90-Day QA Roadmap

Role: First QA Engineer
Product: AI-powered Chatbot Platform

Objective

Establish a scalable QA foundation that ensures product reliability, AI response quality, performance stability, and fast release cycles without compromising quality.

🗓️ Days 0–30: Foundation & Risk Assessment

🎯 Goal:

Understand the product deeply, identify risks, and establish basic quality structure.

---

1. Product & Technical Deep Dive

- Understand system architecture (frontend, backend, AI model integration)
- Identify:
    - Core business flows
    - AI-dependent features
    - Data sources
    - Deployment pipeline
- Review existing logs, bug reports, customer feedback

Deliverable:

- High-level QA Test Strategy document
- Risk assessment matrix

---

2. Define Quality Standards

Establish:

- Definition of Done (DoD)
- Bug severity & priority matrix
- Release readiness checklist
- Acceptance criteria validation process

Align with:

- Engineering
- Product
- Leadership

---

3. Manual & Exploratory Testing Setup

Create:

- Smoke test checklist
- Critical user flow test cases
- AI prompt validation checklist
- Edge case coverage list

Focus on:

- Core chatbot interactions
- Safety compliance
- Performance bottlenecks

---

4. Basic Automation Framework Setup

API Automation:

- Use Playwright API testing (already implemented)
- Cover critical endpoints

Structure:

- Modular test architecture
- Environment configuration support
- Reporting setup

---

5. CI/CD Integration (Initial)

Integrate into GitHub Actions:

- Run API tests on every PR
- Fail build if critical tests fail
- Publish test report artifacts

This ensures quality gates from Day 30.

🗓️ Days 30–60: Automation & AI Evaluation Framework

🎯 Goal:

Scale automation and introduce AI-specific evaluation.

---

1. Expand Test Coverage

- Increase API coverage to 70–80%
- Add integration tests
- Add regression suite
- Add negative testing

---

2. AI Evaluation Framework

Build Golden Dataset:

- Critical prompts
- Edge case prompts
- Adversarial inputs
- Business-sensitive queries

Automate:

- Multi-run testing
- Semantic similarity scoring

- Hallucination detection rules
- Latency tracking
- Token usage monitoring

Generate:

- AI Quality Score Dashboard

---

3. Performance & Load Testing

Implement:

- Concurrent request testing
- Latency benchmarking
- Stress testing under traffic spikes

Tools:

- Playwright
- k6 or similar lightweight load tool

---

4. Test Data & Environment Strategy

- Create test data management plan
- Separate staging vs production validation
- Mock external dependencies when required

---

5. Shift-Left Quality Process

- Join PR reviews
- Review acceptance criteria before development
- Identify testability gaps early

🗓️ Days 60–90: Optimization, Scaling & Governance

🎯 Goal:

Build long-term scalable quality infrastructure.

---

1. Advanced Automation

- Increase automation coverage >85%
- Parallel execution
- Retry mechanism for flaky AI tests
- Stability monitoring

---

2. Regression & Drift Monitoring

For AI features:

- Version comparison testing
- Drift detection alerts

- Trend analysis dashboard
- Automated nightly golden dataset runs

---

3. Security & Compliance Testing

Add:

- Prompt injection testing
- Input validation checks
- Rate limiting validation
- Role-based access validation

---

4. Release Governance

Establish:

- Automated regression before release
- AI quality score threshold for release approval
- Rollback validation checklist
- Monitoring and alerting integration

---

5. Quality Metrics Dashboard

Track:

- Defect leakage rate

- Automation coverage %
- AI hallucination rate
- Test stability index
- Release quality score

Present monthly quality review to leadership.

Frameworks & Tools I Would Set Up

Area	Tools
API Automation	Playwright
AI Evaluation	Custom prompt runner + similarity scoring
CI/CD	GitHub Actions
Reporting	Playwright HTML + Custom dashboard
Performance	k6
Version Control	GitHub
Defect Tracking	Jira (if available)

Long-Term Vision

Build a scalable AI Quality Engineering practice where:

- AI responses are measurable
- Regression risk is minimized
- Releases are automated and predictable

- Quality metrics are visible to leadership
- AI drift is proactively monitored