**INTRODUCTION:** Our project focused on creating an affirmations app with the goal of promoting positivity and enhancing people's daily morale through uplifting affirmations.

We successfully created a user-friendly program that effectively serves its intended purpose.

An overview of the report's roadmap can be found within the group activity log but to summarise ([GROUP ACTIVITY LOG](https://docs.google.com/spreadsheets/d/1iSRKo-YPGSkqCuGZ1z7IPP3QrfpjmXzfEP-I16K0ocM/edit?usp=sharing)):

* What we wanted was to create an affirmation app
* What we managed to do was establish an app featuring a navigation tab, allowing users to access different sections of the app. We were able to integrate a save button that saves the users favourite quotes to a database. One of our favourite features is the mood tracker which allows the user to click on the emoji that relates to their mood, this will be stored on a separate page with the date and time
* What we didn’t manage to do but wanted to do was have a share feature so that favorited affirmations could be shared on social platforms. We also wanted the app to take over the users homescreen at a particular time so that the user is reminded in a mre upfront manner to check in with themselves. Lastly, we wanted to have a downloadable easily accessible free app

**BACKGROUND:** In our finalised product, you are able access different pages using a toggle/menu navigation bar which is always accessible to the left of the window. By making it always accessible, we did not need to install a back button.

Pages that can be accessed: **‘Home’**, where a welcome greeting is presented along with our logo. **‘About’**, where users can understand the app better. **‘Affirmation’** where the user is able to generate an affirmation using a button. The affirmation is generated from the Zen quotes API. The user is also able to record affirmations using the ‘save affirmation button’. These affirmations are stored in a database. **‘Mood’**, where the user inputs their mood which is saved and can be reviewed in **‘My Mood'**. Similarly, **‘**💙**’** is where the user can review saved affirmations. And lastly, **‘Support’**, where the user has access to external support. Within this page is a link to a website where you type in your country and need (e.g mental health, abuse, bullying, loneliness, self harm etc), you are then presented with an array of numbers and information catered to that specific need. We wanted to be a beacon of light but also provide our users with additional direction if needed.

If this was a published app, we would make it freely available for download. This would allow the app to be more accessible to a variety of people who would benefit from wellness support without any financial restrictions.

**SPECIFICATIONS AND DESIGN:**

**Functional:**API connection - provides positive quotes. The randomly generated affirmations are not generated by us but instead come from the ZenQuotes API. When this function is called, at the click of a button, the affirmations are displayed to the user on the affirmations page

Mood tracker - place to store user inputted mood for user to refer back to after having selected a mood from a list of emojis. Each recorded mood entry will be stamped with the date and time it was recorded. The user will also have the option to clear their recorded mood history at their convenience

Navigation bar - essentially a toolbar that allows the user to go from page to page with ease for better user experience

Saved affirmations (‘🩵’) page - allows the user to view the affirmations they have saved after having clicked the saved affirmations button.

**Non-functional:**Usability - The interface should be user-friendly, allowing users to easily navigate and interact with the application. We did this by using a simple navigation bar

Performance - The application should respond to user interactions. The API fetches affirmations

Reliability - The app should be reliable however we have an exception handling error built into our code so that users are made aware if the API is down. We are relying on an external API and so we have no control over its reliability. Mood records and saved affirmations should be securely safely on SQL

Compatibility - The application should be compatible with popular web browsers such as Chrome, Firefox, and Safari. Needs wifi to open URL on support page and to access API

User Support - The application should provide user support within the about page

**Design**:User Interface - consistent colour scheme and font with a layout that is aesthetically pleasing. The calming colour scheme echos what our app represents: positive guidance to support and uplift users ‘mood’

Navigation - The navigation bar/sidebar should allow users to easily switch between different pages. Pages are also designed in a uniform manner with thoughtfully designed headers to indicate the purpose of the page

Affirmation Page - The Affirmation page should display a random positive affirmation fetched by the API. Users should be able to click a button to fetch a new affirmation and have access to a save button to save their favourites.

