De-Stress Me: A Personal Digital Assistant for Stress Detection and Treatment Recommendation

Yoon Kyung Shon
Department of Computer Science
University of Maryland

William Ingold
Department of Computer Science
University of Maryland

Hyemi Song
Department of Computer Science
University of Maryland

Geonsun Lee
Department of Computer Science
University of Maryland

Irtaza Shahid
Department of Computer Science
University of Maryland

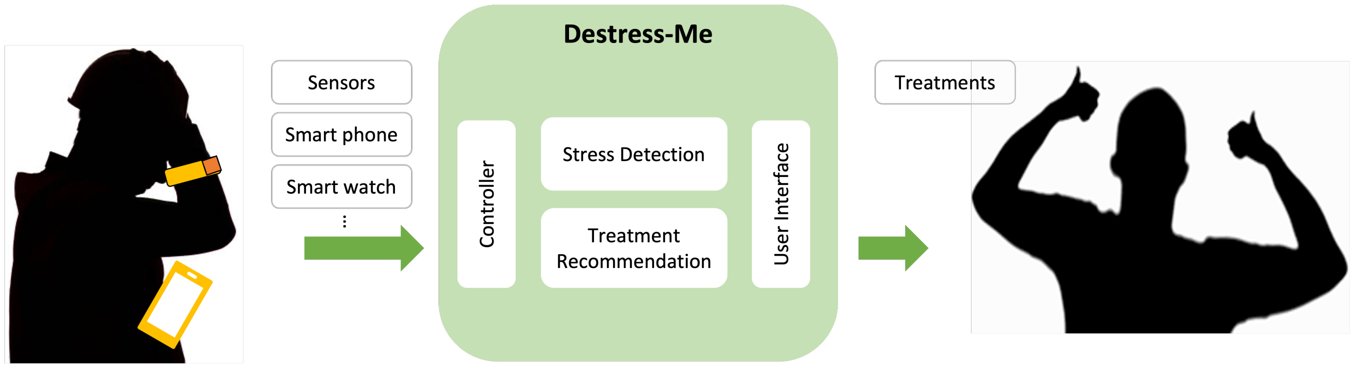


Figure 1: A representative application and overview of *De-Stress Me*.

ABSTRACT

Stressful situations are critical to one's physical and mental health if not handled properly. To suggest effective stress-relief treatments, it is essential to understand users' context information. For better assistance in users' stress management, we present *De-Stress Me*, a stress detection system that exploits contextual information and recommends treatments accordingly. The architecture of *De-Stress Me* is designed in a scalable and extensible way based on the actor model. The detection system takes five different contextual information including the user's heartbeat and blood pressure which is collected from personal devices such as a smart watch. Once the system determines the user's stress level based on a predefined discrete steps, the recommendation engine additionally retrieves the user's environmental context and delivers the appropriate treatment. *De-Stress Me* is presented in a form of a web app where the user can view their current stress level and get notified with treatment recommendations.

1 INTRODUCTION

Stress is an escalated state of a human body in response to challenging conditions. We all face stress during our daily lives: while working on some critical task, thinking about some future scenario, worrying about some problem etc. Short-term exposure to stress might increase productivity for a short duration, but constant stress can cause mental and physical disorders such as cardiovascular diseases, hypertension, diabetes, cancer, headaches, depression, anxiety, and insomnia. A recent study has shown that stress results in accidents, diminished productivity, and medical, legal, and insurance costs which is costing United States around $300 Billion every year [14]. Considering these drastic effects of stress, we are envisioning a personal digital assistant that can constantly monitor user's stress level, and based on the user's context and detected stress level able to recommend some activities or actions that can help in reducing/controlling the stress level.

The goal of our Personal Digital Assistant (PDA) is to recommend appropriate treatments based on detected users' stress level. To achieve this, we assume our system continuously detecting users' stress and determine its severity out of five predefined stress levels. Note that such stress levels are determined heuristically. Depending on the detected stress level, the system will provide a set of recommendations. For example, if the stress level is 1, the system will recommend the user to simply stand up or meditate fora few minutes.

On the other hand, for extreme cases that reach stress level 5, the system would have to ask the user to adjust their work schedule or reach out to their therapist for medical assistance.

