Accolade

Sivakumarreddy Sangu
sivakumarreddy@vt.edu
Virginia Tech
Blacksburg, VA, USA

Yaswanth Chakiri
yaswanth22@vt.edu
Virginia Tech
Blacksburg, VA, USA

Samhitha Pentaparthy
samhithap@vt.edu
Virginia Tech
Blacksburg, VA, USA

Akshay Reddy Narra
akshayreddy@vt.edu
Virginia Tech
Blacksburg, VA, USA

ABSTRACT

An organization's incentive system, which includes both monetary and non-monetary benefits, has become crucial for monitoring employees' performance. In software development, we talk a lot about hard skills and frequently forget about the role of soft skills, which can be a big element in organizational success. Every software engineer is ultimately a human, and an individual's performance and productivity suffer from a lack of desire. Due to a lack of personalization and a constrained scope of acknowledged contributions, current standard methods of motivation and engagement are insufficient. Along with their contributions to the team, employees' contributions to the software development community need to be acknowledged. Additionally, manual distribution and tracking of rewards can be time-consuming and impractical, especially in larger or more complex environments. To address this problem, the Accolade application is a personalized and dynamic reward system that can effectively incentivize desired behaviours and increase user motivation. It will effectively track employee contributions, determine appropriate reward points, and provides analytics to measure employee performance and engagement. The application dynamically identifies employee contributions to open-source communities like stack overflow using the user ID and StackAPI library. Also, the application provides a kudos feature where colleagues can send rewards to each other. Finally, it consists of a reward analytics dashboard where employees and their managers can effectively keep track of contributions throughout time. The ultimate goal of the software is to improve employee satisfaction and performance by acknowledging their efforts in an effective way.

KEYWORDS

Dynamic reward system, Kudos feature, Rewards, Stack API, Dashboard, Employee satisfaction

1 INTRODUCTION

Due to the COVID pandemic, many companies are working remotely. Organizations that want to survive and develop themselves in the global pandemic need to produce a variety of solutions and enable their employees to work effectively. Accolade Software is one of the strategies for attracting and retaining suitable employees and also facilitating them to enhance their performance in a corporation. Rewarding employees is related to the motivation of the workforce of an organization for better performance. Accolade Software will influence employees' behavior and attitude toward their job if the rewards satisfy their needs and help them to succeed in their personal goals.
In organizational discourse, the maintenance of a reward strategy that guarantees organizational well-being is often emphasized. This is because the primary concern of reward management is how it helps employees achieve higher levels of performance, ensure retention, and increase production in the organization (Armstrong and Stephen, 2005).

The term "reward" is frequently discussed in the organizational literature. It refers to what an organization provides to employees in response to their contribution and performance and what employees want (Agarwal, 1998). Armstrong (2012) averred that a reward is something that recognizes a person‘s contribution. He argued that people are financially rewarded for the job they do (basic pay) and often for their level of performance, competence, or skill (contingent or variable pay) or their services on the job (service-related pay). Krietner and Kinicki (2007) agree with these views that a reward is a form of compensation to the employee for doing the assigned work well, which may come in the form of financial and non-financial incentives.

Accolade's purpose is to attract, retain, and motivate high-quality people to the organization and to develop policies and practices that support the achievement of business objectives. Appreciating and rewarding employees for their performance is important not only to achieving organizational goals but also to maintaining relationships with talented employees in the organization (Sabo, 2011).

Silbert (2005) concluded that it is important for rewards to have a lasting impression. It continues to prove the employee's understanding of the employee and why they are valuable in the organization.

2 MOTIVATING EXAMPLE

A reward system for open-source contributions by an employee can provide several benefits for both the company and its employees. First and foremost, it can encourage employees to participate and contribute to open-source projects, which can be especially important for companies that prioritize open-source as part of their business strategy. By offering rewards for open-source contributions, employees are more likely to see the value in contributing and feel motivated to do so.

Rewarding employees for their open-source contributions can also help show that their efforts are valued and appreciated by the company. This recognition and appreciation can lead to higher job satisfaction and engagement among employees, which can have positive effects on overall employee morale and retention. Additionally, companies that have a strong open-source culture and actively encourage and reward their employees for contributing to open-source projects may have a competitive advantage in attracting and retaining talented employees who are passionate about open-source.