Mood Page - The Mood page should present users with emoji buttons representing different moods (Happy, Okay, Sad, Angry). When clicked, the data is stored

My Mood - The My Mood page should display a history of recorded time stamped mood entries

My Affirmations (‘🩵’) - The My Affirmations page should display a list of affirmations that the user has saved so that users can view their saved affirmations at their convenience

About and Support -The Support and About pages should provide information about the purpose of the application and offer resources for emotional well-being

**Architecture**: Frontend - The frontend has been developed using Tkinter as our framework for creating the graphical user interface. Each page of the application is presented as a separate frame, allowing for modular development

Backend - The backend has been built using Python with a modular structure where each page's logic is encapsulated in separate functions. Classes have been used for testing code

Database - We used a relational database (MySQL) to store users input: mood records and saved affirmations

API Integration - Implement a function to save fetched affirmations to the database.

File Handling - used to record a readable text file within the My Mood page where mood is recorded

Error Handling and Logging - Implement error handling to catch and handle exceptions

**IMPLEMENTATION AND EXECUTION: Roles**:

**Poppy** in charge of SQL related data, setting up the database where information is stored as the user navigates and interacts with the app.

**Bobbie** in charge of mood input GUI, creating buttons allowing pop up with mood confirmed and mood tracking.

**Maggie** in charge of affirmation API linking up, creating initial app pop up box and design

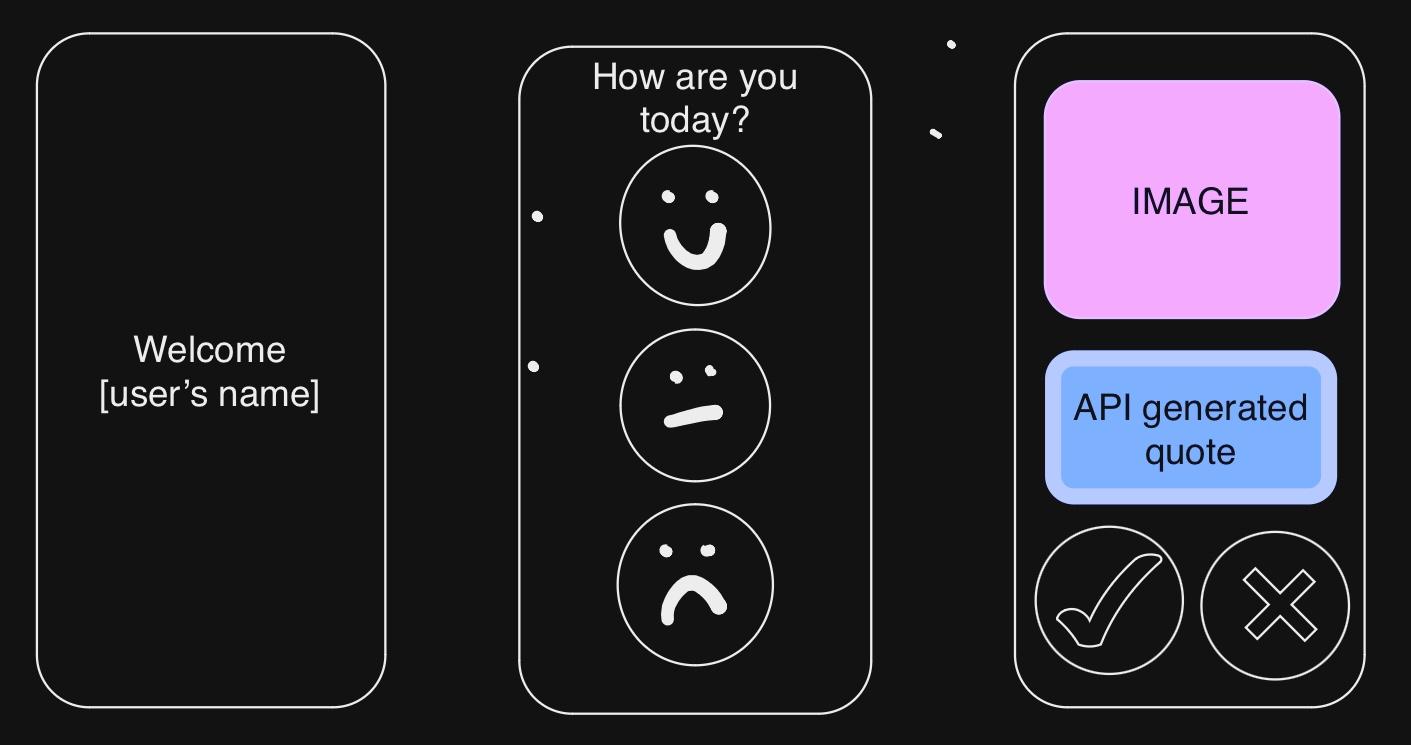
**Harmony** in charge of app layout and design, transfering code into layout