We envision *De-Stress Me* to assist graduate students with a busy work schedule as an example of our user. Fig 1 is demonstrating our working scenario, where a person is feeling stressed, and his smart watch is measuring his biomarkers, and then our system is getting this contextual information, estimating the stress level and then recommending appropriate action to reduce the stress level.

At this stage of development, *De-Stress Me* makes the following contributions:

- *De-Stress Me* utilizes a model that detects user's stress level based on context information in addition to sensor data.
- *De-Stress Me* is the first stress-relief treatment recommendation system to provide context-based personalized treatments.
- We provide a stress treatment PDA architecture that is both scalable and extensible.

2 RELATED WORK

Stress plays a critical role in our lives by impacting our physiological and psychological well-being. Moderate stress occasionally may boost our productivity, but prolonged exposure to stress has detrimental effects [12, 15]. That's why development of a system that can monitoring stress level is an active area of research for the past two decades. The standard approach for stress monitoring relies on extraction of cortisol level from the saliva, or requiring user to complete questionnaire, which is inconvenient for continuous/long term stress monitoring [8]. To bridge this gap, researchers are exploring IOT based solutions for the development of non-obtrusive stress monitoring systems that can measure stress level continuously without effecting user's daily activities.

Recently, various wearable devices are developed to infer stress level by measuring various biomarkers such as smart belts [13], shirts [16], smart watches [3], and skin attachabel sensors [9]. Smartphones are also considered for monitoring stress level by analyzing user behavior [4, 5]. Recent work [11] have shown the possibility of photoplethysmogram-based HRV extraction using smartphone which is then used for stress detection. WiStress [7] has proposed a passive stress monitoring system which relies on wireless signals for bio-marker measurements, and machine learning algorithm for stress prediction. In this project, we are building on top of these technologies, where we are capturing data from smartphones and wearables to estimate the stress level, and then designed a recommendation system to guide user in controlling or reducing their stress level.

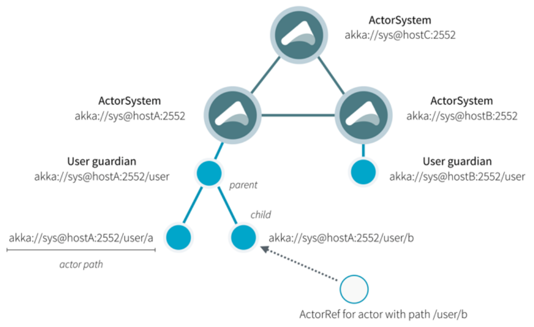

ActorSystem
akka://sys@hostC:2552
ActorSystem
akka://sys@hostA:2552
ActorSystem
akka://sys@hostB:2552
User guardian
akka://sys@hostA:2552/user
User guardian
akka://sys@hostB:2552/user
parent
child
akka://sys@hostA:2552/user/a
actor path
akka://sys@hostA:2552/user/b
ActorRef for actor with path /user/b

Figure 2: Actor System.

3 CORE INTUITIONS AND PRIMERS

3.1 Features

The selection of features which we measure and then use for stress detection is a crucial part for the project. Selected features should have strong relation with the stress level. After careful consideration, we have shortlisted the following 5 features: heart rate, blood pressure systolic, blood pressure diastolic, busyness, and sleep hours. A study has shown that in a state of stress, body releases an adrenaline hormone which in turn increase the heart rate [1]. Another study on the effects of psychological stress has proved that blood pressure increases during stress [6]. Hence, we have decided to include heart rate, and blood pressure measurements. We also included number of sleep hours as a detection feature because sleep deprivation is reported to be associated with increase blood pressure and stress hormone level [10]. In addition to that we have added one more feature representing busyness of a user because long list of tasks, and constant work can increase the level of stress hormones. So, we have finalized these 5 features for stress level detection.

3.2 Actor Model

For the implementation of our *De-Stress Me* system which requires multiple modules to run in parallel and communicate with each other, we are using Akka. Akka is a set of open source libraries for designing scalable and resilient systems that spans over processor cores and networks [2]. Akka provides a level of abstraction to write parallel, concurrent and distributed systems. Akka relies on actor model, where different actors perform operations independently and communicate with each other through messages. Actors can define a specific message format for themselves, actions upon reception of next message, and create other actors. The use of actor model make it easier to write modules independently that can work as a single concurrent system. It also enables varying interaction patterns between modules, encourages simplistic design, and offers the ability to gracefully handle errors. Fig 2 shows different actors communicating to each other and creating other actors so have parent-child relation.

