iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 12 **Issue:** V **Month of publication:** May 2024

DOI: https://doi.org/10.22214/ijraset.2024.62883

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue V May 2024- Available at www.ijraset.com

GitHub Profile Rating and Analysis Using Machine Learning

Jayesh Punjabi[1], Neeraj Shilwant[2], Vinit Ghadge[3], Shubam Sonawane[4], Dr. Bhagyashree Tingare[5]
[1, 2, 3, 4]*UG Scholar,* [5]*Assistant Professor, Department of AI and Data Science, D.Y. Patil College of Engineering, Pune*

Abstract: ***Assessing developer engagement and impact in the changing world of software development is an important task for employers, partners, and the developers themselves. GitHub, the leading code hosting and collaborative development platform, provides rich data for such testing. This article introduces GitHub Profile Analyzer, a new tool that uses artificial intelligence (AI) and data science to analyse and evaluate GitHub profiles. The system uses a gradient boosting regressor model to evaluate profiles based on a variety of metrics, including user activity, data retention, and engagement history, resulting in a total score between 0 and 5. Improve the hiring process and identify developers, their strengths and areas for improvement. This article covers the GitHub API documentation process, predefined documentation, model development, model training and evaluation, and model deployment using the Flask framework. Preliminary results show that the tool is accurate and reliable in assessing developer knowledge and highlight its implications for modern software development practices.***
Keywords: Gradient Boosting Regressor, GitHub, Repository, GitHub API, Flask framework.

I. INTRODUCTION

In the dynamic and collaborative world of software development, the ability to evaluate and understand developer engagement is critical. GitHub is one of the most popular platforms for hosting and managing code repositories and is an important resource for developers worldwide. As of 2024, GitHub has over 100 million repositories and supports millions of users, providing valuable personal and collaborative information. This information is invaluable to many stakeholders, including employers looking to hire technical engineers, project managers looking for partners, and business developers looking to evaluate their skills. The GitHub Profile Analyzer project was created to meet the need for a powerful tool that can analyse and evaluate GitHub profiles. The tool uses the power of artificial intelligence (AI) and data science to provide a comprehensive assessment of developers based on their GitHub projects. Leveraging a gradient boosting regressor model, the tool evaluates profiles based on a variety of metrics such as user activity, repository content, and service history, resulting in a rating from 0 to 5. Artificial intelligence (AI) has revolutionized many fields. Machines that perform tasks that normally require human expertise. In this project, artificial intelligence, especially from machine learning, plays an important role in analysing large amounts of data and making informed predictions. Data science, on the other hand, combines information logging, statistical analysis, and machine learning to gain insights from complex data. Together, these technologies form the backbone of the GitHub Profile Analyzer and enable accurate and effective evaluation of the developer profile.

The aim of this project is not only to help employers make informed hiring decisions, but also to support developers in understanding their strengths and areas for development. Additionally, open-source project managers can benefit from this tool by identifying potential candidates that can meet their needs. The broader GitHub community will also benefit, as the tool promotes transparency in coding practices and a culture of continuous improvement. This article describes the development and implementation of the GitHub Profile Analysis Tool, showing all phases of the project, from data collection and preprocessing to design, evaluation and export. Providing an overview of the process and workflow, this article focuses on the effectiveness and potential impact of the GitHub Profile Analyzer in the software development world. The facts support the importance of this study. According to GitHub's own statistics, user activity on the platform has increased significantly over the past few years; Millions of pull requests, issues, and commits are created every day. These collaborations represent a wealth of information that, when analysed correctly, can be beneficial to productivity and collaboration. GitHub Profile Analyzer leverages this capability and provides a powerful tool for today's developers.

