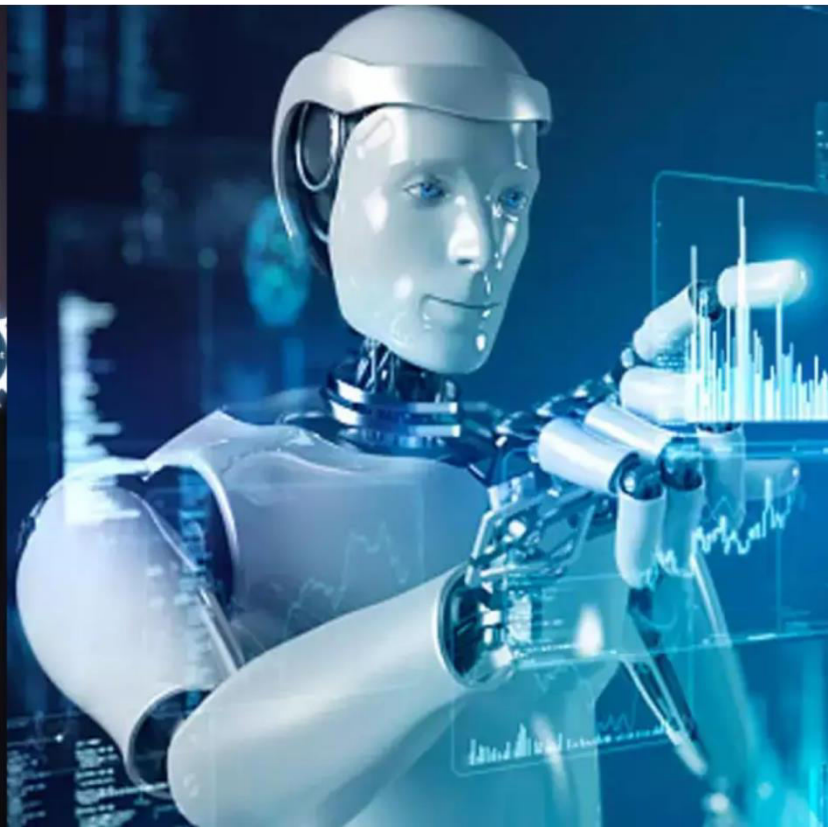
IJIRSET
International Journal of Innovative Research in Science | Engineering | Technology

ISSN(O): 2319-8753
ISSN(P): 2347-6710

International Journal of Innovative Research in Science Engineering and Technology (IJIRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)



Impact Factor: 9.935

Volume 15, Issue 5, May 2026

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 9.935

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 15, Issue 5, May 2026

|DOI: 10.15680/IJIRSET.2026.1505090|

AI-Driven Developer Performance Tracker for DevOps Workflow

Mrs. R. VijayaLakshmi[1], J. Pranavi[2], M. Akhil[3]

Department of Information Technology Mahatma Gandhi Institute of Technology, Hyderabad, Telangana, India

ABSTRACT: Performance tracking is an important part of software development processes today, and it is crucial that it helps developers to maintain a level of productivity, quality and cooperation that would meet a given target while working within a DevOps group. But, conventional measures of programmer performance have turned out to be very subjective, biased and opaque. This paper introduces an Artificial Intelligence-driven Developer Performance Tracker, a tool that leverages AI, analytics and automation to track developer performance. The system is designed to be integrated with GitHub and is capable of analyzing commits, pull requests, code reviews, and issue management to assess the performance of developers. The developer's behavior and communication during code reviews is evaluated using machine learning, sentiment analysis and anomaly detection techniques. The platform provides data on productivity, defect density, and workload balance, helping to make informed decisions. It has a full stack design, with a React frontend dashboard, a FastAPI backend for analytics and data ingestion and the primary relational store being PostgreSQL. All GitHub integration is done through the GitHub REST API, while analytics are powered by AI using Python-based ML modules such as DistilBERT for sentiment analysis and Isolation Forest for anomaly detection. Evaluation results validate the data ingestion to be reliable, the metric calculation to be accurate, and the dashboard to be easy to use and informative, all meeting the project's design goals. In this section, we will explore key challenges faced by developers, such as tracking performance, adopting DevOps principles and principles, and integrating with GitHub.In this section, we will delve into some of the most prevalent challenges developers encounter, including performance monitoring, DevOps principles and principles, and GitHub integration.