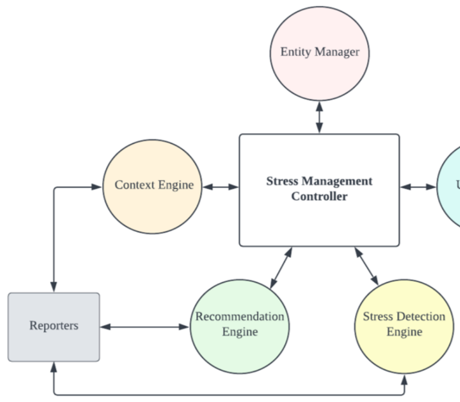


Figure 3: *De-Stress Me* architecture design.

4 SYSTEM DESIGN

The system design of *De-Stress Me* comprises of the following 6 modules: Controller, Entity Manager, Context Engine, Detection Engine, Recommendation Engine, and User Interface. Fig 3 shows the *De-Stress Me* architecture. Now, we discuss each module in more detail.

4.1 Controller

Our *De-Stress Me* system design supports various personal devices and with any given data from devices, the system sorts out contexts which are useful to measure stress level. Based on collected context data, the system periodically performs machine learning methods to detect stress level and gets the estimated level. After detection is finished, the corresponding measurement for each stress level is recommended to the user and it's shown via user interface by pop-up notification

In order to support asynchronous performance for each engine and scalability of the system, we implemented a controller to manage work flows between all engines and managers as in Fig 3. First, the Controller generates all engines that it need for stress detection flow. Later when another engine such as privacy engine is added for user data protection, we can simply add to our system by updating controller. After the controller gets message back from all engines, it starts periodic detection and recommendation procedure. Finally the controller sends result data to the front-end, so that user can be notified with current stress level and get appropriate treatment at exact point of time.

4.2 Entity Manager

Recent development of IoT wearable devices enables them to provide not just one but various kinds of health data to an end user. Therefore at the end user side, collected data from devices can be overlapped and excessive. To prevent this problem, the controller gets knowledge of system-connected device lists from the entity manager and sends this information over to the context engine to sort out only necessary data for stress detection. In our current system's implementation, we used device databases instead of actual data collection process. The device database is consisted with smart watch, smart phone, blood pressure cuff, and calendar tables. Each table contains different data that only the device can support.

4.3 Context Engine

In order to track the current context of the user, we have created the *StressContextEngine*. This houses a range of what we have called *reporters*, which report on the current context and environment. Each *reporter* is a single actor responsible for a single type of contextual information.

The *StressContextEngine* offers a way to abstract away interfacing with numerous types of entities and allows focus to remain upon the data, rather than how it is obtained. This allows a simpler interface to interact with for both the detection and recommendation engines. Meanwhile, due to the Actor Framework, supervision and orchestration of how the *reporters* operate or are handled upon error falls upon the StressContextEngine, without the outside system knowing or caring how it is done.

As shown in Fig 4, *De-Stress Me* is currently designed to operate with the following *reporters*: *Blood Pressure*, *Heart Rate*, *Sleep*, *Media*, *Location*, *Scheduler*, *Busyness*, and *Medical History*. Each reporter is tied to a specific *entity*, which it relies upon for its respective data (heart rate, blood pressure, etc.). Fig 4 has a simple visualization of the organization of this module, whereby each pink box is a reporter and blue boxes are entities. These *reporters* operate asynchronously, querying their entities for pertinent, fresh data. Each *reporter* offers a query command to request its latest data. Many also offer the ability to subscribe to any new events that occur; useful for values that may update often, such as the user's heart rate.

References for these *reporter* actors are shared with the rest of the system. This way any who may require specific data may directly communicate with the respective reporter. Upon its creation, a message with the references are forwarded both to the recommendation and detection engines. Additionally, since data requirements may develop overtime for these engines, this list of references provides a flexible way of sharing any number of reporters with the system - even across different hosts.