Contributing to open-source projects can also help build a sense of community within the company and the broader open-source community. Rewarding employees for their contributions can further reinforce this sense of community and encourage collaboration and knowledge-sharing. This can lead to improved communication and problem-solving skills among employees, which can ultimately benefit the company as a whole.

Overall, implementing a reward system for open-source contributions can help foster a culture of innovation and collaboration within a company, while also providing tangible benefits to employees who participate. By recognizing and rewarding open-source contributions, companies can encourage employee engagement, build a sense of community, and gain a competitive advantage in the tech industry.

3 BACKGROUND WORK

The Accolade application is built using various technologies and tools, including the Flask web framework, PostgreSQL database management system, Power BI analytics platform, Stack API, and GitHub code hosting platform.

Flask is a popular Python web framework that is lightweight and easy to use. It was used in the Accolade application to handle requests and responses from the front-end of the application. Flask provides a simple and flexible way to build web applications, and its use in the Accolade application allowed for fast and efficient development.

PostgreSQL is a powerful open-source database management system that provides excellent reliability, scalability, and performance. It was used in the Accolade application to store employee data and reward points. PostgreSQL's ability to handle large amounts of data and provide advanced query capabilities made it an ideal choice for the Accolade application.

Power BI is a cloud-based business analytics platform that provides interactive visualizations and business intelligence capabilities. It was used in the Accolade application to create the reward analytics dashboard, which allows employees and managers to track contributions over time. Power BI's ability to connect to various data sources and provide real-time analytics made it an ideal choice for the Accolade application.

The Stack API is a RESTful web service that allows developers to access the Stack Overflow community data. It was used in the Accolade application to track employee contributions to open-source communities like Stack Overflow. The Stack API provided an efficient and convenient way to gather information on employee contributions, which could then be used to assign appropriate reward points.

GitHub is a web-based code hosting platform that provides version control and collaboration features. It was used in the Accolade application to manage the codebase and allow for easy collaboration between team members. GitHub's ability to provide a centralized repository for code and provide advanced version control capabilities made it an ideal choice for the Accolade application.

The integration of these various technologies and tools was critical in the development of the Accolade application. Flask was used to create the web application, PostgreSQL was used to store data, Power BI was used to create the reward analytics dashboard, Stack API was used to track employee contributions, and GitHub was used for version control and collaboration. The various technologies were integrated to create a seamless experience for users and ensure efficient development and management of the application.

4 IMPLEMENTATION

4.1 ARCHITECTURE

Our software will use a layered architecture, specifically the Model-View-Controller (MVC) pattern. This pattern separates the application into three main logical components: the model, the view, and the controller. Each of these components is built to handle specific aspects of application development. MVC helps create largely scalable and extensible projects, so it is widely used in industry-standard web development frameworks.

Model
The model component handles the interaction with the database to perform all data-related logical operations. That can represent either the data being transferred between the view and the controller components or any other business-related logic data. An Employee model, for example, will retrieve employee information from a database, manipulate it, and then either update the reward information in the database or use it to render the data on the webpage.

View
The view component handles the user interface of the application. It consists of a template and data and produces a response for the browser. It receives data from the controller of the MVC, packages it, and presents it to the browser for display. For example, the employee view page will include all the UI components, such as buttons, dropdowns, etc., that the final user interacts with.

Controller
Controllers act as an interface between the Model and View components to process all the business logic and incoming requests, manipulate data using the Model component, and interact with the

Views to render the final output. For example, the employee controller will handle all the interactions and inputs from the employee view and update the database using the employee model.

