COMP3000 Computing Project 2024/2025

KANO: A Mood Analysis Voice Journal

Links

Source code: *https://github.com/Esther-Skillman/KANO-A-Mood-Analysis-Voice-Journal*

Contents

KANO: A Mood Analysis Voice Journal 1
- Links 1

Project Vision 2

Technological Architecture 3
- Initial Architecture 3
  - Mobile Application UI 3
  - Back-End Server 3
  - AI Processing 3
- Languages and Technologies 3
- Development Environment 4
- Repository Layout 4
- Test Environment 5

Risk Plan 5
- Assessment and Mitigation 5
- Risk Matrix 6

Initial Product Backlog 6
- Overview 6
- MVP 7

High Level Plan 7
- Proposed Gantt Chart 7

Project Management Tools 8

Keywords 8

Project Vision

FOR those who have an interest in their own emotional wellbeing, journaling and embrace new technologies involving AI. With these core characteristics, the intended clients' demographics would be largely age 16-25 with their increasing usage of AI. Additionally, they would be expected to already be existing members of the journaling community, allowing for an easy integration of their current journaling habits to the niche that is voice journalism with mood analysis. Furthermore, the culture would be English speaking and the same as the one represented in the primary dataset(s) used. Other demographics such as gender, economic and social class are intended to not impact the client target audience, but this could be revised later during research and development.

WHOSE problem is addressed is an individuals' inability to conduct an unbiased reflection of one of their many emotional expressions, in this case, speech. According to Whilst the common method of reflection for today is writing, these entries can be written over, once, twice or even three times with technology, obscuring a person's true emotions to how they think they should feel as opposed to what they actually feel. Voice entries help remove this obscurity, as individuals can't 'rewrite' their speaking as easily, providing a raw entry of sorts but likewise adds a significant key benefit of listening to not only what they say but HOW they say it. Whilst humans are able to discern the emotions of speech with minimal effort, such as anger, happiness or fear, this can fall short when they're the ones analysing themselves. The process can become bias, thrown behind the lens of their own perspective and this is when technology can intervene - providing an unbiased report to aid an individual in their own reflection.

THE PRODUCT KANO which stands for 'Keep A New Outlook' will be a mood analysis voice journal that will use AI to discern the emotions of a person's voice entry.

IT IS A mobile application that will initially address android mobiles, but compatibility with android and iPhone will be explored during development. Server-side options will be researched and reviewed further into development, dependant on many factors such as computational power, availability of server technologies and greatest suitability for mobile compatibility.

THAT which is the fundamental premise is a user will record themselves speaking and the AI will conduct a live analysis after they've finished, giving numerical and qualitative feedback on what emotion(s) the user displayed during the voice entry. This premise works to achieve the overall goal of the application being a tool for a user's mental wellbeing, as they're able to reflect on an unbiased and numerical picture of the emotion(s) they portray. This will enable them to approach their mental health with a more objective viewpoint and keep a record of the frequency and trends of their emotions over time.

Technological Architecture

Initial Architecture

The architecture of the KANO project will involve three key layers: the user interface (UI), the backend server for computational tasks, and AI processing for Speech Emotion Recognition (SER). (Ref Fig 1)

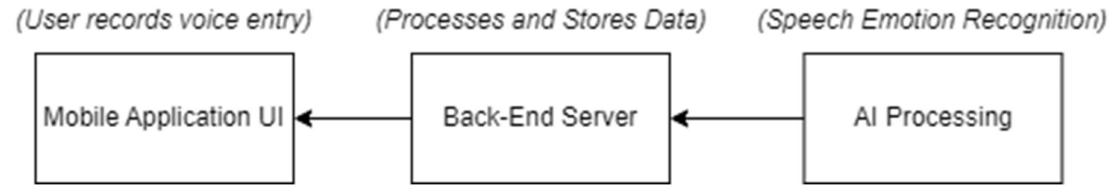

Figure 1

Mobile Application UI

The mobile application will act as the front end, allowing users to record voice entries. These entries will then be sent to the backend for processing with an SER model (AI).

Back-End Server

The backend server will primarily be responsible for interacting with the UI, storing user data and sending the AI voice data to return a qualitative and quantitative analysis to then send to the UI. User data must also be securely saved, suggesting an encryption method may be used

AI Processing

Research and development within the project will reveal how the AI processing will interact with the back-end server to the point they may either be completely independent or merged entities. Treating AI processing as a separate entity for now, this will be where the voice data is processed on a pre-trained SER model to gather a mood analysis and is then sent back to the UI.

Languages and Technologies

AI and Machine Learning

Python will be used for the SER (Speech Emotion Recognition) model development, likely utilising libraries like TensorFlow and PyTorch for deep learning and machine learning tasks. Access to university servers may be used to support increased demand for computational speed and power (Ref Risk: R8).

