Preprints.org

Article

Not peer-reviewed version

Design of Cloud-based Real-Time Eye Tracking Monitoring and Storage System

Mustafa Can Gursesli , Mehmet Emin Selek , Mustafa Oktay Samur , Mirko Duradoni , Kyoungju Park , Andrea Guazzini , Antonio Lanatà *

Posted Date: 22 June 2023

doi: 10.20944/preprints202306.1589.v1

Keywords: data management; cloud computing; RESTful API; eye-tracking; web portal

Preprints.org is a free multidiscipline platform providing preprint service that is dedicated to making early versions of research outputs permanently available and citable. Preprints posted at Preprints.org appear in Web of Science, Crossref, Google Scholar, Scilit, Europe PMC.

Copyright: This is an open access article distributed under the Creative Commons Attribution License which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 June 2023 doi:10.20944/preprints202306.1589.v1

Disclaimer/Publisher's Note: The statements, opinions, and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions, or products referred to in the content.

Article

Design of Cloud-based Real-Time Eye Tracking Monitoring and Storage System

Mustafa Can Gursesli [1,2], Mehmet Emin Selek [3], Mustafa Oktay Samur [4], Mirko Duradoni [2], Kyoungju Park [6], Andrea Guazzini [2,5], Antonio Lanatà [1,*]

1 Department of Information Engineering, University of Florence, 50139 Florence, Italy
2 Department of Education, Literatures, Intercultural Studies, Languages and Psychology, University of Florence, 50135 Florence, Italy
3 Department of Mining Engineering, Istanbul Technical University, Istanbul 34467, Turkey
4 Department of Electrical and Electronics Engineering, Bilgi University, 34060 Istanbul, Turkey
5 Centre for the Study of Complex Dynamics, University of Florence, 50019 Sesto Fiorentino, Italy
6 Department of Computer Science and Engineering, Chung-Ang University, Seoul 06974, Republic of Korea
* Correspondence: antonio.lanata@unifi.it

Abstract: The rapid development of technology has led to the implementation of data-driven systems whose performance heavily relies on the amount and type of the data itself. In the latest decades, in the fields of bioengineering data management, among others, eye-tracking data has become one of the most interesting and essential components for many medical, psychological, and engineering research applications. However, despite the large usage of eye-tracking data in many studies and applications, a strong gap is still present in the literature regarding real-time data collection and management, which led to strong constraints for the reliability and accuracy of on-time results. To address this gap, this study aims to introduce a system that enables the collection, processing, real-time streaming, and storage of eye-tracking data. The system is developed by using Java programming language, WebSocket protocol, and Representational State Transfer (REST), improving the efficiency in transferring and managing eye-tracking data. Results were computed in two test conditions, i.e., local and online scenarios, within a time window of 100 seconds. The experiments conducted for this study were carried out by comparing the time delay between two different scenarios. Even if preliminary, results showed a significantly improved performance of data management systems in managing real-time data transfer. Overall, this system can significantly benefit the research community by providing real-time data transfer and storing the data, enabling more extensive studies using eye-tracking data.

Keywords: data management; cloud computing; RESTful API; eye-tracking; web portal

---

1. Introduction

In recent decades, technology has become a crucial element of human life, leading to various innovative and convenient advancements across numerous fields, including health [1,2], entertainment [3,4], social media [5,6], physics [7,8] and chemistry [9,10]. While these innovations have positively impacted human life, they also demanded several technological requirements. These requirements, including computational power [11], internet access [12], electricity [13], data [14] and other factors [15,16], have become increasingly crucial in both academia and industry sectors. Data and its management, in particular, became the center node for solving these technological challenges, and their relevance has been further increased by the growth of machine learning methods and AI applications [14,17,18]. However, despite the need for huge data, research applications still lack suitable and effective data collection systems. Moreover, since many studies, especially those regarding a large part of the population moved to mobile applications, real-time data became the strongest constraint to solve [19,20]. These conditions have led researchers to focus on improving and creating novel data collection systems to facilitate the technological advances in the research activity. These data collection systems

© 2023 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 June 2023 doi:10.20944/preprints202306.1589.v1

are widely used in many studies as different as brain signals [21], earthquakes [22], weather conditions [23], etc.

In this context, Representational State Transfer (REST), the most widely used Web-based architecture in both academic literature and industry, has been introduced in 2000 as a Ph.D. thesis by Roy Fielding [24] for leading the design and development of the architecture of an Internet-scale distributed hypermedia system. It facilitates caching of components to reduce user-perceived latency, enforce security, and encapsulate legacy systems [24]. REST employs the Hyper Text Transfer Protocol (HTTP) to enable communication between clients and servers. Its structure provides several advantages, including modifiability and statelessness, which enhanced interoperability [25]. These advantages bring substantial benefits to the management of real-time data.

In addition to the solutions and limitations associated with real-time data management, it is widely recognized that the real-time management of diverse data types presents unique challenges due to their high density and rapid flow rates [26]. Particularly eye-tracking data exemplifies this complexity due to its highly dynamic and rapidly changing nature [27–29]. Furthermore, an eye-tracking pattern is an indirect measure of the complex biological system behind it, which requires high-cost computational methods for analysis with models, creating a major problem for the smooth real-time streaming of data.