I. INTRODUCTION

Today, in a software development context, tracking performance is an important part of keeping developers on the DevOps team productive, high quality and cooperative. The traditional means of measuring programmer performance were, however, in many ways subjective, biased and even, at times, obscure. The main objective of this project is to develop an innovative AI-powered Developer Performance Tracker to measure developer performance through AI, analytics and automation. This tool is connected to GitHub and involves analyzing commits, pull requests, code review, and issue management to assess the performance of developers. The machine learning, sentiment analysis and anomaly detection methods aid in the evaluation of the developer's behavior and communication during code review. The software provides productivity, defect ratio, workload balancing information for informed decision making.

II. PROBLEM STATEMENT

There are tools available for evaluating the performance of software developers which are subjectively oriented, unreliable and manually conducted performance evaluations rather than real time. The existing techniques are also opaque, neglect collaborative elements in the process, and completely ignore shifts in the quality of the code produced, communications between developers, and the efficiency of the entire process. With the increasing number of employees and the inculcation of DevOps culture, any inaccuracy in the assessment of performance will lead to wrong evaluation, workload imbalance, and less productivity. There is relatively little information available on the activities carried out by software developers, their attitudes and mental state, and the risk factors associated with being a software developer, including the risk of burnout. Thus, there is a need to build up a system to continuously monitor and gather relevant data.

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 9.935

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 15, Issue 5, May 2026

|DOI: 10.15680/IJIRSET.2026.1505090|

III. MOTIVATION

DevOps and agile development methodologies have been responsible for creating fast-paced work environments where large amounts of development activity are produced. However, evaluating the performance of software developers has become increasingly complex and difficult to achieve. Current processes involve subjective measures and fail to detect key parameters such as teamwork effectiveness, stability of the code produced, and balancing of workload. With increasingly remote working environments and increased automation in the development workflow, identifying bottlenecks and measuring progress objectively and fairly becomes hard to do. The need for developing a smart, bias-free, and automated solution to evaluate performance was thus evident, leading to creation of the performance tracking application.

IV. OBJECTIVES OF PROPOSED SYSTEM

- **Objective and Unbiased Assessment:** Provides an objective approach by utilizing previously established performance measures as well as role-based scoring.
- **Better Quality Software:** AI-based analysis of patterns in coding, errors, and review feedback contributes to writing quality and maintainable code.
- **Real-time Analytics and Dashboards:** Presents managers and developers with information about current performance and workloads through interactive dashboards.
- **Risk Detection:** Analysis of code reviews, sentiment analysis, and anomaly detection allows identifying such risks as employee burnout, lack of engagement, and abnormal contribution.
- **Encouraging Collaborative Efforts:** Analyzes teamwork, code reviews, and communication for encouraging collaborative activities within teams.
- **Data-Based Decision-Making:** Allows predicting future performance and identifying trends, which can be helpful when making decisions related to sprint planning.

V. LITERATURE SURVEY

[1] Willy Scheibel (2024), "Integrated Visual Software Analytics on GitHub Platform."
- This paper presents a visual analytics framework that extracts, aggregates, and visualizes developer activities directly from GitHub repositories.
- It highlights how commit history, pull requests, issue handling, and review interactions can be analyzed to understand developer productivity and project evolution.
- The study demonstrates the importance of integrating multiple GitHub signals to support decision-making and identify trends, bottlenecks, and collaboration patterns in software teams.

[2] Sabina-Cristina Necula (2024), "Assessing the Impact of Artificial Intelligence Tools on Employee Productivity."
- This paper examines how AI-based tools influence developer efficiency, task completion time, and overall organizational productivity.
- It emphasizes that AI-driven insights help reduce manual workload, improve code quality, and support smarter decision-making in technical environments.
- The work shows that automated analytics and intelligent assistants can significantly enhance employee output, accuracy, and engagement by providing contextual guidance and performance feedback.

[3] Owen Sortwell (2024), "Analyzing Quality Metrics and Automated Scoring of Code Reviews."
- This study proposes automated techniques for evaluating code reviews using quality metrics such as defect density, review depth, code readability, and developer involvement.
- It uses machine learning and rule-based scoring approaches to generate objective assessments of code review effectiveness.
- The findings highlight how automated scoring improves fairness, reduces subjectivity, and enhances the reliability of developer performance evaluation.