©IJRASET: All Rights are Reserved

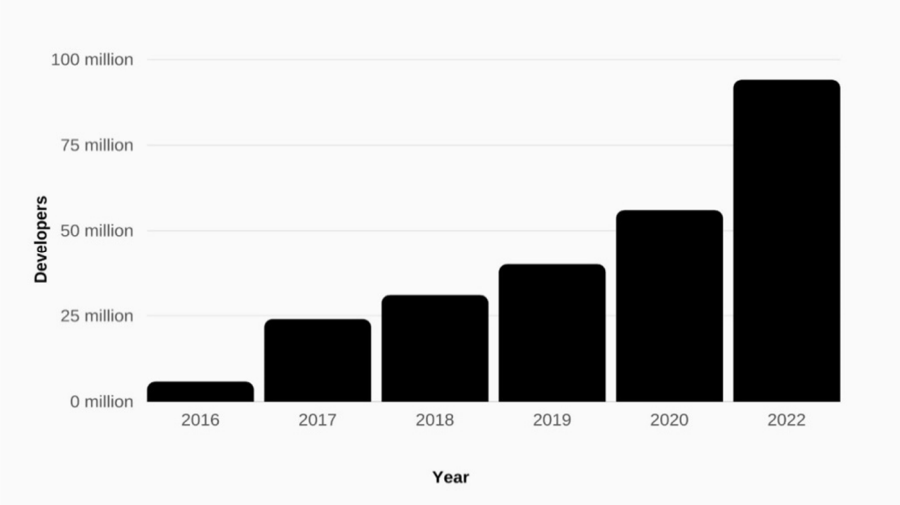


Figure : GitHub's number of developers

II. EXISTING SYSTEM

In the field of GitHub analytics, many systems have been developed to measure and analyse user data and repositories. These systems use a variety of data points provided by GitHub's comprehensive API to gain insight into the developer's activities, contributions, and overall impact. Understand these existing frameworks, their strengths and limitations, provide key points for development and deployment in GitHub Profile Analyzer.

1) *GitHub's Built-in Insights and Statistics:* The platform's engagement provides a snapshot of historical users, including engagement, pull requests, and other important factors. Additionally, GitHub provides a traffic analysis that provides information on the number of views and clones a repository has received. This feature helps data warehouse owners understand the scope and breadth of their operations. Another useful feature is the dependency graph, which analyses the repository's dependencies to help maintainers better manage and secure their work. These built-in tools integrate into GitHub, making data quickly accessible and providing necessary metrics for repositories and user management. However, this tool is only used for simple analysis and cannot perform quality or comprehensive evaluation of user data, which is important for in-depth evaluation of developer influence and involvement.
2) *Gitalytics:* Gitalytics is a third-party tool designed to provide in-depth information about GitHub repositories and user activity. It focuses on metrics such as failure frequency, pull request efficiency, and issue resolution time. Gitalytics provides more detailed information than GitHub's built-in tools, especially when it comes to product metrics, and is useful for team evaluation and project control. Despite its benefits, Gitalytics is primarily geared towards group and site analysis rather than individual user ratings. In addition, Gitalytics' premium features often require paid subscriptions; This may limit access to individual developers or small groups who could benefit from profile analysis but may not be able to source expensive equipment.
3) *CodeClimate:* CodeClimate integrates with GitHub to provide quality code analysis and engineering metrics. It addresses security controls, metrics, and code complexity, provides insights to improve the codebase, and helps teams maintain high standards forever through collaboration. rather than the overall quality of user activity and engagement. The tool does not provide a comprehensive evaluation process necessary to evaluate a developer's overall impact and potential on GitHub.
4) *GitPrime (now Pluralsight Flow):* GitPrime, now part of Pluralsight Flow, provides detailed analysis of engineering results by checking GitHub documentation. Indicates areas for improvement, providing insight into individual and team performance. This makes it a great tool for managers to understand and improve team performance. Although GitPrime offers comprehensive productivity analysis, it focuses more on team dynamics and productivity metrics. The high registration fees associated with GitPrime can also be a deterrent to smaller teams or developers who would benefit from detailed documentation but cannot afford the tools.
5) *Octoboard:* Octoboard provides reporting and dashboard for GitHub metrics. It consolidates data into easy-to-view reports with a user-friendly interface and customizable dashboards. Automatic reporting in Octoboard saves time for users who need regular updates and see details of their work. But Octoboard is also limited compared to dedicated analysis tools. It does not provide a depth index or profile index, which is important for detailed analysis of a user's GitHub profile and future potential.