Mobile Application

JavaScript (subject to change) will likely be used within the Android Studio environment for front-end development. The app will be initially developed for Android, with a possible future expansion to iOS if project management goes as planned.

Backend
JavaScript (subject to change) will likely be used for server-side handling, processing requests from the UI to the AI and maintaining/sending user data dependant on data storage choice
Data Storage
This is largely undecided as it is heavily dependent on the outcomes of development, although an option that will likely be explored is SQL Query for SQL databases, and the technologies may include cloud-based storage.
Dataset
Kaggle datasets have been and will be used for initial training and model evaluation, allowing myself as a developer to familiarise myself with key concepts of SER with smaller datasets such as CREMA-D, RAVDESS, SAVEE and TESS. The actual dataset(s) used will be explored with the help of my tutor to establish a suitable set to train a SER model.

Development Environment

IDE: Development will be done primarily in Android Studio for the mobile application and IDEs such as VS Code or PyCharm for Python-based AI development.
Version Control: GitHub will be used to manage version control.

Repository Layout

Initial Repository Layout:

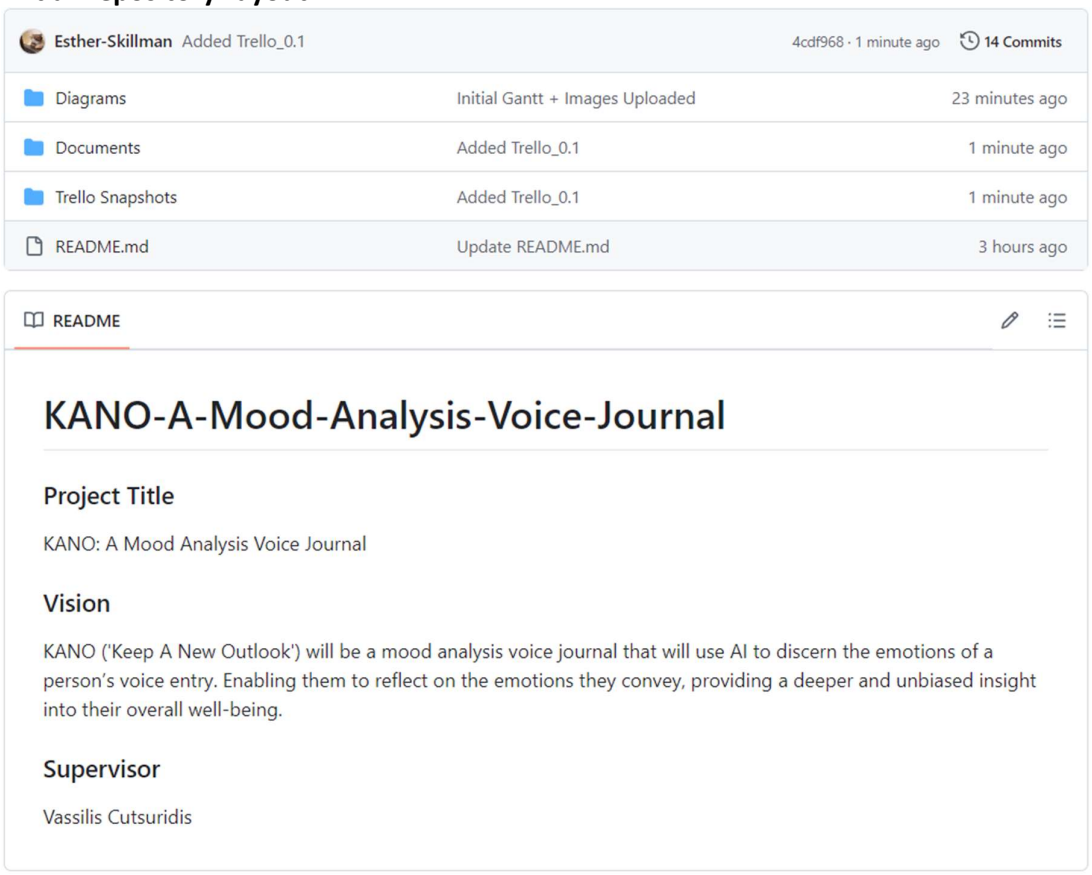

Figure 2

Future Repository Layout:
Will include source code in following development, expecting to populate the repository with files such as “front-end”, “back-end”, “SER AI” etc. through a thorough incremental-based work cycle as evidenced by frequent commits.

Test Environment

Testing will be naturally incorporated into the process with the AI, as it learns to recognise emotions from a given audio dataset. Unit testing and integration testing may also accompany both client and server sides of the product, ensuring the mitigation of common bugs and errors.

