electronics

MDPI

Article

A Recommendation System Based on a Microservice Architecture to Avoid Workplace Stress

Fátima Rodrigues [1,2,*], Francisco Pinelas [1], Simão Ferreira [3], Matilde Rodrigues [3] and Nuno Rocha [3]

[1] Institute of Engineering, Polytechnic of Porto, 4249-015 Porto, Portugal
[2] INESC TEC, Institute for Systems and Computer Engineering, 4200-465 Porto, Portugal
[3] TBIO, School of Health, Polytechnic Institute of Porto, 4200-465 Porto, Portugal
* Correspondence: mfc@isep.ipp.pt

Abstract: Stress in the workplace is a major problem that affects people of all ages, backgrounds, and occupations. It can contribute to various health problems, from anxiety to insomnia, among others. Workplace stress significantly impacts employee well-being and productivity. Current stress-management approaches, while valuable, primarily address stress after it has occurred. This highlights the critical need for proactive systems capable of anticipating individual stress and preventing negative health consequences. This research presents the design and initial implementation of a novel microservice-based recommendation system for proactively mitigating workplace stress among computer users. The system leverages predicted stress levels to deliver timely, personalized, and easily implemented interventions. This study focuses on evaluating the system's architecture, core functionalities, and initial performance using a content-based filtering approach. A pilot study demonstrated the system's feasibility, highlighting areas for future development.

Keywords: recommendation system; content-based filtering; microservice architecture; rule management system

Academic Editors: Shiho Kim and Yagiz Nalcakan

Received: 24 February 2025
Revised: 26 March 2025
Accepted: 1 April 2025
Published: 3 April 2025

Citation: Rodrigues, F.; Pinelas, F.; Ferreira, S.; Rodrigues, M.; Rocha, N. A Recommendation System Based on a Microservice Architecture to Avoid Workplace Stress. *Electronics* **2025**, *14*, 1446. https://doi.org/10.3390/electronics14071446

1. Introduction

Prolonged computer use is linked to significant increases in workplace stress, impacting productivity and employee well-being [1,2], resulting in lost productivity and increased healthcare expenses [3]. This paper addresses the critical need for intelligent, proactive stress-management systems specifically designed for this growing population of computer-based workers.

The substantial economic costs associated with work-related stress underscore the critical need for proactive interventions [3]. While existing stress-management techniques offer some benefits, they often lack the ability to anticipate and address stress before it escalates [4–7], particularly for individuals engaged in prolonged computer-based work.

This research addresses this critical need by presenting the design and initial implementation of a novel microservice-based recommendation system for proactive stress management among computer users. This system leverages a previously developed, independent machine learning stress detection module [8] (details of which are beyond the scope of this paper) to deliver timely and personalised interventions. The system's unique microservice architecture promotes scalability, flexibility, and maintainability. The stress detection module uses a non-intrusive approach, collecting physiological data via a webcam to capture HRV, PERCLOS, and facial expressions. This method avoids potential biases from self-reporting and offers greater suitability for unobtrusive monitoring in typical office environments.

This research focuses on the design, implementation, and initial evaluation of a novel microservice-based recommendation system for proactive stress management among computer users. The primary objectives are to (1) develop a functional recommendation system with a robust architecture, (2) evaluate the system's core functionalities and initial performance using a content-based filtering approach, and (3) identify key areas for future development and improvement.

The results presented are from a limited-scale pilot study, highlighting the need for a larger-scale evaluation to fully assess the system's effectiveness. The system itself functions as a web application operating synchronously and asynchronously within a microservice-oriented architecture to ensure flexibility and scalability. It uses predicted stress levels to suggest interventions, adapting to users' feedback and individual profiles. A content-based filtering approach is employed to personalise recommendations.

This paper organises the remaining sections as follows: The second section starts by presenting several filtering techniques typically employed in recommendation systems, along with some recommendation systems designed to alleviate stress. In the next section, we define the problem in relation to our particular recommendation system. The fourth section presents the architectural solution. The solution's implementation is covered in the fifth section. The next section evaluates the recommendation system based on a pilot solution with 22 volunteers, succeeded by the discussion section, while the final section culminates with the achieved goals and future work.

2. Background

Recommendation systems are software applications that utilise specialised algorithms to provide suggestions to users within a given context, aiming to offer options that are likely to be of interest or relevance to the user [9]. The main approaches for recommendation systems include content-based filtering, collaborative filtering, and knowledge-based filtering. A content-based recommendation system is based on items similar to those a given user has liked in the past. A collaborative filter makes recommendations based on items owned by users whose tastes are similar to those of the given user. Knowledge-based filtering models the user profile to identify the correlation between their preferences and existing products, services, or content through inference algorithms [10]. Often, one type of filtering classifies a recommendation system. However, there are cases where recommendation systems benefit the most from using more than one filtering technique, resulting in hybrid recommendation systems.

Recommendation systems for stress detection in the workplace represent a specialised application of machine learning technologies aimed at improving employee well-being by identifying stress levels and suggesting interventions. Although there have been many studies on the use of recommendation systems in health-related solutions, stress-specific recommendation systems, especially in the workplace context, are still relatively new and not as common as more general wellness programmes or generic stress-management tools. The article [11] provides a thorough and well-documented analysis of health recommendation systems based on 73 published studies. Out of these studies, only two specifically focus on recommendation systems related to stress. One of the works to alleviate stress includes the recommendation of books to read [12] and other meditative audiobooks [13].

Various studies have explored the use of mobile recommendation systems for stress management. The authors in the work [14] investigate the use of smartphones as a widely accepted method of coping with stress. The solution employs a microintervention writing process and a machine learning-based recommendation engine to adapt widely used web applications to help with stress management. Following a four-week duration, participants conveyed an increased level of consciousness about their stress, a reduction in symptoms

of depression, and acquired novel techniques to manage stress. The recommendations generated by the machine learning system also resulted in the adoption of more effective coping strategies. Clarke and colleagues in [15] describe a smartphone application that uses a smart band to detect and assess stress levels, providing recommendations for appropriate actions. The recommendation module employs a Q-learning algorithm to determine the optimal combination of interventions. After conducting 1000 episodes of testing, the algorithm successfully decreased the frequency of interventions to only one. In the paper [16], the authors proposed an innovative approach to decompose tensors from many perspectives, which can be used to model stress and user behaviour using diverse data. This approach enables the tracking of customised stress levels and the modelling of user behaviour over time. It can simultaneously model stress and behaviour using numerous sources of data. In another study [17], they develop a student system that uses smartphones to forecast stress levels and suggest suitable partners. These studies jointly emphasise the potential of mobile recommendation systems to deliver tailored and efficient stress-management support.

Cvetkovic and colleagues [18] devised a method to supervise the handling of physical, mental, and environmental stress. The system gathers data, performs analysis, and offers recommendations to the user via a smartphone app, utilising wearable sensors and environmental data. The system comprises three modules: one for measuring physical activity, one for tracking mental stress, and a third for monitoring ambient quality. Each module uses machine learning algorithms to assess sensor data that is particular to its designated duty and incorporates expert knowledge to provide recommendations targeted at improving the user's physical, mental, and environmental well-being. The assessment shows that the system can accurately detect sedentary, stressful, and unhealthy environments. The system primarily provides personalised recommendations for older people who have sedentary, demanding jobs in an office setting and are at greater risk due to generally less robust health.