4.4 Detection Engine

The purpose of this module is to estimate the stress level so that it can be used to provide appropriate recommendations to the user. For this purpose, we are relying on 5 different measurements that provide the contextual information about

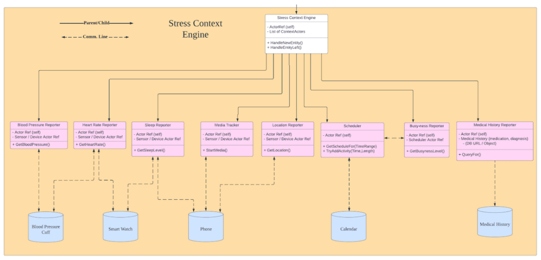


Figure 4: Context Engine.

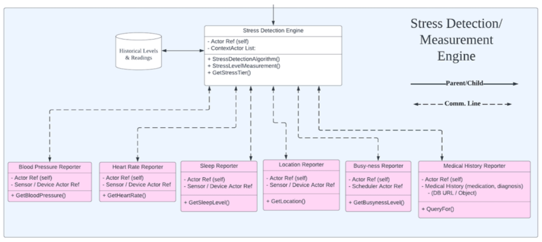


Figure 5: Detection engine.

the user. Then we have trained a machine learning model to predict the current stress level. Fig 5 shows the parts of a detection engine.

To get the contextual information about the user, we are relying the following 5 features: sleep-hour, busyness, bp-systolic, bp-diastolic, and heart-rate. All of these features has some correlation with the stress level. Human body tend to pump blood rapidly in a state of stress to prepare our body for fight or flight response, so heart rate and blood pressure both increase with the stress level. Less numbers of sleep hours is a major factor contributing to the stress in the body. It is natural to feel stressed when you have more number of tasks in your todo list.

To get the latest measurements of the above mentioned features, We have implemented a separate reporter using Akka actor system discussed in previous section. Upon request from the detection engine, each reporter grabs a latest measurement and returns to the detection engine. The benefit of using actor model here is that multiple reporters can work in parallel and with different frequencies without any interruption.

Now, that detection engine has contextual information, the next step is to predict the current stress level. For this purpose, we refer to the field of machine learning that has shown a great potential in finding/extracting the hidden relation from the data. We have formulated our stress detection problem as a traditional multi-class classification problem by defining stress in 5 different levels. After experimenting with multiple machine learning algorithms, we have found logistic regression is providing us the maximum accuracy of 95%. We have trained our model on 200 data points, and tested on 20 data points.

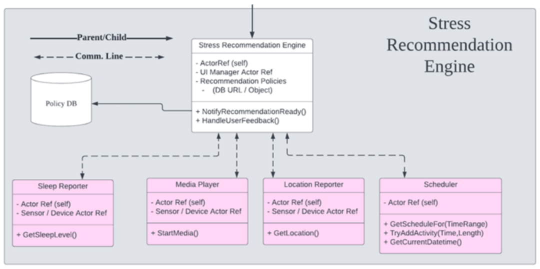


Figure 6: Recommendation engine.

After getting the estimated stress level, detection engine sends that value to the controller so that it can be further used in providing appropriate recommendation.

4.5 Recommendation Engine

To suggest users context-appropriate stress relief treatments, our recommendation engine takes two factors into account; the detected stress level of the user and the current status of the user. As seen in Fig. 6, the engine is structured in a way that it communicates with the controller to receive user's stress level and a set of reporters to retrieve user's status.

First, we set up a treatment database with predefined treatment recommendations. Each treatment considers three conditions; stress level, user's sleep status, and user's current location. For our current implementation, we simplified the sleep status as **good** if user slept for more than 6 hours and **bad** if it is any less than that. Based on the sleep status, treatments that involves physical activity will be excluded. Additionally, we take into account whether the user is currently located in a **public** or a **private** location to distinguish treatments that require solitary space.

The workflow of the recommendation engine is as follows: With the determined stress level from the detection engine, the controller passes on relevant information to the recommendation engine including the user's past and current stress level. Once the recommendation engine receives the stress level from the controller, it first determines whether the stress level has changed from the previous message. If so, the engine calls reporters that provides the environmental information of the user's status. Such information is critical to elect treatments that can be performed by the user immediately. The sleeping hours of the user is collected from the *Sleep Reporter* and the engine determines whether the sleep status is good or bad. Similarly, it differentiate private and public spaces from the *Location Reporter*'s data such as office, home, or gym. With the three parameters, the engine fetches the corresponding treatment from the treatment database and sends it back to the controller.

