};
```

6.3 Load Testing Script (Artillery.io)

File: `artillery-load-test.yml`

```yaml
config:
  target: "wss://api.partyflow.game"
  phases:
    # Scenario 1: Steady State
    - duration: 120  # 2min ramp-up
      arrivalRate: 5  # 5 new players/sec → 600 players total
      name: "Ramp-up to 50 rooms"
    - duration: 600  # 10min steady state
      arrivalRate: 0  # maintain existing connections
      name: "Steady state load"

  # WebSocket specific config
  ws:
    reconnect: true
    reconnectDelay: 1000  # 1sec between retries
    maxRetries: 3

  processor: "./load-test-processor.js"

  # Monitoring plugins
  plugins:
    expect: {}
    metrics-by-endpoint: {}

  # Performance thresholds
  ensure:
    maxErrorRate: 1  # <1% errors
    p95: 200  # P95 latency <200ms
    p99: 500  # P99 latency <500ms
```

```yaml
scenarios:
  # Host creates room and runs game
  - name: "Host Flow"
    weight: 10  # 10% of users are hosts
    engine: ws
    flow:
      # 1. Create room
      - send:
          channel: "room.create"
          data:
            name: "Test Room {{ $randomString() }}"
            hostName: "Host{{ $randomNumber(1, 1000) }}"
      - think: 1

      # 2. Wait in lobby (15sec for players to join)
      - think: 15

      # 3. Start game
      - send:
          channel: "game.start"
      - think: 2

      # 4. Play 3 rounds
      - loop:
        # Each round: 60sec gameplay + 10sec scoring
        - think: 60  # Round active time
        - send:
            channel: "round.end"
        - think: 10  # Scoring screen
        - send:
            channel: "round.next"
        count: 3

      # 5. End game
      - send:
          channel: "game.end"

  # Player joins and plays
  - name: "Player Flow"
    weight: 90  # 90% of users are players
    engine: ws
    flow:
      # 1. Join room
      - send:
          channel: "room.join"
          data:
```

```
            code: "{{ roomCode }}"  # Injected by processor
            name: "Player{{ $randomNumber(1, 1000) }}"
      - think: 2

      # 2. Wait in lobby
      - think: 13  # Host waits 15sec, we join ~2sec after

      # 3. Play 3 rounds (simulate swipes)
      - loop:
        # Each round: swipe every 3-5sec for 60sec
        - loop:
          - send:
              channel: "word.swipe"
              data:
                direction: "{{ $randomChoice(['correct', 'skip']) }}"
                timestamp: "{{ $timestamp() }}"
          - think: "{{ $randomNumber(3, 5) }}"  # 3-5sec between swipes
          count: 15  # ~15 swipes per 60sec round

        # Wait for scoring + next round
        - think: 10
        count: 3
```

File: `load-test-processor.js`

```
// Maintains pool of active room codes for players to join
const activeRooms = [];
const roomCapacity = new Map(); // Track player count per room

module.exports = {
  // Before scenario starts
  beforeScenario: (context, events, done) => {
    // Listen for room creation events
    events.on('room.created', (data) => {
      if (activeRooms.length < 50) { // Max 50 rooms for test
        activeRooms.push(data.code);
        roomCapacity.set(data.code, 1); // Host counts as 1 player
        console.log(`Room created: ${data.code} (${activeRooms.length} total
rooms)`);
      }
    });

    // Listen for player join events
    events.on('player.joined', (data) => {
      const count = roomCapacity.get(data.roomCode) || 0;
      roomCapacity.set(data.roomCode, count + 1);
```

