Kotesh Kumar Yelamati

Software Development Engineer in Test

+19089526096 ◇ koteshyelamati45@gmail.com ◇ United States ◇ https://www.linkedin.com/in/koteshyelamati/
◇ https://github.com/koteshyelamati

SUMMARY

Detail-oriented SDET with experience in test automation, REST API validation, and CI/CD using Jenkins, Docker, and Git. Proficient in Python, JavaScript, and Node.js with working knowledge of PyTest, Selenium, and Playwright. Skilled in writing unit/integration tests, building full-stack apps, and supporting Agile teams to deliver high-quality software.

SKILLS

Programming Languages	Python, JavaScript, Java, C, SQL, HTML, CSS
Testing & QA Tools	PyTest, Selenium, Unit Testing, TDD, Postman, JMeter, Playwright (learning)
Web & App Frameworks	Flask, Django, Node.js, React.js, Angular, Express.js
Cloud Platforms	AWS (EC2, S3, Lambda, RDS), Microsoft Azure
CI/CD & DevOps	Git, GitHub Actions, Jenkins, Docker, Kubernetes, AWS CodePipeline, Bash Scripting
API Development & Testing	RESTful APIs, Postman, Swagger, Flask-RESTful, FastAPI
Big Data & Streaming	Spark (PySpark), Kafka, Hadoop (basic), Redshift, Glue, S3 (for ETL)
Monitoring & Automation	CloudWatch, Shell Scripting, Logging Pipelines, Schedulers
BI & Reporting Tools	Power BI, Tableau, Excel, SSRS
Software Practices	Agile/Scrum, TDD, Version Control (Git), Debugging, Code Reviews

WORK EXPERIENCE

Software Development Engineer in Test (SDET) Intern Nov '2021 — Apr '2022
Sree Tech Solutions Hyderabad, India

- Designed and implemented **automated test suites** (unit, regression, smoke) using **PyTest**, **Postman**, and **JavaScriptbased Playwright** scripts to validate both REST APIs and web UI workflows.
- Contributed to **CI/CD pipelines using GitHub Actions**, integrating unit, regression, and smoke tests for continuous validation of cloud-deployed services (**AWS Lambda**).
- Developed test cases for modern **JavaScript applications**, contributing to UI validation and cross-browser testing using **Playwright** and **basic DOM manipulation scripts**.
- Executed **load and performance tests** to identify bottlenecks and optimize response times, supporting productionreadiness of key service endpoints.
- Conducted **accessibility and security testing** on web interfaces using open-source tools and manual validation techniques, helping improve compliance and hardening efforts.
- Collaborated with cross-functional teams in an **Agile/Scrum environment**, contributing to sprint planning, exploratory testing, and quality reviews.
- Investigated test failures and production issues to perform **root cause analysis**, implementing corrective test coverage and documenting outcomes.
- Mentored peers on test automation best practices, SDLC process adherence, and test script optimization for reusability and clarity.

Software QA & Automation Engineering Jun '2018 — Jun '2019
Tata Power Solar Systems Bengaluru, India

- Operated and maintained **automated production systems** (200MW, 100MW, 80MW PV lines) with a strong focus on **process reliability, fault diagnosis**, and continuous performance improvement — skills directly transferable to software system debugging.
- Performed **root cause analysis** and **PLC-level troubleshooting** of complex subsystems (servo drives, pneumatic controls, automation interfaces), demonstrating a **structured problem-solving approach** akin to isolating defects in test environments.
- Identified opportunities for performance tuning and **system-level optimization**, analogous to reducing runtime of automated test scripts or backend load testing.
- Documented issue resolution procedures, preventive maintenance steps, and improvement plans — reflecting experience in writing **clear technical documentation** and test cases.

PROJECTS

CI/CD Test Automation Pipeline – GitHub Actions, Jenkins, Docker, PyTest Link

- Built a Python-based Flask microservice with health check and mock endpoints.
Developed automated test suites using PyTest and tracked coverage with Codecov.
Configured GitHub Actions to run tests on pull requests and Dockerized Jenkins for scalable CI/CD.
Enabled Slack notifications for real-time feedback on build and test outcomes.
Key Skills: Python, Flask, PyTest, GitHub Actions, Jenkins, Docker, CI/CD, Codecov, Slack API