4.6 User Interface

4.6.1 Back-end. The back-end is structured into three key segments: *StresWebsever*, *StressWebHandler*, and *WebRoutes*.

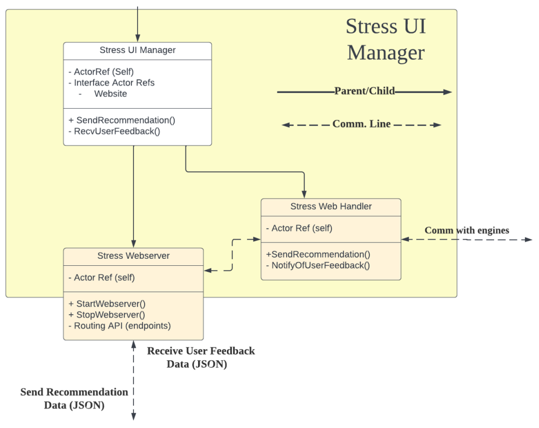


Figure 7: User Interface architecture.

These three segments are handled by the *StressUIManager*, which acts as a supervisor to them.

Upon creation by the *StressManagementController*, as stated earlier, the *StressUIManager*'s responsibility is to spawn and maintain an accessible web server, which is handled by the *StresWebsever*, for an end user to interact with during the life cycle of the application. *StressWebsever* remains simplistic, merely listening for new HTTP connections and serving said clients. It not only serves up typical front-end needs, but also functions as the application's RESTful API.

Due to potentially other aspects of the PDA, or an evolving front-end experience, the *StressWebHandler* and *WebRoutes* form *De-Stress Me*'s API and other web routes. At this stage, the API remains simplistic, having a API single endpoint for data. This endpoint returns back aggregated data from all reporters, the detection engine, and the recommendation engine in the form of a JSON object. This is presented to the user for their knowledge, however, the amount of data presented can easily be diminished in the future. The key information the front-end requires is when a new stress level is detected and any potential recommended treatments we have for the user.

4.6.2 *Front-end.*

The front-end side of *De-Stress Me* is implemented as a web-app using React Redux. On the landing page, *Dashboard*, the user can view their real-time physiological data including heartbeat rate, blood pressure, and sleeping hours. A pop-up window will be displayed when there is a change in the detected stress level. The user will proceed to see a stress-relief treatment recommendation to resolve their stress. To reinforce our detection and recommendation engine, we also attempt to collect user's reaction to the presented information. For example, the user may click *decline* if they conclude that the detected stress is inconsistent with their perceived level of stress (as shown in Fig. 8-(b)). For the recommendation engine, we aim to collect whether the user found the treatment helpful in a rate of 5 (Fig. 8-(c)).

We also provide a *My Profile* page where the user can input their daily emotional status which cannot be collected through sensor data yet is highly relevant to user's stress level. Also, there is a data entry field where the user can type in trivial medical conditions that are not recorded in the official medical report such as a slight fever. In the *All Treatments* page, the user is allowed to view the entire list of treatment recommendations so that they can rank their preference toward each action. Screenshots of the application are shown in Fig. 8.

5 EVALUATION PLANS

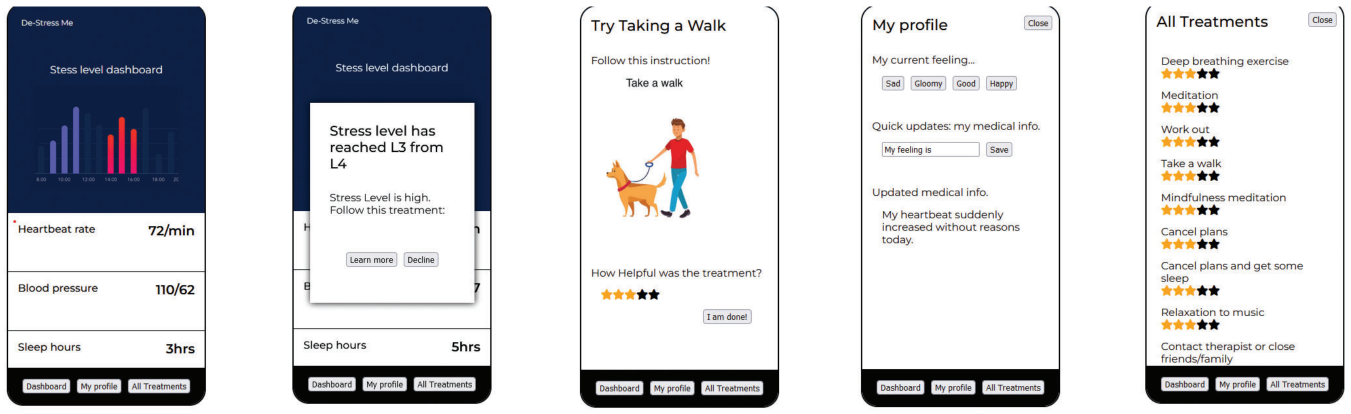
We plan to evaluate *De-Stress Me* through a series of phases for 14 days in total; we would be able to observe how effective our system is in a natural setting. We design a with-in subject study with three consecutive filed phases, assigning 7 days each.