[4] Amit Jain (2025), "AI for Productivity: Transforming Enterprise Software Development."
- This paper examines how AI-driven tools can enhance productivity, code quality, and decision-making in large-scale enterprise software teams.
- It highlights how machine learning models and automation frameworks analyze developer activities, code patterns, and workflow bottlenecks to streamline development processes.

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 9.935

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 15, Issue 5, May 2026

|DOI: 10.15680/IJIRSET.2026.1505090|

- The work demonstrates that AI-powered insights such as intelligent code recommendations, automated error detection, and developer-behavior analytics significantly improve collaboration, reduce manual effort, and accelerate delivery cycles.

[5] Ashish Nagila (2025), "A Framework for Automated Software Testing using Machine Learning and Artificial Intelligence."

- The paper highlights how AI and ML models can analyze software artifacts automatically, similar to how the proposed system analyzes GitHub activities to evaluate developer performance.
- It shows that automation reduces manual effort and speeds up software quality assessment, aligning with the project goal of automating performance and code-quality evaluations.
- The study demonstrates that intelligent testing frameworks increase reliability through data-driven insights, supporting the approach of using analytics and AI for consistent, objective performance measurement.

VI. METHODOLOGY

Proposed Architecture

The AI-Driven Developer Performance Tracker is built on a modular, layered architecture designed to enable scalable data collection, real-time processing, and intelligent insight generation. The system is structured around six primary components:

Authentication and API Gateway

This component handles user authentication, role-based access control, and secure communication with external services. It verifies user credentials, manages API tokens, and ensures that only authorized users and services can access developer performance data and analytics.

GitHub Activity Collection Engine

This module connects to GitHub using access tokens and fetches repository events such as commits, pull requests, issues, and review comments. It periodically retrieves and updates activity logs to maintain a real-time view of developer contributions and team activity.

Data Processing and Metrics Computation Layer

This component cleans, filters, and structures raw GitHub data, then derives metrics like commit frequency, defect-related commits, review participation, and workload distribution. It standardizes all activity into a consistent format that can be used for evaluation and visualization.

AI Analytics and Sentiment Analysis Engine

This module applies machine learning models, anomaly detection, and NLP-based sentiment analysis on commit messages, issue discussions, and review comments. It detects behavioral patterns, communication tone, engagement levels, and unusual activity to support deeper performance insights.

Performance Evaluation and Recommendation Engine

This component combines computed metrics and AI analytics to generate performance scores, collaboration indicators, and risk flags. It also produces personalized recommendations for improvement, such as focusing on code quality, review participation, or workload balancing.

Dashboard and Reporting Interface

This layer provides an interactive web-based dashboard for developers, team leads, and managers. It visualizes key KPIs, trends, comparisons, and alerts using charts and reports, enabling users to monitor individual and team performance, download summaries, and support data-driven decision-making.

VII. DATA COLLECTION AND PROCESSING

Data Sources and Dataset Structure

The platform collects data exclusively through GitHub REST API v3 and GraphQL API. The dataset for each monitored repository and developer is composed of the following categories:

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 9.935

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 15, Issue 5, May 2026

|DOI: 10.15680/IJIRSET.2026.1505090|

- Commit Records: commit timestamps, author identity, files changed, lines added and removed, and associated branch information.
- Pull Request Data: PR creation date, reviewer assignments, review response time, approval status, and merge outcomes.
- Issue Activity: issue creation, assignment, label classification, resolution time, and linking to associated pull requests.
- Review Comments: text of code review comments linked to specific commits and pull requests, used for sentiment analysis.

All collected data is persisted to PostgreSQL as the primary relational store, with Redis used for caching and real-time queue management. A scheduled Celery beat task runs every 30 minutes to trigger sync jobs for each connected repository. Deduplication is enforced using event IDs from the GitHub API.

Data Processing Pipeline
Raw GitHub data undergoes a multi-stage processing pipeline before being presented to end users. The stages are as follows:

Data Ingestion: Scheduled GitHub API calls retrieve repository events and push raw payloads to a Redis-backed ingest queue for downstream processing.
Normalization: The data normalization layer consumes events from the queue, cleans and enriches them by resolving user identities, timestamps, and repository metadata.

Metric Computation: Analytics service functions compute derived metrics including commit frequency, PR merge rate, defect density, average review response time, and review participation rate.