E-Commerce QA Automation Suite – JavaScript & Playwright Link

- Created Playwright test cases for key user flows like product search, filters, cart, and checkout.
  Configured cross-browser automation across Chromium, Firefox, and WebKit.
  Integrated nightly regression testing via CI pipeline and fixed flaky test behaviors.
  Improved overall test stability and reduced manual QA workload.
  Key Skills: JavaScript, Playwright, E2E Testing, Cross-Browser Testing, CI/CD, Regression Testing

Full-Stack Web App QA – React & Django REST API Testing Link

- Developed comprehensive PyTest coverage for Django REST APIs with validation of response codes and payloads.
  Built Jest unit tests for React components to ensure UI integrity.
  Used Postman for scenario-based API validation and integrated GitHub Actions for test automation.
  Delivered 90% test coverage across frontend and backend layers.
  Key Skills: React, Django, PyTest, Playwright, REST APIs, Postman, GitHub Actions, Test Automation

Task Management App – React + MongoDB + TDD Link

- Built a full-stack task manager with reusable React components and a Node.js REST API backend.
  Wrote backend tests using Mocha and Chai to validate all CRUD logic.
  Developed Jest tests to validate component states, interactions, and UI flows.
  Applied TDD principles to guide feature development from the start.
  Key Skills: React, Node.js, MongoDB, Mocha, Chai, Jest, REST API, Test-Driven Development

Performance Testing Framework – API Load Testing with Locust & JMeter Link

- Simulated hundreds of concurrent users to validate backend API performance under heavy load.
  Used Locust and JMeter to capture throughput, error rates, and latency metrics.
  Created visual reports to identify bottlenecks and scalability risks.
  Recommended performance enhancements based on detailed test data.
  Key Skills: Locust, Apache JMeter, Load Testing, API Performance, Metrics Analysis, Scalability Testing

BugTracker360 – Full-Stack Bug Tracking System (React, Node.js, MongoDB) Link

Built a real-time issue tracking app with React frontend and Node.js + MongoDB backend.

- Developed REST APIs for bug reporting, user roles, and project management.
- Implemented unit and API tests using Jest and Mocha for stability.
- Automated test runs with GitHub Actions for continuous integration.

Key Skills: React, Node.js, Express, MongoDB, REST API, Jest, Mocha, GitHub Actions, CI/CD

Lung Cancer Identification System (Academic Project)

- Developed a Lung Cancer Identification System to improve the accuracy of lung cancer identification using the K-Nearest Neighbour (KNN) Algorithm compared with the Logistic Regression Algorithm.
- Improved model precision and integrated advanced pre-processing steps for data handling.

EDUCATIONAL QUALIFICATION

Master in Data Analytics, Indiana Wesleyan University (GPA: 3.5/4.0)	Sep '2023 — Apr '2025
Bachelor in Electronics and Communication Engineering, SSE (GPA: 7.4/10)	Aug '2019 — Aug '2022

CERTIFICATIONS

Introduction to Cloud, IBM Developer Skills Network

Prompt Engineering for Everyone, IBM Developer Skills Network.

R for Data Science, IBM Developer Skills Network

SQL and Relational Databases 101, IBM Developer Skills Network

PUBLICATIONS

Published Paper: Lung Cancer Identification System to Improve the Accuracy Using Novel K Nearest Neighbour in Comparison with Logistic Regression Algorithm IEEE ICECONF 2023 — https://doi.org/10.1109/ICECONF57129.2023.10084340

As part of an IEEE-published research project, I built a machine learning system to detect lung cancer using an optimized **KNearest Neighbors (KNN)** algorithm and compared its accuracy against **Logistic Regression** and other classifiers. I was responsible for **cleaning and preparing the data**, including handling missing values, removing outliers, and standardizing features — laying the groundwork for a reliable and reproducible testing pipeline.

The project was developed using **Python** on **Ubuntu Linux**, with libraries like **scikit-learn**, **NumPy**, and **pandas**, simulating a real-world, test-driven development setup.

I split the data into **training, validation, and test sets**, tuned hyperparameters like “K” using the validation set, and then evaluated model performance using **accuracy, precision, recall, and F1-score** on the test data.

To ensure reliability, I performed **comparative testing** between multiple models (KNN, Decision Tree, Random Forest) and used **confusion matrices** to analyze errors and misclassifications — much like how bugs and edge cases are tracked during software QA cycles.

The experience strengthened my understanding of **test automation**, result validation, and how detailed analysis and iteration can significantly improve **system accuracy and reliability**, which directly mirrors best practices in software testing and quality engineering.