In this regard, eye tracking can be defined as, is a powerful research tool for studying various topics such as marketing [30], attention [31], perception [32], psychopathology [33], computer vision [34], and decision-making [35,36]. Eye tracking provides insight into the neural mechanisms at the base of exploring strategy of visual stimuli [37]. The eye-tracking technology greatly advanced in recent years, achieving greater precision and accuracy, even in real-world environments [34,38,39]. The history of eye tracking can be traced back to the late 1800s, with improvements in terms of comfort, wearability, and performance for a long time measure of eye movements [36,40,41]. Since then, there have been important advancements that have led to the development of increasingly sophisticated eye-tracking systems [42–44].

Although physical eye-tracking devices have improved and become easier to use, the real-world employment of these devices is still not widespread due to their high cost [42–44]. This practical issue has prompted researchers to find different solutions, and many eye-tracking models using webcams have been developed [34,39,45]. Many of these models are eligible for implementation locally on the user's devices and streamed to different platforms via the internet with the help of web servers. Researchers integrate their models into web platforms to reach larger audiences and collect more data. Unfortunately, there is a gap in the literature regarding web-based streaming and storage systems that can be integrated with real-time eye-tracking models.

This study aims to introduce a system that allows the collection, processing, real-time streaming, and storage of eye-tracking data with REST architecture implementation. The manuscript is structured as follows: In section 1, data necessity, REST, real-time data, and eye-tracking are reported. Section 2 presents Materials and Methods, System Design, Representational State Transfer and Application Programming Interface, WebSocket, Database Server, Docker, WebGazer.js, Hardware Implementation, and Experiment. In Section 3, experimental results are drawn. Section 4 discusses on achieved experimental results compared with those presented in the literature. Lastly, Section 5 provides a conclusion of the entire study and possible future research directions.

2. Materials and Methods

The system's architecture consists of two software modules, i.e., the frontend and the backend. The former is responsible for interfacing with the user, acquiring information, preprocessing, streaming live data, and transferring to the backend. The latter is the invisible part and includes applications, servers, and databases. In our structure, three different interconnected platforms are designed to collect, process, stream, and store eye movements during an experimental session. These platforms are the Experiment Platform, the Real-Time Results Platform, and the Database Management Platform (see

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 June 2023 doi:10.20944/preprints202306.1589.v1

Figure 1). Experiment and Real-Time Results Platforms are located in the frontend whilst the Database Management Platform is located in the backend. Frontend and backend communicate through HTTP protocol. Lastly, data gathered from this study was analyzed by using Python Version 3.10.0 with Matplotlib library version 3.5.3. All details of the system and data flow are reported in the following sessions.

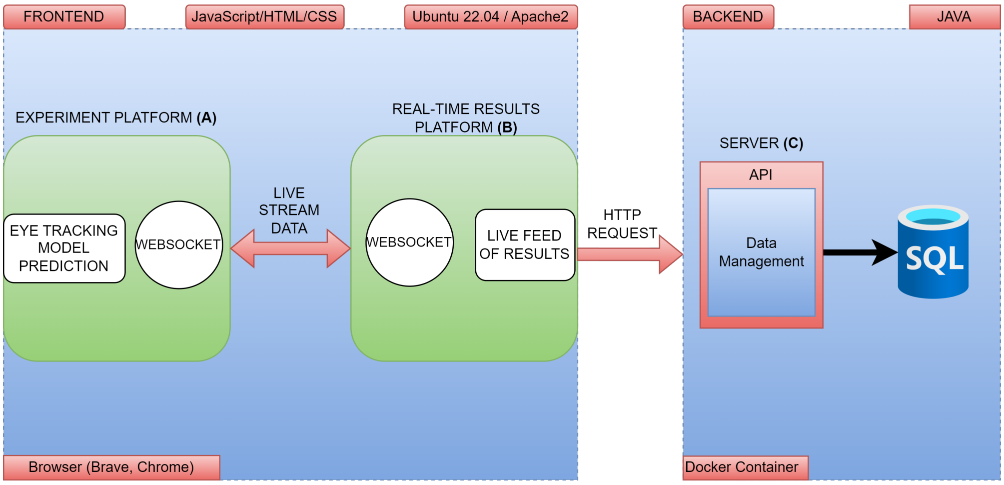

Figure 1. The Tripartite Components of the System Design: Experiment Platform (A), Real-Time Results Platform (B), and Server (C).

2.1. System Design

The system is designed with three components under two main modules (see Figure 1). The experimental platform (A) and the real-time results platform (B) are in the front-end module. In the backend module, there is a Server (C) component. The frontend development uses JavaScript, HyperText Markup Language (HTML), and Cascading Style Sheets (CSS), which allows the creation of a user-friendly interface. This interface displays the live data stream and the evaluation report of the eye-tracking system. To display the participant's live video stream, HTML5 video tags are used.

The frontend is specifically designed to integrate of eye-tracking model and provide a clear presentation of its live results. Moreover, data transfer between the Experiment Platform (A) and Real-time Results Platform (B) is carried out via WebSocket. Subsequently, this data is transmitted to the backend server via an HTTP request and stored in the database.

The backend structure is built using Spring Boot, which is a framework of Java language, providing a robust and scalable data processing and management platform. At the end of data collection on the frontend side, the collected data are sent to the backend service by HTTP requests. The backend service aims to process and manage data for the database storage. In addition, the backend provides a data management API that facilitates read and write operations to the SQL database.