Prior to phase 1, participants will be invited to an entry session, where they read and sign a consent form of our study to approve collection of personal data. Next, a simple online survey will be provided regarding the basic demographic information and one's stress-related perception. While the participants are completing the survey, we will be installing our *De-Stress Me* application to each participant's phone. The subjects will be given instructions on how to use the app and the overall flow of the study.

Phase 1: Baseline. The first 7 days will be done to collect a baseline data of the user's physiological data, the average stress level according to that, how users' handle stressful situations, and finally whether there are changes in perceived stress level after their reaction to it. Apart from the automatically collected data from the user's smart watch and phone, users' will be asked to fill out a daily survey on their perceived stress level of the day and how they managed to handle it.

Phase 2: De-Stress Me. In the beginning of phase 2, users will receive an email to remind users how to use *De-Stress Me*. With the recommendation engine active, users will receive notifications when there are changes in their stress level/treatments and also will be able to check their current status through the mobile application. Users will be given the freedom to optionally add their own desired stress treatment to utilize our customization function. During this phase, we will record the app usage time, personal physiological data, changes in stress level, and users' satisfaction on the provided treatments.

After completing phase 2, participants will receive an exit survey form via email. The survey will focus on the usability of the app and whether the subject experienced changes in handling a stressful situation. Upon completion



Stress level has reached L3 from L4

Stress Level is high. Follow this treatment:

Follow this instruction!

Take a walk

How Helpful was the treatment?

My current feeling...

Quick updates: my medical info.

My feeling is

Updated medical info.

My heartbeat suddenly increased without reasons today.

Deep breathing exercise

Meditation

Work out

Take a walk

Mindfulness meditation

Cancel plans

Cancel plans and get some sleep

Relaxation to music

Contact therapist or close friends/family

Figure 8: User Interface for *De-Stress Me*. (a) The dashboard displays user's real-time physiological data. (b) A pop-up window shows up when a change in stress level is detected. When user clicks on *learn more*, (c) a stress-relief treatment is suggested to the user. There are (d) *My Profile* tab and (e) *All Treatments* tab where the user can manually input their status and preferred stress-relief treatments.

of the survey, subjects will be compensated with a 20USD gift card.

Participants will be further invited to a semi-structured interview to gain deeper qualitative insights to the user experience. All interviews will be audio recorded and coded with categorical codes. A thematic analysis will be conducted afterwards.

5.1 Participant Recruitment

Participants will be recruited through a university mailing list. We will conduct a screening survey to recruit subjects who are (1) graduate students; (2) uses a smartphone and has a mobile data plan with at least 2GB/month (to ensure the collected data can be transferred to our server real-time); (3) able to physically attend the initial entry session.

6 CONCLUSION

In this paper, we presented *De-Stress Me*, a personalized stress detection system that proposes context-based treatments which correspond to the detected stress level. Utilizing the actor model, the framework of *De-Stress Me* is designed in a way so that it is **scalable**, multiple devices and entities can be attached to the system, and **extensible**, additional contexts can be added or contexts collected from the system can be used in a different stress-related system. For stress level detection, we used a total of 5 features to define the level of stress and achieved 95% of accuracy by using logistic regression. Based on the detected stress level and additional contextual information for the user's current status, an appropriate treatment is recommended and pushed to the *De-Stress Me* application's UI.

