Altering Heartbeat Using AI Selected Music
Development Plan

Team Members:
May Wandyez, Jeremy John, Ahmad Shah

Tables of Contents:

Project Overview

If your heart beats too fast, you die[1]. If your heart beats too slow, you die[2]. The solution to this problem is to control a person's heartbeat. A person's heartbeat naturally synchronizes to the tempo of music due to a phenomenon known as sympatho-respiratory coupling[3]. Being able to predict what a listener's heartbeat will be before they listen to a piece of music would mean that it would be possible to then select a piece of music to change a listener's heartbeat towards a desired heartbeat. This project aims to create a machine learning model that is trained to predict what the change in a user's heartbeat will be, and integrate this model into a desktop application that selects music for the user to listen to based upon what the user wishes their heart rate to be. The potential impact of this project is that it can mitigate the health risks imposed by unsafe heart rates by allowing a user to control their heart rate with technology.

Project Purpose, Scope, Objectives

The purpose of this project is to allow a user to control their heart rate using music. The goal of this project is to develop an application that allows the user to control their heart rate using music. The scope of this project is limited to developing a desktop application for steering heart rate using a machine learning model that connects to ANT+ communication protocol enabled heart rate monitors. This scope does not encompass developing a mobile application, using non-ANT+ enabled heart rate monitors, or monitoring heart health information.

The project objectives will consist of developing a prototype desktop application that alters a user's heart rate through selecting music that will bring the user's heart rate closer to a target heart rate selected by the user. When the user opens the application, it will automatically connect to the user's wearable ANT+ compatible heart rate monitor and scan for mp3 files. The user will then select their desired target heart rate from a series of predefined choices (e.g. higher, lower). The application will then generate the user's predicted change in heart rate for all music files scanned using a machine learning algorithm trained to predict heart rate. The application will then select whichever mp3 file gets the user closest to their target heart rate and automatically play it for the user. This process will repeat for the target heart rate until the

---

[1] Ventricular tachycardia occurs when the ventricles of the heart beat abnormally fast - leading to insufficient blood flow to the rest of the body as the heart pumps inefficiently at high heart rates. Around 300,000 deaths occur in the United States every year from ventricular tachycardia. Tang PT, Shenasa M, Boyle NG. Ventricular Arrhythmias and Sudden Cardiac Death. Card Electrophysiol Clin. 2017 Dec;9(4):693-708.

[2] Ventricular bradycardia occurs when the heart beats too slow, which can lead to heart failure, fainting, and cardiac arrest Scherlag, B. J., Kabell, G., Harrison, L., & Lazzara, R. (1982). Mechanisms of bradycardia-induced ventricular arrhythmias in myocardial ischemia and infarction. *Circulation*, *65*(7), 1429–1434. https://doi.org/10.1161/01.cir.65.7.1429

[3] Watanabe, K., Ooishi, Y., & Kashino, M. (n.d.). Heart rate responses induced by acoustic tempo and its interaction with basal heart rate. PubMed Central (PMC). https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5339732/

user stops the process.

Objective:	**Measurement:**
Get user heart rate.	Live feed of user heart rate accessible on desktop.
Get music characteristics.	Length, tempo, and pitch of any given .mp3 file accessible.
Predict change in heart rate.	Predictions successfully predict direction of change in user heart rate with 70% accuracy on test data.
Choose song for user.	Application chooses song. Chosen song moves user's heart rate in direction of target heart rate.

Team Organization

The team will communicate primarily over the messaging application discord. A dedicated communications channel within discord has been reserved for assigning tasks for all team members. The team consists of the following team members in the following roles:

May Wandyez - Team Lead/Documentation lead :
May Wandyez is the team lead. The Team Lead's responsibilities consist of ensuring that all project deliverables occur on time, scheduling meetings with the class GTA, determining immediate project priorities, distributing workload, and handling internal team conflicts.
The documentation lead's responsibilities include taking all notes during class and ensuring that these are available to all team members. This responsibility also extends to making sure that internal project documentation meets the required standards set by the Senior Capstone Project.

Jeremy John - Front End Lead
Jeremy John is the Front End Lead. The Front End Lead's responsibilities extend to making all decisions regarding the design and structure of the front end of the application. The Front End lead is tasked with making sure that the functions defined in the backend operate properly, and that the user experience is streamlined and simplified. The Front End Lead will also make sure that the user interface is friendly and easy to use.