Database management involves the use of a SQL database for the efficient storage and management of data. The database provides scalability and ease of retrieval and analysis of stored data. The backend communicates with the database using Java Database Connectivity (JDBC), a Java API designed to access and manage databases. This seamless integration enables effective data handling within the system. Dockerization plays a major role in encapsulating the various components of the system. It involves separating the frontend, backend, and database into separate Docker containers. Each container can be deployed independently, allowing for easy scalability based on the application's needs. Dockerization also provides a secure and isolated environment for each component, ensuring the stability and security of the overall system.

The system flow can be briefly described as follows: Experiment Platform (A) captures the eye movements and positions of the subject through an embedded model, converts them into coordinates, and sends them to the Real-time Results Platform (B). These two websites communicate to each other

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 June 2023 doi:10.20944/preprints202306.1589.v1

via WebSockets and provide data flow by constantly listening to exchanged messages from A to B and vice-versa. The eye-tracking model placed on the experimental platform initiates data collection and its results are then transmitted to a real-time results platform and streamed to the backend via HTTP requests. Following the end of the data collection session, the eye-tracking data, streamed instantaneously on the Real-time Results Platform (B), is sent via HTTP request to the Server (C), which constitutes the last stage of the data flow in the backend.

2.2. Representational State Transfer (REST) and Application Programming Interface (API)

Representational State Transfer (REST) is designed to develop web services based on precise standards and limitations to grant an expandable and adaptable cross-data transaction over the internet [24]. RESTful API (Application Programming Interface) is an interpretation of the REST architecture that provides access to and actions on resources using HTTP. In a RESTful API, the server doesn't store data about the user between requests; instead, each request has all the data the server needs to process it. RESTful APIs follow a set of constraints, such as client-server architecture, and a consistent interface, among others, to ensure that they are reliable, scalable, and easy to maintain [46,47]. REST has become popular among developers due to its simplicity and flexibility. In addition, RESTful APIs have evolved into a standard for web services development and are actively used by many large companies such as Google, Twitter, etc.

2.3. WebSocket

WebSocket is a communication protocol that belongs to the application layer in the Transmission Control Protocol/Internet Protocol model (TCP/IP) [48]. Due to the popularity and prevalence of HTTP, WebSocket uses HTTP constructs for the initial connection between a client and a server [49] and provides persistent communication so that both the client and the server can send messages at any time. Compared to traditional real-time web communication, the WebSocket protocol saves a lot of network bandwidth and server resources, and the real-time performance is significantly improved [50]. It is helpful for real-time applications such as online games, financial trading platforms, eye tracking, and Internet of Things (IoT) - based applications that support server push technology [51,52].

2.4. Database Server (SQL)

SQL (Structured query language) is a fourth-generation declarative programming language for relational DBMSs (Database management systems) and is a method used to communicate with and manipulate databases [53]. The MySQL database stores and retrieves data via the REST API. The stored procedures and functions are designed as a security layer to perform operations that would receive queries from the API for SQL processing in the database [54].

There are many parameters to consider when evaluating database performance. In particular, NoSQL databases outperform SQL databases regarding write speed and scalability. NoSQL databases perform better when dealing with large scalability requirements and facilitating rapid data updates. [55]. However, SQL databases better manage complex relationships and multiple client scenarios [55]. The characteristics of SQL, structure, and capability to maintain data integrity make them suitable for scenarios involving relational data tables (such as the study carried out). Due to the anticipated availability of multiple user results and relational data in this system, the choice of database system tended

2.5. Docker

Docker is a technology that enables container virtualization, which can be compared to a highly efficient virtual machine due to its lightweight nature [56,57]. It encompasses a modular architecture comprising multiple integral components that interact harmoniously to facilitate the process of "Containerization." At the core of Docker is the Docker Engine, which provides the runtime environment for containers [57]. Docker Images, read-only templates that serve as container building

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 June 2023 doi:10.20944/preprints202306.1589.v1

blocks, utilize a layered file system and copy-on-write mechanism for efficient image management [57]. When a Docker Image is instantiated, it becomes a Docker Container, which offers a lightweight and secure execution environment [57]. Docker Containers can be easily created, started, stopped, and deleted, providing flexibility in managing application instances [58]. To facilitate image sharing and distribution, Docker Registries, such as Docker Hub, host a vast collection of pre-built images [59]. Additionally, organizations can establish private registries tailored to their specific image requirements [59]. The modular architecture of Docker, along with its components, enables scalable and flexible application deployment across various environments.

2.6. WebGazer.js

WebGazer.js is a JavaScript-based eye-tracking algorithm. This algorithm allows real-time display of eye-gaze locations on the web using webcams on notebooks and mobile phones [39,60] This tool aims to utilize eye-tracking systems, which are currently only used in controlled environments and experiments, to enable people to use them in their daily lives [39,60]. WebGazer.js consists of two core elements. These are a pupil detector and a gaze estimator. The pupil detector detects the position of the eye and pupil through the webcam. At the same time, the gaze estimator uses regression analysis to estimate where the individual is looking on the screen [39,60]. The gaze estimator applies a regression analysis through a calibration based on mouse clicks and mouse movements.

2.7. Hardware Implementation

In the system, three Dockers virtual environments were used to perform experiments, stream real-time eye movements, and store these data. In order to carry out online experiments, two separate 1 physical AMD Central Processing Units (CPUs), 1 GB Random Access Memory (RAM), and 25 GB Solid State Disk (SSD) hardware were used for the frontend where the eye-tracking model runs and for live feed eye-tracking data. Furthermore, to store the data and manage the backend, 2 physical Intel CPUs, 2 GB RAM, and 25 GB SSD hardware were used. The locations of servers where Dockers are used are located in Frankfurt, Germany for online experiments. The local experiments were conducted with Intel i5 8600k CPUs and 16 GB RAM. Lastly, in both systems, eye-tracking data is collected in X and Y coordinates, while the data time in seconds is stored in Year:Month:Day:Hour:Minute:Second:Millisecond.

