🚀 [YOUR NAME]

Senior SDET / Automation Engineer

📧 your.email@example.com | 🔗 GitHub (https://github.com/yourprofile) | 💼 LinkedIn (https://linkedin.com/in/yourprofile)

👤 PROFESSIONAL SUMMARY

✦ **5+ years** building scalable test automation frameworks across **web, API, and mobile** platforms
✦ Specialized in **CI/CD pipeline integration**, reducing deployment cycles by **60%** through strategic quality gates
✦ Proven track record implementing testing strategies for **microservices architectures** with **85%+ code coverage**
✦ Strong advocate for **shift-left testing** practices and **infrastructure-as-code** for test environments

---

🛠️ CORE TECHNICAL SKILLS

🎯 Automation & Frameworks

- Playwright • Cypress • Selenium WebDriver
- REST Assured • Postman • Newman
- Cucumber • SpecFlow • BDD

🧪 Testing Strategy

- E2E • Integration • Contract (Pact)
- Performance (k6, JMeter)
- Visual Regression (Percy, Chromatic)

☸️ DevOps & Infrastructure

- GitHub Actions • Jenkins • GitLab CI
- Docker • Kubernetes
- AWS • GCP • Azure

💻 Languages & Tools

- JavaScript/TypeScript • Python • Java
- Git • Grafana • DataDog
- TestRail • Xray

💡 AUTOMATION & TESTING PHILOSOPHY

🔺 **Test Pyramid in Practice:** Prioritize fast unit tests (70%), strategic integration tests (20%), focused E2E scenarios (10%)

⚡ **CI/CD First:** Every test should run in pipeline; if it doesn't run automatically, it doesn't exist

📊 **Quality Metrics That Matter:** Focus on defect escape rate, mean time to detection, and deployment frequency over vanity metrics

🔧 **Maintainability:** Page object models, reusable components, self-healing locators, clear naming conventions

---

🗂 SELECTED PROJECTS

🎯 Project 1: Multi-Tier Test Automation Framework

❌ **Problem:**
E-commerce platform had **3-hour regression cycles** blocking daily releases; **40% manual test coverage**

☑ **Solution:**

- Built **Playwright-based framework** with parallel execution across **10 workers**, reducing runtime to **25 minutes**
- Implemented **contract testing with Pact** to catch breaking API changes pre-deployment
- Created **reusable component library** cutting test authoring time by **50%**
- Integrated **visual regression testing** catching **15 UI bugs** before production

🔧 **Tech Stack:** `Playwright` • `TypeScript` • `Docker` • `GitHub Actions` • `Allure Reports` • `PostgreSQL`

🔲 **Outcomes:**
✓ **85% automation coverage**
✓ **Zero critical defects** in 6-month period

→ View on GitHub

🎯 Project 2: API Testing & Contract Validation Pipeline

❌ **Problem:**
Microservices team experiencing **frequent integration failures** in staging; **no API test coverage**

☑ **Solution:**

- Designed **contract-first testing approach** with Pact for **8 microservices**
- Built **REST Assured test suite** with **200+ scenarios** covering happy paths, edge cases, error handling
- Implemented **automated schema validation** and backward compatibility checks in PR pipeline
- Created **test data management strategy** using Docker-based test databases

🔧 **Tech Stack:** `REST Assured` • `Pact` • `Java` • `Gradle` • `Jenkins` • `WireMock` • `MongoDB`

🔲 **Outcomes:**
✓ **95% API coverage**
✓ Integration issues caught **3 stages earlier**

→ View on GitHub

🔄 CI/CD & QUALITY GATES EXPERIENCE

🏗 **Pipeline Architecture:** Designed **4-stage quality gates** (unit → integration → E2E → smoke tests) with fail-fast mechanisms

⚡ **Performance Testing:** Integrated **k6 load tests** in CI; automated alerts for response time degradation **>200ms**

🔒 **Security Scanning:** Embedded **SAST/DAST tools** (SonarQube, OWASP ZAP) blocking merges on critical vulnerabilities

🚀 **Deployment Confidence:** Implemented **canary deployment testing** and synthetic monitoring in production

☘️ **Infrastructure:** Maintained **Dockerized test environments**; Kubernetes-based parallel test execution

---

🏆 NOTABLE ACHIEVEMENTS

🚀 Reduced regression test execution time from **4 hours to 18 minutes** through parallel execution strategy

🔲 Achieved **40% reduction in production defects** through implementation of contract testing

🔧 Built **self-service test data generation tool** adopted by **5 engineering teams**

👥 Mentored **3 junior SDETs** on framework design patterns and CI/CD best practices

---

🎓 CERTIFICATIONS & CONTINUOUS LEARNING

✓ **AWS Certified Developer Associate**

✓ **ISTQB Advanced Test Automation Engineer**

🔍 **Currently exploring:** AI-assisted test generation, chaos engineering practices

---

📬 LET'S CONNECT

🔗 **GitHub:** github.com/yourprofile (https://github.com/yourprofile)
💼 **LinkedIn:** linkedin.com/in/yourprofile (https://linkedin.com/in/yourprofile)
✉️ **Email:** your.email@example.com
🌐 **Portfolio Site:** yourportfolio.dev (https://yourportfolio.dev) *(optional)*

📋 QUICK CUSTOMIZATION CHECKLIST

Before distributing your portfolio:

☐ Replace `[YOUR NAME]` with your actual name
☐ Update all contact information (email, GitHub, LinkedIn)
☐ Replace project descriptions with your actual projects
☐ Update all tech stacks to match your experience
☐ Replace placeholder GitHub links with real repository URLs
☐ Quantify all achievements with real metrics
☐ Update certifications section with your credentials
☐ Proofread for typos and formatting consistency
☐ Test all links before sending
☐ Export to PDF and verify formatting

Pro Tip: Keep your GitHub repositories clean with detailed README files and commit history. Recruiters will click those links!