In another study [19], the authors examined past trip data to quantify driving stress and its influence on various driving habits. A Long Short-Term Memory Fully Convolutional Network (LSTM-FCN) was employed to forecast the corresponding stress level of the driver. Subsequently, they found a correlation between stress and driving behaviour and devised an intelligent suggestion system for cab firms to choose drivers for future trips. The created model was evaluated on the UAH-DriveSet dataset, achieving 95% accuracy in predicting stress levels and improving driving quality and enjoyment.

The objective of Tong et al.'s research [20] was to improve the support available to online workers who typically face a lack of adequate assistance from therapists and learning materials in managing their stress levels. Home Sweet Office (HSO) is a browser-based application that provides stress micro-interventions. These stress micro-interventions are derived from various therapeutic approaches, including somatic, positive psychology, metacognitive, and cognitive-behavioural techniques (including CBT). While their study incorporated CBT, our recommendation system currently includes a set of behavioural interventions categorised as somatic, positive psychology, metacognitive, and cognitive-behavioural approaches. During a four-week field study, researchers conducted a comparison of random interventions, machine-recommended interventions, and self-proposed interventions by participants. The aim was to investigate the effectiveness of different content and suggestion approaches. The study's primary result indicated that both machine-recommended and self-proposed interventions led to significantly better outcomes than random selection. Furthermore, the machine-recommended approach offered access to a wider variety of therapeutic activities compared to self-selected options.

Existing stress-management systems, while offering valuable insights, often suffer from limitations that our system addresses. For example, many mobile-based approaches [14,15] lack sufficient workplace integration and may not capture workplace-specific stressors, while systems focused on specific demographics [18,19] lack generalisability. Even systems with diverse interventions [20] may lack real-time stress detection.

Our microservice-based system overcomes these limitations by (1) seamlessly integrating into computer-based workflows, (2) incorporating real-time stress detection for timely interventions, (3) providing personalised recommendations through content-based filtering and user feedback, (4) offering scalability through its microservice architecture, and (5) employing a holistic approach combining stress detection, personalised recommendations, and user feedback.

Integrating real-time stress detection with a recommendation system presents several significant challenges:

- Data Synchronisation: Timely communication of stress-level data to the recommendation system is critical for generating timely interventions. Our microservice architecture utilises asynchronous communication to ensure efficient data flow.
- Latency Management: Minimising delays between stress detection and recommendation delivery is crucial for maintaining intervention relevance. This is addressed by careful optimisation of inter-service communication and intelligent task scheduling.
- Contextual Awareness: Recommendations must avoid disrupting critical work activities such as meetings or addressing urgent tasks. Our rule management system incorporates contextual factors (workload, meeting schedules, deadlines) to guide appropriate recommendations.
- Privacy Concerns: Balancing accurate stress detection with employee privacy requires careful consideration, especially in workplace settings. We implement robust data anonymisation and access control mechanisms to protect user privacy.
- Adaptive Learning: Continuously refining recommendations based on real-time data and user feedback is essential to maximise intervention efficacy. Our system uses a content-based filtering approach enhanced with user feedback for continuous improvement.

In summary, while existing stress-management solutions offer valuable insights, they often lack the proactive, personalised, and seamlessly integrated approach offered by our novel microservice-based recommendation system. By addressing critical challenges in real-time stress detection, data management, and user engagement, our system provides a more comprehensive and effective solution for proactive workplace stress management among computer-based workers.

3. Problem Definition

This recommendation system addresses the multifaceted challenges inherent in managing workplace stress for employees engaged in computer-based work. Computer workers face numerous specific stressors: prolonged sedentary behaviour leading to physical health issues, screen-induced cognitive loads causing mental fatigue, and limited opportunities for physical activity during the workday. Furthermore, collecting sensitive data while respecting employee privacy necessitates a non-intrusive approach. Personalising recommendations and managing the dynamic nature of stress require adaptive algorithms. Finally, ensuring sustained user engagement and effectively handling new users with minimal data present additional hurdles.

To overcome these challenges, our system employs several key strategies: personalised interventions based on detailed user profiles and exercise characteristics; a hybrid approach to address data sparsity in new users, gradually shifting from generalised rec-

ommendations to a fully personalised content-based approach as user data accumulates; an integrated feedback mechanism to continually refine recommendations; and a robust architecture designed to ensure privacy, maintain engagement, and adapt to the dynamic nature of stress.

Finally, ensuring sustained user engagement and effectively handling new users with minimal data present additional hurdles. To address this, the system is designed with a focus on minimising user burden, a key factor in successful technology adoption [21] and recommendation system effectiveness [22]. Beyond minimising effort, a good recommendation system should also exhibit qualities such as relevance, providing suggestions that align with the user's current needs, and diversity, exposing users to a range of options beyond their immediate interests [23]. The system also aims for accuracy in predicting user preferences and novelty in its recommendations, increasing user engagement and satisfaction. This approach provides tailored stress-management support within the constraints of a typical office environment, aiming to deliver relevant, diverse, accurate, and novel recommendations with minimal user effort.

4. System Architecture

The recommendation system employs a microservice architecture implemented in Java to optimise scalability, flexibility, and maintainability. It is composed of six main components: a graphical user interface (GUI), a core service, a rule service, a recommender service, a stress detection service, and a message broker. Figure 1 illustrates the system's microservice architecture. This diagram combines elements of UML component diagrams with standard system architecture diagramming conventions to clearly represent the system's structure, components, and data flow. Microservices are depicted as rectangular boxes, with arrows indicating data flow or communication. Labels on the arrows specify the communication type (e.g., "stress level", "recommendation"). The diagram emphasises inter-service interactions and dependencies rather than low-level implementation details.

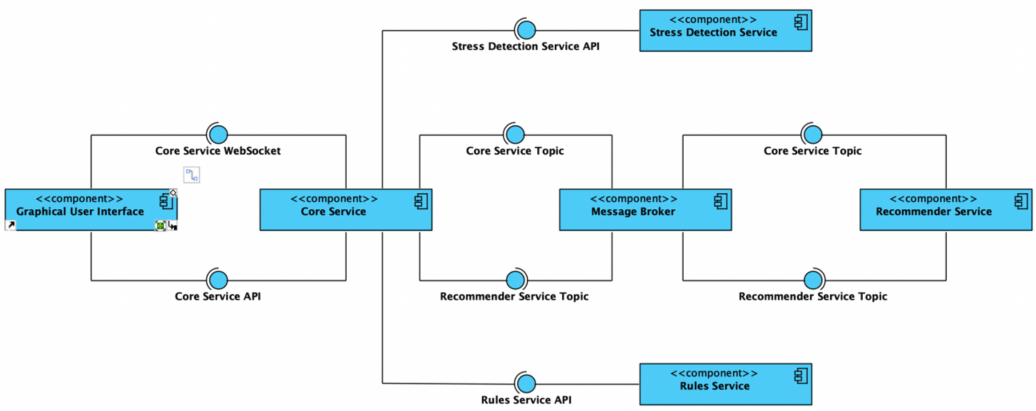

Figure 1. Microservices oriented architecture.

This architecture leverages several key technologies to enhance scalability, flexibility, and resilience:

- Service Discovery: Consul enables dynamic service discovery, eliminating the need for hardcoded endpoints and facilitating loose coupling between services.
- Load Balancing: Spring Cloud Load Balancer distributes incoming requests across multiple service instances, optimizing resource utilization and enhancing system availability.
- Fault Tolerance: Hystrix circuit breakers prevent cascading failures by providing fallback mechanisms when services become unavailable.