6) *Limitations of Existing Systems:* Existing systems identified by GitHub, although useful, present some limitations:

a) *No Profile Evaluation:* Most The system focuses on a specific aspect, such as good numbers, productivity, or traffic analysis. They do not provide an exact metric of a user's GitHub profile; this can provide a more comprehensive measure of developer influence and involvement.

b) *Limited Predictive Ability:* Most of these tools do not use advanced learning models to predict future outcomes or availability based on previous data. Predictive capabilities are crucial to understanding and predicting a developer's future performance.

c) *Usability and Cost:* Advanced features in third-party tools are often located behind paywalls, making these features difficult to use for little-to-no developers or teams. For those who can make the most of a detailed analysis, high registration fees can be a significant problem.

d) *Integration and Usability:* While some tools are uploaded to GitHub, others require additional configuration and may not fit seamlessly into the developer's work. Ease of integration and usability are important to ensure that developers can use these tools effectively without significantly impacting their workflows.

III. PROPOSED METHODOLOGY

The process begins with the user entering their GitHub username into the web interface to initiate the data recovery process. This username is sent to the GitHub API via POST request; This API retrieves information about the user's public repository, including commit history and specific details of Users such as accountants and corporate affiliates. The system manages the API price limit quite well, making the data consistent and useful.

After the object is returned, it goes through a pre-emptive period where the JSON response from the API is analysed and cleaned properly and accurately. These steps include filtering irrelevant data, resolving inconsistencies, and modelling specific workflows to facilitate integration. The preliminary data is fed into a machine learning algorithm, such as a random forest regressor or gradient boosting regressor, which analyses the data to predict the numbers based on various users and characteristics in the storage location.

Deadline involves generating a score that represents the calculation of the user's maximum storage space. The results are presented in an intuitive way, including visual representations such as graphs or charts that show the hierarchy of activity for different storage areas. This process allows customers to better understand their coding habits and their contributions to the GitHub ecosystem.

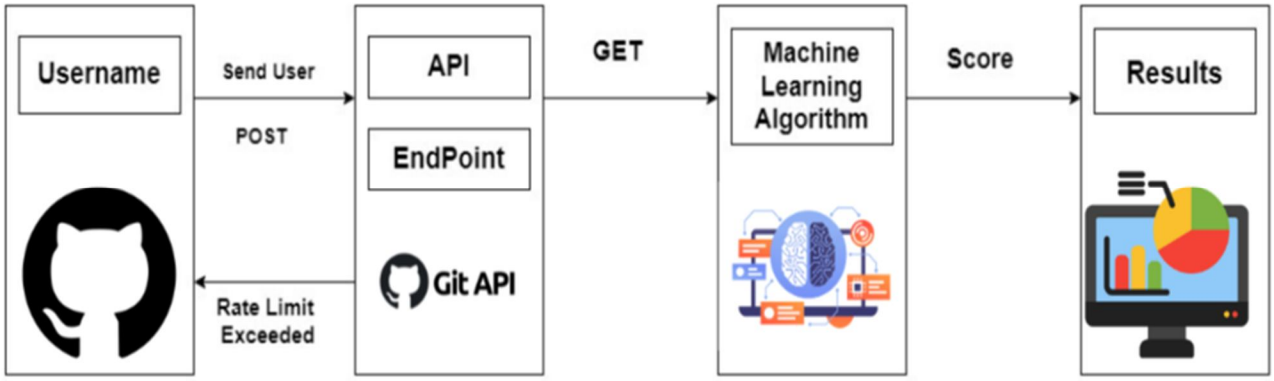


Figure: Project Flow

This section covers the complex research methods used to analyse commit counts of the top public repositories for GitHub customers. Explores in detail data collection methods, analytical methods used, and design methods. The project flow is described in a diagram that shows the sequence of steps from customer input to final output.

A. *Documentation: A Multidisciplinary Approach*

The primary focus is on the history of commits associated with individual users of GitHub's public repositories. This project adopted a two-pronged data collection strategy using the GitHub REST API and Kaggle's Gitrater dataset.