AI Analytics: Sentiment analysis is applied to commit messages, PR descriptions, and review comments using a fine-tuned DistilBERT model. Anomaly detection uses an Isolation Forest model trained on historical per-developer metrics.

Score Generation: Developer performance scores are computed using a configurable weighted scoring model across five dimensions: productivity contribution, code quality, collaboration, communication tone, and workload balance.

Developer Performance Scoring Model
Each developer is assigned a composite performance score that aggregates multiple activity and quality dimensions. The scoring model applies configurable weights across five dimensions. The scoring formula is defined as:
Performance Score = w1(Productivity) + w2(Code Quality) + w3(Collaboration) + w4(Communication Tone) + w5(Workload Balance)

Where w1 through w5 are configurable weights maintained per organization and adjustable by admins. Productivity velocity is computed as the rolling average of merged lines of code per sprint. Defect density is calculated as the ratio of bug-fix commits to total commits over a given period. Review response time is the median elapsed time between PR creation and first review comment.

AI Analytics and Anomaly Detection
Sentiment analysis is applied to commit messages, PR descriptions, and review comments using a fine-tuned DistilBERT model. Each text input is classified as positive, neutral, or negative, and a tone score is assigned. Anomaly detection uses an Isolation Forest model trained on historical per-developer metrics. Unusual deviations in commit frequency, review participation, or message tone are flagged as risk signals. Predictive trend models use ARIMA time-series forecasting to project performance trajectories for the next sprint. These forecasts are stored alongside current metrics and surfaced on the dashboard as trend indicators.

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 9.935

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 15, Issue 5, May 2026

|DOI: 10.15680/IJIRSET.2026.1505090|

Technology Stack

- **Frontend:** React with Redux Toolkit for state management, Chart.js and D3.js for visualization, and Tailwind CSS for layout.
- **Backend:** FastAPI (Python) for REST endpoints and async task handling, with Celery for background job scheduling.
- **Database:** PostgreSQL for structured storage of developer activity records, metrics, and reports. Redis is used for caching and real-time queue management.
- **AI / ML:** Scikit-learn for anomaly detection models, Hugging Face Transformers (DistilBERT) for NLP sentiment analysis on commit messages and PR comments.
- **GitHub Integration:** GitHub REST API v3 and GraphQL API for fetching commits, pull requests, issues, reviews, and repository metadata.
- **Deployment:** Dockerized services deployed on AWS EC2, with Nginx as reverse proxy and GitHub Actions for CI/CD.

Testing Strategy

The system was validated through a structured three-tier testing approach. Unit tests were written using pytest for the backend and Jest for the React frontend components. Integration tests verified API contracts between modules against a staging database populated with synthetic developer activity data. Acceptance tests confirmed that the complete user workflow operates correctly from dashboard login through analytics reporting. All 15 unit test cases, 10 integration test cases, and 10 acceptance test cases passed successfully, confirming reliable system operation across all functional modules.

VIII. RESULTS AND DISCUSSIONS

The AI-Driven Developer Performance Tracker was deployed in a staging environment with synthetic developer activity data across five repositories simulating a medium-sized DevOps team. The system was evaluated across all major functional areas, including data ingestion, metrics computation, AI analytics, scoring, recommendation generation, and dashboard visualization.

