[[1]](#footnote-1)

**CodeRewards: An Intelligent Code Review Application**

*Author: Balaji Rajan T S  
 Co Author: Bhanu Chella*

**Wells Fargo**

**ABSTRACT**

CodeRewards is an innovative and comprehensive code review application designed to streamline the code review process and enhance collaboration among software developers. By automating various tasks, incorporating AI-driven evaluation, and providing comprehensive tracking and compensation systems, CodeRewards aims to improve code quality, promote effective reviews, and recognize the efforts of reviewers. This white paper provides a detailed overview of the features, functionality, architecture, and benefits of CodeRewards.

1. **INTRODUCTION**

CodeRewards addresses the challenges faced during code reviews by introducing automation, AI-powered analysis, and a performance tracking system. The application integrates with popular code hosting platforms like GitHub or Bitbucket, enabling seamless parsing of pull requests. Leveraging predefined rules and supervisor/peer relationships, CodeRewards automatically assigns appropriate default code reviewers. Furthermore, it notifies all subscribers of new pull requests, facilitating timely review feedback.

2. **FEATURES AND FUNCTIONALITY**

2.1 **Pull Request Parsing**: CodeRewards utilizes APIs from GitHub or Bitbucket to extract essential information from pull requests. It comprehensively analyzes code changes, comments, and review history, enabling a thorough evaluation.

2.2 **Automatic Reviewer Assignment**: Leveraging predefined rules and supervisor/peer relationships, CodeRewards intelligently assigns default code reviewers for each pull request. This ensures that reviews are assigned to individuals with relevant expertise and a proper understanding of the developer's work.

2.3 **Notification System**: CodeRewards employs a robust notification system that broadcasts notifications to all subscribers when a new pull request is available for review. Users can customize their notification preferences, such as email, push notifications, or in-app notifications, ensuring that reviewers stay informed.

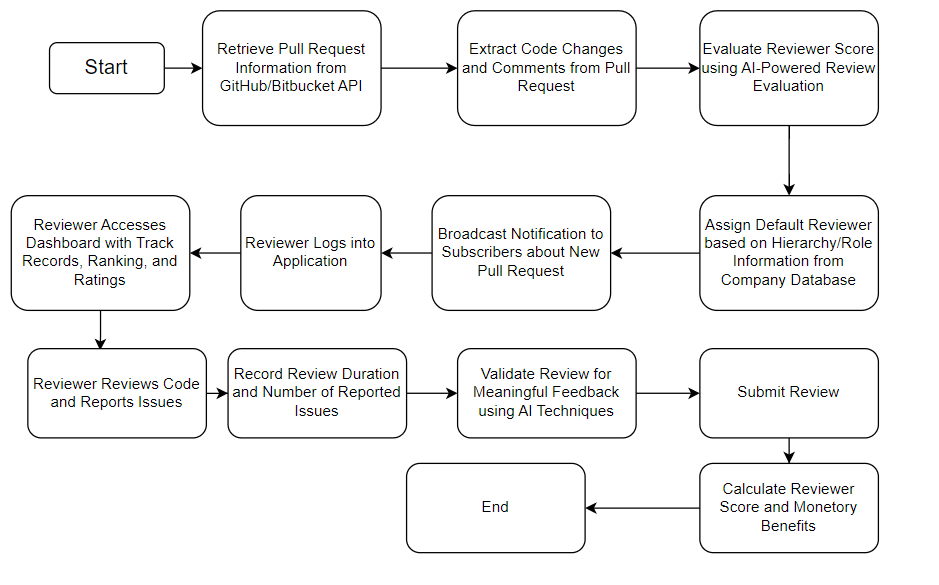
2.4 **Comprehensive Dashboard**: CodeRewards offers a user-friendly dashboard for reviewers, providing a centralized hub for all review-related activities. Reviewers can access pull requests assigned to them, view their review history, track performance metrics, and manage their settings efficiently.

2.5 **Performance Tracking**: CodeRewards records the time spent by each reviewer on reviewing a pull request and tracks the number of reported code issues. These metrics are valuable for evaluating reviewer performance, identifying areas of improvement, and recognizing outstanding contributions.

2.6 **AI-Powered Review Evaluation**: CodeRewards employs intelligent AI algorithms to evaluate the quality and effectiveness of code reviews. The AI model analyzes the content and structure of review comments, identifies repetitive or unhelpful feedback, and offers suggestions for improvement. This ensures that reviews are meaningful, constructive, and beneficial for the development process.