**Layefa** in charge of sorting most of the meetings, checking availability and setting up zoom calls, creating mood logo, starting code on save button

**All** Reviewing each other's code, playing around with it and suggesting improvements

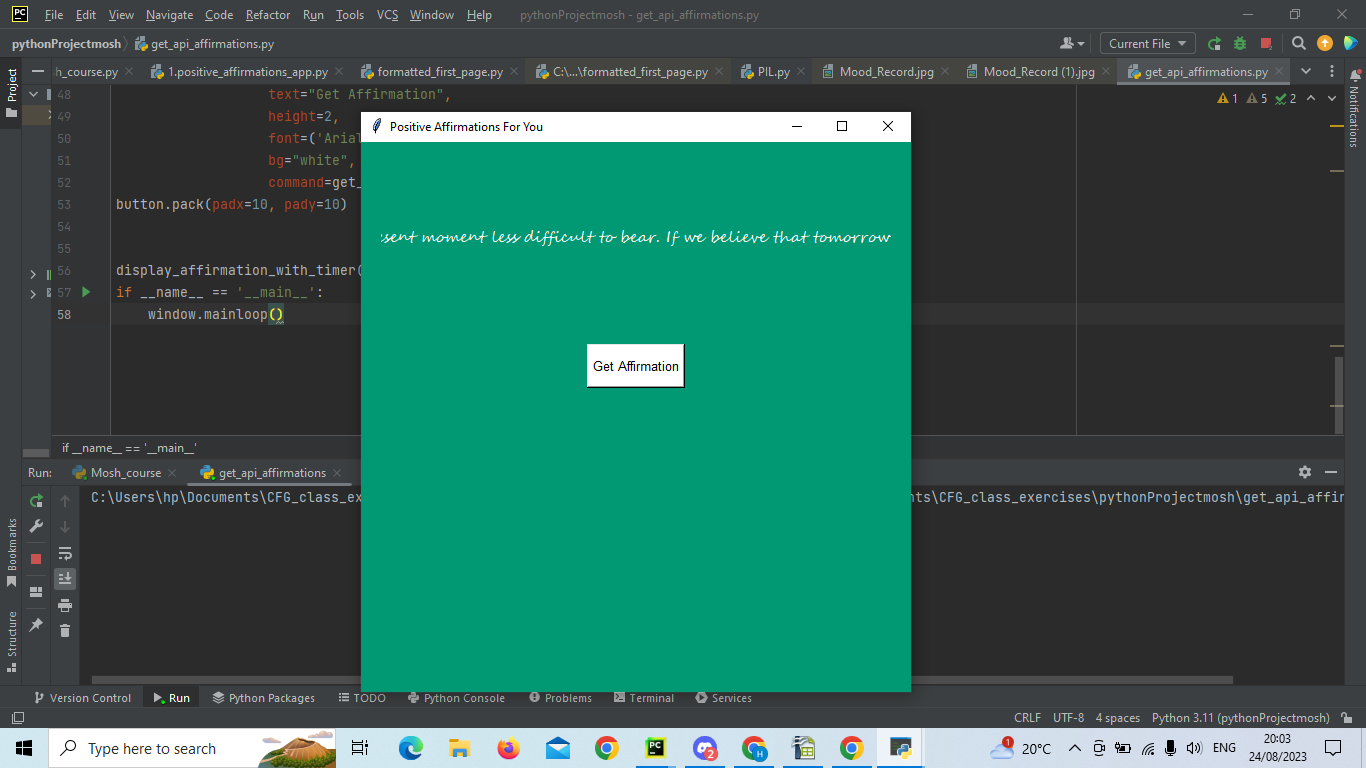
**Agile**: Meet at least once a week as a whole group. More times in smaller groups depending on availability. Our meetings were used as checkpoints for checking in/updating team progress on specific parts of the project.