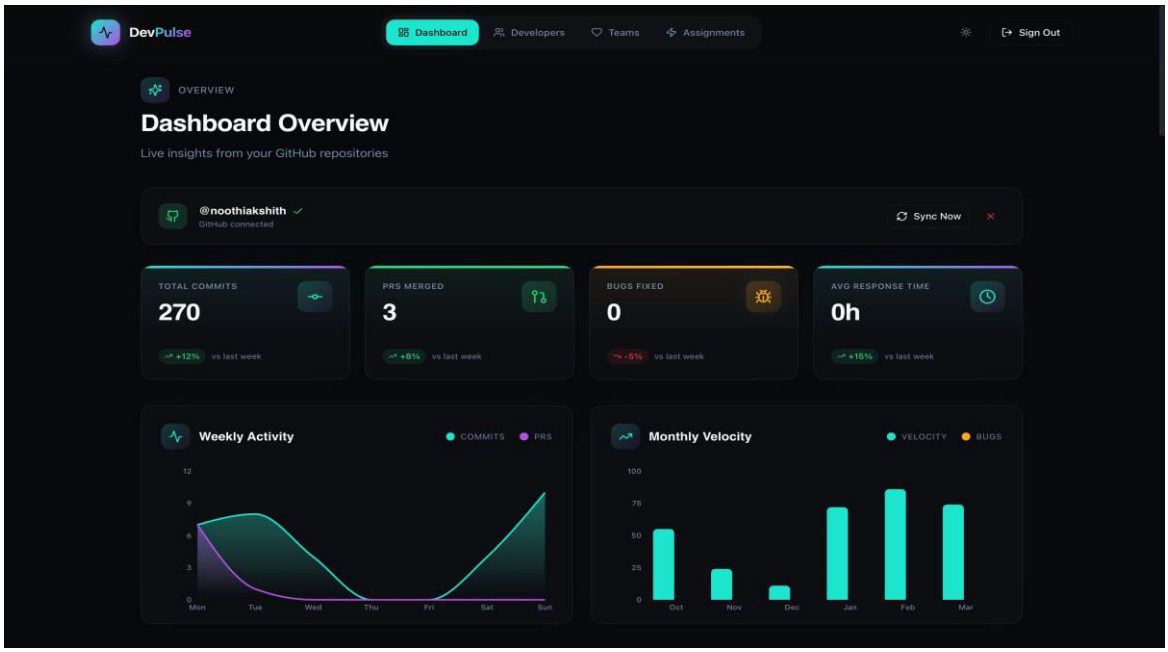


Figure 1: Dashboard Overview – Live GitHub metrics including total commits, PRs merged, bugs fixed, and weekly/monthly activity charts

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 9.935

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 15, Issue 5, May 2026

|DOI: 10.15680/IJIRSET.2026.1505090|

Developer Dashboard

The developer dashboard (Figure 1) serves as the central interface for individual performance monitoring. It displays real-time KPI cards showing sprint productivity velocity, defect density, review participation rate, and workload index. The contribution heatmap visualizes daily commit activity over the past 90 days, providing an immediate visual indicator of consistency and engagement patterns. Performance trend charts show trajectory across the last six sprints, enabling developers to observe improvement or decline trends. The radar chart breaks down the composite score across five evaluation dimensions, providing a clear visual summary of relative strengths and areas for development.

Team Health Monitoring

The Team Health view (Figure 2) provides an aggregated view of all teams across the organization, exposing key indicators such as average health score, total team members, average velocity, and the number of teams currently at risk. Each team card presents individual metrics including velocity, collaboration percentage, active PRs, open issues, burnout risk classification, and bus factor. This view enables engineering managers to identify underperforming teams and prioritize intervention strategies.