```
    });

    done();
  },

  // Assign room code to player
  setRoomCode: (context, events, done) => {
    // Find room with <10 players
    const availableRoom = Array.from(roomCapacity.entries())
      .find(([code, count]) => count < 10);

    if (availableRoom) {
      context.vars.roomCode = availableRoom[0];
    } else {
      // Fallback: pick random room (might be full, will test error
handling)
      context.vars.roomCode = activeRooms[
        Math.floor(Math.random() * activeRooms.length)
      ] || 'TESTROOM';
    }

    done();
  },

  // Add timestamp for race condition testing
  timestamp: (context, events, done) => {
    context.vars.timestamp = Date.now();
    done();
  }
};
```

6.4 Load Test Monitoring & Metrics

Metrics Dashboard (Grafana):

Panel	Metric	Query	Aler Thre
WebSocket Latency	Time: swipe sent → ack received	`histogram_quantile(0.95, rate(ws_message_latency_ms[5m]))`	P95 >20
Message Loss Rate	% of messages not	`(ws_messages_sent - ws_messages_acked) / ws_messages_sent * 100`	>0.1

Panel	Metric	Query	Aler Thre
	acknowledged		
Active Connections	Current WebSocket connections	`sum(ws_connections_active)`	>12((12(targ
CPU Usage	Server CPU percentage	`100 - (avg(rate(node_cpu_seconds_total{mode="idle"}[5m])) * 100)`	>70'
Memory Usage	Server RAM usage	`node_memory_MemTotal_bytes - node_memory_MemAvailable_bytes`	>2G
DB Query Time	P95 query latency	`histogram_quantile(0.95, rate(db_query_duration_ms[5m]))`	P95 >10(
Error Rate	5xx responses / total	`rate(http_requests_total{status=~"5.."}[5m]) / rate(http_requests_total[5m])`	>1%

Real-Time Monitoring During Test:

```
# Terminal 1: Run Artillery test
artillery run load-test-steady-state.yml --output report.json

# Terminal 2: Watch server metrics
watch -n 1 'curl -s http://api.partyflow.game/metrics | grep ws_connections'

# Terminal 3: Monitor database
psql -h db.partyflow.game -c "SELECT COUNT(*) FROM active_connections;"
```

Post-Test Analysis:

```
# Generate HTML report
artillery report report.json --output report.html

# Key metrics to review:
# - scenarios.completed (should be >95% of scenarios.created)
# - errors (should be <1%)
# - http.response_time.p95 (<200ms)
# - http.response_time.p99 (<500ms)
```

6.5 Load Test Success Criteria Summary

Test Scenario	Duration	Target Load	Pass Criteria
Steady State	10min	50 rooms, 500 players	Latency P95 <200ms, 0% msg loss, CPU <70%
Spike Test	8min	0→100 rooms in 2min	All connects <3sec, no crashes, graceful recovery
Reconnect Storm	5min	150 simultaneous reconnects	100% success within 15sec, no data loss

Pre-Launch Requirement: All 3 scenarios must pass before public launch.

---

7. Edge Case Paranoia Checklist

7.1 Network & Connectivity

- [ ] WiFi → Mobile Data switch mid-game (Android auto-switches network)
- [ ] Airplane mode enabled → disabled (iOS kills all connections, must reconnect)
- [ ] Router restart during active game (all players disconnect simultaneously)
- [ ] VPN connection toggle (IP address changes mid-session)
- [ ] Proxy/firewall blocking WebSocket upgrade (corporate networks, some hotels)
- [ ] IPv4 vs IPv6 connection behavior (some ISPs prefer one over other)
- [ ] Captive portal redirect (coffee shop WiFi requires login mid-game)
- [ ] Network throttling (slow 3G, 2G fallback, test performance)
- [ ] Packet loss simulation (test with 5%, 10%, 20% packet loss)

7.2 Device States

- [ ] Phone call interruption during player's turn (iOS/Android handle differently)
- [ ] Low battery mode enabled (iOS throttles background tasks, timers may slow)
- [ ] App backgrounded for 1min / 5min / 30min (test various background durations)
- [ ] Device auto-lock during Explainer's turn (screen dims then locks)
- [ ] Screenshot taken during gameplay (some games pause, should we?)
- [ ] Notification banner appears (push notification overlays UI)
- [ ] OS update prompt mid-game ("iOS 18 is ready to install")
- [ ] Device storage full (can't save game state to localStorage)
- [ ] Device overheating (CPU throttled, performance degrades)
- [ ] Accessibility features enabled (VoiceOver, screen reader, font scaling)

7.3 Browser Behavior

- [ ] Browser tab switched (does game pause or continue in background?)
- [ ] Browser refreshed during active round (F5 or pull-to-refresh)
- [ ] Browser dev tools opened (affects performance, layout inspector active)
- [ ] Browser back button pressed from game screen (should prevent or confirm?)
- [ ] Multiple tabs open to same room (same user, different devices)
- [ ] Browser "request desktop site" toggled on mobile (UI breaks?)
- [ ] Browser cache cleared mid-game (localStorage, session data lost)
- [ ] Browser extensions interfere (ad blockers, privacy tools block WS?)
- [ ] Browser memory limit reached (tab crashes on low-RAM devices)
- [ ] Incognito/private browsing mode (localStorage disabled, session-only)

7.4 Timing Edge Cases

- [ ] Timer reaches 00:00 but client clock skewed by +2sec (NTP sync failed)
- [ ] Explainer swipes last word at T=00:00.050sec (server timestamp vs client)
- [ ] Host clicks "Next Round" during scoring animation (state transition race)
- [ ] Two rounds end simultaneously (if multi-room bug, invalid state)
- [ ] Game started exactly at midnight (date rollover, timestamp wraparound?)
- [ ] Daylight saving time change during active game (timezone shift)
- [ ] Leap second occurs during timer countdown (rare but possible)
- [ ] Timer animation frame drops (low FPS on old devices, timer appears frozen)
- [ ] System clock manually changed by user mid-game (time travel exploit?)

7.5 Data Boundary Cases

- [ ] Player name = empty string ("")
- [ ] Player name = 50 character emoji string (🎉🎊🎈 repeated)
- [ ] Player name = SQL injection attempt (`'; DROP TABLE players;--`)
- [ ] Player name = XSS payload (`<script>alert('xss')</script>`)
- [ ] Room code = "0000" (all zeros, valid but unusual)
- [ ] Room code = "AAAA" (all same letter)
- [ ] Room with 1 player (host only) tries to start game (should block)
- [ ] Room with 11 players (exploit: bypass 10 player limit)
- [ ] Word pack with 0 words (corrupted data or empty pack)
- [ ] Word pack with duplicate words (same word appears 5 times)