1) *GitHub REST API:* Users enter their GitHub username into the web interface. Using the GET /users/:username, GET /users/:username/followers and GET /users/:username/following endpoints, the system can track user history contribution, follower count, inclusion, affiliate, etc. It stores user-specific information. The system maintains cost limit exceptions as shown in the flowchart.
2) *Gitrater Dataset from Kaggle:* Other user data: Gitrater data set has been enriched by providing features such as number of repositories, number of followers, number of contributions and contribution stability analysis. This information includes GitHub API information that compares usernames to improve identification of specific users. These two methods enable accurate data collection, providing a better understanding of the factors that influence referral behaviour.

B. *Data Preprocessing: Laying a Strong Foundation*

The first step is to make sure the data consumed by the GitHub API and Gitrater datasets is clean and good.

1) *Transferring and processing JSON responses:* For GitHub API files, JSON responses are carefully parsed to extract relevant information.
2) *Eliminate missing results:* Resolve any inconsistencies or errors in the resulting data to maintain the integrity of the analysis.
3) *Standardized Format:* The format of user data is standardized to facilitate seamless integration of the two data sets.

C. *Most Popular Public Repositories Review*

The purpose is to review the most active repositories in the Search list. These data centres have improved performance, which holds the key to uncovering patterns in behaviour. Best 'n' public repositories; determined by metrics such as date, number of forks, or their order in the list that is retrieved.

D. *Measuring commitment activities*

Once the top warehouse has been determined, the process of identifying the commitment should be done Consider all commitments for each selected storage location.

1) *Iterating over commit history:* This involves using the GitHub API to iterate over the commit history of each repository and saving any changes.
2) *Bulk commit counts:* Alternatively, you can improve performance by using batch commits in the GitHub API (if available).

E. *Visualization: Illuminating Insights with Diagrams*

Representation of data is essential to explain hidden patterns and relationships.

1) *Pie and Bar Charts:* These views display storage names and counts, providing a clear understanding of the relevant levels.
2) *Distribution and relationship Matrix:* This offering explores the relationship between the count and the consumer, such as the count or origin collaboration history.

F. *Machine Learning Models*

Use two powerful regression models to predict numbers based on customer and store location characteristics.

1) *Random Forest Regressor:* Builds multiple decision trees and averages their predictions to reduce the number of changes and make them more accurate. Get insight into which user characteristics have the biggest impact on computing.
2) *Gradient Boosting Regressor:* Build a regression model by correcting errors in previous models to increase accuracy. Its ability to achieve high accuracy on complex data makes it suitable for predictive computing.

IV. RESULTS

This section provides a detailed overview of profile analysis to understand activity patterns in GitHub repositories. Insights from previous data, insights, and performance evaluations of machine learning models (Random Forest Regressor and Gradient Boosting Regressor) are discussed in depth.

A. *Data Preprocessing and Visualization:*

Initially there is a preprocessing of the data provided by the GitHub API and Gitrater datasets. This step is important to ensure the accuracy and consistency of the data.

JSON responses from the GitHub API are carefully parsed to extract important information such as commit history and user details. Resolve discrepancies by filtering custom storage facilities and handling missing values appropriately. Gitrater data were also checked for variance and techniques such as imputation were used to deal with missing values.

User information is standardized to seamlessly integrate information from two sources. This process facilitates further analysis with additional user-specific information such as follower count and interaction history. Visualizations, including exploded charts and line charts, provide first-hand information. This information forms the basis for a deeper analysis by revealing the relationship between the number of users and customer characteristics such as the number of followers or interaction history.

B. Browse the best public data repositories

Top Check out the side. 'n' is an important step in defining public repositories. The focus is on warehouses with significant improvements. Two main points are used:

1) *Community Share Tests:* The number of stars and forks obtained from the GitHub API indicates the level of interest and implementation of the community.
2) *Repository Order:* An easier way is to select the repositories by order in the search form, giving a comprehensive overview of the various root codes.

By focusing on the best sources of data, analysis can focus on the most important sources and discover important patterns in behaviour.