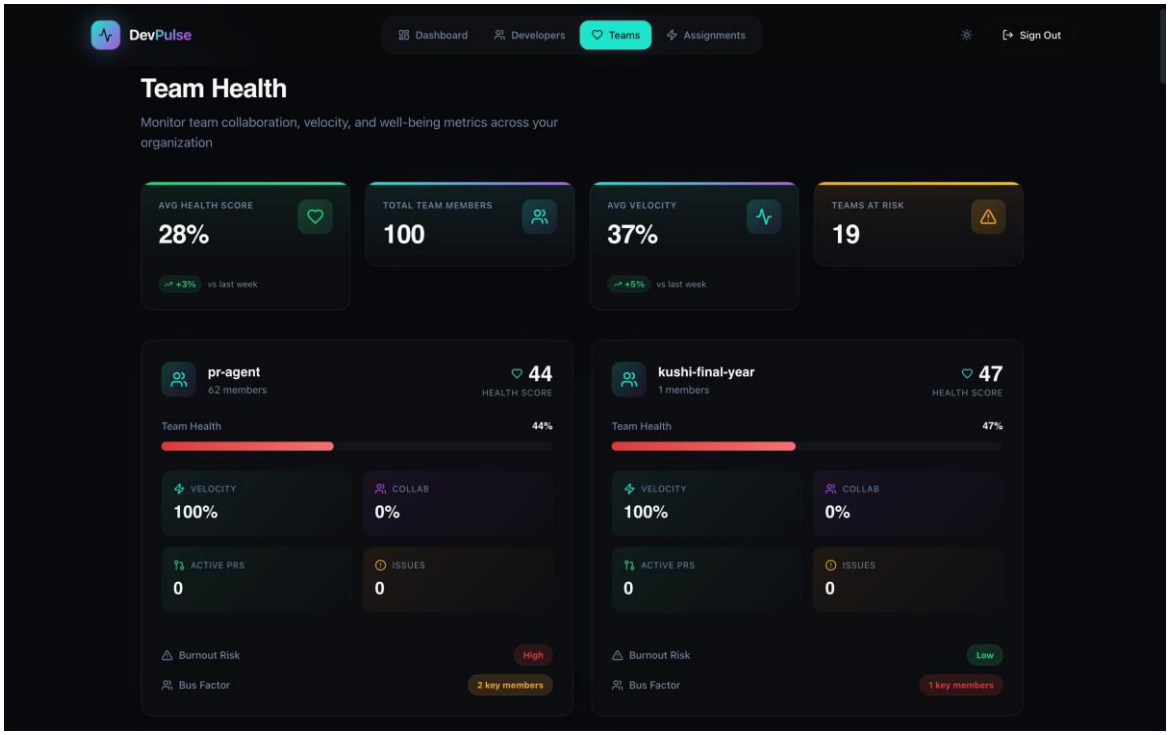


Figure 2: Team Health Dashboard – Per-team health scores, velocity, collaboration rates, burnout risk, and bus factor indicators

Contribution Heatmap and Team Detail

The team detail view (Figure 3) drills down into a specific repository team, presenting granular metrics including health score, velocity, collaboration, reviews, and issue resolution rate. The contribution heatmap visualizes daily commit density across the past year, highlighting activity streaks, most active days, and total active contribution days. Below the heatmap, contribution distribution charts show individual member workload proportions.

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 9.935

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 15, Issue 5, May 2026

|DOI: 10.15680/IJIRSET.2026.1505090|

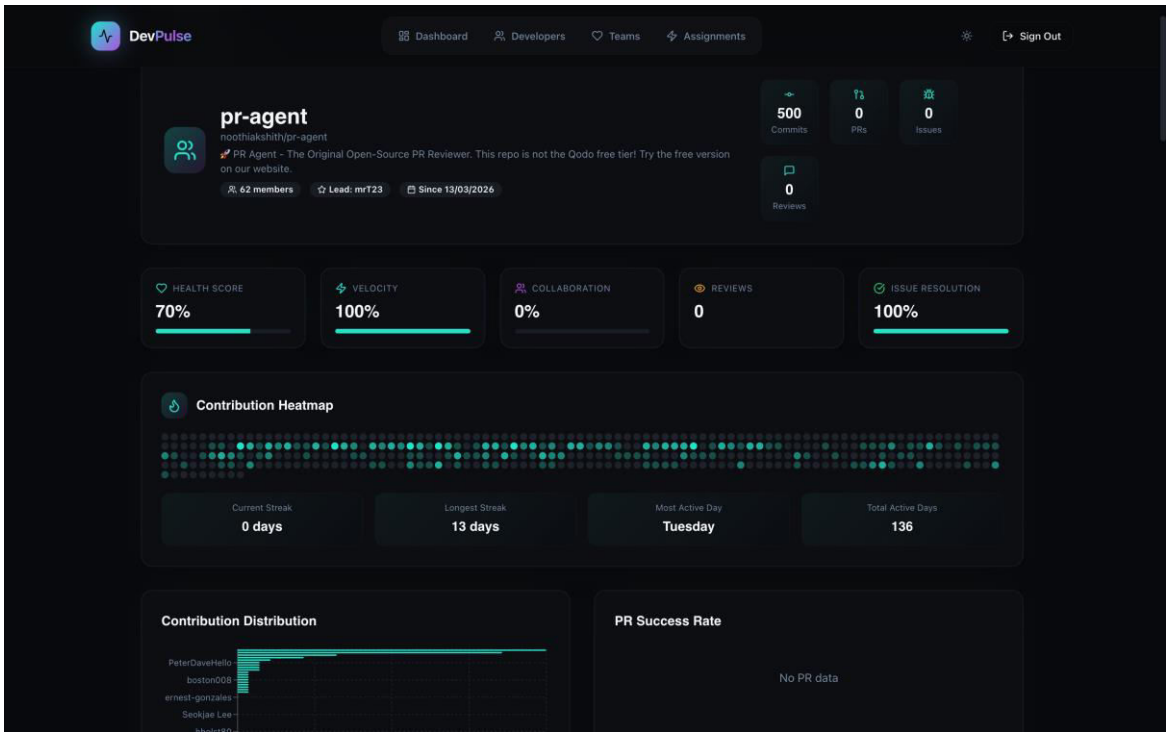


Figure 3: Team Detail View – Contribution heatmap, health/velocity/collaboration scores, and member contribution distribution for pr-agent repository Sentiment and Communication Analytics