- Asynchronous Communication: Kafka facilitates efficient and near real-time data exchange between the stress detection and recommendation services, preventing performance bottlenecks. This asynchronous design ensures that fluctuations in one service do not directly impact the other.
- Data Flow Optimization: Lightweight JSON messages streamline data transfer between services, minimizing overhead and ensuring efficient processing.

The stress detection service publishes stress events to a Kafka topic, and the recommendation service consumes these events to generate personalized interventions. This approach enables efficient, near real-time responses to stress while maintaining system stability and scalability.

5. Implementation

This section presents and describes the process involved in the implementation of the components of the recommendation system architecture. The components that require implementation as part of the solution are the GUI, the core service, the rule service, and the recommender service. Although not implemented in the same way as the rest of the components, the message broker needed integration with some solution components and is described.

5.1. Graphical User Interface

A web-based application's usability, visual consistency, and aesthetics are crucial. For that reason, the user interface design resulted from collaboration with the stakeholders, who shared their preferences and intentions for the application regarding its visual experience. We used the interface design tool Figma [24] to further facilitate this collaboration. This tool is a web-based application that allows multiple people to work together at the same time around the graphical design of a solution or product. Furthermore, it provides cascading style sheets (CSS) directly from the user interface, meaning that most of the time, it is possible to obtain the right CSS for the interface components when developing the graphical user interface.

The user interface is designed to guide users through a seamless experience, from initial registration and profile creation to receiving personalized stress-reduction recommendations. The portal consists of several key pages: a sign-in/sign-up page, a profile form for collecting user information, a recommendation history page displaying past suggestions, and detailed recommendation cards providing access to resources and feedback mechanisms. Figure 2 illustrates the sign page when accessing the application.

Through this page, users may either sign in or navigate to the sign-up page. Upon successful log-in, the application redirects users to a page that highlights the importance of collecting their personal data, enabling them to make relevant recommendations. On this page, users can either choose to navigate to the sign page or to the form page. By choosing to navigate to the form page, users are asked to answer a wide range of questions about their well-being, such as questions about eating and sleeping habits, physical activity, social interaction, and past health-related conditions. As shown in Figure 3, users can select one of the many options available for each question.

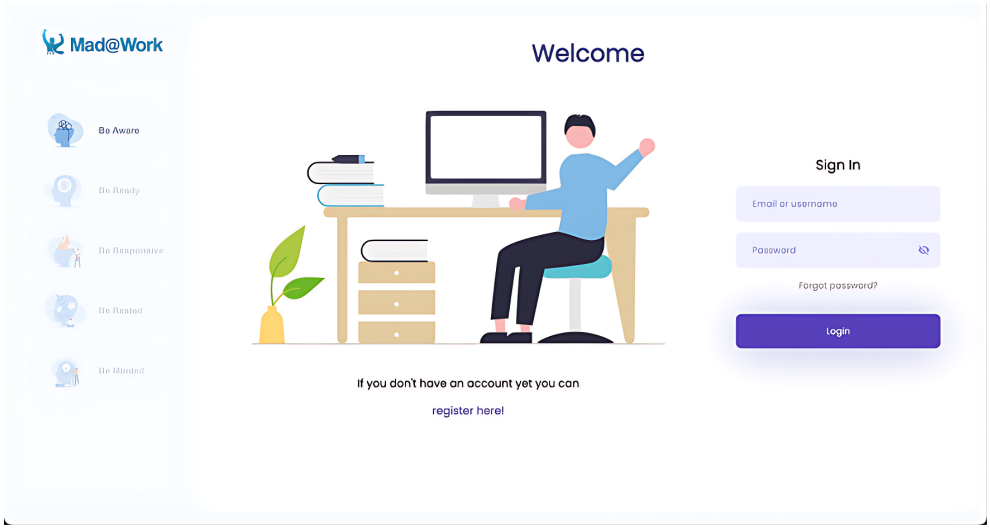

Figure 2. Sign-in page.

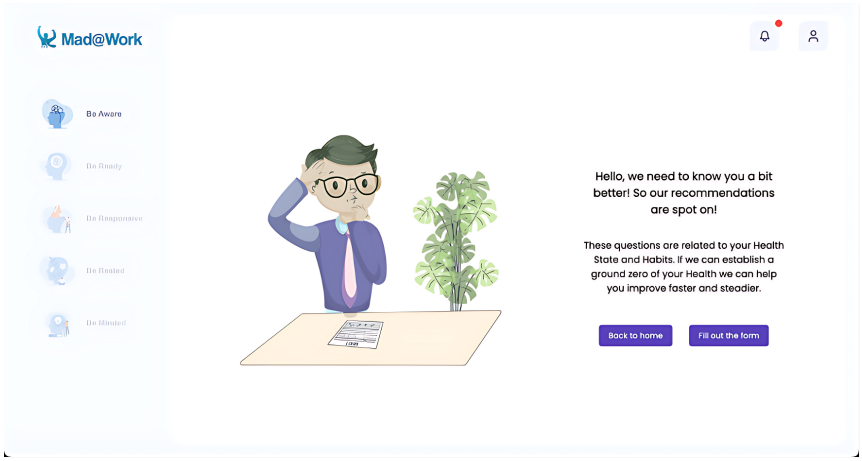

Figure 3. Form page.

The backend saves this information each time users interact with the form by selecting one of the answers. In this way, if users choose to close the web application or their authentication token expires, the application can retrieve and use the information related to the questions that users previously answered.

The home page for a web-based application that primarily focused on providing tailored recommendations to its users was defined as a list of past recommendations that users can interact with and learn more about. We could create an even more engaging and richer home page for the application. However, in the current state of the application, Figure 4 shows what the homepage looks like.

By clicking on the Be Aware icon, users can navigate to the home page from anywhere within the application and return to their previous page by clicking on the icon again. Once on the home page, users can view all previous recommendations made by the application. As shown in Figure 4, the only information users see is a brief description of what was recommended and the order in which it was recommended. However, by clicking on a recommendation from the list, a card appears and displays additional details related to the recommendation that was clicked, as represented in Figure 5.

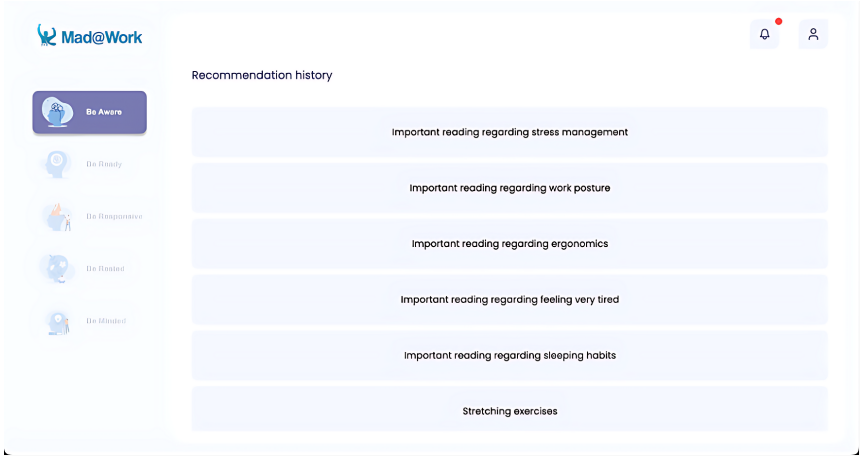

Figure 4. Recommendation history page.