C. Machine Learning Model Performance

This analysis uses two powerful regression models: random forest regressor and gradient boosting regressor to predict numbers based on customer and location characteristics. The performance of these models provides important information about the factors affecting export performance.

1) *Random Forest Regressor:* The model has a mean square error (MSE) of 0.35142797713709574 and an R-squared score (R^2) of 0.7933087501614843 on the test data. These measurements show little or no good ability to predict the count. The main score shows that user characteristics such as number of followers and number of interactions affect the number of predictions. This interpretation gives us an understanding of what makes redirects work. shows a slight improvement in prediction accuracy compared to the random forest regressor. To avoid overfitting, careful hyperparameter tuning and regularization are required, indicating the sensitivity of the model to these parameters.

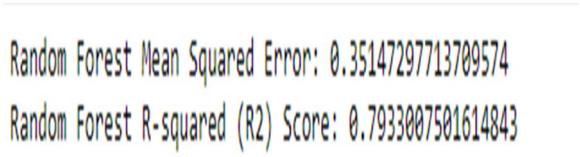


Figure : Performance Metrices (Random Forest)

2) *Gradient Boosting Regressor:* The MSE of the model is 0.324895444345211 and the R-squared (R^2) score is 0.889520180186819; This shows a slightly better prediction accuracy compared to the random forest regressor. To avoid overfitting, careful hyperparameter tuning and regularization are required, indicating the sensitivity of the model to these parameters.

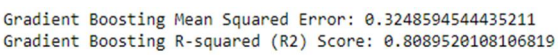


Figure : Performance Metrices (Gradient Boosting)

D. Username Input Field

In the first step of the review, users will be asked to enter their GitHub username. This entry is designed to store the username used as a key identifier when retrieving relevant data from the GitHub API.

Users can enter a valid username and begin the process of retrieving GitHub files and submitting projects for self-checking. This step is important because it forms the basis of the entire rescue and operation.



Figure: Username Field

E. User Profiles and Statistics

After submitting the username, the system will retrieve and display the GitHub user's profile information and some statistics. This includes the user's avatar, name, number of followers, number of followers, and number of followers. Statistical information such as total contributions, open counts, download requests and other contributions are also provided. This overview provides a snapshot of users' activities and contributions on GitHub, forming the basis for a deeper analysis of development patterns.



Figure: User Profile and Statistics

F. Statistics Graphs

The third step is graphs showing the statistics of users' GitHub activities. These charts include:

1) *Reposes by language:* The first chart showing the distribution of repositories across different languages. This helps understand user experience and interest in different languages.
2) *Commits per Reposes:* Pie chart/donut chart showing the processing count for each warehouse. This report provides information about the strength of the user's contribution to their project. These visual agents help identify trends and patterns in users' development behaviour, providing insight into their coding practices and project involvement.

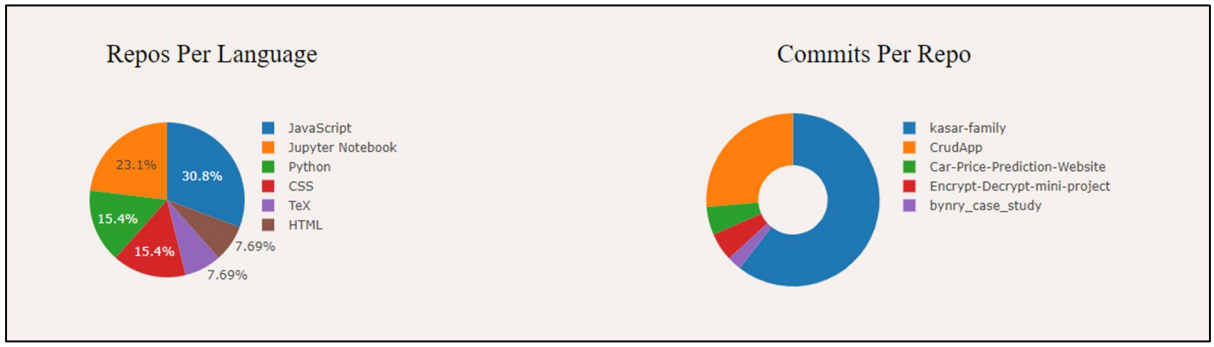


Figure: Statistical Charts

G. Rating Report

The final step is to provide a detailed report showing the results of the analysis. The report summarizes all data and insights into one unified view, highlighting key metrics such as total commitments, peak reserves, and key results. It also includes an assessment of user impact per employee, as well as an overview of their overall work on GitHub. The report aims to provide a clear and structured description of the user's development, facilitating an understanding of their contributions and areas of expertise.