2.8. Experiment

Experimental sessions were carried out to assess the reliability of the proposed system architecture, comparing two different scenarios: Local Implementation (LI) and Online Implementation (OI). The local scenario involved configuring the system on the local computer, while the online scenario consisted of configuring the system on the online server.

In order to measure the time delay of both scenarios (LI and OI), the timestamps of each platform were collected during a 100 seconds time window. The delay was computed as subtracting the timestamp value of the experiment platform (A) from the timestamp received on the real-time results platform (B) (B-A), (i.e., arrival time - starting time). Of note, platform (A) sends data to platform (B) at the frequency of 1HZ (see Figure 1).

Moreover, Console.log() function was used to visualize data in the experiment. Specifically, Console.log() is a function that allows the data given into the function to be seen outside the code environment. This function allowed us to capture the precise timestamps indicating the arrival and starting time of data effectively.

3. Results

This study performed a series of statistical analyses to evaluate the difference between the delays of LI and OI. In order to perform the analyses correctly, firstly, the Shapiro-Wilk test was applied to determine whether the delay data were normally distributed. According to the results of the

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 June 2023 doi:10.20944/preprints202306.1589.v1

Shapiro-Wilk test, both the LI delay data (Shapiro-Wilk test statistic = 0.370, $p < 0.05$) and the OI delay data (Shapiro-Wilk test statistic = 0.322, $p < 0.05$) did not fit a normal distribution. Time difference distributions are shown in Figure 2.

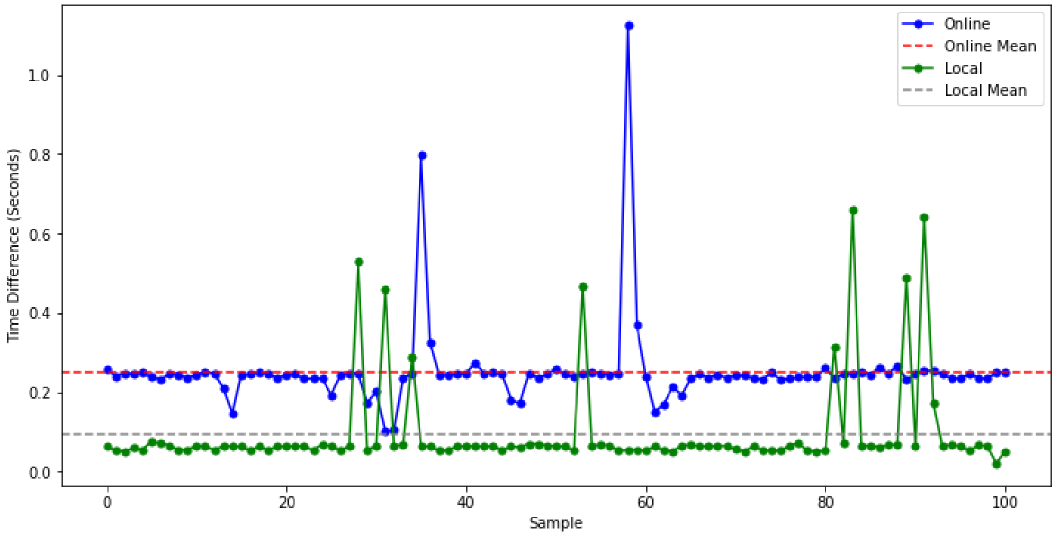

Figure 2. Comparison of time differences between the Online and Local systems.

Based on these results, it was concluded that parametric statistical tests could not be used and the Mann-Whitney U test, a non-parametric test, was preferred. The results of the Mann-Whitney U test showed a statistically significant difference between Local and Online latency (U = 794.0, $p < 0.05$). Table 1 shows Mann-Whitney U test results.

Table 1. Mann-Whitney U Test Results Regarding the Local and Online Conditions

Condition	U-Statistic	P-Value
Local vs Online	794.0	3.317643×10^{-25}

According to Descriptive Statistics, the MAD value for the LI delay was 0.004, the median value was 0.064, the minimum value was 0.020 and the maximum value was 0.660. Similarly, the MAD value for the OI delay was 0.006, the median value was 0.244, the minimum value was 0.101 and the maximum value was 1.123. All results of descriptive statistics are shown in Table 2.

Table 2. Descriptive Statistics

Condition	MAD	Median	Min	Max
Local	0.004	0.064	0.020	0.660
Online	0.006	0.244	0.101	1.123

These findings indicate that there is a statistically significant difference between LI and OI delay and that there is a significant difference in their performance.