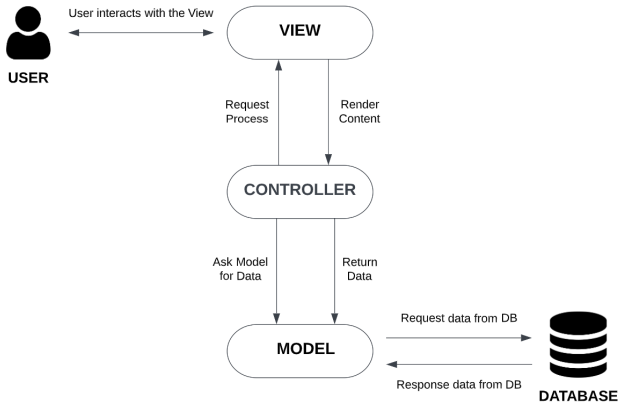

Figure 1: MVC Architecture

4.1.1 *Guidelines and Constraints.*

Clear separation of concerns: The MVC pattern emphasizes the clear separation of concerns between the model, view, and controller components.
Reusability: The code should be designed with reusability in mind, using libraries and modules that can be easily integrated with other parts of the application.
Flexibility: The MVC pattern provides a great deal of flexibility when it comes to making changes or updates to the application.
User Experience: The view should be intuitive and easy to use, with clear navigation and functionality aligned with the user's goals.
Performance: The model should be efficient and optimized for data storage and retrieval, while the view and controller should be designed to minimize latency and maximize responsiveness.

4.2 DESIGN

Behavioral design patterns would be relevant for developing the Accolade application because the patterns focus on communication and interaction between objects in the system. These patterns are concerned with how objects collaborate and delegate responsibilities to each other to achieve specific behaviors that would be suitable to implement with Model-View-Controller(MVC) architecture.

4.3 DEVELOPMENT

The Accolade application was developed using the Agile software development methodology. This allowed for an iterative and incremental approach to development, ensuring that the application was developed in an efficient and effective manner. The development team consisted of four developers, who took up different roles as Scrum Master, Developer and QA, etc.

The development process involved several phases, including planning, design, implementation, testing, and deployment. During the planning phase, the team defined the project scope, identified project requirements, and developed a project plan. The design phase involved the development of wireframes, mockups, and user interface designs. The implementation phase involved the actual development of the application using the Flask framework, PostgreSQL database management system, and Stack API.

The Accolade application consists of several key components, including:

User Authentication: The application uses a secure login system to ensure that only authorized users can access the system. Users can log in using their email and password.

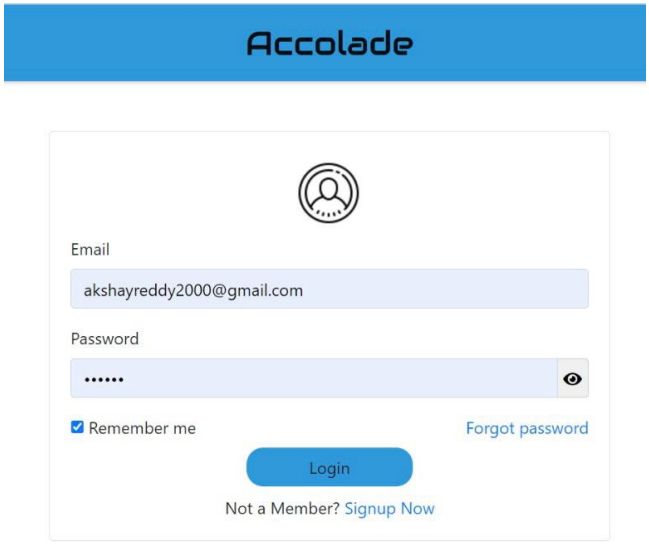

Figure 2: User Authentication Page

Dashboard: The application dashboard provides an overview of the user's rewards points, contributions, and kudos. The dashboard provides quick access to the user's contribution history, and the reward analytics dashboard.

Contribution Tracking: The application tracks user contributions to open-source communities like Stack Overflow using the Stack API. The contributions are then used to assign appropriate reward points to the user.

Reward Analytics Dashboard: The reward analytics dashboard provides a detailed analysis of user contributions, rewards points, and kudos. The dashboard provides insight into the user's performance, engagement, and overall satisfaction with the application.

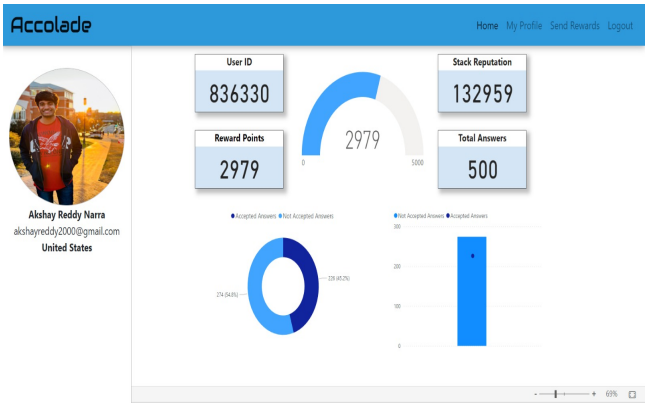

Figure 3: Dashboard Page

Kudos Feature: The kudos feature allows users to send rewards to their colleagues as a way of acknowledging their contributions. The feature is designed to encourage a culture of recognition and collaboration within the organization.