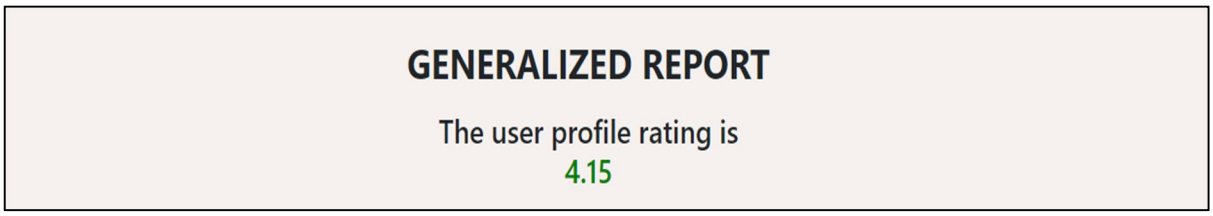


Figure: Generalized Report.

V. CONCLUSION

GitHub's user activity analytics system, which uses the GitHub API and machine learning models, is a powerful way to provide deep insight into developer behaviour and project engagement. By analysing user data, storage, and transmission history, the system can identify patterns and important patterns, such as users and relationships between users. The system works by pre-processing and normalizing data, then training and analysing learning models such as random forests and gradient boosting regressors to predict numbers based on users and characteristics of storage facilities.

The system can be widely used to understand and improve product development, track progress, and identify stakeholders in the open-source community. As AI technology continues to advance, integrating multiple models and techniques can improve prediction accuracy and provide better insights. The scheme demonstrated its effectiveness in analysing and predicting GitHub work, achieving critical accuracy and delivering useful information. Overall, GitHub's user effort analysis using the GitHub API and machine learning has the potential to be a powerful tool for developers, project managers, and engineers to better understand and improve the software development process.

VI. ACKNOWLEDGEMENT

We are grateful for the guidance from Dr. Bhagyashree Tingare, Assistant Professor, Department of Artificial Intelligence and Data Science, D.Y. Patil College of Engineering, Akurdi. Her valuable guidance, constructive feedback and tireless support were instrumental in the success of this project. Her expertise and passion influenced our research, and we are grateful for her advice.