4. Discussion

The demand for data has witnessed a substantial increase in recent years due to factors such as rapid technological advancements, growing interest in AI from both the private sector and researchers, and the proliferation of diverse research in the literature [61–64]. However, it is widely acknowledged that data collection systems, expected to keep up with these demands, are falling limited. This study aims to develop a system that facilitates the data collection process for various studies, particularly in the academic domain, while simultaneously enabling real-time observation and streaming of the collected data.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 June 2023 doi:10.20944/preprints202306.1589.v1

Presently, REST is extensively employed in academic research across various fields, including case generation [65], methodologies [66], biological data [67], and machine learning [68], etc. Furthermore, prominent companies like Google, Amazon, Twitter, and Reddit also utilize this architecture. As part of this study, REST enables instantaneous streaming of the collected data. However, to avoid restricting researchers solely to internet-based usage, the system incorporates the Dockerization technique, allowing for local implementation. Consequently, tests were conducted in local and online (server-based) configurations. A significant difference was found between the time it took for the eye-tracking model data to reach the results page in the locally configured system compared to the same system configured online. Numerous performance bottlenecks such as Internet latency [69], computer configuration [70], and server location [71] present considerable challenges that are difficult to mitigate. Although the latency experienced online is significantly higher than that of the local configuration, it is believed that the experimental online latency is not substantial enough for users to discern [72] (the delay values shown in Figure 2).

The system presented in this study, which is based on several techniques, serves the purpose of real-time streaming and storage of eye-tracking data. However, it is crucial to highlight the flexibility of the proposed system that can be adapted for collecting and analyzing other data types in different experimental settings. For instance, physiological data [73,74], mathematical simulations [75,76], AI models for emotion recognition [77,78], and psychological tests [79,80] are among the potential experiments and data that can be included in this system. Simultaneously, the system allows real-time tracking of users' eye movements, enabling streaming over the Internet without being limited to a single task.

Furthermore, future studies would involve a larger number of participants and more extensive data, for a deeper understanding of the system's capabilities and limitations. In addition, the proposed architecture fosters strong collaboration between researchers adopting similar platforms, enabling to an incredibly flexible data exchange and sharing. Moreover, data storage via the Internet is also expected to increase accessibility, thereby encouraging further research and discovery in various fields.

5. Conclusions

This study reported on an approach to data collection and experimentation that demonstrates the intricacies of a multi-purpose system for both online and local applications. This study highlights the fundamental importance of data in scientific endeavors and calls for further exploration of alternative data collection techniques.

Author Contributions: Conceptualization, M.E.S., M.O.S., A.L., A.G., M.C.G., K.P. and M.D.; methodology, A.L., and M.C.G.,; investigation, M.E.S., M.O.S., A.L., M.C.G.; data curation, M.E.S., M.O.S., M.C.G.; writing—original draft preparation, M.C.G., M.O.S., M.E.S. and M.D.; writing—review and editing, A.G., M.C.G., M.O.S., K.P. and A.L.; supervision, A.G., A.L., M.D., and K.P. A. Guazzini and A. Lanata are equally responsible for this study. All authors have read and agreed to the published version of the manuscript.

Funding: Thanks to DigitalOcean (digitalocean.com) and Oliver Mensah for providing us with servers for tests and various other experiments.

References

1. Chaudhry, B.; Wang, J.; Wu, S.; Maglione, M.; Mojica, W.; Roth, E.; Morton, S.C.; Shekelle, P.G. Systematic review: impact of health information technology on quality, efficiency, and costs of medical care. *Annals of internal medicine* **2006**, *144*, 742–752.
2. Buntin, M.B.; Burke, M.F.; Hoaglin, M.C.; Blumenthal, D. The benefits of health information technology: a review of the recent literature shows predominantly positive results. *Health affairs* **2011**, *30*, 464–471.
3. Martucci, A.; Gursesli, M.C.; Duradoni, M.; Guazzini, A. Overviewing Gaming Motivation and Its Associated Psychological and Sociodemographic Variables: A PRISMA Systematic Review. *Human Behavior and Emerging Technologies*, *2023*, e5640258. Publisher: Hindawi, doi:10.1155/2023/5640258.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 June 2023 doi:10.20944/preprints202306.1589.v1