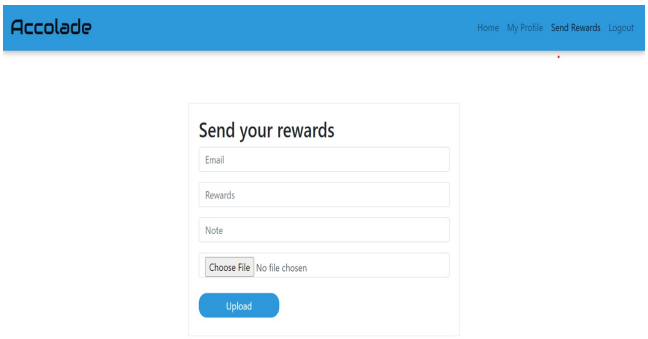

Figure 4: Send Rewards Page

The implementation of the Accolade application involved an iterative and incremental approach to development, ensuring that the application was developed in an efficient and effective manner. The application consists of several key components, including user authentication, dashboard, contribution tracking, kudos feature, and reward analytics dashboard.

4.4 TESTING

The testing phase of the Accolade application involved manual testing to ensure the quality and functionality of the application. The testing phase was crucial to ensuring that the application was meeting the requirements of the project and was free of bugs and issues.

The testing phase included the following types of testing:
Functional testing was used to test the functionality of the application. The team manually tested each feature of the application, including user authentication, dashboard, contribution tracking, kudos feature, and reward analytics dashboard, to ensure that they were working as expected.
Usability testing was used to test the usability of the application. The testing team tested the application's user interface to ensure that it was user-friendly and intuitive. The testing team also tested the application's accessibility to ensure that it was accessible to users with disabilities.
Compatibility testing was used to test the compatibility of the application with different devices and browsers. The testing team tested the application on different devices and browsers to ensure that it was functioning correctly.

During the testing phase, the team identified several bugs and issues with the application. These bugs and issues were logged using the GitHub, and were assigned to the appropriate developer for resolution. The team also conducted regression testing to ensure that the fixes did not introduce any new issues.
The testing phase also involved user acceptance testing (UAT), which involved testing the application with actual end-users. The UAT phase was crucial to ensuring that the application was meeting the needs and expectations of the end-users. The team received feedback from the end-users and made necessary changes to the application.

5 DEPLOYMENT

Accolade can be deployed on external cloud platforms like Azure or AWS. However, handling large amounts of data and querying it efficiently could be challenging. To tackle this issue, a NoSQL database like Amazon DynamoDB could be a suitable solution due to its scalability, reliability, and security features. To simplify the deployment process, the blue-green deployment approach could be used in conjunction with continuous integration and deployment tools like GitHub Actions or GitLab CI/CD. After the successful deployment of the Accolade application, it is crucial to implement monitoring and maintenance techniques to ensure its optimal performance, detect any issues, and keep it up-to-date with evolving needs. To achieve this, a preventative maintenance approach could be used, which involves implementing measures to avoid potential problems from arising.

6 RELATED WORK

Awardco[5] is software that recognizes employee engagement through reward recognition. Admins can manage peer nominations, reward budgets, approval workflows, onboarding programs, and service awards, and can build custom-branded catalogs for reward items such as charity donations, gift cards, and more. Key features of Awardco include manager-to-peer recognition, performance management, recognition tracking, rewards points, and nominations.

Fond[6] is a SaaS employee recognition solution designed to help employees redeem corporate perks and rewards and share achievements with team members. The application allows HR teams to create personalized service award catalogs for employees, track their work anniversaries, and award commemorative plaques and trophies.