Had a simple design in mind: a welcome page and a page where a positive quote could be retrieved. With this idea we asked for external feedback quite early on so that we knew exactly what users wanted and could go from there.

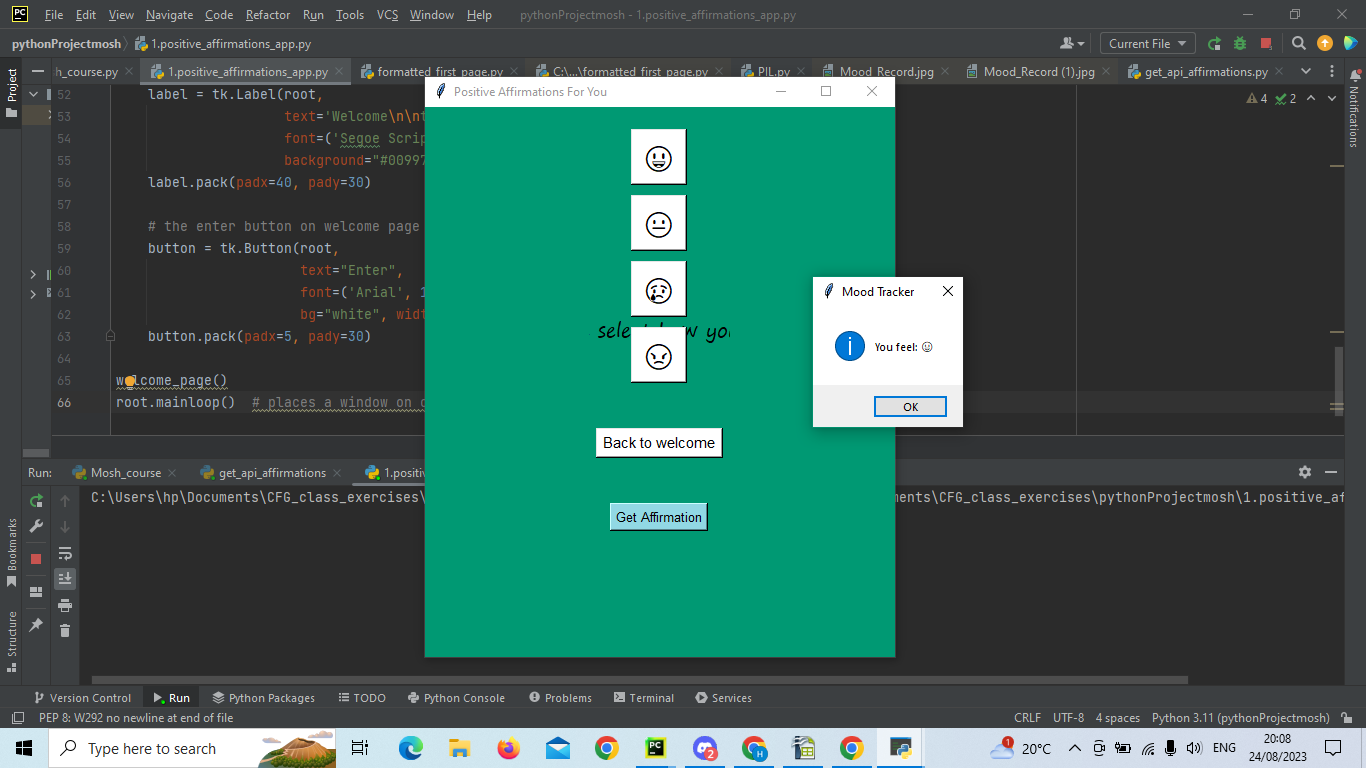
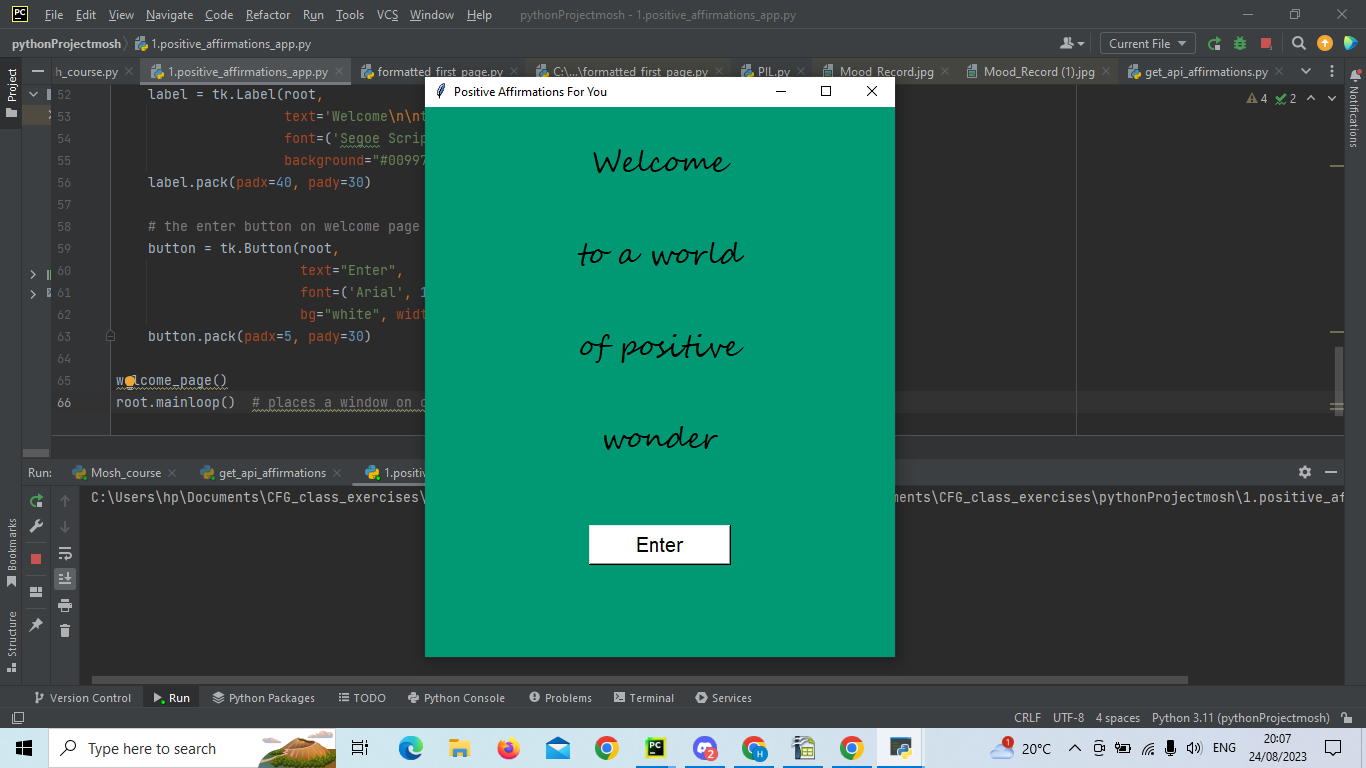