4. Rauterberg, M. Positive Effects of Entertainment Technology on Human Behaviour. Building the Information Society; Jacquart, R., Ed. Springer US, 2004, IFIP International Federation for Information Processing, pp. 51–58. doi:10.1007/978-1-4020-8157-6_8.
5. Duradoni, M.; Spadoni, V.; Gursesli, M.C.; Guazzini, A. Development and Validation of the Need for Online Social Feedback (NfOSF) Scale. *Human Behavior and Emerging Technologies*, *2023*, e5581492. Publisher: Hindawi, doi:10.1155/2023/5581492.
6. Carr, C.T.; Hayes, R.A. Social media: Defining, developing, and divining. *Atlantic journal of communication* **2015**, *23*, 46–65.
7. Kadish, K.M.; Ruoff, R.S. *Fullerenes: Chemistry, Physics, and Technology*; John Wiley & Sons, 2000. Google-Books-ID: SQRugQM4p9QC.
8. Nicollian, E.H.; Brews, J.R. *MOS (Metal Oxide Semiconductor) Physics and Technology*; John Wiley & Sons, 2002. Google-Books-ID: IgUVEAAAQBAJ.
9. Noll, W. *Chemistry and Technology of Silicones*; Elsevier, 2012. Google-Books-ID: 5J3YS3dXA6kC.
10. Whistler, R.L.; BeMiller, J.N.; Paschall, E.F. *Starch: Chemistry and Technology*; Academic Press, 2012. Google-Books-ID: pvAzqk2pAIsC.
11. Hwang, T. Computational power and the social impact of artificial intelligence. *arXiv preprint arXiv:1803.08971* **2018**.
12. Yaqoob, I.; Ahmed, E.; Hashem, I.A.T.; Ahmed, A.I.A.; Gani, A.; Imran, M.; Guizani, M. Internet of things architecture: Recent advances, taxonomy, requirements, and open challenges. *IEEE wireless communications* **2017**, *24*, 10–16.
13. SG Andrae, A. New perspectives on internet electricity use in 2030. *Engineering and Applied Science Letter* **2020**, *3*, 19–31.
14. Williams, P.H.; Margules, C.R.; Hilbert, D.W. Data requirements and data sources for biodiversity priority area selection. *Journal of biosciences*, *27*, 327–338. doi:10.1007/BF02704963.
15. Navajas, J.; Barsakcioglu, D.Y.; Eftekhar, A.; Jackson, A.; Constandinou, T.G.; Quiroga, R.Q. Minimum requirements for accurate and efficient real-time on-chip spike sorting. *Journal of neuroscience methods* **2014**, *230*, 51–64.
16. Chaudhary, N.; Weissman, D.; Whitehead, K.A. mRNA vaccines for infectious diseases: principles, delivery and clinical translation. *Nature reviews Drug discovery* **2021**, *20*, 817–838.
17. Vidgen, B.; Derczynski, L. Directions in abusive language training data, a systematic review: Garbage in, garbage out. *Plos one*, *15*, e0243300. Publisher: Public Library of Science, doi:10.1371/journal.pone.0243300.
18. Raupach, M.R.; Rayner, P.J.; Barrett, D.J.; DeFries, R.S.; Heimann, M.; Ojima, D.S.; Quegan, S.; Schmullius, C.C. Model–data synthesis in terrestrial carbon observation: methods, data requirements and data uncertainty specifications. *Global Change Biology*, *11*, 378–397. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1365-2486.2005.00917.x, doi:10.1111/j.1365-2486.2005.00917.x.
19. Farmer, A.; Gibson, O.; Hayton, P.; Bryden, K.; Dudley, C.; Neil, A.; Tarassenko, L. A real-time, mobile phone-based telemedicine system to support young adults with type 1 diabetes. *Informatics in primary care* **2005**, *13*.
20. Gradl, S.; Kugler, P.; Lohmüller, C.; Eskofier, B. Real-time ECG monitoring and arrhythmia detection using Android-based mobile devices. 2012 annual international conference of the IEEE engineering in medicine and biology society. IEEE, 2012, pp. 2452–2455.
21. Teplan, M.; others. Fundamentals of EEG measurement. *Measurement science review* **2002**, *2*, 1–11.
22. Boore, D.M.; Smith, C.E. Analysis of earthquake recordings obtained from the Seafloor Earthquake Measurement System (SEMS) instruments deployed off the coast of southern California. *Bulletin of the Seismological Society of America* **1999**, *89*, 260–274.
23. Xue, M.; Wang, D.; Gao, J.; Brewster, K.; Droegemeier, K.K. The Advanced Regional Prediction System (ARPS), storm-scale numerical weather prediction and data assimilation. *Meteorology & Atmospheric Physics* **2003**, *82*.
24. Fielding, R.T. *Architectural styles and the design of network-based software architectures*; University of California, Irvine, 2000.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 June 2023 doi:10.20944/preprints202306.1589.v1