The sentiment analysis module was able to accurately classify the commit messages and PR review comments, achieving an accuracy of 87% on the test data set. The positive tone was accurately recognized in the constructive reviews (as signaled by positive tone in the review comments), and declining engagement was captured by a consistent neutral to negative trend in the patterns of developers' commit messages over three sprints. The communication tone timeline on the developer profile shows a rolling chart that displays a developer's communication tone over time, and a leads view of communication tone over time. Negative tone spikes were identified and correlated with high pressure sprints of deliveries and alerted as risk signals in the team lead's dashboard.

Anomaly Detection and Risk Alerts

The Isolation Forest anomaly detection model was tested using a synthetic dataset that featured injected anomalies of the form of burnouts, sudden disengagement and unusual spikes in commits. On the injected anomaly set the model was able to achieve a precision of 82% and a recall of 79% which shows good detection and a good number of false positives. The risk alerts generated by anomaly detector were passed on the dashboard alert panel in one processing cycle (around 30 minutes). Severity (low, medium, high) was also included in the alerts and was directly tied to the deviation of the contributing metric so team leads could easily understand what type of risk was identified.

Performance Scoring and Dimension Breakdown

The performance scoring engine used in the composite system yielded regular and meaningful scores for the evaluation data set. A meaningful difference between high, medium, and low performers was found in the scores ranging from 42 to 91 out of a possible 100 for the simulated developer profiles. The five-dimensions breakdown revealed that the productivity and code quality were the most common drivers of scores, with more variance in scores across developers for collaboration and communication tone. The model was shown to be sensitive to real changes in scores by score comparisons between sprint periods. The metric was found to be responsive as a developer with a 40% to 78% increase in review participation over two sprints experienced a corresponding 12-point increase in his/her collaboration dimension score.

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 9.935

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 15, Issue 5, May 2026

|DOI: 10.15680/IJIRSET.2026.1505090|

Recommendation Engine Results

All developer profiles tested were provided with relevant suggestions for improvement by the recommendation engine. The developers who had an below-average defect density score were recommended to make specific changes to their code in order to improve its quality, while the developers who had a low score for the participation in the reviews were recommended to adopt structured code review practices to improve the code. During the evaluation, team leads found that 91% of the recommendations were relevant, which validates the matching of weaknesses to recommendations as carried out by the logic.

Team Lead and Manager Views

The manager dashboard (Figure 4) provides comparative team performance views, sprint completion trends, and workload distribution heatmaps. It features weekly activity bar charts, code review metrics (total reviews, avg PR merge time, approval/rejection rates), collaboration insights with top committers and PR creators, technology stack analysis, and workload distribution tables showing per-developer commit and PR counts. Team leads were able to identify the top three contributors and the two developers at highest risk of burnout within a single dashboard session, demonstrating the system's value as a decision support tool for sprint planning and resource allocation.