Ahmad Shah - Back End Lead
Ahmad Shah is the Back End Lead. The Back End Lead is responsible for ensuring that the various backend processes communicate with each other and the front end without issue. This responsibility extends to ensuring that the various backend processes run in parallel - not just in series, to ensure faster operation. Lastly, the Back End Lead is also responsible for creating scripts to ensure features are created and correctly implemented.

Problem Resolution Policies

The project members have agreed on strict problem resolution policies.These policies are intended to ensure that all team members are aware of what is expected of them and that all team members have a voice in ensuring team members are treated equally. The policies are listed in order of severity: failure to meet deadlines, failure to attend meetings, and all other problems.

Tasks assigned within the team are deliberately designed to be able to be completed within a small time period; therefore, failure to complete a task is automatically considered to be a breach of trust by the offender. If a team member fails to satisfactorily complete their tasks as assigned as determined by the team lead, all team members shall be notified by email within 24 hours of the deadline by the team lead. This email shall concisely explain what the task was, what was expected of the team member, and why what was turned in did not meet those expectations. The offending team member is required to explain within 24 hours via email to all members why they failed to complete their task, and how they propose that the task be completed so as to not hamper development of the project. If any team member is dissatisfied with this explanation, they shall email all other team members within 24 hours explaining why this explanation is unsatisfactory, and schedule a meeting with the GTA to bring the failure to their attention. Should this occur three times then any team member can move for expelling the team member from the team. This process assures that all disputes on deadlines can be completed within a 72-hour time period with a satisfactory paper trail to justify all decision making.

The team meeting schedule has been immaculately designed to fit within the schedules of all team members - it is a contract for all members to be there to contribute to ensure the project is completed. Failure to adhere to the team meeting schedule by missing team meetings without prior excused absence, agreed upon absence, extraordinary circumstances, or illness shall result in an automatic strike. For each strike the team member shall be required to explain to all other team members within 24 hours via email the reason for their absence, and how they will make up for the missed meeting. The team lead shall then notify the team members of the strike via email within 24 hours of the missed meeting. Upon the agreement of all team members, the strike can be revoked. The accumulation of three strikes or more shall result in an automatic meeting with the GTA to decide if the team member’s absences were justified, and if the team member is capable of contributing to the project sufficiently to justify remaining on the team. If in the opinion of all other team members and the GTA that the offending team member is unable to

remain a part of the team, the team member shall be removed from the team. Due to meetings occurring every weekday, this means that all meeting related issues will be resolved within a 72-hour time period.

Any other problem occuring between team members must be brought to the team's attention via email to all team member's. The complainant must make clear what they believe the issue to be, and how they wish to address it. The team members then have 24 hours to respond to this issue brought to attention with their own email which must include specific plans to resolve the problem, or concise reasoning as to why the issue is not a priority. If the issue is not resolved after this process is repeated twice the team will meet with the GTA for final judgment and the team shall be compelled to follow whichever plan of action is selected. This process ensures that all severe issues will be addressed within a 48-hour time period.

Project Plan:

Weekly Meeting Schedule:
To accomplish broad objectives, the team has established a weekly meeting schedule. This schedule has been mutually agreed upon by all team members so that all can attend regularly. This schedule has also been built with redundancy in mind - meaning that if something should happen to any team member during the week then there will be ample time during the week (and if necessary the weekend) to make up for any failure.

Weekly Meeting Schedule					
Day:	**Monday**	**Tuesday**	**Wednesday**	**Thursday**	**Friday**
Meeting Time:	12:00pm-4:00pm	4:30pm-6:00pm	12:00pm-4:00pm	4:30pm-6:00pm	12:00pm-5:00pm
Location:	In Person	In-Person	In-Person	In-Person	In-Person
Purpose:	Feature hackathon to develop features in person together. Assign tasks for midweek review.	Presentation and document review meeting. Practice for presentations.	Midweek-meeting. Ensure goals from Monday are met, assign new tasks for Friday. Merge prior tasks on github.	GTA meeting and progress review from midweek goal determination.	Feature hackathon - complete all features not yet finished from midweek task assignment. Merge completed tasks on github

Project Schedule:
A project plan has been developed which includes what general features of the project must be done and by when. All project deadlines are deliberately set a few days before in class Senior Capstone deadlines to ensure time for quality control and to avoid a 'last minute scramble' should deliverables take more time than initially allocated. It is extremely important that the schedule be followed or else the project will not be completed on time.

