Cloud-Driven Image Matching Game: A Case Study in Continuous Deployment

Malireddy Charan Kumar Reddy, Katragadda MeghaShyam, Kandlapalli Aravind Sai,K Venkatesh, Shinu M Rajagopal
Department of Computer Science and Engineering,
Amrita School of Computing, Bengaluru,Amrita Vishwa Vidyapeetham, India
bl.en.u4cse21121@bl.students.amrita.edu, bl.en.u4cse21085@bl.students.amrita.edu, bl.en.u4cse21087@bl.students.amrita.edu,
bl.en.u4cse21097@bl.students.amrita.edu, shinu_mr@blr.amrita.edu

***Abstract*—This project explores the development of an image matching game leveraging Amazon Web Services (AWS) infrastructure. The game aims to engage users by presenting them with a series of image pairs that they must match correctly within a specified time limit. AWS services such as S3 for storage, Lambda for serverless computing, and DynamoDB for data management are integrated to ensure efficient handling of images, game logic, and user interactions. Additionally, the game incorporates an "I am" feature that dynamically adjusts game difficulty based on user performance, providing a personalized and adaptive gaming experience. This feature utilizes AWS AI/ML services like Amazon Rekognition for image analysis and AWS Lambda for real-time adjustment of game parameters, enhancing user engagement and enjoyment. The image matching game demonstrates the practical application of AWS services in creating a scalable and interactive gaming experience. By leveraging cloud-based resources, the game achieves flexibility in handling varying user loads and maintains high availability and performance. The "I am" feature enhances gameplay by tailoring challenges to individual user capabilities, fostering a more personalized gaming journey. Overall, this project showcases the effectiveness of AWS for developing engaging and adaptive gaming applications while highlighting the potential of AI-driven features to enhance user experience and retention.**

***Index Terms*—Interactive Image Matching Game, Cloud Computing Deployment, Cognitive Skill Enhancement, Web-based Application, Seamless Accessibility**

I. Introduction

In the realm of interactive entertainment, digital games have evolved into powerful platforms for engagement and learning. This project focuses on the development of an image matching game, leveraging the robust capabilities of Amazon Web Services (AWS) to deliver a seamless and engaging user experience. Image matching games are popular for their simplicity and cognitive challenges, making them ideal for showcasing the integration of cloud computing and artificial intelligence (AI) technologies.

AWS provides a comprehensive suite of services that enable developers to build scalable and resilient applications without the upfront costs and complexities of traditional infrastructure management. By harnessing AWS services such as S3 for storing game assets, Lambda for executing backend logic, and DynamoDB for managing game data, this project ensures efficient handling of game mechanics and user interactions. Moreover, the incorporation of an "I am" feature, powered by AWS AI/ML services like Amazon Rekognition, illustrates how real-time image analysis can dynamically adjust gameplay difficulty based on individual user performance, enhancing both challenge and enjoyment. This introduction sets the stage for exploring how AWS empowers developers to create innovative gaming experiences that are responsive, adaptive, and scalable to diverse user demands.

II. Related Work

Ramón López-Viana et al. [1] In an IoT edge computing context, this paper proposes a proof-of-concept implementation of GitOps, a DevOps methodology that integrates version control and automation to infrastructure management. The authors point out that although Kubernetes-based solutions have been the primary focus of GitOps, edge computing for the Internet of Things may also benefit from its ideas, as both have similarities such as size and highly dispersed workloads. To assess the viability of GitOps at the edge, the authors note that experimental data is necessary because this adoption environment is different from the original one for GitOps. The paper details the deployment of a GitOps workflow spanning Linux servers and a Raspberry Pi-based edge cluster running KubeEdge, utilizing technologies from the Cloud Native Computing Foundation (CNCF) ecosystem, such as GitLab for continuous integration (CI), Nexus OSS for artifact registration, and ArgoCD for continuous distribution. Presenting their findings and lessons learned from this proof-of-concept, the authors draw attention to the difficulties and constraints associated with implementing GitOps concepts in the context of IoT edge computing.