25. Costa, B.; Pires, P.F.; Delicato, F.C.; Merson, P. Evaluating a Representational State Transfer (REST) architecture: What is the impact of REST in my architecture? 2014 IEEE/IFIP Conference on Software Architecture. IEEE, 2014, pp. 105–114.
26. Cho, G.Y.; Lee, S.J.; Lee, T.R. An optimized compression algorithm for real-time ECG data transmission in wireless network of medical information systems. *Journal of medical systems* **2015**, *39*, 1–8.
27. Kroner, A.; Senden, M.; Driessens, K.; Goebel, R. Contextual encoder–decoder network for visual saliency prediction. *Neural Networks* **2020**, *129*, 261–270.
28. Skaramagkas, V.; Giannakakis, G.; Ktistakis, E.; Manousos, D.; Karatzanis, I.; Tachos, N.S.; Tripoliti, E.; Marias, K.; Fotiadis, D.I.; Tsiknakis, M. Review of eye tracking metrics involved in emotional and cognitive processes. *IEEE Reviews in Biomedical Engineering* **2021**, *16*, 260–277.
29. Black, M.H.; Chen, N.T.; Iyer, K.K.; Lipp, O.V.; Bölte, S.; Falkmer, M.; Tan, T.; Girdler, S. Mechanisms of facial emotion recognition in autism spectrum disorders: Insights from eye tracking and electroencephalography. *Neuroscience & Biobehavioral Reviews* **2017**, *80*, 488–515.
30. Wedel, M.; Pieters, R. A review of eye-tracking research in marketing. *Review of marketing research* **2017**, pp. 123–147.
31. Srivastava, N.; Nawaz, S.; Newn, J.; Lodge, J.; Velloso, E.; M. Erfani, S.; Gasevic, D.; Bailey, J. Are you with me? Measurement of Learners' Video-Watching Attention with Eye Tracking. LAK21: 11th International Learning Analytics and Knowledge Conference. Association for Computing Machinery, 2021, LAK21, pp. 88–98. doi:10.1145/3448139.3448148.
32. Borys, M.; Plechawska-Wójcik, M. Eye-tracking metrics in perception and visual attention research. *EJMT*.
33. Iacono, W.G.; Lykken, D.T. Eye Tracking and Psychopathology: New Procedures Applied to a Sample of Normal Monozygotic Twins. *Archives of General Psychiatry*, *36*, 1361–1369. doi:10.1001/archpsyc.1979.01780120091011.
34. Krafka, K.; Khosla, A.; Kellnhofer, P.; Kannan, H.; Bhandarkar, S.; Matusik, W.; Torralba, A. Eye Tracking for Everyone. Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 2176–2184.
35. Fiedler, S.; Glöckner, A. The Dynamics of Decision Making in Risky Choice: An Eye-Tracking Analysis. *Frontiers in psychology*, *3*.
36. Holmqvist, K.; Nyström, M.; Andersson, R.; Dewhurst, R.; Jarodzka, H.; Weijer, J.v.d. *Eye Tracking: A comprehensive guide to methods and measures*; OUP Oxford, 2011. Google-Books-ID: 5rIDPV1EoLUC.
37. Pfeiffer, U.J.; Vogeley, K.; Schilbach, L. From gaze cueing to dual eye-tracking: Novel approaches to investigate the neural correlates of gaze in social interaction. *Neuroscience & Biobehavioral Reviews*, *37*, 2516–2528. doi:10.1016/j.neubiorev.2013.07.017.
38. Duchowski, A.T. *Eye Tracking Methodology: Theory and Practice*; Springer-Verlag, 2017.
39. Papoutsaki, A.; Laskey, J.; Huang, J. *SearchGazer: Webcam Eye Tracking for Remote Studies of Web Search*; Proceedings of the 2017 conference on conference human information interaction and retrieval, 2017. Pages: 26, doi:10.1145/3020165.3020170.
40. Aslin, R.N.; McMurray, B. Automated Corneal-Reflection Eye Tracking in Infancy: Methodological Developments and Applications to Cognition. *Infancy*, *6*, 155–163. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1207/s15327078in0602_1, doi:10.1207/s15327078in0602_1.
41. Marino, J. Reading Screens: What Eye Tracking Tells Us about the Writing in Digital Longform Journalism. *Literary Journalism Studies* **2016**, *8*.
42. Niehorster, D.C.; Hessels, R.S.; Benjamins, J.S. GlassesViewer: Open-source software for viewing and analyzing data from the Tobii Pro Glasses 2 eye tracker. *Behavior Research Methods*, *52*, 1244–1253. doi:10.3758/s13428-019-01314-1.
43. Kortman, B.; Nicholls, K. Assessing for Unilateral Spatial Neglect Using Eye-Tracking Glasses: A Feasibility Study. *Occupational Therapy in Health Care*, *30*, 344–355. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/07380577.2016.1208858, doi:10.1080/07380577.2016.1208858.
44. Mele, M.L.; Federici, S. Gaze and eye-tracking solutions for psychological research. *Cognitive processing*, *13*, 261–265. doi:10.1007/s10339-012-0499-z.
45. Lu, F.; Sugano, Y.; Okabe, T.; Sato, Y. Head pose-free appearance-based gaze sensing via eye image synthesis. Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012), 2012, pp. 1008–1011. ISSN: 1051-4651.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 June 2023 doi:10.20944/preprints202306.1589.v1