REFERENCES

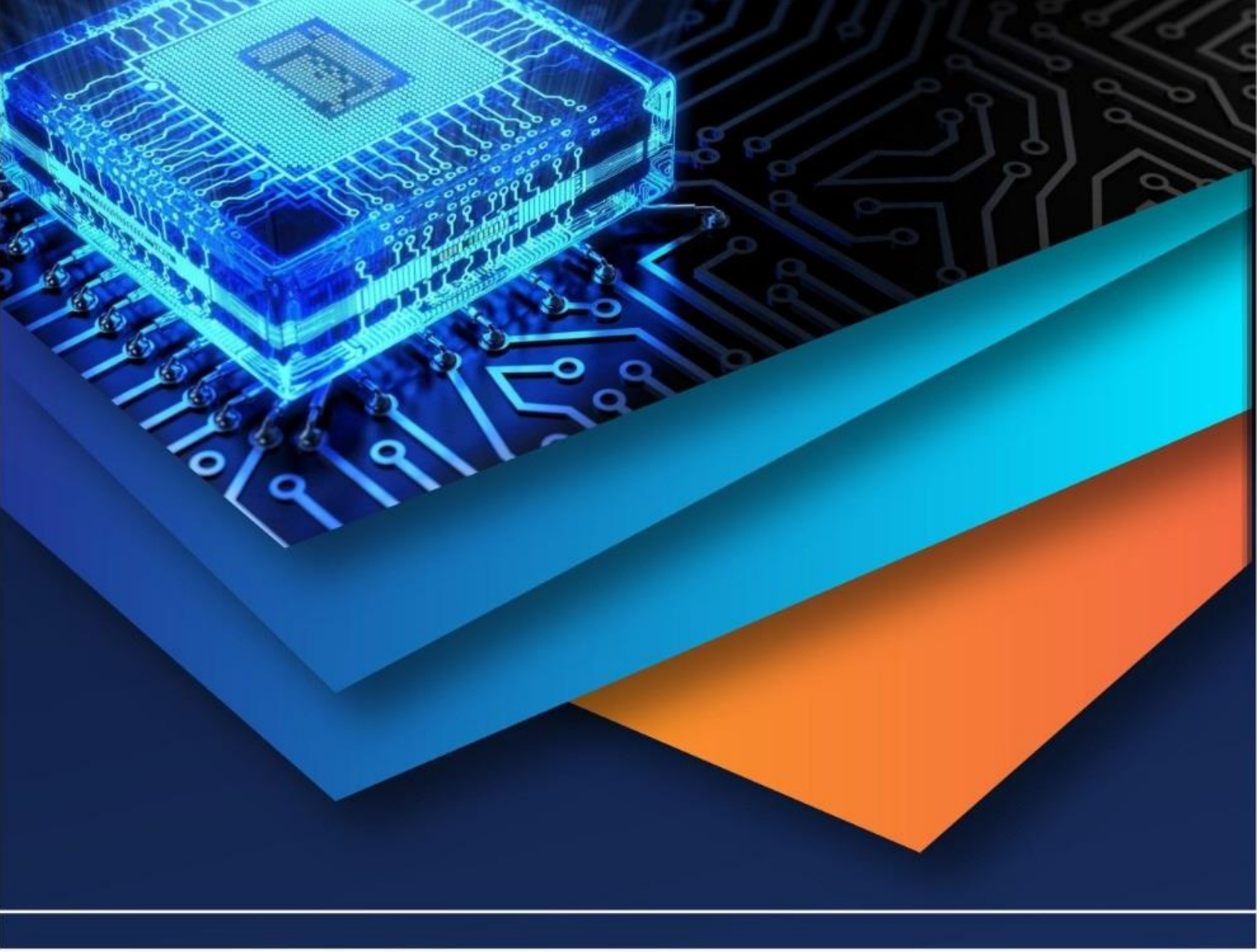
[1] V. Bandara et al., "Fix that Fix Commit: A real-world remediation analysis of JavaScript projects," 2020 IEEE 20th International Working Conference on Source Code Analysis and Manipulation (SCAM), Adelaide, SA, Australia, 2020.

[2] D. Gonzalez, T. Zimmermann, P. Godefroid and M. Schaefer, "Anomalicious: Automated Detection of Anomalous and Potentially Malicious Commits on GitHub," 2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP), Madrid, ES, 2021.

[3] R. T. R. Jayasekara, K. A. N. D. Kudarachchi, K. G. S. S. K. Kariyawasam, D. Rajapaksha, S. L. Jayasinghe and S. Thelijjagoda, "DevFlair: A Framework to Automate the Pre-screening Process of Software Engineering Job Candidates," 2022 4th International Conference on Advancements in Computing (ICAC), Colombo, Sri Lanka, 2022.

[4] A. Giri, A. Ravikumar, S. Mote and R. Bharadwaj, "Vritthi - a theoretical framework for IT recruitment based on machine learning techniques applied over Twitter, LinkedIn, SPOJ and GitHub profiles," 2016 International Conference on Data Mining and Advanced Computing (SAPIENCE), Ernakulam, India, 2016.

[5] F. Chatziasimidis and I. Stamelos, "Data collection and analysis of GitHub repositories and users," 2015 6th International Conference on Information, Intelligence, Systems and Applications (IISA), Corfu, Greece, 2015.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)