Neelam Singh et al. [2] The paper offers a thorough walkthrough of utilizing Jenkins, Ansible, and Kubernetes to create a completely automated pipeline for continuous integration and deployment. The authors discuss about how DevOps techniques help to speed up software development and delivery without sacrificing quality. The CI/CD pipeline method is shown, emphasizing the functions of Ansible for continuous deployment and Jenkins for continuous integration. The implementation specifics are also covered in the paper, which includes GitHub integration for automated code distribution, Jenkins and Ansible server configuration, and Kubernetes cluster setup. The writers illustrate Jenkins' effec-

tiveness and scalability by contrasting it with other well-known CI/CD systems. For CI/CD chores, the suggested architecture is demonstrated to be robust and efficient, offering benefits like full automation, scalability, and rapid update delivery.

Robert Botez et al [3] Cloud computing and DevOps approaches are covered in detail in the paper "Implementation of a Continuous Integration and Deployment Pipeline for Containerized Applications in Amazon Web Services Using Jenkins, Ansible and Kubernetes" . The three models of cloud services—Infrastructure as a Service, Platform as a Service, and Software as a Service—are highlighted in their progression. In contemporary IT settings, where development and operations teams collaborate to improve flexibility and scalability, the paper highlights the significance of DevOps and Continuous Integration/Continuous Deployment approaches. The authors compare the features and applications of many CI/CD tools, such as Jenkins, Ansible, TeamCity, Bamboo, Puppet, and Chef. Jenkins is used for continuous integration and deployment (CI) and Ansible is used for continuous integration and delivery (CD) of an AWS web application. The pipeline consists of actions like launching the containerized application, adding new resources to the Kubernetes cluster, and identifying modifications to the source code. The pipeline's efficiency and dependability are demonstrated by experimental results. It incorporates six different technologies and guarantees zero downtime.

Len Bass et al. [4] The last link in a software supply chain, the deployment pipeline, is shown as having an engineering process for security in this study. The three ways that a pipeline may be subverted, according to the authors, are via deploying an erroneous image, avoiding the pipeline, and getting into the production environment from another environment. They concentrate on the first case and provide a method to fortify the pipeline by distinguishing between reliable and unreliable elements, simulating their interactions, and breaking down unreliable elements into parts that are reliable and unreliable. Applying this procedure to a pipeline using Jenkins, Chef, Docker, and AWS, the authors discover that although they can't fully protect the pipeline, they can harden it by limiting component access privileges and network connection.

Indika Perera et al. [5] This paper offers a thorough review of the literature on the application of pipeline automation for Continuous Delivery and Continuous Integration in Agile software project management. It goes over the main advantages of using CICD techniques, including risk reduction, better product quality, quicker time to market, and increased efficiency and productivity. The primary elements of a CICD pipeline, including as build tools, automation tools, test automation frameworks, version control systems, and monitoring tools, are also reviewed in this paper. It also looks at other deployment tactics and how to incorporate them into the CICD strategy, including rolling deployments, dark launches, blue-green deployments, and feature flags. The review of the literature emphasizes the significance of scaling identification, load testing, and benchmarking as essential CICD pipeline processes to guarantee the functionality and stability of the deployed product.

Ganesh Semalty et al.[6] In the context of continuous integration (CI) and continuous deployment (CD), when frequent system releases necessitate a great deal of emphasis on the most recent deployed or modified system, the article presents the Last in First emphasis (LIFF) technique for system deployment. In order to speed up resolution times, machine learning may assist assess tickets and forecast which system components will be impacted. However, imbalanced datasets from too many tickets for recently updated components might compromise the accuracy of the model. For categorization, the study makes use of natural language processing (NLP) methods including TF-IDF and LinearSVC. The study suggests adopting oversampling methods, such as the Synthetic Minority Oversampling Technique (SMOTE), to create synthetic samples for minority classes in order to handle unbalanced datasets. This method increases recall and F1-score. To guarantee constant accuracy across folds, stratified K-fold cross validation is also employed.

Rudolf Ramler et al. [7] An extensive case study of an Austrian online business company's usage of automated testing in its continuous delivery pipeline is presented in this article. It goes over all of the important pipeline steps, such as commit, acceptance test, user acceptance test, and capacity test. The usage of several technologies and frameworks, including Selenium, Jenkins, Silk4Net, and Mantis, is highlighted throughout the paper's discussion of the tasks, roles, and responsibilities for each step. It highlights the value of automated testing as a crucial component of creating and maintaining a successful continuous delivery pipeline and disseminates the knowledge gained from the company's more than six years of hands-on experience in this field.