Testing the entirety of the project will include alpha and beta testing encapsulated in a UAT towards the end. Alpha testing being done with myself as the developer and my supervisor overseeing my project. Beta testing on the other hand will involve end users to receive more detailed feedback, likely including a survey to gather qualitative and quantitative information back. This data will be analysed and considered to implement new deliverables.

Risk Plan

Assessment and Mitigation

Num	Identified Risk	Effect	Likelihood	Impact	Risk Ranking	Mitigation Strategies
R1	Unrealistic Time Estimates	Too little or too much time is allocated to a task, that could mean deliverables are met later, impacting the overall critical path. On the other hand, too much time could encourage scope screep if critical path appears to move ahead of schedule.	Almost certain	Crucial	Very high	Incremental development, supervisor feedback, analysis of past work ethic/projects, multiple time estimation techniques, revision of schedule after each sprint
R2	Personnel shortfalls	Lack of skills and knowledge could increase likelihood of poor time efficeny and increase the duration of meeting a deliverable	Almost certain	Moderate	High	Supervisor teaching and resources, training via online courses, university support e.g. writing cafe
R3	Missed deadlines for development	Delays critical path, features, documentation and sprints are not met on time. This butterfly effect could impact final deadline deliverable	Possible	Major	High	Schedule contingency time for possible overruns, requirements revision, assess critical path, revise and extend schedule for similar developments
R4	Personnel unexpected event (illness etc.)	Deliverables are met late and work productivity and efficency could go down.	Unlikely	High	Moderate	Schedule potential sick days, assess impact to critical path, keep work incremental and spread out, supervisor meeting to evaluate next steps
R5	Product doesn’t meet requirements	Client (end users) produce feedback on product that increases the workload, wether it be to add more, less or start over to meet requirements.	Unlikely	High	Moderate	Prototyping, supervisor feedback, consistent evaluation of requirements during development, software research, User testing and feedback (UAT)
R6	Gold plating	Can accumulate an increased workload, taking more time and could cause a delay to the critical path	Likely	Low	Low	Assess design time-cost, prototyping, requirements scrubbing
R7	Development too technically challenging	Time is consumed having to solve a particular challenge in the product, making it hard to tell if and when a deliverable is met until the problem is fixed	Possible	High	High	Technical analysis with supervisor feedback, technical training, requirements scrubbing
R8	Lack of computational power and speed	Processes involved with machine learning could take prolonged periods of time on developer's device, postponing development of other devlierables until the processes have been completed	Almost certain	High	Very High	University servers, use other university resources (computers etc.)

Figure 3

Risk Matrix

Risk Exposure			
Ref	Likelihood	Impact	Risk Exposure
R1	5	5	25
R2	4	3	12
R3	3	4	12
R4	3	3	9
R5	2	4	8
R6	2	4	8
R7	3	3	9
R8	5	3	15

Risk Assessment Matrix					
	1	2	3	4	5
1					
2					
3					
4					
5					

Ignore	Consider	Take Action

Initial Product Backlog

Overview *(subject to change):*

- Front-End: Android Application
- Back-end: Handles validation and management of data to AI and data storage
- Voice Recording Feature: Basic UI for recording voice entries.
- Emotion Feedback UI: Display the results of the SER analysis to the user.
- SER Model: Trained AI to recognise emotions in speech
- Data Storage: Saves recorded entries

MVP *(subject to change):*

- Fully trained SER Model
- Minimum UI to interact with SER Model
- Ability to record (at least) one entry
- Analysis on (at least) one entry

