AI Risk Repository Chatbot: Comprehensive Testing & Feedback Plan

# Document Color Coding

■ Black Text: UI/UX User Feedback & Interface Testing

■ Dark Gray Text: Developer Performance & Technical Testing

■ Medium Gray Text: External Tester Performance Evaluation

# Executive Summary

This comprehensive plan integrates three critical evaluation streams for the AI Risk Repository Chatbot: user interface experience, technical performance metrics, and response quality assessment. Each colored section represents a different testing approach, enabling parallel execution while maintaining clear ownership and accountability.

# 1. Unified Testing Framework

## 1.1 Testing Dimensions

**User Interface & Experience:**

* Entry point optimization (direct navigation vs context)
* Interface design patterns (embedded chat vs floating widget)
* Mobile responsiveness and cross-browser compatibility
* User journey from discovery to task completion
* Visual design and information architecture

**Technical Performance:**

* Response latency (P50, P95, P99 percentiles)
* System accuracy and factual correctness
* Database query efficiency and optimization
* Token usage and cost optimization
* Load handling and concurrent user support
* Error rates and recovery mechanisms

**Response Quality & Usability:**

* Answer completeness and relevance
* Language clarity and technical appropriateness
* Citation quality and source reliability
* Query understanding and intent recognition
* Comparative performance vs alternatives

## 1.2 Success Metrics Overview

|  |  |  |  |
| --- | --- | --- | --- |
| Category | Metric | Target | Testing Type |
| User Experience | Task Completion Rate | >80% | UI/UX |
| User Experience | Time to First Interaction | <10s | UI/UX |
| User Experience | Mobile Usability Score | >4/5 | UI/UX |
| Performance | Response Time | <2s P95 | Developer |
| Performance | Accuracy Rate | >95% | Developer |
| Performance | Uptime | >99.9% | Developer |
| Quality | User Satisfaction | >4/5 | Tester |
| Quality | Citation Relevance | >90% | Tester |
| Quality | Query Success Rate | >75% | Tester |

# 2. Integrated Testing Methodology

## 2.1 Phase 1: Setup & Preparation (Week 1)

**UI/UX Preparation:**

* Recruit 10-12 diverse user participants
* Prepare user journey scenarios
* Set up screen recording and analytics tools
* Create task completion checklists

**Developer Setup:**

* Create ground truth dataset (100+ test queries)
* Build automated test suite infrastructure
* Configure performance monitoring tools
* Establish baseline performance metrics

**Tester Preparation:**

* Develop evaluation rubrics and scoring sheets
* Create test query categories and examples
* Brief testers on system capabilities and limitations
* Prepare feedback collection instruments

## 2.2 Phase 2: Testing Execution (Weeks 2-3)

**UI/UX Testing Sessions:**

* 45-minute moderated user sessions
* Think-aloud protocol for interface navigation
* Task-based scenarios (finding information, exploring domains)
* Post-session interviews and surveys
* Heatmap and clickstream analysis

**Developer Performance Testing:**

* Run automated test suite across all query categories
* Execute load testing (10, 50, 100, 200 concurrent users)
* Stress test with edge cases and malformed queries
* Monitor resource usage and optimization opportunities
* Validate citation accuracy and accessibility

**Tester Quality Evaluation:**

* Systematic testing of basic, complex, and edge queries
* Domain-specific knowledge assessment
* Response quality scoring on 5-point rubric
* Comparative analysis with ChatGPT/Claude
* Documentation of issues and improvement suggestions

## 2.3 Phase 3: Analysis & Synthesis (Week 4)

All three testing streams converge for integrated analysis:

* Cross-reference UI frustrations with technical bottlenecks
* Map user expectations to actual system capabilities
* Identify correlation between performance metrics and user satisfaction
* Prioritize improvements based on impact across all dimensions
* Create unified action plan with clear ownership

# 3. Comprehensive Test Categories

## 3.1 Query Complexity Levels

### Basic Queries (All Teams Test)

• "What types of AI risks exist?"

• "How many AI risks are in the repository?"

• "Tell me about privacy risks"

*UI/UX Focus: Ease of input, result presentation*

*Developer Focus: Response time, accuracy*

*Tester Focus: Completeness, clarity*

### Complex Analytical Queries

• "Compare privacy risks to security risks"

• "What are the unintentional post-deployment discrimination risks?"

• "How do different domains intersect?"

*UI/UX Focus: Information organization, navigation*

*Developer Focus: Multi-service coordination, token usage*

*Tester Focus: Analytical depth, coherence*

### Edge Cases & Error Scenarios

• Ambiguous queries: "AI?"

• Out-of-scope requests: "How to prevent risks?"

• Technical stress: Very long queries, special characters

• Language variations: Typos, non-English queries

*UI/UX Focus: Error messaging, graceful degradation*

*Developer Focus: Error handling, system stability*

*Tester Focus: User guidance, helpful alternatives*

# 4. Integrated Data Collection

## 4.1 Quantitative Metrics

|  |  |  |
| --- | --- | --- |
| Testing Type | Metrics Collected | Collection Method |
| UI/UX | Click paths, time on task, abandonment points | Analytics tools, session recordings |
| UI/UX | User satisfaction scores, preference rankings | Surveys, interviews |
| Developer | Response times, error rates, resource usage | Automated monitoring, logs |
| Developer | Accuracy scores, citation validity | Automated validation scripts |
| Tester | Quality ratings, completeness scores | Evaluation rubrics |
| Tester | Comparative rankings, trust metrics | Feedback forms |