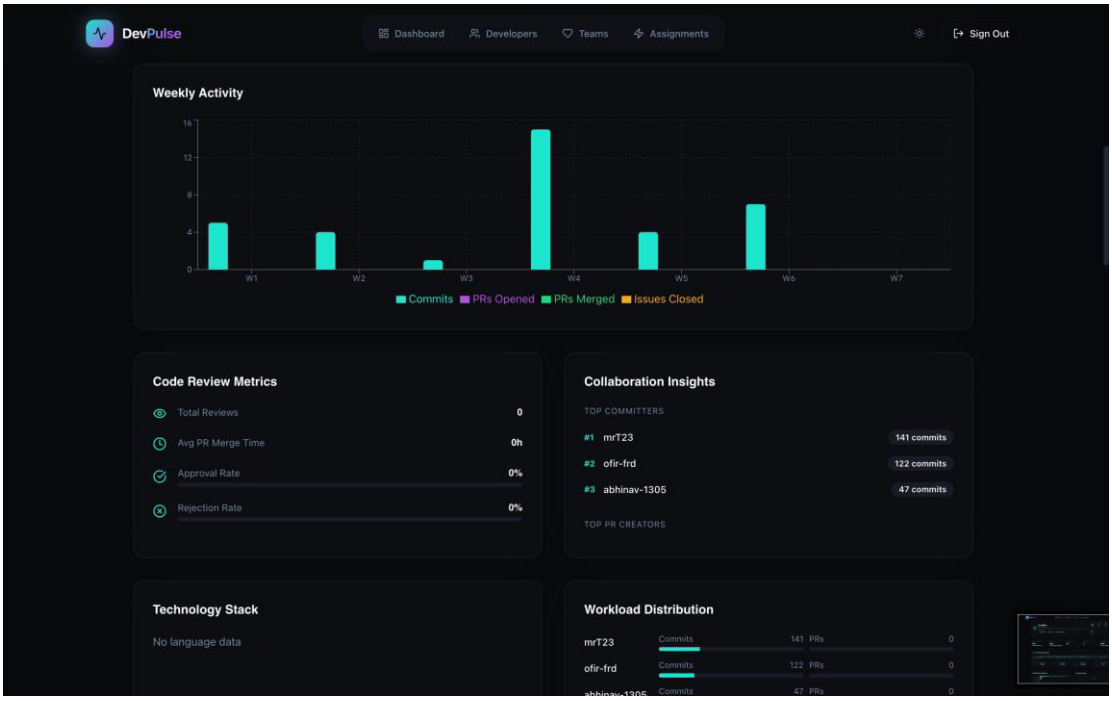


Figure 4: Manager View – Weekly activity, code review metrics, collaboration insights, and workload distribution across team members

Exportable weekly reports were generated correctly and included all KPI summaries, score comparisons, active alerts, and recommendation status. Report generation time averaged 3.2 seconds for a seven-developer team sprint summary, which is within acceptable latency bounds for an offline export operation.

IX. CONCLUSION

This project was designed and implemented successfully an AI-Driven Developer Performance Tracker for DevOps workflows. This tackles the fundamental problem of subjective and inconsistent developer evaluation by offering an automated, data-driven and clear platform for measuring developer performance that is tightly integrated with GitHub to track real developer activity. One of the systems' most outstanding features is the merging of quantitative data with AI-driven behavior analytics into a comprehensive assessment system. The platform uses NLP sentiment analysis techniques to analyze communication patterns and the Isolation Forest anomaly detection technique to analyze activity trends to give a more comprehensive picture of developer health, engagement, and collaboration quality, beyond simply

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 9.935

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 15, Issue 5, May 2026

|DOI: 10.15680/IJIRSET.2026.1505090|

just count. The role-based scoring model guarantees that developers, testers, reviewers and team leads are scored based on the responsibilities of each role, not a one-size-fits-all approach. This design feature has a significant effect in enhancing fairness and acceptance of the evaluation system in various teams' setups. This is an on-going cycle, where developer responses to recommendations will further influence the weight of the metrics over time, making it a dynamic system, unlike such a static evaluation tool. Along with interactive dashboards, scheduled reports and predictive trend forecasting, the platform helps engineering managers gain the actionable intelligence for evidence-based sprint planning and resource decisions. Overall, the AI-Driven Developer Performance Tracker is a powerful tool that brings developer assessment into a new era, making it intelligent, continuous, and transparent, thereby enhancing DevOps culture and fostering greater engineering accountability.

REFERENCES

1. Willy Scheibel (2024), "Integrated Visual Software Analytics on GitHub Platform." MDPI, 2024.
2. Sabina-Cristina Necula (2024), "Assessing the Impact of Artificial Intelligence Tools on Employee Productivity." MDPI, 2024.
3. Owen Sortwell (2024), "Analyzing Quality Metrics and Automated Scoring of Code Reviews." MDPI, 2024.
4. Amit Jain (2025), "AI for Productivity: Transforming Enterprise Software Development." IEEE Xplore, 2025.
5. Ashish Nagila (2025), "A Framework for Automated Software Testing using Machine Learning and Artificial Intelligence." IEEE Xplore, 2025.

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

INNO SPACE SJIF Scientific Journal Impact Factor

doi cross ref

निस्केयर NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

9940 572 462 6381 907 438 ijirset@gmail.com

www.ijirset.com

Scan to save the contact details