"An Empirical Study of the Characteristics of Open Source Software Volunteers and Their Work" by Kevin Crowston and James Howison. This study investigates the motivations and characteristics of open source contributors and how they differ from traditional software developers. The authors found that open source contributors are motivated by a variety of factors, including the desire to learn new skills, gain recognition, and contribute to a community. (Crowston and Howison, 2005)

"Designing Rewards for Peer Code Review in Open-Source Software Projects" by David Redmiles, Michael D. Ernst, and Andrew J. Ko. This paper describes a study of how peer code review can be incentivized in open source projects through the use of rewards. The authors found that rewards such as badges and rankings can be effective in motivating contributors to participate in code reviews. (Redmiles et al., 2014)

"Rewards and Recognition in Free/Libre Open Source Software Communities: A Literature Review" by Georg J.P. Link and Dirk Riehle. This literature review examines the use of rewards and recognition in open source communities and discusses their impact on motivation, participation, and project success. The authors conclude that rewards and recognition can be effective in motivating contributors and improving project outcomes, but that they must be carefully designed and implemented to avoid negative consequences. (Link and Riehle, 2010)

7 DISCUSSION

7.1 LIMITATIONS

Limited recognition of contributions: The application relies on the use of the Stack API library to identify employee contributions to open-source communities like Stack Overflow. While this is a valuable source of information, it may not fully capture all of an employee's contributions to the software development community.

Manual reward distribution: Although the application tracks employee contributions and determines appropriate reward points, the actual distribution of rewards is still a manual process. This can be time-consuming and may lead to delays in reward distribution.

Dependence on external APIs: The application relies on external APIs such as the Stack API library and the GitHub API, which may be subject to changes in their terms of service or availability. This could impact the functionality of the application.

Dependence on accurate user data: The application relies on accurate user data, including user IDs from external platforms such as Stack Overflow and GitHub. If there are errors or inaccuracies in the user data, it could impact the effectiveness of the application.

Due to time constraints, we could not focus on implementing flow of approval to manager for sending the rewards to the peers.

7.2 FUTURE WORK

While the Accolade application is currently robust and fully functional, there are potential areas for improvement that we have identified. These include:

Enhancing GitHub tracking: We propose incorporating the ability to track and acknowledge contributions made by team members on GitHub. By integrating GitHub data, the application would provide a more comprehensive view of employees' open-source contributions, further recognizing their efforts.

Improving dashboard interactivity: Our plan involves enhancing the existing reward analytics dashboard by incorporating more interactive filters. This would allow employees and managers to explore and analyze contributions in a more customizable and dynamic manner, gaining deeper insights into individual and team performance.

Mobile app integration: Recognizing the growing reliance on mobile devices, we suggest developing a mobile app version in addition to the existing web support. This expansion would provide employees with greater accessibility and convenience, enabling them to track their contributions and receive recognition on the go.

8 CONCLUSION

The importance of a personalized and dynamic reward system, as demonstrated by the Accolade application, in motivating desired actions and encouraging employee motivation and engagement cannot be emphasized. The capacity of the application to effectively monitor and measure employee contributions, provide appropriate rewards, and provide performance and engagement statistics is priceless. The Accolade application may continue to improve, giving a complete and versatile method to recognizing and inspiring staff, by implementing proposed enhancements such as refining GitHub tracking, improving dashboard interaction, and introducing mobile app support. Finally, firms can promote increased engagement, productivity, and overall success by effectively appreciating and rewarding employees' efforts. The Accolade application is an effective tool in this endeavor, resulting in great consequences for both people and the firm as a whole.

9 REFERENCES

[**1**] Crowston, K., Howison, J. (2005). An Empirical Study of the Characteristics of Open Source Software Volunteers and Their Work. IEEE Transactions on Software Engineering, 31(6), 481-494.
[**2**] Redmiles, D. F., Ernst, M. D., Ko, A. J. (2014). Designing Rewards for Peer Code Review in Open-Source Software Projects. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) (pp. 1724-1735).
[**3**] Link, G. J. P., Riehle, D. (2010). Rewards and Recognition in Free/Libre Open Source Software Communities: A Literature Review. In 2010 Fourth International Conference on Research Challenges in Information Science (RCIS) (pp. 295-304). IEEE.
[**4**] Hamid Mcheick and Yan Qi, 'Dependency of components in MVC distributed architecture'.
[**5**] Awardco Tool: https://www.award.co/ Steve Sonnenberg - Founder CEO
[**6**] Fond Tool: https://www.fond.co/ Taro Fukuyama - CoFounder CEO, Sunny Tsang - CoFounder