2.7 **Review Submission and Scoring**: Reviewers can easily submit their feedback through the intuitive interface provided by CodeRewards. The application records all relevant information, including the reviewer's comments, suggestions, and ratings. Based on the AI evaluation and predefined criteria, CodeRewards calculates a reviewer's score, reflecting their review quality, timeliness, and consistency.

2.8 **Monetory Benefits Calculation**: CodeRewards incorporates a monetory benefits system that utilizes the reviewer's score as a basis for determining their compensation or rewards. The scoring system considers various factors, such as review quality, timeliness, reviewer's expertise, and predefined compensation parameters. This ensures a fair and transparent process for recognizing and rewarding reviewers for their valuable contributions.



3. **ARCHITECTURE**

CodeRewards follows a client-server architecture to deliver its robust functionality. The client-side encompasses a web-based interface and mobile applications to ensure flexibility and accessibility. The server-side architecture includes the following components:

3.1 **User Management**: This component handles user authentication, profile management, and permissions, ensuring secure access to the application.

3.2 **Pull Request Integration**: CodeRewards can integrate with GitHub or Bitbucket APIs to extract relevant information from pull requests by leveraging the available API endpoints and authentication mechanisms provided by these platforms. Here's an overview of the integration process:

3.2.1. **Authentication and Authorization**:

CodeRewards needs to authenticate itself with the GitHub or Bitbucket API to access the necessary resources. This typically involves generating API keys or OAuth tokens that grant CodeRewards the required permissions to interact with the repositories and pull requests.

3.2.2. **API Endpoint Access**:

Once authenticated, CodeRewards can use the appropriate API endpoints provided by GitHub or Bitbucket to retrieve pull request data. The APIs typically offer endpoints to access repositories, pull requests, comments, and review details.

3.2.3. **Fetching Pull Request Information**:

CodeRewards can make API calls to retrieve the list of repositories and their respective pull requests. This information can be used to populate the application's dashboard and notify reviewers about new pull requests for review.

3.2.4. **Extracting Code Changes and Comments**:

For each pull request, CodeRewards can retrieve the specific code changes and associated comments. The APIs usually provide endpoints to fetch the files modified in a pull request, as well as any review comments or discussions.

3.2.5. **Analyzing Review History**:

CodeRewards can access the review history of pull requests, including information about reviewers, their comments, and any actions taken. This allows the application to track the progress of reviews, evaluate reviewer performance, and generate metrics for analysis.

3.2.6. **Continuous Synchronization**:

To ensure that CodeRewards stays up-to-date with the latest pull request data, the application can periodically poll the APIs or set up webhook integrations. Webhooks enable real-time notifications of updates, such as new pull requests or changes in review status, ensuring timely information retrieval.

3.3 **Automatic Reviewer Assignment**: The automatic reviewer assignment component utilizes predefined rules and supervisor/peer relationships to intelligently assign default code reviewers for each pull request. This ensures that reviews are directed to the most suitable individuals. To integrate with the Wells Fargo's hierarchy and role information from the Wells Fargo's database for Automatic Reviewer Assignment in CodeRewards, the application can follow these steps:

3.3.1. **Establish Database Connection**:

CodeRewards needs to establish a connection to the Wells Fargo's database, which contains the hierarchy and role information of employees. This typically involves configuring the necessary credentials, connection settings, and ensuring compatibility with the database system used by the Wells Fargo.

3.3.2. **Retrieve Hierarchy and Role Data**:

CodeRewards queries the Wells Fargo's database to retrieve the hierarchy and role information of employees. This data may include details such as employee names, job titles, reporting relationships, and team structures. The specific tables and fields to be queried depend on the structure and organization of the Wells Fargo's database.

3.3.3. **Mapping Roles to Reviewer Permissions**:

CodeRewards maps the roles defined in the Wells Fargo's database to reviewer permissions within the application. For example, if a specific role in the database represents a senior developer, CodeRewards can assign corresponding reviewer permissions, such as the ability to review complex or critical code changes.

3.3.4. **Defining Reviewer Assignment Rules**:

Based on the retrieved hierarchy and role information, CodeRewards defines rules for automatic reviewer assignment. These rules can consider factors such as the author's position, the reviewer's position, team associations, and project expertise. For instance, a rule might dictate that pull requests from junior developers should be assigned to senior developers or their direct supervisors.

3.3.5. **Algorithmic Assignment Logic**:

CodeRewards implements an algorithmic logic that takes the defined rules and hierarchy information as inputs. This logic determines the appropriate default code reviewer(s) for each pull request based on the author's role, team structure, and other relevant factors. The algorithm can be designed to prioritize supervisors, peers, or individuals with specific expertise, depending on the specific requirements of the Wells Fargo.