- [ ] Score = 999+ (UI overflow? Does leaderboard fit?)
- [ ] Score = negative (exploit: swipe down multiple times)

- [ ] 10 players all named "Player 1" (display collision in lobby)
- [ ] Player name with RTL characters (Hebrew, Arabic text direction)
- [ ] Player name with zero-width characters (invisible name exploit)

7.6 WebSocket Protocol Edge Cases

- [ ] Client sends malformed JSON message (`{invalid json}`)
- [ ] Client sends non-JSON message (binary data, plain text)
- [ ] Client floods server with 1000 msg/sec (rate limiting test)
- [ ] Server sends message to disconnected client (ghost connection)
- [ ] Ping/pong timeout (30sec no activity, keepalive test)
- [ ] WebSocket frame size >64KB (large word list, image upload)
- [ ] Binary data sent instead of text frame (wrong message type)
- [ ] Client doesn't respond to PING (dead connection detection)
- [ ] Server restart during active games (state recovery, graceful shutdown?)
- [ ] Load balancer switches WS connections mid-session (sticky session test)
- [ ] TLS certificate expires mid-game (rare, but test error handling)
- [ ] DNS resolution fails temporarily (reconnect with cached IP?)

7.7 Security & Abuse Edge Cases

- [ ] Replay attack (attacker replays captured swipe messages)
- [ ] Man-in-the-middle (attacker intercepts WS messages, can't decrypt with TLS)
- [ ] Room code brute force (attacker tries 10,000 codes to join random rooms)
- [ ] Score manipulation (modified client tries to send fake score)
- [ ] Timer manipulation (client sends "time's up" early)
- [ ] Impersonation attack (player B claims to be player A)
- [ ] DOS via room creation (create 10,000 rooms to exhaust DB)
- [ ] Word pack piracy (free user tries to unlock paid pack)
- [ ] Cross-site WebSocket hijacking (malicious site opens WS to our server)

---

8. Test Execution Plan & Timeline

8.1 Development Phase Testing

Timeline: Weeks 1-8 (during MVP development)

Week	Milestone	Testing Activity	Owner	Deliverable
W1-2	Module 1 (Room & Lobby) complete	• Unit tests for room CRUD • Integration test: create → join flow • Manual test on 2 devices	Dev Team + QA	Test report (pass/fail per AC)
W3-4	Module 2 (Core Game Loop) complete	• Unit tests for game logic • Disconnect test (DC-002) • Race condition test (RC-001) • Timer sync test (DI-002)	Dev Team + QA	Edge case test results
W5-6	Module 3 (Word Packs) + Module 4 (Dual-Screen Sync)	• Integration test: pack unlock flow • Desync detection test (DI-001, DI-003) • Cross-device manual test (phone + tablet)	QA Lead	Cross-device test matrix
W7	Module 5 (PWA) complete	• PWA install flow (iOS + Android) • Offline fallback page • Service Worker caching	QA Lead	PWA checklist
W8	All modules integrated	• Full regression test (all ACs) • Smoke test on 7 device matrix • Exploratory testing (find edge cases)	QA Lead	Regression report + bug list

---

8.2 Pre-Launch Testing Phase

Timeline: Weeks 9-10 (before soft launch)

Day	Activity	Description	Success Criteria
Day 1-2	Automated E2E Test Suite	• Playwright scripts for critical paths • Automated disconnect tests • Run on BrowserStack (iOS + Android)	100% critical paths pass
Day 3-4	Load Testing	• Run Artillery steady state test (50 rooms) • Run spike test (0→100 rooms) • Run reconnect storm test	All 3 scenarios pass
Day 5	Security Audit	• Penetration testing (room code brute force) • Input validation tests (XSS, SQLi) • Rate limiting tests	No P0/P1 vulnerabilities
Day 6-7	Cross-Browser Compatibility	• Full test on 7 devices in matrix • iOS Safari + Android Chrome priority • Test PWA install on real devices	>95% pass rate
Day 8-9	Beta User Testing	• Recruit 20 beta testers (10 groups) • Provide feedback form • Monitor Sentry errors	<5 P0/P1 bugs reported
Day 10	Bug Fixes & Retest	• Fix all P0/P1 bugs from beta • Retest affected areas • Sign-off for launch	All P0/P1 bugs resolved

---

8.3 Post-Launch Monitoring

Timeline: Ongoing after launch

Frequency	Activity	Tool	Alert Threshold
Real-time	Error monitoring	Sentry	Any P0 error occurs
Real-time	WebSocket connection health	Grafana dashboard	>5% disconnect rate
Real-time	Server performance	CloudWatch / Datadog	CPU >80%, Memory >3GB

Frequency	Activity	Tool	Alert Threshold