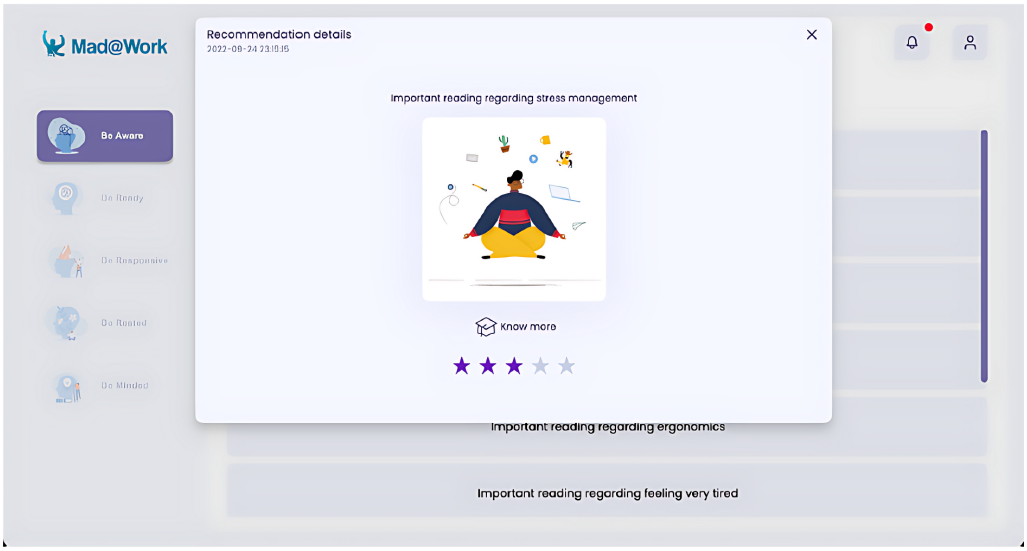

Figure 5. Recommendation card.

By hovering with their mouse cursor on the Know More icon, users can see a list of PDF documents that they can click on. Clicking on a document from this list opens a new tab in the browser that displays the corresponding content. These documents provide advice and discuss the benefits of adopting the proposed behaviors.

This card not only allows for a more detailed vision of the recommendation but also provides a rating system with which users can interact by clicking on any of the stars at the bottom. Users can rate a recommendation from one to five stars, with one indicating the user's dislike or lack of interest in it and five indicating that the user genuinely enjoyed it or found it very helpful, and the system considers these ratings for future recommendations. The star-rating system's concise design is consistent with the recommendation system's emphasis on minimizing user burden.

It is important to acknowledge that even with explicit training, user ratings remain subject to individual interpretation and potential biases. While users were instructed to rate recommendations based on their perceived impact on stress reduction, other factors (such as the enjoyment of an activity) might still influence their ratings.

In most of the figures related to the graphical user interface, it is possible to see an icon that resembles a bell and that is often associated with notifications. Users receive application recommendations through notifications. When users receive notifications, a red dot appears in the upper left corner of the icon, and when their mouse hovers over the

icon, they can see the latest notifications sent from the backend, as shown in Figure 6. Once users have seen all the notifications, the red dot disappears.

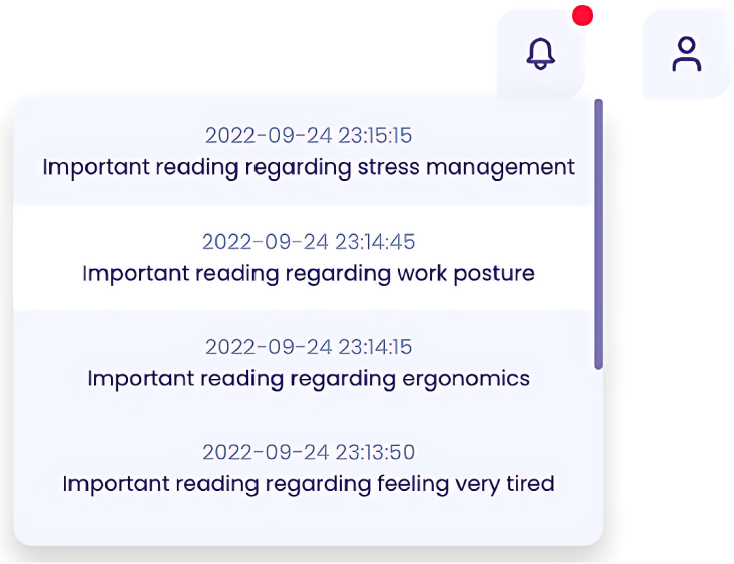

Figure 6. Notifications.

5.2. Message Broker

In a microservice-oriented architecture, integrations with other services sometimes lead to unexpected bottlenecks due to the blocking nature of synchronous requests. Thus, we added a message broker to allow the solution's services to communicate asynchronously via API instead of HTTP.

The use of Apache Kafka for asynchronous communication between microservices presents several challenges related to timely and reliable data transfer. We carefully configured topic partitions to allow parallel processing of stress data, reducing queuing delays. The stress detection module uses Kafka's low-latency producer configuration to minimize the time between stress event detection and message publication. The recommendation service employs a fine-tuned consumer polling strategy, balancing between frequent polls for timely data retrieval and avoiding excessive CPU usage. We also implemented intelligent message batching on the producer side to optimize throughput without significantly impacting latency. For critical stress events requiring immediate attention, we used Kafka streams to process data in near real time as it flowed through the system.

With these solutions, we achieved a balance between system responsiveness and scalability, ensuring that stress interventions can be delivered promptly while maintaining the overall system's ability to handle increasing loads. However, it is important to note that the complete elimination of latency in a distributed system is challenging, and ongoing monitoring and optimization are necessary to maintain performance as the system scales.

5.3. Core Service

The core service is the interface between the frontend and backend components, directly receiving and responding to every request from the GUI. In this manner, we can facilitate user-application interaction while separating the logic pertaining to recommendations and rules.

When a recommendation is requested, the core service retrieves the user's current stress level and profile information, applies relevant business rules from the rule management system, uses a content-based filtering algorithm to select appropriate exercises, and then returns the top-ranked exercise recommendation to the user.

The core service acts as the central component of the recommendation system, integrating various microservices and providing essential functionalities through its API. Its primary responsibilities include:

- User management
- Exercise management
- Recommendation generation
- Feedback collection and processing

The core service is built using the Spring Boot framework, which facilitates rapid development and easy integration with other microservices. It utilizes a RESTful API design to expose its functionalities to other system components and external clients. The core service exposes several crucial key API endpoints:

- User-related endpoints: For user registration, profile management, and authentication.
- Exercise management endpoints: To add, update, and retrieve exercise information.
- Recommendation endpoints: To generate and deliver personalized exercise recommendations.
- Feedback endpoints: To collect and process user feedback on recommended exercises.

The core service orchestrates communication and data flow between the various microservices in the system. It communicates with the stress detection module to receive real-time stress-level data; it interacts with the rule management system to apply business rules for recommendation generation; and it uses the data access layer for the persistent storage of user profiles, exercise data, and feedback information.

Handling real-time data from the stress detection module presents several challenges, namely ensuring timely data synchronization and managing potential latency issues. We employed Kafka's message queuing capabilities to address these, creating a robust and efficient asynchronous data flow. This approach ensures that fluctuations in the stress detection service do not directly impact the recommendation service while still providing near-real-time responses. Error handling and data validation mechanisms were incorporated to guarantee data integrity.

5.4. Rules Service

As mentioned above, the solution required the inclusion of a questionnaire for users to respond to. One could implement this questionnaire in a traditional way, hard-coding the questions and the resulting dependencies from users' answers, or in a more generic, configurable, and approachable manner that would allow both developers and non-developers to make changes. For this purpose, we used a rule management system.