Our work has several limitations. Our current detection model relies on limited training data. For improved detection accuracy, we would need to utilize or collect a larger dataset that contains contextual data required for our system as well as signal data such as EKG to detect minute heart rate variability. While we designed the application user interface to collect manual input from the user's side, our current implementation does not train on this data. To summarize, we would like to examine state-of-the-art machine learning algorithms and datasets for an enhanced detection engine. An extensive exploration on stress treatment methods with the assistance of experts would also be critical for practicality. Based on our evaluation plan described in section 5, we would like to validate the effectiveness and usability of *De-Stress Me*. Another direction would be investigating ways to secure user's personal data by implementing privacy preserving measurement, secure data transmission, and encryption of the data.

REFERENCES

[1] How does everyday stress impact your heart?, Aug 2018.
[2] Akka. Akka/akka: Build highly concurrent, distributed, and resilient message-driven applications on the jvm.
[3] Anliker, U., Ward, J. A., Lukowicz, P., Troster, G., Dolveck, F., Baer, M., Keita, F., Schenker, E. B., Catarsi, F., Coluccini, L., et al. Amon: a wearable multiparameter medical monitoring and alert system. *IEEE Transactions on information technology in Biomedicine 8*, 4 (2004), 415–427.
[4] Bauer, G., and Lukowicz, P. Can smartphones detect stress-related changes in the behaviour of individuals? In *2012 IEEE International Conference on Pervasive Computing and Communications Workshops* (2012), IEEE, pp. 423–426.
[5] Bogomolov, A., Lepri, B., Ferron, M., Pianesi, F., and Pentland, A. Daily stress recognition from mobile phone data, weather conditions and individual traits. In *Proceedings of the 22nd ACM international conference on Multimedia* (2014), pp. 477–486.
[6] Gasperin, D., Netuveli, G., Dias-da Costa, J. S., and Pattussi, M. P. Effect of psychological stress on blood pressure increase: a meta-analysis of cohort studies. *Cadernos de saude publica 25*, 4 (2009), 715–726.

[7] Ha, U., Madani, S., and Adib, F. Wistress: Contactless stress monitoring using wireless signals. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies 5*, 3 (2021), 1–37.

[8] Hellhammer, D. H., Wüst, S., and Kudielka, B. M. Salivary cortisol as a biomarker in stress research. *Psychoneuroendocrinology 34*, 2 (2009), 163–171.

[9] King, Z. D., Moskowitz, J., Egilmez, B., Zhang, S., Zhang, L., Bass, M., Rogers, J., Ghaffari, R., Wakschlag, L., and Alshurafa, N. Micro-stress ema: A passive sensing framework for detecting in-the-wild stress in pregnant mothers. *Proceedings of the ACM on interactive, mobile, wearable and ubiquitous technologies 3*, 3 (2019), 1–22.

[10] Leproult, R., Copinschi, G., Buxton, O., and Van Cauter, E. Sleep loss results in an elevation of cortisol levels the next evening. *Sleep 20*, 10 (1997), 865–870.

[11] Liu, I., Ni, S., and Peng, K. Happiness at your fingertips: Assessing mental health with smartphone photoplethysmogram-based heart rate variability analysis. *Telemedicine and e-Health 26*, 12 (2020), 1483–1491.

[12] Marcovecchio, M. L., and Chiarelli, F. The effects of acute and chronic stress on diabetes control. *Science signaling 5*, 247 (2012), pt10–pt10.

[13] Mishra, V., Sen, S., Chen, G., Hao, T., Rogers, J., Chen, C.-H., and Kotz, D. Evaluating the reproducibility of physiological stress detection models. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies 4*, 4 (2020), 1–29.

[14] Mohney, G. Stress: What it costs u.s. economy, Jan 2018.

[15] O'Donovan, A., Tomiyama, A. J., Lin, J., Puterman, E., Adler, N. E., Kemeny, M., Wolkowitz, O. M., Blackburn, E. H., and Epel, E. S. Stress appraisals and cellular aging: A key role for anticipatory threat in the relationship between psychological stress and telomere length. *Brain, behavior, and immunity 26*, 4 (2012), 573–579.

[16] Taelman, J., Adriaensen, T., van Der Horst, C., Linz, T., and Spaepen, A. Textile integrated contactless emg sensing for stress analysis. In *2007 29th annual international conference of the IEEE Engineering in Medicine and Biology Society* (2007), IEEE, pp. 3966–3969.