## 4.2 Qualitative Insights

**UI/UX Qualitative Data:**

* User journey pain points and delights
* Mental models and expectations
* Feature requests and design preferences

**Developer Observations:**

* Performance bottlenecks and optimization opportunities
* System behavior under stress conditions
* Resource utilization patterns

**Tester Feedback:**

* Response quality issues and patterns
* Trust and credibility factors
* Comparison with user expectations

# 5. Unified Priority Matrix

## 5.1 Issue Classification

Issues are prioritized based on impact across all three testing dimensions:

|  |  |  |  |
| --- | --- | --- | --- |
| Priority | Criteria | Example Issues | Timeline |
| Critical | Affects all dimensions, blocks core functionality | System crashes, major inaccuracies, unusable interface | 48 hours |
| High | Significant impact on 2+ dimensions | Slow responses affecting UX, poor mobile experience | 1 week |
| Medium | Notable impact on 1 dimension | Suboptimal layouts, minor accuracy issues | 2 weeks |
| Low | Minor improvements, nice-to-have | Formatting preferences, additional features | Backlog |

## 5.2 Cross-Functional Impact Analysis

Each identified issue is evaluated for its impact across all testing dimensions:

**Example: "Response takes 5+ seconds for complex queries"**

* UI/UX Impact: Users abandon tasks, poor experience
* Technical Impact: Performance SLA violation, resource strain
* Quality Impact: Perceived as unreliable, reduces trust

→ Classification: HIGH PRIORITY - affects all dimensions

# 6. Integrated Timeline & Resources

## 6.1 Week-by-Week Breakdown

|  |  |  |  |
| --- | --- | --- | --- |
| Week | UI/UX Activities | Developer Activities | Tester Activities |
| Week 1 | Recruit participants, prepare scenarios | Build test suite, create datasets | Develop rubrics, brief testers |
| Week 2 | Conduct user sessions (5) | Run automated tests, load testing | Begin systematic evaluation |
| Week 3 | Complete sessions, analyze data | Stress testing, optimization | Complete evaluations, comparative analysis |
| Week 4 | Synthesize findings | Performance report | Quality assessment report |
| Week 5 | Unified analysis and recommendations | All teams collaborate | All teams collaborate |

## 6.2 Resource Requirements

**UI/UX Testing:**

* 10-12 user participants (45 min each)
* 2 UX researchers/facilitators
* Recording and analysis tools

**Developer Testing:**

* 1-2 developers for test implementation
* CI/CD pipeline access
* Performance monitoring infrastructure

**Tester Evaluation:**

* 6-8 domain experts and researchers
* 2 coordinators for session management
* Evaluation tools and forms

# 7. Integrated Deliverables

## 7.1 Individual Reports

**UI/UX Deliverables:**

* User journey maps with pain points
* Interface design recommendations
* Usability test results and videos

**Developer Deliverables:**

* Performance baseline report
* Load test results and analysis
* Optimization recommendations

**Tester Deliverables:**

* Quality assessment scores
* Comparative analysis report
* Improvement suggestions

## 7.2 Unified Deliverable

Final Comprehensive Report including:

* Executive summary with key findings across all dimensions
* Integrated priority matrix with cross-functional impacts
* Unified improvement roadmap with clear ownership
* Success metrics dashboard combining all KPIs
* Best practices and lessons learned
* Recommendations for continuous improvement

# 8. Definition of Success

## 8.1 Minimum Viable Performance

**• User Experience:** Can complete basic tasks without assistance

**• User Experience:** Mobile interface is functional

**• Technical:** Response time <5s for 95% of queries

**• Technical:** System uptime >99%

**• Quality:** Accuracy >85% for factual queries

**• Quality:** Users rate experience >3/5

## 8.2 Target Performance

**• User Experience:** Intuitive interface requiring no training

**• User Experience:** Seamless mobile experience

**• Technical:** Response time <2s for 95% of queries

**• Technical:** System uptime >99.9%

**• Quality:** Accuracy >95% for all queries

**• Quality:** Users rate experience >4.5/5

# 9. Risk Mitigation Strategy

**• Risk:** Low participant engagement → Mitigation: Recruit 20% extra participants, offer incentives

**• Risk:** Test environment instability → Mitigation: Use staging environment, have rollback plan

**• Risk:** Tester bias → Mitigation: Use diverse tester pool, structured rubrics

**• Risk:** Timeline slippage → Mitigation: Build in buffer time, parallel execution where possible

**• Risk:** Conflicting feedback → Mitigation: Use priority matrix, data-driven decisions

# 10. Immediate Next Steps

**Day 1-2:** Approve this comprehensive plan *[All Teams]*

**Day 3-5:** Begin participant recruitment *[UI/UX Team]*

**Day 3-5:** Set up test infrastructure *[Developer Team]*

**Day 3-5:** Finalize evaluation rubrics *[Tester Team]*

**Day 6-7:** Conduct pilot tests *[All Teams]*

**Week 2:** Begin full testing execution *[All Teams]*

# Questions for Stakeholder Discussion

1. What is the relative priority between UI/UX, performance, and quality?

2. Are there specific user groups or use cases to prioritize?

3. What is the budget allocation across the three testing streams?

4. Should we coordinate with the main airisk.mit.edu site team?

5. What level of performance degradation is acceptable during peak usage?

6. How should we handle conflicting recommendations across teams?

*MIT FutureTech - AI Risk Repository*

*Comprehensive Testing & Feedback Plan | November 2024*