46. Wang, S.; Keivanloo, I.; Zou, Y. How do developers react to restful api evolution? Service-Oriented Computing: 12th International Conference, ICSOC 2014, Paris, France, November 3-6, 2014. Proceedings 12. Springer, 2014, pp. 245–259.
47. Richardson, L.; Ruby, S. *RESTful Web Services*; "O'Reilly Media, Inc.", 2008. Google-Books-ID: XUaErakHsoAC.
48. Berners-Lee, T.J. Information management: A proposal. Technical report, CERN, 1989.
49. Cassetti, O. Websockets and their integration in enterprise networks. *Citeseer* **2011**.
50. Hu, Y.; Cheng, W. Research and implementation of campus information push system based on WebSocket. 2017 12th International Conference on Intelligent Systems and Knowledge Engineering (ISKE), 2017, pp. 1–6. doi:10.1109/ISKE.2017.8258720.
51. Soewito, B.; Christian.; Gunawan, F.E.; Diana.; Kusuma, I.G.P. Websocket to Support Real Time Smart Home Applications. *Procedia Computer Science*, *157*, 560–566. doi:10.1016/j.procs.2019.09.014.
52. Hale, M. Eyestream: An Open WebSocket-based Middleware for Serializing and Streaming Eye Tracker Event Data from Gazepoint GP3 HD Research Hardware. *Journal of Open Source Software*, *4*, 1620. doi:10.21105/joss.01620.
53. Codd, E.F. A Relational Model of Data for Large Shared Data Banks. *Communications of the ACM*, *13*.
54. Kern, C.; Kesavan, A.; Daswani, N. *Foundations of security: what every programmer needs to know*; Springer, 2007.
55. Khan, W.; Kumar, T.; Zhang, C.; Raj, K.; Roy, A.M.; Luo, B. SQL and NoSQL Database Software Architecture Performance Analysis and Assessments—A Systematic Literature Review. *Big Data and Cognitive Computing* **2023**, *7*, 97.
56. Anderson, C. Docker [Software engineering]. *IEEE Software* **2015**, *32*, 102–c3. doi:10.1109/MS.2015.62.
57. Martin, J.P.; Kandasamy, A.; Chandrasekaran, K. Exploring the support for high performance applications in the container runtime environment. *Human-centric Computing and Information Sciences* **2018**, *8*, 1–15.
58. De Benedictis, M.; Lioy, A. Integrity verification of Docker containers for a lightweight cloud environment. *Future generation computer systems* **2019**, *97*, 236–246.
59. Chamoli, S. Docker Security: Architecture, Threat Model, and Best Practices. Soft Computing: Theories and Applications: Proceedings of SoCTA 2020, Volume 2. Springer, 2021, pp. 253–263.
60. Slim, M.S.; Hartsuiker, R.J. Moving visual world experiments online? A web-based replication of Dijkgraaf, Hartsuiker, and Duyck (2017) using PCIbex and WebGazer.js. *Behavior Research Methods*. doi:10.3758/s13428-022-01989-z.
61. Das, S.; Nayak, G.K.; Saba, L.; Kalra, M.; Suri, J.S.; Saxena, S. An artificial intelligence framework and its bias for brain tumor segmentation: A narrative review. *Computers in Biology and Medicine* **2022**, p. 105273.
62. Wilson, C. Public engagement and AI: A values analysis of national strategies. *Government Information Quarterly* **2022**, *39*, 101652.
63. Lee, J.C.; Chen, X. Exploring users' adoption intentions in the evolution of artificial intelligence mobile banking applications: the intelligent and anthropomorphic perspectives. *International Journal of Bank Marketing* **2022**.
64. Dogan, M.E.; Goru Dogan, T.; Bozkurt, A. The use of artificial intelligence (AI) in online learning and distance education processes: A systematic review of empirical studies. *Applied Sciences* **2023**, *13*, 3056.
65. Arcuri, A. RESTful API automated test case generation. 2017 IEEE International Conference on Software Quality, Reliability and Security (QRS). IEEE, 2017, pp. 9–20.
66. Ehsan, A.; Abuhaliqa, M.A.M.; Catal, C.; Mishra, D. RESTful API testing methodologies: Rationale, challenges, and solution directions. *Applied Sciences* **2022**, *12*, 4369.
67. Miller, M.A.; Schwartz, T.; Pickett, B.E.; He, S.; Klem, E.B.; Scheuermann, R.H.; Passarotti, M.; Kaufman, S.; O'Leary, M.A. A RESTful API for access to phylogenetic tools via the CIPRES science gateway. *Evolutionary Bioinformatics* **2015**, *11*, EBO–S21501.
68. Gossett, E.; Toher, C.; Oses, C.; Isayev, O.; Legrain, F.; Rose, F.; Zurek, E.; Carrete, J.; Mingo, N.; Tropsha, A.; others. AFLOW-ML: A RESTful API for machine-learning predictions of materials properties. *Computational Materials Science* **2018**, *152*, 134–145.
69. Briscoe, B.; Brunstrom, A.; Petlund, A.; Hayes, D.; Ros, D.; Tsang, J.; Gjessing, S.; Fairhurst, G.; Griwodz, C.; Welzl, M. Reducing internet latency: A survey of techniques and their merits. *IEEE Communications Surveys & Tutorials* **2014**, *18*, 2149–2196.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 June 2023 doi:10.20944/preprints202306.1589.v1

70. Henning, J.L. SPEC CPU2000: Measuring CPU performance in the new millennium. *Computer* **2000**, *33*, 28–35.
71. Charyyev, B.; Arslan, E.; Gunes, M.H. Latency comparison of cloud datacenters and edge servers. GLOBECOM 2020-2020 IEEE Global Communications Conference. IEEE, 2020, pp. 1–6.
72. Stetson, C.; Cui, X.; Montague, P.R.; Eagleman, D.M. Motor-sensory recalibration leads to an illusory reversal of action and sensation. *Neuron* **2006**, *51*, 651–659.
73. Shu, Y.S.; Chen, Z.X.; Lin, Y.H.; Wu, S.H.; Huang, W.H.; Chiou, A.Y.C.; Huang, C.Y.; Hsieh, H.Y.; Liao, F.W.; Zou, T.F.; others. 26.1 A 4.5 mm 2 Multimodal Biosensing SoC for PPG, ECG, BIOZ and GSR Acquisition in Consumer Wearable Devices. 2020 IEEE International Solid-State Circuits Conference-(ISSCC). IEEE, 2020, pp. 400–402.
74. Soufineyestani, M.; Dowling, D.; Khan, A. Electroencephalography (EEG) technology applications and available devices. *Applied Sciences* **2020**, *10*, 7453.
75. Shakhmurov, V.B.; Kurulay, M.; Sahmurova, A.; Gursesli, M.C.; Lanata, A. Interaction of Virus in Cancer Patients: A Theoretical Dynamic Model. *Bioengineering* **2023**, *10*, 224.
76. Graw, F.; Perelson, A.S. Modeling viral spread. *Annual review of virology* **2016**, *3*, 555–572.
77. Khaireddin, Y.; Chen, Z. Facial emotion recognition: State of the art performance on FER2013. *arXiv preprint arXiv:2105.03588* **2021**.
78. Koolagudi, S.G.; Rao, K.S. Emotion recognition from speech: a review. *International journal of speech technology* **2012**, *15*, 99–117.
79. Li, X.; Liu, Y.; Mao, J.; He, Z.; Zhang, M.; Ma, S. Understanding reading attention distribution during relevance judgement. Proceedings of the 27th ACM international conference on information and knowledge management, 2018, pp. 733–742.
80. Cox, W.M.; Fadardi, J.S.; Pothos, E.M. The addiction-stroop test: Theoretical considerations and procedural recommendations. *Psychological bulletin* **2006**, *132*, 443.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.