3.3.6. **Automatic Assignment Execution**:

Whenever a new pull request is created, CodeRewards triggers the automatic reviewer assignment process. The application retrieves the relevant information from the pull request, such as the author's role or team, and applies the assignment algorithm to determine the default reviewer(s) for that particular pull request.

3.3.7. Reviewer Assignment Notifications:

CodeRewards sends notifications to the assigned default reviewers, informing them about the new pull request that requires their review. The notifications can be delivered through email, in-app notifications, or other preferred communication channels.

By integrating with the Wells Fargo's hierarchy and role information from the Wells Fargo's database, CodeRewards ensures that automatic reviewer assignment aligns with the organizational structure and roles. This streamlines the review process, ensures that reviews are assigned to appropriate individuals, and promotes efficient collaboration within the development teams.

3.4 **Notification Engine**: The notification engine broadcasts notifications to subscribers through various channels, such as email, push notifications, or in-app notifications. This component facilitates prompt communication and ensures that reviewers are aware of new pull requests requiring their attention.

3.5 **AI Review Evaluation**: AI-Powered Review Evaluation in CodeRewards is a sophisticated component that utilizes machine learning algorithms and natural language processing techniques to assess the quality and effectiveness of code reviews. It aims to provide intelligent suggestions, identify repetitive or unhelpful feedback, and enhance the overall review process.

3.5.1. **Machine Learning Model Training**:

CodeRewards employs machine learning techniques to train a model using a large dataset of code reviews. The dataset consists of annotated reviews that are labeled with attributes such as clarity, helpfulness, and relevance. The model is trained to learn patterns and correlations between the review content and these attributes.

3.5.2**. Natural Language Processing (NLP) Analysis**:

The AI component utilizes NLP techniques to analyze the content and structure of review comments. It performs tasks such as text tokenization, part-of-speech tagging, and sentiment analysis to gain a deeper understanding of the review content.

3.5.3. **Review Quality Assessment**:

The trained model evaluates the quality of each review based on various factors, including clarity, relevance to the code changes, and overall helpfulness. The model assigns a score or rating to each review, indicating its quality level.

3.5.4. **Intelligent Suggestions**:

CodeRewards's AI component provides intelligent suggestions to reviewers to improve the quality and effectiveness of their feedback. It identifies common issues, offers recommendations for specific code patterns or practices, and highlights areas where the review could be more constructive or insightful.

3.5.5. **Redundancy and Repeated Feedback Detection**:

The AI component also helps detect redundant or repeated feedback in code reviews. It analyzes the review comments and identifies instances where similar comments or suggestions have been provided multiple times. This helps streamline the review process and avoids unnecessary duplication of feedback.

3.5.6. **Continuous Learning and Improvement**:

The AI model in CodeRewards is designed to continuously learn and improve over time. It can be fine-tuned based on user feedback, updated with new training data, and adapted to specific programming languages or codebases. This allows the AI component to become more accurate and effective in evaluating reviews as it gains more experience.

By leveraging AI-Powered Review Evaluation, CodeRewards enhances the quality of code reviews by providing meaningful insights, intelligent suggestions, and reducing repetitive feedback. This component improves the efficiency and effectiveness of the review process, empowering reviewers to provide valuable feedback that helps improve code quality and promotes collaboration among developers.

3.6 **Performance Tracking**: The performance tracking component records reviewer metrics, including review duration and the number of reported issues. These metrics are essential for evaluating reviewer performance, generating analytics, and tracking improvements over time.

3.7 **Reviewer Scoring and Compensation**: The reviewer scoring and compensation component calculates reviewer scores based on the AI evaluation and predefined criteria. It takes into account factors such as review quality, timeliness, consistency, expertise, and other predefined parameters. The scores obtained serve as the basis for determining reviewer compensation or rewards, ensuring a fair and motivating system.

Reviewer Scoring and Compensation in CodeRewards is a component that calculates reviewer scores based on predefined criteria and determines their corresponding compensation or rewards. It aims to establish a fair and motivating system that recognizes and incentivizes reviewer performance.

1. **Reviewer Score Calculation**:

CodeRewards assigns a score to each reviewer based on various factors. These factors can include the quality of their reviews, timeliness in completing reviews, consistency in providing feedback, adherence to review guidelines, expertise in specific programming languages or domains, and other predefined parameters.