Germano Veiga et al. [8]The Robot Operating System (ROS) ecosystem is the specific subject of the paper's extensive literature review, which examines the state of continuous deployment (CD) and integration (CI) methods in the robotics industry today. The bulk 82.7% of the 150 well-known ROS repositories that the authors study use some kind of continuous integration (CI), but only about 45% make use of unit or other tests, and even fewer (6% and 4.7%) make use of code quality and coverage measures. Interestingly, the authors note that despite the possible advantages of this strategy for robotic applications, no repository in their research incorporates simulation-based testing into their continuous integration process. In order to enhance the methodical evaluation of robotic software dependability and behavior in simulated settings, the study then suggests a cloud-based robotic simulation framework as an addition to current CI/CD pipelines.

Enrique Moguel et al [9] The problems of building and delivering quantum services in comparison to traditional cloud services are addressed, and the study proposes a process for the continuous deployment of quantum services. In order to enable the automated development of quantum service code, the authors suggest expanding the OpenAPI definition to include custom characteristics for designing quantum circuits

and providers. Using DevOps techniques like Continuous Integration and Continuous Deployment, this method seeks to bring the standards of developing quantum services closer to those of traditional cloud services. The authors examine the creation and delivery of quantum services using a variety of quantum algorithm implementations in order to verify the viability of the suggested procedure. The study makes it easier for software developers to construct and implement quantum services by contributing to the expanding efforts to apply DevOps approaches and Service-Oriented Computing ideas to the field of quantum computing.

III. Proposed Methodology

Using HTML, JavaScript, and CSS, the project intends to create an interactive picture matching card game with an emphasis on improving players' cognitive abilities through difficult memory tests and captivating graphics. During the development process, a user-friendly interface is created utilising CSS for style, JavaScript for game logic, and HTML for structure. Furthermore, the game programme is deployed on a cloud server for scalability and accessibility, and cloud storage solutions are utilised for game assets and user data. These integrations of cloud computing technologies extend to hosting and deployment. Developing a user-friendly interface, putting complex game logic into practice, guaranteeing dependability across devices and network circumstances, and enabling smooth updates via a continuous deployment pipeline are among the challenges. Code modifications are immediately saved in an S3 bucket by connecting GitHub to AWS CodePipeline, which simplifies execution and lets users play the game without interruption from any location. All things considered, the project shows how cloud computing may enhance web-based applications' efficiency, scalability, and accessibility while giving users a fun and instructive gaming experience.

A. Conceptualization and Game Design

This cloud-based image matching game's key objective is to give players of all ages an enjoyable and engaging experience. It is created as an interactive and instructive emoji matching task. The objective of the game is to match pairs of emojis in a certain amount of time or movements, which helps players improve their memory and cognitive abilities while having fun. The game is meant for a wide range of players, from casual players to those looking for a fun pastime, so it's both approachable and attractive. The game provides enhanced scalability, stability, and accessibility through the use of cloud computing technologies, enabling users to conveniently access and enjoy the game from any location with an internet connection.

B. Development Framework Selection

HTML is used to structure web pages, JavaScript is used to implement game logic and interactivity, and CSS is used for style and visual upgrades in the development of the cloud-based emoji matching game. Preprocessors for CSS, such as SASS or LESS, are also taken into consideration to provide more effective style and maintainability. With the help of these technologies, a highly engaging and visually beautiful user experience that appeals to a broad audience may be created.