Rule management systems offer mechanisms to distinguish business logic from application logic, facilitating scalability and maintenance as business logic progresses [25]. We can represent business logic as the outcome of fulfilling a set of constraints. There are rule management systems apart from Drools [26] that can enhance this configurability and approachability. However, being open-source and easily integrated with Java made it the perfect candidate for this solution.

This application directly connects business rules with questionnaire configurations. Considering the configurability and accessibility that Drools offers, its capacity to read and understand spreadsheets enables it to formulate business rules in an XLS file format, as illustrated in Figure 7.

RuleSet	rules				
Import	com.wellbeing.questionnaireservice.services.RulesHandler, com.wellbeing.questionnaireservice.models.Question				
Variables	java.util.List questions				
RuleTable Question					
NAME	CONDITION	CONDITION	CONDITION	CONDITION	ACTION
	RulesHandler	RulesHandler	$question : Question		
	!$rulesHandler.contains (questions, $param)	$rulesHandler.contains (questions, $param)	$question.getText().equalsIgnoreCase($param)	$param.equalsIgnoreCase($question.getAnswerText())	questions.add(new Question($param));
ID	Current Question				New Question
question16_1	"What kind of activity do you usually		"Do you exercise regularly?"	"Yes"	"What kind of activity do you usually practice?"

Figure 7. Business rules sample.

Figure 7 illustrates that the spreadsheet conforms to a particular format. Nonetheless, it remains true that it is more straightforward to comprehend and sustain than having the same logic written in code. Upon initialization of the Drools container, the rules service offers an API that enables other services within the solution's architecture to make HTTP queries. This API comprises merely two endpoints. The first endpoint's job is to obtain the rules spreadsheet from the rules folder and change it into a format that Drools can read. This makes it easier to do a more technical analysis of the logic behind the business rules. Based on the business rules illustrated in Figure 7, the answer from the initial endpoint pertaining to those rules is displayed in Figure 8.

```
rule "question16_1"
    when
        $rulesHandler : RulesHandler(!$rulesHandler.contains(questions, "What kind of activity do you usually practice?"))
        $question : Question($question.getText().equalsIgnoreCase("Do you exercise regularly?"), "Yes".equalsIgnoreCase
            ($question.getAnswerText()))
    then
        questions.add(new Question("What kind of activity do you usually practice?"));
end
```

Figure 8. Parsed business rules sample.

By simplifying the rule "question 16_1" in Figure 8, we obtain the following expression:
WHEN the list of questions does not contain the question: "What kind of activity do you usually practise?"
AND the current question is: "Do you exercise regularly?"
AND the current question is answered with: "Yes"
THEN added the question to the list of questions: "What kind of activity do you usually practise?"

The second endpoint depends on the files located in the services and models folders to process and provide a response to any request. This API receives a list of Java objects of type Question, initializes a Drools container session, and, for each object in the list, injects it as a variable into the parsed business rules, executing all rules and modifying the original list prior to returning it. For each rule, there exists a corresponding sequence of action contingent upon the fulfilment of all prerequisites. This application often leads to the addition or removal of questions from the list of Question objects that users can respond to through the GUI.

5.5. Recommender Service

The success of our recommendation system depends on the relevance and efficacy of the interventions offered to users. Given the system's focus on computer workers and the involvement of health professionals in its development, the selection and ordering of recommendations were meticulously determined by three health specialists on the research team (co-authors of this study). Their expertise was pivotal in choosing interventions known to effectively address the unique stressors experienced by computer users. This process ensured that the interventions would be both relevant and practical.

5.5.1. Recommendation Selection and Prioritization

Approximately 20 distinct categories of recommendations were included, encompassing strategies for:

- Physical Activity: Short bursts of movement, stretching exercises, and mindful movement breaks.
- Stress Reduction: Deep breathing exercises, progressive muscle relaxation techniques, and guided meditation.
- Improved Resilience: Techniques for building coping mechanisms and self-compassion.
- Better Sleep Habits: Tips and resources for optimizing sleep hygiene.
- Ergonomic Adjustments: Guidance on proper posture, workstation setup, and regular breaks to mitigate repetitive strain injuries.
- Time Management: Strategies to improve task prioritization and minimize multitasking to prevent burnout.
- Mindful Self-Care: Prompts for hydration, nutrition, and taking pauses throughout the workday.

The order of recommendations presented also reflects a strategic approach; higher-priority interventions (e.g., short breaks) appear early, with other strategies introduced over time. This initial order is refined iteratively through ongoing user feedback by analyzing ratings and usage patterns.

We accompany each recommendation with a collection of supplementary resources to enhance user understanding and application. These resources include peer-reviewed research articles on specific stress-reduction techniques (in PDF format), links to relevant and credible websites and online resources, short videos demonstrating specific exercises (e.g., stretching and mindful movements), audio files of guided meditations, and infographics summarizing key aspects of stress-management strategies.

5.5.2. Content-Based Filtering and Recommendation Properties

To implement content-based filtering, we defined the following properties for each recommendation: social interaction, nutrition, health condition, physical activity, sleep, mental health, and individual factors. These properties serve as a representation of the recommendations' characteristics and enable the comparison of recommendations and users. The same properties also represent user interests. However, by default, the solution does not know which properties are associated with each user. Instead, when users interact with the application through the GUI, they can rate recommendations. The application assigns its properties to the users who rate a recommendation positively. Only then will it be possible to calculate similarities between both recommendations and those of the users. Similarly, when users receive recommendations that do not please them and decide to rate such recommendations badly, some of the common properties are removed from users' interests to avoid similar future recommendations.

5.5.3. Content-Based Recommendation Service Implementation

The limited availability of user data at the system's initialization drove the choice of a content-based filtering approach. Even with sparse data, this approach allows for personalized recommendations, which we iteratively refine through user feedback. We leverage Apache Spark's open-source library within a Spring Boot application to implement this filtering method. The implementation uses Spark's API to perform the following steps:

1. Data Retrieval: A query retrieves recommendations (and associated properties) not yet provided to the user.
2. Data Vectorization: The properties for each recommendation and the user profile are converted into numerical vectors using techniques like count vectorization. By

representing items and user profiles as vectors in a high-dimensional space, the recommendation system can efficiently measure the similarity between the user and the recommendations.

3. Similarity Calculation: The cosine similarity between the user vector and the recommendation vectors is calculated using Spark's column Similarities method that uses the cosine similarity.
4. Recommendation Ranking: Recommendations are ranked in descending order of similarity to the user vector, with the highest-ranked recommendation presented first.

To build the initial dataset, SparkService runs a query in the database that obtains all the identifiers of recommendations that have not yet been provided to the user in the solution, as well as each corresponding property. Considering that the goal is to understand which recommendation is most similar to the interests of the users, this query also obtains the user properties and places them, along with the user identifier, on the query result, as shown in Figure 9.

Data Output | Explain | Messages | Notifications

	recommendation_id character varying (255)	recommendation_properties character varying (255)
1	5b2d376c-ca4f-4060-91fd-d9d81bb8f559	Physical activity
2	5b2d376c-ca4f-4060-91fd-d9d81bb8f559	Individual
3	rec_id_1	Physical activity
4	rec_id_1	Sleep
5	rec_id_1	Individual
6	rec_id_1	Health condition
7	rec_id_2	Health condition
8	rec_id_2	Physical activity
9	rec_id_2	Sleep
10	rec_id_2	Individual

Figure 9. Initial dataset query subset.