Hourly	Synthetic monitoring	Checkly (uptime check)	Any endpoint down
Daily	User analytics review	Amplitude / Mixpanel	D1 retention <40%
Weekly	Bug triage meeting	Jira / Linear	Prioritize P1/P2 bugs
Monthly	Regression test suite	Playwright (automated)	<95% pass rate

---

8.4 Testing Resource Allocation

Role	Time Commitment	Responsibilities
QA Lead (you)	100% (full-time)	• Manual testing on devices • Write test cases • Coordinate beta testing • Sign-off for launch
Backend Dev	20% (8hrs/week)	• Write unit tests • Fix bugs found in QA • Support load testing setup
Frontend Dev	20% (8hrs/week)	• Write Playwright E2E tests • Fix UI bugs • Optimize performance
DevOps	10% (4hrs/week)	• Set up Artillery load tests • Configure monitoring alerts • Support infrastructure testing
PM	5% (2hrs/week)	• Review test reports • Prioritize bug fixes • Coordinate beta testers

Total Team Testing Effort: ~1.5 FTE for 10-week MVP cycle

---

9. Definition of Done — Per Module

9.1 Module 1: Room & Lobby System

Functional Requirements:

- [ ] All Acceptance Criteria in PRD pass manual testing
- [ ] Room code generation produces unique 4-character codes (tested 10,000 times, no collisions)
- [ ] Room code validation accepts valid formats (A-Z, 0-9, case-insensitive)
- [ ] Invalid room codes show appropriate error messages
- [ ] Room capacity enforced (max 10 players, 11th player gets "Room full" error)
- [ ] Lobby updates in real-time on all devices when player joins/leaves

Edge Cases Tested:

- [ ] DC-001 (Host disconnect in lobby) handled gracefully — all players redirected
- [ ] 4 players join simultaneously — no race conditions, all join successfully
- [ ] Host deletes room while players joining — join fails with clear error
- [ ] Room expires after 60min of inactivity — cleanup job removes from DB

Non-Functional:

- [ ] Join flow completes in <1sec (P95 latency)
- [ ] Lobby screen renders correctly on 5+ devices tested
- [ ] No memory leaks after 20 room creates/joins (Chrome DevTools heap snapshot)

Test Evidence:

- [ ] Test report document with pass/fail results for each AC
- [ ] Video recording of 4-device join flow (no issues)
- [ ] Load test: 50 rooms created concurrently (<2sec each)

9.2 Module 2: Core Game Loop

Functional Requirements:

- [ ] Round timer counts down accurately across all devices (tested with stopwatch)
- [ ] Swipe gestures recognized correctly (up = correct, down = skip)
- [ ] Score updates propagate to all devices within 200ms
- [ ] Round transitions smoothly (scoring screen → next round)
- [ ] Game ends after 3 rounds, final scores displayed

Edge Cases Tested:

- [ ] DC-002 (Explainer disconnect mid-round) recovery works — timer pauses, reconnect within 15sec
- [ ] RC-001 (Simultaneous swipe) server correctly arbitrates — first swipe wins
- [ ] RC-003 (Swipe at T=0) server rejects late swipes correctly
- [ ] DI-001 (Score desync) auto-recovery — force sync if >0 point difference
- [ ] DI-002 (Timer drift) auto-correction — client resets to server time if >1.5sec drift

Non-Functional:

- [ ] 3-round game completes on 3 different device combinations (iOS+iOS, iOS+Android, Android+Android)
- [ ] Swipe response time <100ms (feels instant)
- [ ] No memory leaks after 10 consecutive games (DevTools check)
- [ ] Frame rate >30fps during gameplay (no jank)

Test Evidence:

- [ ] Automated E2E test (Playwright) passes for full game flow
- [ ] Race condition test results (RC-001 tested 100 times, 0 double-counts)
- [ ] Timer sync test results (tested with 5sec clock skew, auto-corrects within 1sec)

9.3 Module 3: Word Pack System

Functional Requirements:

- [ ] Default word pack (500 words) available immediately
- [ ] Premium word packs displayed in shop with prices
- [ ] Pack purchase flow works (Stripe payment integration)
- [ ] Purchased pack unlocks for entire room (all players can use)
- [ ] Pack selection persists across reconnect

Edge Cases Tested:

- [ ] RC-005 (Concurrent purchase) handled — second player sees "Already purchased"
- [ ] Pack with 0 words (corrupted data) — graceful error, doesn't crash app
- [ ] Free user tries to access paid pack — blocked with upgrade prompt

Non-Functional:

- [ ] 100-word pack loads in <1sec

- [ ] Shop screen renders correctly on small screens (iPhone SE)
- [ ] Payment flow complies with PCI-DSS (card data never touches our servers)

Test Evidence:

- [ ] Purchase flow tested with Stripe test cards (successful + failed payments)
- [ ] Concurrent purchase test results (no double-charge)

9.4 Module 4: Dual-Screen Sync

Functional Requirements:

- [ ] Host (tablet) and players (phones) show synchronized game state
- [ ] Word cards appear in same order on all devices
- [ ] Score updates visible on all devices simultaneously
- [ ] Timer synchronized across devices (±1sec acceptable)

Edge Cases Tested:

- [ ] DI-003 (Word order desync) auto-corrects — hash mismatch triggers re-sync
- [ ] Phone + tablet tested simultaneously — identical game state confirmed (screenshot comparison)
- [ ] Network latency simulated (200ms delay) — still syncs within 500ms

Non-Functional:

- [ ] Latency <200ms for score update propagation (P95)
- [ ] Tested on iOS + Android + Desktop simultaneously (3+ devices)

Test Evidence:

- [ ] Video recording showing phone + tablet in sync
- [ ] Desync test results (hash mismatch recovery tested 50 times)

9.5 Module 5: PWA Installation

Functional Requirements:

- [ ] Install banner shows after 2 visits + 5min engagement
- [ ] "Add to Home Screen" creates app icon on home screen

- [ ] App opens in standalone mode (no browser chrome)
- [ ] Offline fallback page displays when no connection
- [ ] Service Worker caches assets for offline use

Edge Cases Tested:

- [ ] Install banner criteria validated (tested on iOS Safari + Android Chrome)
- [ ] App works offline for 5min (play cached games, then sync when online)
- [ ] Service Worker updates correctly (new version deployed, old cache cleared)

Non-Functional:

- [ ] App icon displays correctly (no pixelation, proper aspect ratio)
- [ ] Splash screen shows during cold start (<2sec)
- [ ] Push notifications work (test with Firebase Cloud Messaging)

Test Evidence:

- [ ] PWA audit (Lighthouse) score >90
- [ ] Screenshots of install flow on iOS + Android
- [ ] Offline test results (app functional without network)

---

10. Bug Severity & Priority Classification

10.1 Severity Definitions

Severity	Description	Example	Response SLA
P0 — Blocker	Complete product failure, affects all users, no workaround	• Server crash / database down • Cannot create or join rooms • Data loss (scores not saved) • Security vulnerability (RCE, SQLi)	Immediate — drop everything, fix within 2hrs
P1 — Critical	Major feature broken, affects most users, difficult workaround	• Host disconnect doesn't notify players • Score desync >5 points • Timer drift >3sec	Urgent — fix within 24hrs

Severity	Description	Example	Response SLA
		• Game stuck in loading state	
P2 — Major	Feature partially broken, affects some users, workaround exists	• Reconnect takes >10sec • UI glitch (overlapping text) • Audio doesn't play (but game works) • Memory leak (after 20+ games)	High — fix within 1 week
P3 — Minor	Cosmetic issue, edge case, minimal impact	• Animation jank • Typo in UI text • Color contrast slightly off • Feature request (not a bug)	Low — fix in next sprint or backlog

10.2 Priority Decision Matrix

Impact → Frequency ↓	High Impact (Breaks core flow)	Medium Impact (Degrades UX)	Low Impact (Cosmetic)
High Frequency (Affects >50% of users)	P0	P1	P2
Medium Frequency (Affects 10-50% of users)	P1	P2	P3
Low Frequency (Affects <10% of users)	P2	P3	P3

Examples:

- High Impact + High Frequency = P0
  "Cannot join room" (100% of players affected, core flow broken)
- High Impact + Low Frequency = P2
  "Game crashes on iPhone 8 with iOS 14" (affects 2% of users, old device)
- Low Impact + High Frequency = P2
  "Button animation stutters" (affects 80% of users, but game still works)

10.3 Bug Triage Process

Weekly Bug Triage Meeting (Fridays, 30min)

Attendees: QA Lead, PM, Backend Dev, Frontend Dev

Agenda:

1. Review all new bugs from past week (Sentry, beta feedback, QA testing)
2. Assign severity (P0-P3) using decision matrix
3. Assign owner (Dev) and due date
4. Prioritize P0/P1 bugs for immediate fix
5. Move P2/P3 to sprint backlog

Bug Tracking Tool: Linear / Jira

Bug Template:

```
**Title:** [Severity] Short description (e.g., "[P0] Cannot join room on iOS Safari")

**Environment:**
- Device: iPhone 13 Pro
- OS: iOS 16.5
- Browser: Safari 16
- Build: v1.2.3

**Steps to Reproduce:**
1. Open app on iOS Safari
2. Tap "Join Room"
3. Enter code "AB12"
4. Tap "Join"

**Expected Result:**
- Player joins lobby within 500ms

**Actual Result:**
- Loading spinner spins forever
- Console error: "WebSocket connection failed"

**Frequency:** 100% reproducible

**Impact:** High (blocks all iOS Safari users from joining)

**Priority:** P0 (blocks core flow, affects 40% of users on iOS)

**Workaround:** None
```

```
**Attachments:**
- Screenshot of error
- Console logs
- Sentry error ID: #12345
```

10.4 Pre-Launch Bug Bar

Launch Blockers (Must Fix):

- [ ] Zero P0 bugs in production
- [ ] Zero P1 bugs affecting >20% of users
- [ ] All edge cases (DC-001 to DC-006, RC-001 to RC-005) pass

Launch Acceptable (Can Launch With):

- [ ] P2 bugs that affect <10% of users (documented workarounds)
- [ ] P3 cosmetic issues (fix post-launch)

Post-Launch Monitoring:

- [ ] Real-time Sentry alerts for any new P0/P1 bugs
- [ ] On-call rotation (dev team) for critical bug fixes
- [ ] Hotfix process (can deploy within 1hr if P0 detected)

---

11. Appendix: Test Tools & Infrastructure

11.1 Test Automation Stack

Layer	Tool	Purpose	Cost
Unit Testing	Jest (JS) / Pytest (Python)	Test individual functions, classes	Free
E2E Testing	Playwright	Automate user flows across browsers	Free
API Testing	Postman / Insomnia	Test REST APIs, WebSocket handshakes	Free
Load Testing	Artillery.io	Simulate 500+ concurrent WebSocket connections	Free (open source)

Layer	Tool	Purpose	Cost
Cross-Browser Testing	BrowserStack	Test on real iOS/Android devices remotely	$99/month
Performance Monitoring	Lighthouse (Chrome DevTools)	Audit PWA, performance, accessibility	Free
Error Tracking	Sentry	Capture crashes, errors in production	$26/month (Team plan)
Monitoring & Alerts	Grafana + Prometheus	Real-time server metrics, custom dashboards	Free (self-hosted)
Synthetic Monitoring	Checkly	Uptime checks, API health	$79/month

Total Monthly Cost: ~$200 (BrowserStack + Sentry + Checkly)

11.2 Test Data Management

Synthetic Test Data:

- 10 pre-generated room codes for testing: `TEST01` to `TEST10`
- 50 test player names: `TestPlayer001` to `TestPlayer050`
- 3 test word packs: "Animals" (50 words), "Tech" (50 words), "Random" (100 words)

Database Reset Script (for staging environment):

```
-- Clear all test data
DELETE FROM rooms WHERE code LIKE 'TEST%';
DELETE FROM players WHERE name LIKE 'TestPlayer%';
DELETE FROM game_sessions WHERE created_at < NOW() - INTERVAL '24 hours';