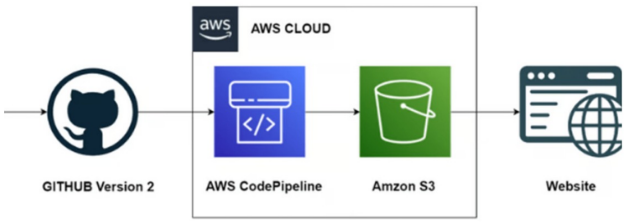

Fig. 1. Workflow diagram

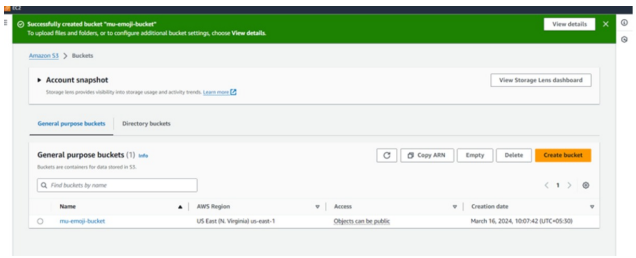

Fig. 2. Creation of S3 bucket

In order to store game assets, such as graphics and code files, an S3 bucket is built on AWS as part of the integration of cloud computing technologies. This offers a dependable and scalable storage option. Users may access game material without any problems because to the S3 bucket's excellent asset management and delivery capabilities. The S3 bucket is connected to AWS CodePipeline, which enables continuous integration and deployment by guaranteeing that code changes are instantly stored and applied without the need for human involvement. This configuration minimizes downtime and enhances user experience overall by enabling quick upgrades and maintenance. To ensure the effective installation, deployment, and maintenance of these technologies, the development environment is set up with tools that enable them, such as version control systems like Git for tracking code changes and collaboration of the game.

C. User Interface Design

Our cloud-based emoji matching game's user interface design prioritizes simple navigation and captivating visuals in order to appeal to a wide range of players, including casual and recreational ones. For convenience and a smooth gaming experience, important components including the timer, score display, game board, and interactive controls are arranged in strategic locations. By utilizing user-centered design concepts, we make sure that visually attractive themes, colors, and animations are included into CSS-crafted visual elements to improve interaction and feedback. The application of responsive design concepts guarantees that the game effortlessly adjusts

to different screen sizes and devices, making it accessible on PCs, tablets, and smartphones.

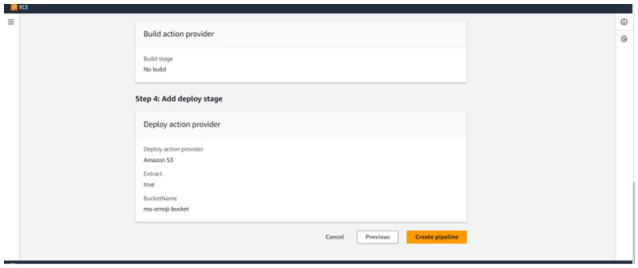

Fig. 3. Continuous deployment pipeline

In order to expedite updates and improvements, our development method now incorporates a continuous deployment pipeline. Code changes, including UI updates for the game, are automatically published to our cloud infrastructure thanks to AWS CodePipeline's build, test, and deployment automation. This configuration enables quick maintenance and iteration, reducing downtime and improving the user experience overall by effectively providing new features and enhancements.

D. Game Logic Implementation

The primary development effort was concentrated on converting the conceptual idea of the picture matching card game into working code during the Game Logic Implementation phase. JavaScript was used to convert the conceptual design into functional code in order to build game logic. This involved outlining the gameplay's guidelines, including the level progression, scoring system, and card matching techniques. Sophisticated algorithms were created to manage the real-time card flipping, matching, and shuffling, guaranteeing smooth gaming and compliance with the desired cognitive difficulties. This cloud-based emoji matching game involves strong JavaScript programming for game logic implementation in order to guarantee smooth gameplay and interactive elements. Among the main features are emoji shuffles, match detection, score updates, and game state management with move counts and time constraints. In order to handle different game circumstances and user interactions with ease, the game logic is modular and reusable. As a player clicks on cards or initiates game events, event handling mechanisms are put into place to react to those actions.

E. Visual Enhancements with CSS

In order to improve both the visual appeal and user experience of our cloud-based emoji matching game, visual enhancements with CSS are essential. The fonts, colors, layouts, and animations of the gaming interface may all be styled and customized using CSS. Thematic designs and color palettes are deliberately selected to produce a unified, aesthetically beautiful environment that goes well with the lighthearted tone of the game. To create seamless and interesting interactions, including card flips or visual feedback when matching emojis, CSS animations and transitions are used. By making the game