However, this query may not return any data. This situation can occur for various reasons: the database does not have any recommendations, the user has already received all recommendations, the database does not contain any recommendations with associated properties, and the user for whom the recommendation is for does not have any associated properties in the database. If any of these situations occur, SparkService returns a checked exception back to RecommendationService, allowing the recommender service to return the most appropriate response.

Then, assuming that the dataset is initialized with data, we need to group the properties by identifier to facilitate vectorization. Once we group the properties by identifier, we use the CountVectorModel from the Apache Spark library to create a vector from each group of properties, as illustrated in Figure 10.

After vectorizing all properties, we use the columnSimilarities method, also from Apache Spark, to calculate the similarity between vectors. However, because this method calculates similarities between columns instead of rows, it was necessary to create a matrix

and transpose it. Finally, we order the similarities of the user recommendations by degree of similarity in descending order to obtain the best recommendation.

recommendation_id	recommendation_properties	feature
rec_id_13	[Physical activity, Sleep, Health condition, Mental health]	(7,[0,1,3,4],[1.0,1.0,1.0,1.0])
rec_id_14	[Individual, Physical activity, Sleep, Mental health]	(7,[1,2,3,4],[1.0,1.0,1.0,1.0])
rec_id_4	[Individual, Physical activity, Sleep, Health condition]	(7,[0,1,2,3],[1.0,1.0,1.0,1.0])
rec_id_16	[Individual, Physical activity, Sleep, Mental health]	(7,[1,2,3,4],[1.0,1.0,1.0,1.0])
rec_id_6	[Social interaction, Health condition, Mental health]	(7,[0,4,5],[1.0,1.0,1.0])
rec_id_1	[Individual, Physical activity, Sleep, Health condition]	(7,[0,1,2,3],[1.0,1.0,1.0,1.0])
rec_id_10	[Individual, Sleep, Health condition, Nutrition]	(7,[0,1,2,6],[1.0,1.0,1.0,1.0])
rec_id_5	[Social interaction, Health condition, Mental health]	(7,[0,4,5],[1.0,1.0,1.0])
5b2d376c-ca4f-4060-91fd-d9d81bb8f559	[Individual, Physical activity]	(7,[2,3],[1.0,1.0])
rec_id_8	[Social interaction, Health condition, Mental health]	(7,[0,4,5],[1.0,1.0,1.0])
rec_id_17	[Individual, Physical activity, Sleep, Mental health]	(7,[1,2,3,4],[1.0,1.0,1.0,1.0])
rec_id_12	[Social interaction, Physical activity, Sleep, Health condition, Montal health]	(7,[0,1,3,4,5],[1.0,1.0,1.0,1.0,1.0])
rec_id_2	[Individual, Physical activity, Sleep, Health condition]	(7,[0,1,2,3],[1.0,1.0,1.0,1.0])
rec_id_9	[Individual, Sleep, Health condition, Nutrition]	(7,[0,1,2,6],[1.0,1.0,1.0,1.0])
rec_id_11	[Social interaction, Sleep, Health condition, Mental health]	(7,[0,1,4,5],[1.0,1.0,1.0,1.0])
rec_id_3	[Individual, Physical activity, Sleep, Health condition]	(7,[0,1,2,3],[1.0,1.0,1.0,1.0])
rec_id_15	[Individual, Physical activity, Sleep, Mental health]	(7,[1,2,3,4],[1.0,1.0,1.0,1.0])
rec_id_7	[Social interaction, Health condition, Mental health]	(7,[0,4,5],[1.0,1.0,1.0])

Figure 10. Vectorized dataset.

The recommender service uses Kafka streams for real-time data processing. This capability allows it to consume stress-level events from the Kafka topic published by the stress detection module and process them as they arrive, ensuring minimal latency. To ensure recommendations are delivered promptly, the service employs asynchronous processing to handle multiple requests concurrently. It uses caching mechanisms to store frequently accessed data (e.g., user profiles and exercise information) for quick retrieval, and it prioritizes critical stress events to ensure they receive immediate attention.

By combining real-time data processing, personalized recommendation generation, and continuous learning, the recommender service can provide timely, relevant stress-reduction interventions tailored to each user's current needs and preferences.

5.5.4. Feedback Integration and Model Updating

User feedback is crucial for refining recommendations over time. Users rate recommendations using a 1–5-star system, and this feedback is stored in the database and used to adjust the recommendation algorithm, improving both relevance and accuracy. Repeatedly low ratings for a particular exercise reduce the likelihood of its recommendation in similar scenarios. The system dynamically updates its model based on accumulated user feedback, enhancing performance and relevance over time.

The following example illustrates how feedback influences future recommendations. Example: Alice's Experience

- Initial Profile: Alice (a 30-year-old software engineer) reported feeling stressed due to tight deadlines and prolonged screen time. She preferred desk-friendly activities.
- Initial Recommendation: A "Quick Neck and Shoulder Stretch" (Exercise A) was suggested.
- Alice's Feedback (2 stars): "Didn't help; felt awkward in the office".
- System Response: Exercise A was flagged as ineffective, tagged with "open office" and "lack of stress relief" and similar exercises were deprioritized in Alice's profile.
- Subsequent Recommendation: A "Mindful Breathing Exercise" (Exercise B) was suggested.
- Alice's Feedback (4 stars): "Great! Easy at my desk; helped me refocus".
- System Response: Exercise B was flagged as effective, tagged as "desk-friendly", "refocusing", and "positive stress relief". Similar exercises were prioritized.
- Future Impact: Alice would receive more mindfulness, desk-based, and focus-enhancing recommendations, with exercises like Exercise A deprioritized.

6. Evaluation

This section presents the evaluation of the implemented solution in detail. The goal is to evaluate whether the implemented solution is of high quality and capable of providing adequate stress alleviation recommendations. In the context of this document, it is assumed that an adequate stress relief recommendation is one that has been positively evaluated by the user who receives it.

6.1. Methodology

To show that the system can provide adequate stress relief recommendations, it is necessary to collect user-generated data. Users' ratings of past recommendations align with these data. We gathered data from a pilot that involved 22 users who volunteered to use the recommendation system and evaluate their recommendations. Therefore, the results are merely demonstrative and not intended for rigorous statistical interpretation. It solely aims to illustrate the system's evaluation process.

6.2. Metrics

An effective recommendation system is expected to provide appropriate recommendations that users enjoy while avoiding recommendations that users dislike. Thus, it is possible to assess the effectiveness of the system by measuring both error metrics and decision support metrics. By measuring error metrics, it is possible to verify how much the recommendation system deviates from users' interests, meaning that the smaller these metrics are, the better. In this manner, we can represent the error as the difference between the user's rating (r) and the predicted rating of the recommendation system (p) for a given recommendation:

$$error = |r - p| \tag{1}$$

By summing all the errors associated with each recommendation for a given user and dividing the result by the number of rated recommendations, we obtain the Mean Absolute Error (MAE):

$$MAE = \frac{\sum_{i=1}^{N} |r_i - p_i|}{N} \tag{2}$$

N represents the total number of recommendations. A lower MAE suggests that the system is more effective in predicting user satisfaction with the interventions.

As of the date of this document, the recommendation system has no concern for a specific rating when providing recommendations, meaning its purpose is to generate the most suitable recommendation based on the information it contains, such as user preferences. However, it is possible to perform a more rigorous assessment by assuming a maximum rating of 5 for all the system recommendations. In this manner, the system will compare all user ratings with the maximum rating of 5.