High Level Plan

Proposed Gantt Chart

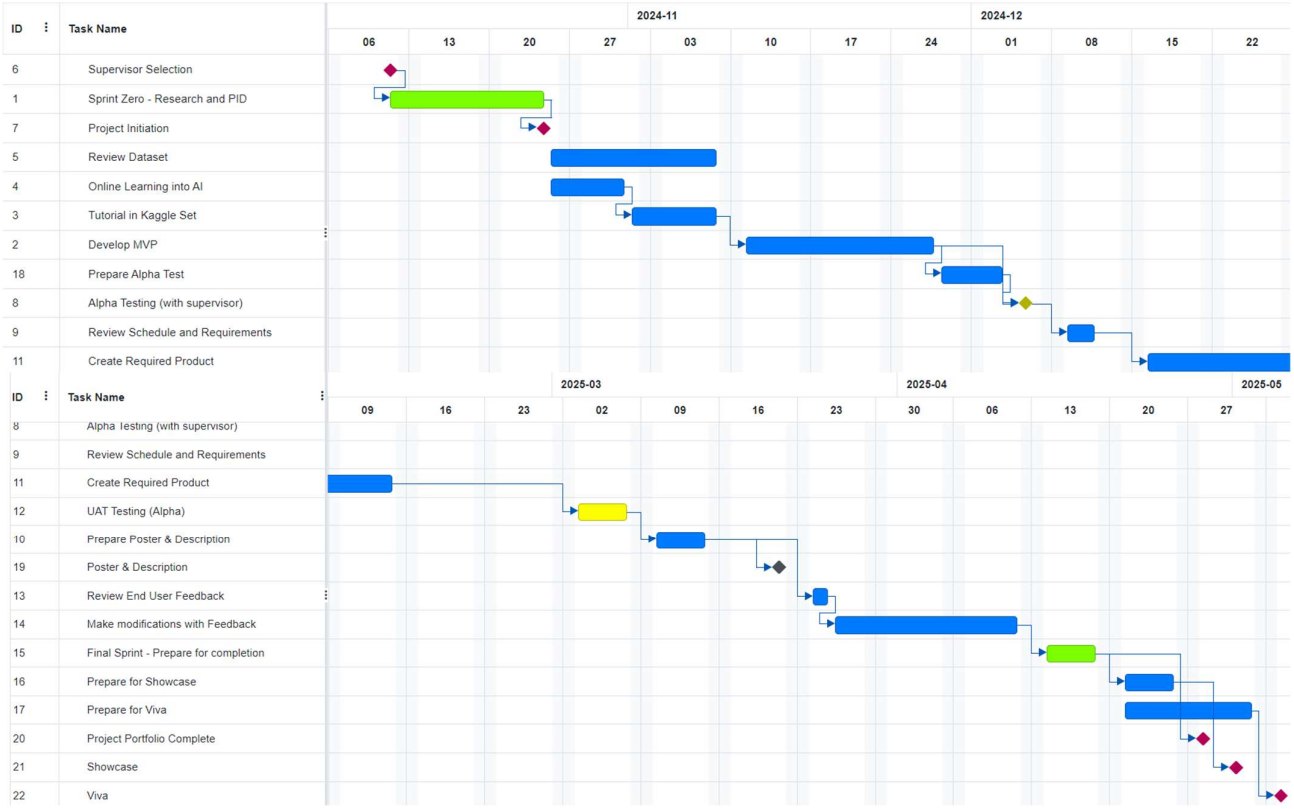

Figure 4

This is the basic overview of how development is expected to unfold. The initial proposal includes key dates of hand-ins, their respective preparation, key stages of development and dates of testing.

Sprints, other than sprint zero and final sprint, have been excluded from the Gantt framework for now due to a lack of knowledge and understanding on how development is expected to progress over the overall 12 sprints for a 24-week period.

Additionally, as the project does progress, time estimates are very likely to change and work packages broken down into finer objectives once the product backlog is finalised.

Project Management Tools

I'll be using Trello (Trello Snapshots on GitHub) and the online Gantt Chart Resource as my main project management tools for the project.

Below is a standard Kanban template board populated with the initial product backlog. Ensuring I manage my agile development process and time effectively.

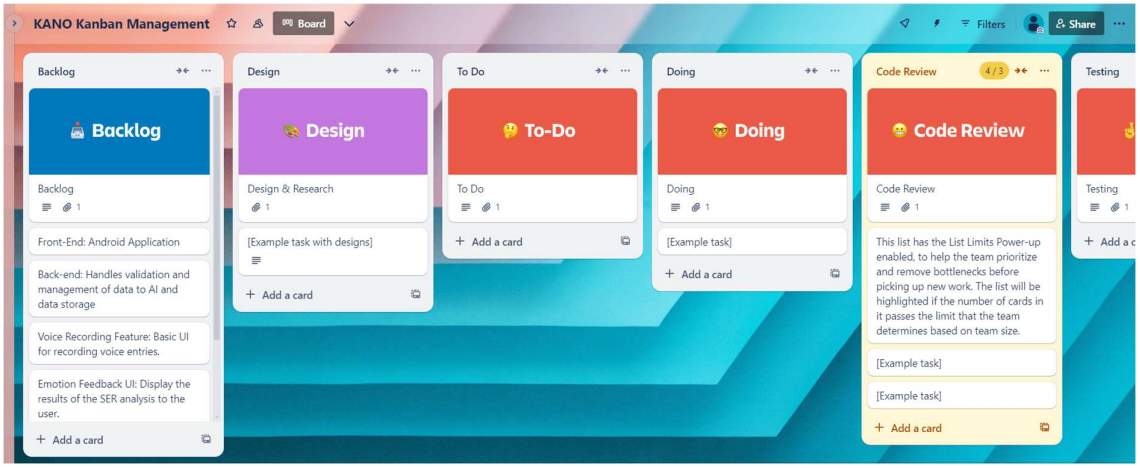

Figure 5

Keywords

voice, speech, SER, journal, dairy, mental health, healthcare, mood, emotion, analysis, AI, machine learning, deep learning, pattern recognition, mobile application, phone, android, iOS