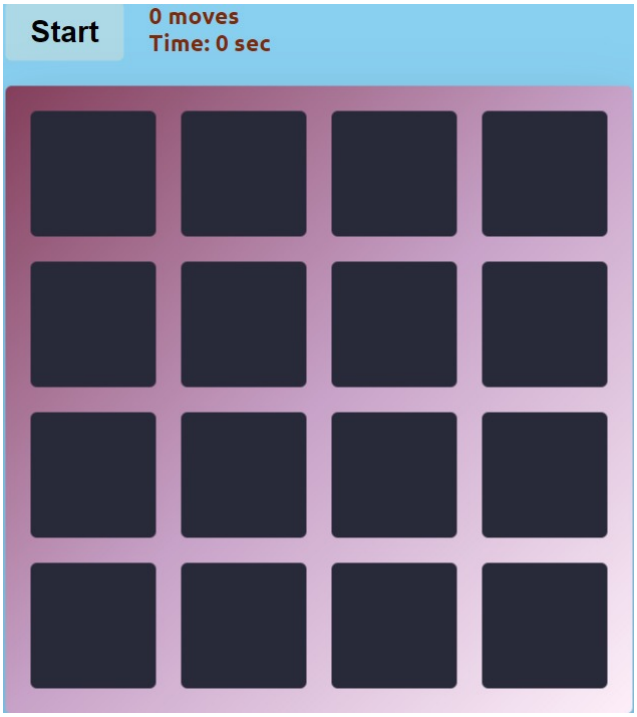

Fig. 4. Emoji matching

fluidly adjust to various screen sizes and devices, responsive design principles maximize accessibility and usability. We simplify the administration and maintainability of stylesheets by utilizing CSS preprocessors such as SASS or LESS. This allows for faster iterations and modifications to the visual presentation of the game. This strategy not only makes the game seem better overall, but it also makes the user experience more engaging and pleasurable, which encourages players to stay in the game longer.

F. Integration of Cloud Computing Technologies

Reliability, scalability, and accessibility of our cloud-based emoji matching game are all greatly improved by the integration of cloud computing technologies. To ensure fast and dependable content delivery, we use AWS to establish an S3 bucket for effective game asset management and storage. Updates and changes may be made smoothly with no downtime thanks to AWS CodePipeline's automation of continuous integration and deployment. By dynamically scaling resources to match user demand, the cloud architecture allows the game to accommodate different traffic volumes. Players may have a seamless and continuous gaming experience because to this configuration, which guarantees that they can access the game from any location with internet access. Through its integration, cloud computing is demonstrated to be able to offer web-based applications a reliable backend that facilitates high availability and smooth maintenance.

G. Continuous Deployment Pipeline

Our cloud-based emoji matching game's Continuous Deployment Pipeline is made to guarantee quick and automatic

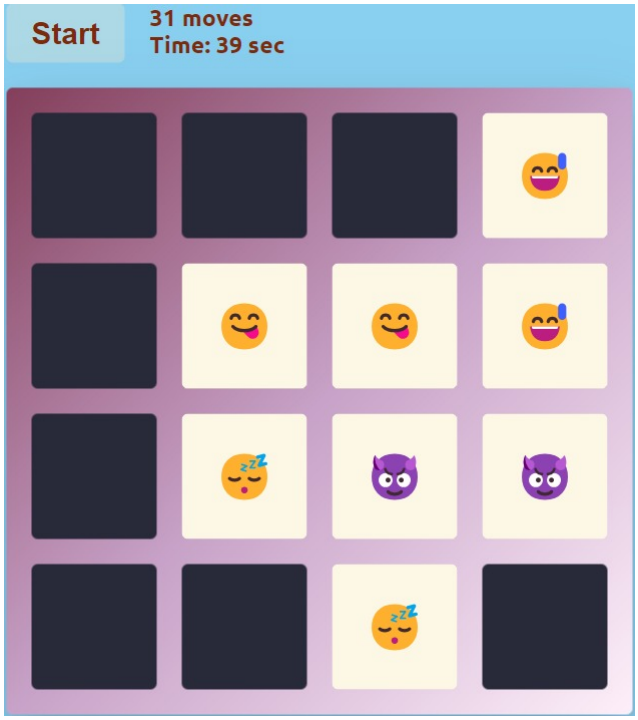

Fig. 5. Matching the emojis

update delivery, improving both the user experience and development process. Building, testing, and deploying code changes are all automated by the pipeline using AWS CodePipeline. Code changes may be easily synchronized through integration with GitHub and then automatically saved in an S3 bucket. By minimizing manual involvement and shortening the deployment time, this configuration guarantees that customers receive updates, bug fixes, and new features in a timely and dependable manner. The continuous deployment pipeline facilitates quick iteration and maintenance by optimizing the deployment process, eventually giving consumers a seamless and uninterrupted gaming experience.