On the other hand, measuring decision support metrics involves evaluating each recommendation as right or wrong and comparing the system's recommendations to users' decisions, as Table 1 illustrates.

Table 1. Confusion Matrix.

	System Recommendation	
	Proposed	**Not Proposed**
accepted	TP	FN
Not accepted	FP	TN

The following outcomes can be provided:

True Positive (TP): recommendations that the user found helpful.
False Positive (FP): recommendations that the user found unhelpful.
False Negative (FN): recommendations not proposed.
True Negative (TN): recommendations not proposed.

Users cannot access recommendations until the system provides them, which is why we exclude both TN and FN outcomes from this evaluation. In the confusion matrix, only the precision metric can be defined to evaluate the recommendation system, which gives the fraction of the recommendations that users liked. However, this is not a problem since it is often considered more important to optimize precision without placing too much emphasis on whether users receive all the possible relevant recommendations, which is represented by the recall metric. The precision is calculated using the following formula:

$$precision = \frac{TP}{TP + FP} \tag{3}$$

Higher precision indicates that the system is more effective in recommending truly helpful interventions.

Given that user ratings for the system's recommendations can range from 1 to 5, a 3-star rating was established as the threshold for distinguishing between positive (TP) and negative (FP) recommendations. This threshold reflects a neutral or satisfactory experience, as per the instructions given to users. A 3-star rating suggests that the recommended exercise provided some benefit but not enough to warrant a higher rating. Ratings below 3 indicate that the recommendation was unhelpful or even detrimental (e.g., increased stress or wasted time), while ratings above 3 imply that the recommendation had a positive impact, regardless of the magnitude of improvement.

6.3. System Evaluation

Table 2 presents the evaluation that the users performed based on the recommendations they received, using the previously defined metrics and the methodology described.

Table 2. User evaluation recommendations.

User	N.Recs.	Max. Score	Min. Score	Av. Score	Std.Dev.	MAE	Precision
U1	4	5	1	3	1.8	2	0.5
U2	3	4	2	2.7	1.2	2.3	0.3
U3	4	5	1	2.8	2.1	2.3	0.5
U4	4	5	2	3	1.4	2	0.5
U5	6	3	1	2.2	0.8	2.8	0.3
U6	3	3	1	2.3	1.2	2.7	0.7
U7	4	3	1	1.5	1	3.5	0.3
U8	3	5	1	2.7	2.1	2.3	0.3
U9	6	3	1	1.7	1	3.3	0.3
U10	6	5	1	3.3	1.5	1.7	0.7
U11	5	4	2	3.6	0.9	1.4	0.8
U12	5	5	1	2.6	1.8	2.4	0.4
U13	3	3	1	2	1	3	0.3
U14	4	5	4	4.5	0.6	0.5	1
U15	4	4	2	3	1.2	2	0.5
U16	6	3	1	2.2	0.8	2.8	0.3
U17	6	5	1	3.5	1.5	1.5	0.8
U18	4	4	1	2.8	1.5	2.3	0.5
U19	7	5	1	3.1	1.5	1.9	0.6
U20	6	5	1	3	1.3	2	0.8
U21	5	5	1	2.2	1.6	2.8	0.2
U22	4	4	1	2.5	1.7	2.5	0.5

According to this initial evaluation, the system achieves an overall MAE of 2.27 (stdev = 0.7) and an overall precision of 0.51 (stdev = 0.2). Despite the performance measures being relatively low, they reflect the assessments of a very small number of users and penalize the system, as the maximum rating was considered in the calculation of the MAE measure. As the system enters production and receives more user evaluations of its recommendations, we anticipate an improvement in its performance.

While these results are not definitive and cannot be generalized to a larger population, they demonstrate the feasibility of our approach and highlight areas for future investigation with a substantially larger user base. The current sample size limits the generalizability of the findings, and the relatively small number of recommendations interacted with by each user also limits the depth of analysis. A larger-scale study would allow for a more robust assessment of user engagement, provide more conclusive evidence of the system's effectiveness, and enable a more detailed analysis of the relationship between user ratings, objective physiological measures, and actual stress reduction.

7. Discussion

This study presented a novel recommendation system designed to mitigate workplace stress proactively among computer users. The evaluation, while providing valuable preliminary insights, highlights several areas for future investigation and refinement.

Despite the concise and efficient design of the star-rating system, user engagement with the evaluation process might decrease over time. Future iterations of the system could explore incorporating gamified feedback mechanisms to enhance engagement and ensure the continued quality of evaluation data. These features would help maintain user motivation and yield more reliable feedback on the system's efficacy over extended periods.

The reliance on self-reported data for evaluating stress reduction presents inherent limitations. User satisfaction, while informative, does not provide a direct measure of actual stress reduction. Other factors (such as individual preferences, enjoyment of the recommended activities, or external influences) might affect user ratings. Therefore, future research should integrate objective physiological data from the stress detection module to provide a more comprehensive and robust assessment of the system's impact on stress levels. The correlation between self-reported stress reduction and objectively measured physiological data will be a focus of future analysis.

This study demonstrates the potential of our microservice-based recommendation system to proactively reduce workplace stress among computer-based workers. Key features contributing to this potential include (1) personalized recommendations tailored to individual preferences and stress patterns, (2) proactive interventions triggered by real-time stress detection, and (3) a seamless integration into the computer-based workflow. A continuous feedback loop further refines recommendations, ensuring ongoing adaptation to user needs. The system's diverse intervention options (physical, breathing, cognitive exercises) cater to individual preferences.

While these initial findings are promising, further research—particularly with larger, more diverse user samples and extended evaluation periods—is needed to fully validate the system's efficacy and address existing limitations. However, this work demonstrates the potential to foster healthier and more productive work environments by delivering relevant and timely stress-reducing interventions.

The potential impact of this system on workplace productivity and employee well-being warrants further investigation. By proactively identifying and addressing stress, the system may reduce absenteeism, improve employee morale, and enhance overall productivity. Quantifying these potential benefits requires a larger-scale evaluation with

a more diverse user group and a longer observation period to gather sufficient data for robust statistical analysis.

The use of physiological data for stress detection raises important ethical considerations, especially within a workplace setting. To address these concerns, our system prioritizes data privacy through data anonymization, user consent, and secure data storage. Furthermore, we ensure transparency by informing users about data collection practices and their rights. However, continued research is necessary to identify, address, and mitigate potential ethical challenges associated with such data collection and usage.

8. Conclusions

This research presented the design and initial implementation of a novel microservice-based recommendation system for proactive workplace stress management among computer users. The system leverages predicted stress levels to provide timely, personalised interventions, prioritising user well-being and minimising additional effort. Initial evaluations, conducted within a limited pilot study, demonstrated the feasibility of developing and deploying such a system. Specifically, the pilot showed the successful creation of a functional system with a robust architecture capable of delivering personalised recommendations.

While these initial results are encouraging, it is crucial to acknowledge the limitations of the small sample size and the preliminary nature of the evaluation. Therefore, the findings should be interpreted as demonstrative of the system's potential rather than as definitive proof of its effectiveness in reducing workplace stress. The observed trends suggest that the system can be successfully implemented and that users are receptive to receiving personalised recommendations, but further research is needed to validate these observations.