2. **Review Quality Assessment**:

One of the key factors in calculating the reviewer score is the quality of their reviews. CodeRewards evaluates the quality using AI-powered review evaluation techniques, as discussed earlier. The AI component assesses the clarity, relevance, helpfulness, and constructiveness of the reviewer's feedback to determine the quality of their reviews.

3. **Timeliness and Consistency**:

CodeRewards considers the reviewer's timeliness in completing reviews as an important aspect. Reviewers who consistently provide timely reviews are rewarded with higher scores. Additionally, the consistency of the reviewer's performance, such as regularly participating in reviews and maintaining a certain level of review activity, also contributes to their score.

4. **Expertise and Domain Knowledge**:

Reviewers with expertise in specific programming languages, frameworks, or domains may receive additional recognition and score boosts. CodeRewards can incorporate this factor by considering the reviewer's experience, certifications, or contributions in relevant areas.

5. **Reviewer Rankings and Ratings**:

CodeRewards maintains a ranking system that ranks reviewers based on their scores. Reviewers with higher scores are ranked more prominently, reflecting their proficiency and contribution to the review process. These rankings can be displayed in the application's dashboard or leaderboard, fostering healthy competition and providing recognition to the top-performing reviewers.

6. **Compensation and Rewards**:

Based on the reviewer scores, CodeRewards determines the compensation or rewards for reviewers. The compensation can take various forms, such as monetary benefits, performance-based bonuses, recognition within the organization, or other incentives. Reviewers with higher scores may be eligible for greater compensation or rewards, creating motivation to consistently deliver high-quality reviews.

7. **Continuous Evaluation and Feedback Loop**:

The Reviewer Scoring and Compensation component operates in a continuous feedback loop. CodeRewards regularly collects feedback from developers, project managers, and stakeholders regarding the quality and impact of the reviews. This feedback is used to refine the scoring criteria and adjust the compensation structure, ensuring an accurate and fair system that aligns with the organization's goals.

By implementing Reviewer Scoring and Compensation in CodeRewards, the application encourages reviewers to provide high-quality reviews, rewards their efforts, and fosters a culture of excellence and recognition. It motivates reviewers to actively participate in the review process, improve their skills, and contribute to the overall code quality and collaboration within the development teams.

4. **BENEFITS**

4.1 **Improved Efficiency**: CodeRewards automates the assignment of code reviews, reducing manual effort and ensuring timely feedback. This streamlines the review process, enabling faster iterations and increased productivity.

4.2 **Enhanced Collaboration**: CodeRewards facilitates collaboration among developers by notifying subscribers of new pull requests. By bringing together reviewers with diverse expertise, it encourages knowledge sharing, fosters effective discussions, and promotes collective code improvement.

4.3 **AI-Driven Evaluation**: CodeRewards's AI-powered review analysis assists reviewers in providing more meaningful and effective feedback. By identifying repetitive or unhelpful feedback, offering suggestions for improvement, and evaluating the overall review quality, it enhances the value of code reviews and drives continuous improvement.

4.4 **Performance Tracking and Recognition**: CodeRewards enables reviewers to track their performance metrics, including rankings, ratings, and historical data. This tracking system provides a comprehensive overview of their review activity, identifies areas for improvement, and recognizes their expertise and contributions.

4.5 **Monetory Benefits**: CodeRewards's scoring system ensures a fair and transparent process for determining reviewer compensation or rewards. By aligning rewards with reviewer scores, it incentivizes reviewers to maintain high-quality reviews consistently, fostering a culture of excellence and recognition.

5. **CONCLUSION**

CodeRewards revolutionizes the code review process by automating key tasks, leveraging AI-powered evaluation, and incorporating a performance tracking and compensation system. It empowers software developers with a comprehensive and intuitive platform that enhances collaboration, improves code quality, and recognizes the efforts of reviewers. With CodeRewards, organizations can establish efficient and effective code review practices, leading to accelerated development cycles, higher-quality software, and increased developer satisfaction.

6. **REFERENCES**

* *Bitbucket API Documentation*. Available at: https://developer.atlassian.com/server/bitbucket/rest/v811
* *Natural Language Processing with Python*. Bird, Steven, et al. O'Reilly Media, 2009.
* *Machine Learning: A Probabilistic Perspective*. Murphy, Kevin P. MIT Press, 2012.
* *Code Review Best Practices*. Fogel, Jason. O'Reilly Media, 2015.
* *Building Machine Learning Powered Applications*: Going from Idea to Product. Kuchana, Emmanuel

1. [↑](#footnote-ref-1)