The establishment of a continuous deployment pipeline aimed to optimise the gaming application's maintenance and update delivery processes. Code updates may be seamlessly deployed to the cloud server thanks to the automation of the deployment procedure provided by integration tools like GitHub and AWS CodePipeline. This strategy reduced interruptions and downtime, giving gamers a seamless and continuous gaming experience.

IV. Results

As evidenced by a number of metrics and assessments, the interactive picture matching card game developed with cloud computing technology produced notable improvements in accessibility, scalability, and performance.

A. Scalability and Performance

This cloud-based emoji matching game performs much better and is more scalable due to the incorporation of cloud computing technology. The game can dynamically scale resources to handle different levels of user traffic by utilizing AWS infrastructure, guaranteeing a responsive and seamless experience even during periods of high usage. Game assets are stored in an S3 bucket, which guarantees quick and dependable content delivery by cutting down on latency and speeding up load times. The responsiveness and efficiency of the game are further improved by ongoing monitoring and performance optimization techniques, giving players a smooth and pleasurable gaming experience.

B. Accessibility and Availability

Our emoji matching game is available to users from any location with an internet connection thanks to cloud-based deployment. Geographical boundaries are lessened by the worldwide dispersion of cloud servers, which offers reliable and quick access to users wherever they may be. The gaming interface's responsive design enhances user convenience and engagement by being compatible with a broad range of devices, including PCs, tablets, and smartphones. By using automatic failover procedures and redundant cloud architecture, high availability is accomplished, reducing downtime and guaranteeing that the game can still be accessed even in the case of hardware or network problems.

C. Deployment Efficiency

The deployment process is greatly streamlined by integrating GitHub with AWS CodePipeline to create a continuous deployment pipeline. Code changes propagate to the live environment fast and reliably because of the automated build, test, and deployment stages. By reducing the need for manual intervention, this configuration minimizes mistakes and speeds up the distribution of updates, bug fixes, and new features. The distribution process's speed allows for quick iterations and ongoing game improvement, guaranteeing that customers always get the newest fixes and improvements. Furthermore, the pipeline's automated testing frameworks uphold high standards, which adds even more to a reliable and dependable gaming experience.

V. Conclusion

The creation and implementation of this cloud-based emoji matching game serves as an example of the numerous benefits that come with using cloud computing technology into contemporary online applications. For the fundamental development of the game, we used HTML, JavaScript, and CSS, which allowed us to produce an interesting and eye-catching user interface. We were able to get great performance and scalability by integrating AWS infrastructure, which made it possible for the game to smoothly handle different traffic volumes. The game's responsiveness and latency were much improved, and gamers were given a seamless and pleasurable gameplay experience, thanks to the dynamic resource allocation and effective asset management made possible by AWS S3 and CodePipeline.