Future research will focus on larger-scale evaluations with diverse demographics and work settings to rigorously assess the system's impact on stress levels, user engagement, and overall well-being. These evaluations will incorporate objective physiological measures alongside user feedback to provide a more comprehensive understanding of the system's effectiveness. Additionally, we plan to expand the recommendation database, integrate ergonomic assessments, and explore gamification strategies to enhance user engagement and optimise the system's performance. These future studies will provide a more robust and generalisable assessment of the system's potential to proactively mitigate workplace stress.

Author Contributions: Conceptualization, F.R.; Software, F.P.; Validation, S.F., M.R. and N.R.; Writing—original draft, F.P.; Writing—review & editing, F.R.; Supervision, F.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by ITEA3, Eureka Cluster on software innovation and COMPETE 2020-POCI-01-0247-FEDER-046168 LISBOA-01-0247-FEDER-046222.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Demissie, B.; Bayih, E.T.; Demmelash, A.A. A systematic review of work-related musculoskeletal disorders and risk factors among computer users. *Heliyon* **2024**, *10*, e25075. [CrossRef] [PubMed]
2. Mirvohid, A. The Impact of Prolonged Computer use on Neurological Health in the IT Sector. *J. Sci. Res. Mod. Views Innov.* **2024**, *1*, 132–134.

3. Pedersen, J.; Graversen, B.K.; Hansen, K.S.; Madsen, I.E.H. The labor market costs of work-related stress: A longitudinal study of 52 763 Danish employees using multi-state modeling. *Scand. J. Work. Environ. Health* **2024**, *50*, 61. [PubMed]
4. Epstein, R.; Aceret, J.; Giordani, C.; Zankich, V.; Zhang, L. On the importance of proactive stress management: A rank ordering and analysis of four cognitive-behavioral competencies of stress management using a large international sample. *SSRN 4648808* **2023**. [CrossRef]
5. Riches, S.; Taylor, L.; Jeyarajaguru, P.; Veling, W.; Valmaggia, L. Virtual reality and immersive technologies to promote workplace wellbeing: A systematic review. *J. Ment. Health* **2024**, *33*, 253–273. [PubMed]
6. Fleming, W.J. Employee well-being outcomes from individual-level mental health interventions: Cross-sectional evidence from the United Kingdom. *Ind. Relations J.* **2024**, *55*, 162–182. [CrossRef]
7. Molek-Winiarska, D.; Kawka, T. Reducing work-related stress through soft-skills training intervention in the mining industry. *Hum. Factors* **2024**, *66*, 1633–1649. [PubMed]
8. Rodrigues, F.; Correia, H. Semi-supervised and ensemble learning to predict work-related stress. *J. Intell. Inf. Syst.* **2024**, *62*, 77–90. [CrossRef]
9. Kuanr, M.; Mohapatra, P. Assessment Methods for Evaluation of Recommender Systems: A Survey. *Found. Comput. Decis. Sci.* **2021**, *46*, 393–421.
10. Roy, D.; Dutta, M. A systematic review and research perspective on recommender systems. *J. Big Data* **2022**, *9*, 59.
11. De Croon, R.; Van Houdt, L.; Htun, N.N.; Štiglic, G.; Abeele, V.V.; Verbert, K. Health Recommender Systems: Systematic Review. *J. Med. Internet Res.* **2021**, *23*, 498–509. [CrossRef] [PubMed]
12. Xin, Y.; Chen, Y.; Jin, L.; Cai, Y.; Feng, L. TeenRead: An adolescents reading recommendation system towards online bibliotherapy. In Proceedings of the 2017 IEEE International Congress on Big Data (BigData Congress), Honolulu, HI, USA, 25–30 June 2017; pp. 431–434.
13. Zaini, N.; Latip, M.F.A.; Omar, H.; Mazalan, L.; Norhazman, H. Online personalized audio therapy recommender based on community ratings. In Proceedings of the 2012 International Symposium on Computer Applications and Industrial Electronics (ISCAIE), Kota Kinabalu, Malaysia, 3–4 December 2012; pp. 318–322.
14. Paredes, P.; Gilad-Bachrach, R.; Czerwinski, M.; Roseway, A.; Rowan, K.; Hernandez, J. PopTherapy: Coping with stress through pop-culture. In Proceedings of the 8th International Conference on Pervasive Computing Technologies for Healthcare, Oldenburg, Germany, 20–23 May 2014; pp. 109–117.
15. Clarke, S.; Jaimes, L.G.; Labrador, M.A. mstress: A mobile recommender system for just-in-time interventions for stress. In Proceedings of the 2017 14th IEEE Annual Consumer Communications & Networking Conference (CCNC), Las Vegas, NV, USA, 8–11 January 2017; pp. 1–5.
16. Wang, C.; Sahebi, S.; Torkamaan, H. STRETCH: Stress and Behavior Modeling with Tensor Decomposition of Heterogeneous Data. In Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, Melbourne, VIC, Australia, 14–17 December 2021; pp. 453–462.
17. Gupta, A.; Misra, S.; Pathak, N. StressAlly: A Smartphone-Based Stress Companion Recommender System for Students. In Proceedings of the GLOBECOM 2023–2023 IEEE Global Communications Conference, Kuala Lumpur, Malaysia, 4–8 December 2023; pp. 504–509.
18. Cvetković, B.; Gjoreski, M.; Šorn, J.; Frešer, M.; Bogdański, M.; Jackowska, K.; Kosiedowski, M.; Stroiński, A.; Luštrek, M. Management of physical, mental and environmental stress at the workplace. In Proceedings of the 2017 International Conference on Intelligent Environments (IE), Seoul, Republic of Korea, 21–25 August 2017; pp. 76–83.
19. Vyas, J.; Das, D.; Das, S.K. Vehicular edge computing based driver recommendation system using federated learning. In Proceedings of the 2020 IEEE 17th International Conference on Mobile Ad Hoc and Sensor Systems (MASS), Delhi, India, 10–13 December 2020; pp. 675–683.
20. Tong, X.; Mauriello, M.L.; Mora-Mendoza, M.A.; Prabhu, N.; Kim, J.P.; Paredes Castro, P.E. Just Do Something: Comparing Self-proposed and Machine-recommended Stress Interventions among Online Workers with Home Sweet Office. In Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems, Hamburg, Germany, 23–28 April 2023; pp. 1–20.
21. Shu, Y.; Zhang, H.; Gu, H.; Zhang, P.; Lu, T.; Li, D.; Gu, N. RAH! RecSys–Assistant–Human: A Human-Centered Recommendation Framework With LLM Agents. *IEEE Trans. Comput. Soc. Syst.* **2024**, *11*, 6759–6770.
22. Rana, A.; Sanner, S.; Bouadjenek, M.R.; Di Carlantonio, R.; Farmaner, G. User experience and the role of personalization in critiquing-based conversational recommendation. *ACM Trans. Web* **2024**, *18*, 1–21.
23. Wang, W.; Zheng, S.; Ali, R.; Li, J. Relevancy or Diversity?: Recommendation Strategy Based on the Degree of Multi-Context Use of News Feed Users. *J. Glob. Inf. Manag. (JGIM)* **2022**, *30*, 1–24.
24. Calonaci, D. *Designing User Interfaces: Exploring User Interfaces, UI Elements, Design Prototypes and the Figma UI Design Tool (English Edition)*; BPB Publications: Noida, India, 2021.

25. van Velzen, R. Business Rules Management Systems. Master's Thesis, Utrecht University, Utrecht, The Netherlands, 2018.
26. Browne, P.; Johnson, P. *JBoss Drools Business Rules*; Packt Publishing Birmingham: Birmingham, UK, 2009; Volume 1847196063.