-- Re-seed test rooms
INSERT INTO rooms (code, host_id, status) VALUES
  ('TEST01', 'test-host-1', 'lobby'),
  ('TEST02', 'test-host-2', 'lobby'),
  ('TEST03', 'test-host-3', 'in_progress');
```

Run before each test suite:

```
npm run test:reset-db
```

11.3 CI/CD Testing Pipeline

GitHub Actions Workflow (`.github/workflows/test.yml`):

```yaml
name: Test Suite

on:
  push:
    branches: [main, develop]
  pull_request:
    branches: [main]

jobs:
  unit-tests:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v3
      - name: Install dependencies
        run: npm install
      - name: Run unit tests
        run: npm run test:unit
      - name: Upload coverage
        run: npm run test:coverage

  integration-tests:
    runs-on: ubuntu-latest
    services:
      postgres:
        image: postgres:15
        env:
          POSTGRES_PASSWORD: test
    steps:
      - uses: actions/checkout@v3
      - name: Run integration tests
        run: npm run test:integration

  e2e-tests:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v3
      - name: Install Playwright
        run: npx playwright install
      - name: Run E2E tests
        run: npm run test:e2e
      - name: Upload screenshots on failure
        if: failure()
```

```
        uses: actions/upload-artifact@v3
        with:
          name: playwright-screenshots
          path: test-results/

  load-tests:
    runs-on: ubuntu-latest
    if: github.ref == 'refs/heads/main'  # Only on main branch
    steps:
      - uses: actions/checkout@v3
      - name: Run Artillery load test
        run: npm run test:load
      - name: Check performance thresholds
        run: npm run test:load:check
```

Test Execution Frequency:

- Unit tests: Every commit (run in <2min)
- Integration tests: Every PR (run in <5min)
- E2E tests: Every merge to main (run in <10min)
- Load tests: Nightly (run at 2am UTC, <30min)

11.4 Test Environment URLs

Environment	URL	Purpose	Database
Local	`http://localhost:3000`	Dev testing, debugging	SQLite (local)
Dev	`https://dev.partyflow.game`	Feature branch testing, integration tests	PostgreSQL (staging DB)
Staging	`https://staging.partyflow.game`	Pre-production, load testing, beta testing	PostgreSQL (prod replica)
Production	`https://partyflow.game`	Live users, smoke tests only	PostgreSQL (prod)

Access Control:

- Local: No auth required
- Dev: Basic auth (username: `dev`, password: set in env)

- Staging: IP whitelist (QA team, dev team, beta testers)
- Production: Public

11.5 Contact & Escalation

QA Lead Contact:

- Name: [Your Name]
- Email: qa@partyflow.game
- Slack: @qa-lead

Escalation Path for Critical Bugs:

1. **P0 Bug Detected** → Immediately notify in `#critical-bugs` Slack channel
2. **Dev Team** → Acknowledges within 15min
3. **PM** → Notified if fix requires >2hrs (may delay launch)
4. **On-Call Rotation** → Dev on-call responds within 30min (evenings/weekends)

Bug Reporting:

- All bugs logged in Linear: `https://linear.app/partyflow/team/QA`
- Sentry errors auto-create Linear tickets for P0/P1 issues
- Beta testers report via Google Form: `https://forms.gle/partyflow-beta-feedback`

Document Revision History

Version	Date	Author	Changes
1.0	2026-02-11	QA Lead	Initial draft — comprehensive test strategy for MVP

Next Steps:

1. **PM Review** — Share this doc with PM for approval (target: 48hrs)
2. **Dev Team Review** — Present test strategy in sprint planning meeting
3. **Tool Setup** — Configure BrowserStack, Sentry, Artillery (Week 1)
4. **Test Case Writing** — Write detailed test cases for Module 1 (Week 2)

5. **Test Execution** — Begin manual testing as each module completes

---

End of Document