Project Schedule	
Deadline:	Objective:
6/1/2024	**Prototype 1 Deadline: A working application.** Live heart rate monitor feed accessible by front end and machine learning model training tool. Training tool has capability to automatically record heart rates and song characteristics at beginning and end of song and automatically saved these results to a .csv file Machine learning model is capable of being compiled off of csv data and predicting change in heart rate. Front end can scan all present music files in the music folder, generate predictions using an instance of the machine learning model, and select a song to achieve a target user heart rate.
6/9/2024	**Software Requirements Specification Deadline.**
6/16/2024	**Proof of Ethics Presentation Deadline.**
6/22/2024	**Design Document Deadline.**
7/2/2024	**Prototype 2 Deadline: More features and prettier.** Machine learning model rebuilt to improve accuracy. Back end has the ability to run audio analysis, heartbeat gathering, and prediction features simultaneously to remove unwanted delays. Front end has integration of 'approach path' feature where users can select how the next song is selected (e.g. gradual increase in tempo, steep increase in tempo, alternating increase and decrease in tempo). Integration of resting heart rate gathering feature by front end.

7/7/2024	**Test Plan Deadline.**
7/20/2024	**Final Version Deadline: Refined features and beautiful.** Application has all previously listed features with a refined UI based on feedback. Test plan has been implemented and most prominent issues resolved.

Configuration Management Plan:

The team will utilize a private github repository for version control of project code. Each team member is expected to create a fork of the main repository for each task assigned, which shall be merged during the weekly github merges on Wednesday and Friday pending code review by the Team Lead.

Tasks and deadlines are assigned by the Team Lead. Tasks will be reviewed by the team lead upon completion, and a new task shall be assigned when completed. Progress will be collectively by all team members reviewed at the Monday, Wednesday, and Friday meetings.

Technologies:

The team will use the following technologies to implement the required features of the project.

Technology Stack		
Technology:	Programming Language	Purpose:
Garmin Vivosmart 4	C, C++, C#	The Garmin Vivosmart 4 is the ANT+ compatible heart rate monitor. The current heart rate of the user is necessary in order to predict the future heart rate of the user.
Garmin USB ANT Stick	C, C++, C#	The Garmin USB ANT stick acts as a radio receiver for ANT+ wireless communications. This USB stick is necessary to receive the heart rate from the Garmin Vivosmart 4.
Windows 11	C, C++, C#	Windows 11 operating system has been identified as compatible with ANT+ communications drivers which are necessary for communicating between the application and the heart rate monitor.
python 3.12	python	python will serve as the general backend for this project.
ANT+ 2.90	C, C++, C#	ANT+ is a wireless communications protocol established by Garmin Ltd. ANT+ allows communication with proprietary

		Garmin devices - including heart rate monitors.
openANT 1.3.1	python	openANT is a python implementation of ANT+ wireless communications protocol - allows python programs to communicate with ANT+ heart rate monitor devices.
pyusb 1.2.1	python	pyUSB is a requirement for openANT and allows python to communicate to USB devices.
librosa 0.10	python	librosa is a python package for audio analysis. librosa allows audio files to be broken down to their base characteristics (tempo, pitch, length). It is necessary to break audio files down to their base characteristics in order for the machine learning model to be able to identify how a music file will impact heart rate.
PyTorch 2.3	python	PyTorch is a python library for construction of machine learning models. PyTorch will be used to create a machine learning model that will predict heart rate based on music characteristics and current user heart rate.
HTML5	HTML5	HTML5 will serve for creating the front end. HTML5 works on nearly every machine, developing the front end in HTML5 ensures it can be reused if backend operations are shifted to a remote server instead of running directly on the user's machine.
Flask 3.0.3	HTML5, python	Flask allows communication between a HTML front end and a python based backend.
Flask-Bootstrap 3.3.7.0	HTML5, python	Allows integration of the popular 'bootstrap' framework into Flask. This allows the frontend to be more responsive and integrate javascript features.