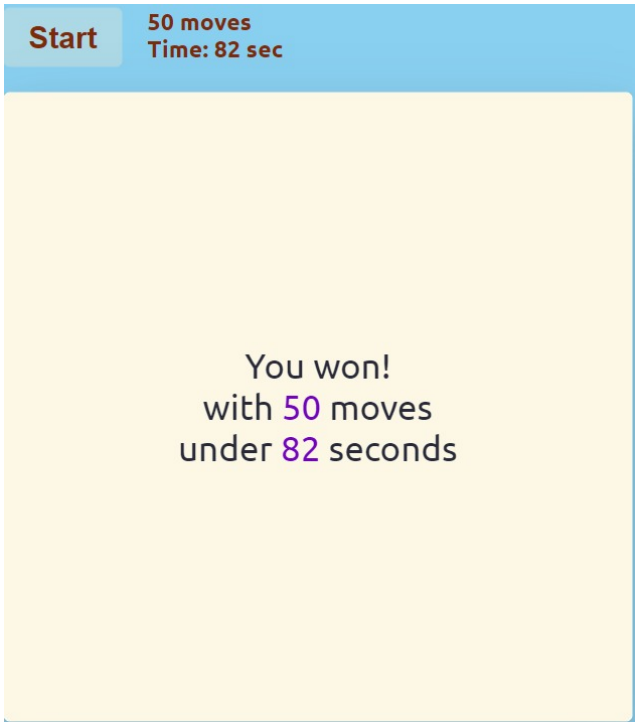

Fig. 6. Moves after completing the matching

Furthermore, our development approach has been completely transformed by the implementation of a continuous deployment pipeline, which allows for quick, dependable updates with no downtime. Through integrated automated testing, this automation not only guarantees that consumers have access to the newest features and enhancements as soon as possible, but it also upholds high standards of quality. Cloud-based hosting has significantly increased the game's availability and accessibility, enabling players to play it from any device, anywhere in the globe. All things considered, this project demonstrates how cloud computing may revolutionize web application development, deployment, and maintenance, offering significant advantages to consumers as well as developers.

REFERENCES

[1]] Srithar, S., et al. "Cost-Effective Integration and Deployment of Enterprise Application Using Azure Cloud Devops." 2022 International Conference on Computer Communication and Informatics (ICCCI). IEEE, 2022.

[2] López-Viana, Ramón, Jessica Díaz, and Jorge E. Pérez. "Continuous Deployment in IoT Edge Computing: A GitOps implementation." 2022 17th Iberian Conference on Information Systems and Technologies (CISTI). IEEE, 2022.

[3] Singh, Neelam, Aman Singh, and Vandana Rawat. "Deploying Jenkins, Ansible and Kubernetes to Automate Continuous Integration and Continuous Deployment Pipeline." 2022 IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI). IEEE, 2022.

[4] Mysari, Sriniketan, and Vaibhav Bejgam. "Continuous Integration and Continuous Deployment Pipeline Automation Using Jenkins Ansible." 2020 International conference on emerging trends in information technology and engineering (IC-ETITE). IEEE, 2020.

[5] Cepuc, Artur, et al. "Implementation of a continuous integration and deployment pipeline for containerized applications in amazon web services using jenkins, ansible and kubernetes." 2020 19th RoEduNet Conference: Networking in Education and Research (RoEduNet). IEEE, 2020.

[6]]Bass, Len, et al. "Securing a deployment pipeline." 2015 IEEE/ACM 3rd International Workshop on Release Engineering. IEEE, 2020.

[7]]Arachchi, S. A. I. B. S., and Indika Perera. "Continuous integration and continuous delivery pipeline automation for agile software project management." 2018 Moratuwa Engineering Research Conference (MERCon). IEEE, 2020.

[8]]Dwivedi, Yogendra Swaroop, Ganesh Semalty, and Amit Moondra. "Predictive Technique To Improve Classification On Continuous System Deployment." 2021 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT). IEEE, 2021.

[9]]Gmeiner, Johannes, Rudolf Ramler, and Julian Haslinger. "Automated testing in the continuous delivery pipeline: A case study of an online company." 2015 IEEE Eighth International Conference on Software Testing, Verification and Validation Workshops (ICSTW). IEEE, 2015.

[10]]Teixeira, Sérgio, Rafael Arrais, and Germano Veiga. "Cloud simulation for continuous integration and deployment in robotics." 2021 IEEE 19th International Conference on Industrial Informatics (INDIN). IEEE, 2021.

[11]]Romero-Alvarez, Javier, et al. "A workflow for the continuous deployment of quantum services." 2023 IEEE International Conference on Software Services Engineering (SSE). IEEE, 2023.

[12] Mowad, Abrar Mohammad, Hamed Fawareh, and Mohammad A. Hassan. "Effect of using continuous integration (ci) and continuous delivery (cd) deployment in devops to reduce the gap between developer and operation." 2022 International Arab Conference on Information Technology (ACIT). IEEE, 2022.

[13] Aydin, Selin, Andreas Steffens, and Horst Lichter. "Automated construction of continuous delivery pipelines from architecture models." 2021 28th Asia-Pacific Software Engineering Conference (APSEC). IEEE, 2021.