This was a sketch of what we wanted out of our design after also taking into consideration feedback from our questionnaire. Multiple people hinted that an image along with the affirmations would be more appealing to them:

Welcome page with personalised message, mood input, image and quote with a dislike/like button.

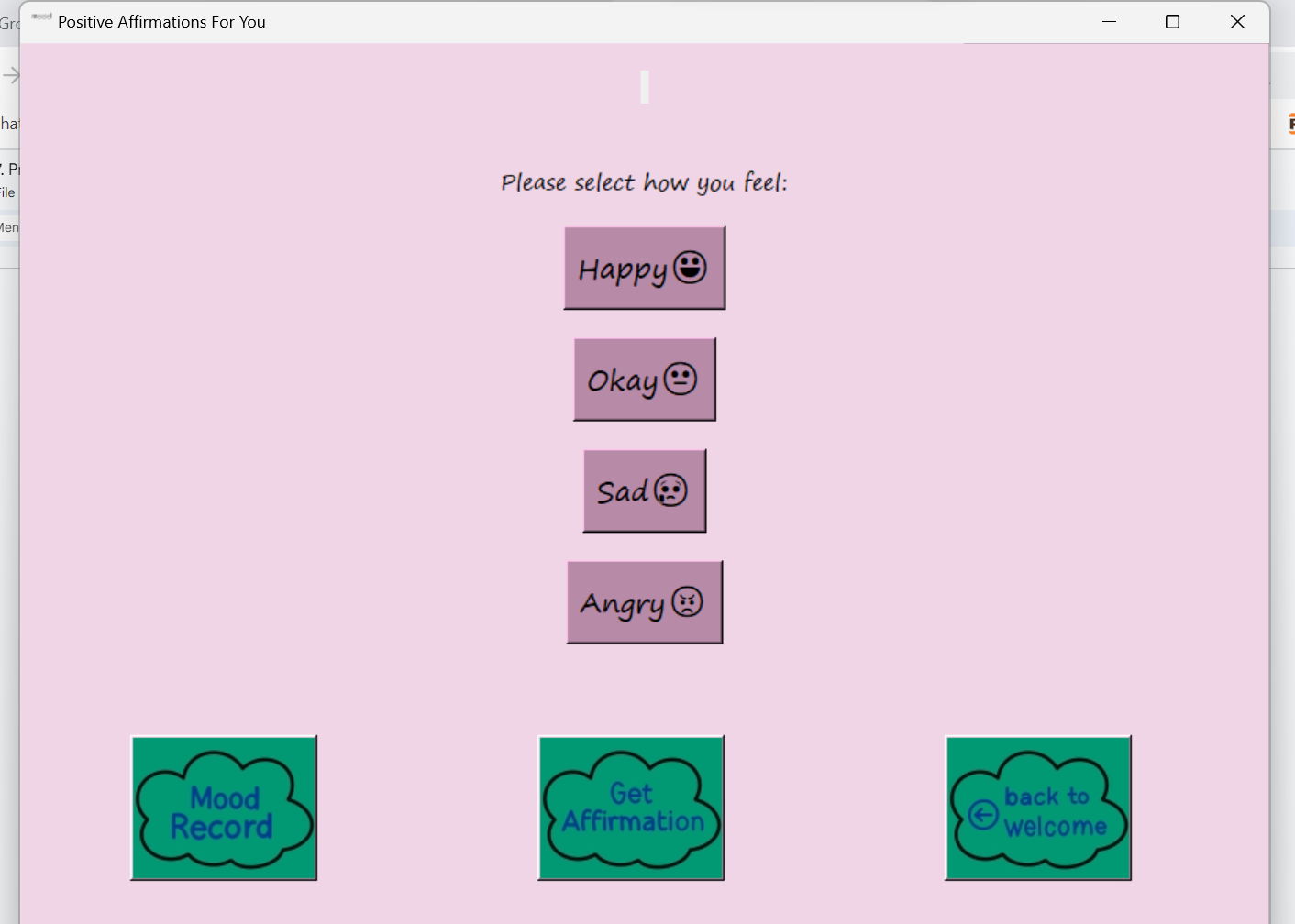
We chose not to input a custom welcome feature into the app because of our time restraint. We decided it was not fundamental in making an effective app which was our priority. The dislike button was not added because it was not needed for the overall functionality of the app. 

Initial implementation: Our first success was creating a page where the API was linked up. Maggie created this on a timer basis. The affirmation changed after 10 seconds. We regrouped and decided that having a timer was not what we wanted in the direction the app was going. It also made the ‘Get Affirmations’ button useless. We wanted our users to be able to enjoy the affirmations at their own pace:



2nd version: We managed to create a welcome page where the user could click ‘enter’, and be directed to a page where mood could be input and saved. We still needed to get the text at the top of page 2 as it was behind our buttons (picture on the right). The back button functioned correctly however we were still working on getting the API affirmations to pop up when that button was clicked:

Code used at this point: *1.positive\_affirmations\_app.py*



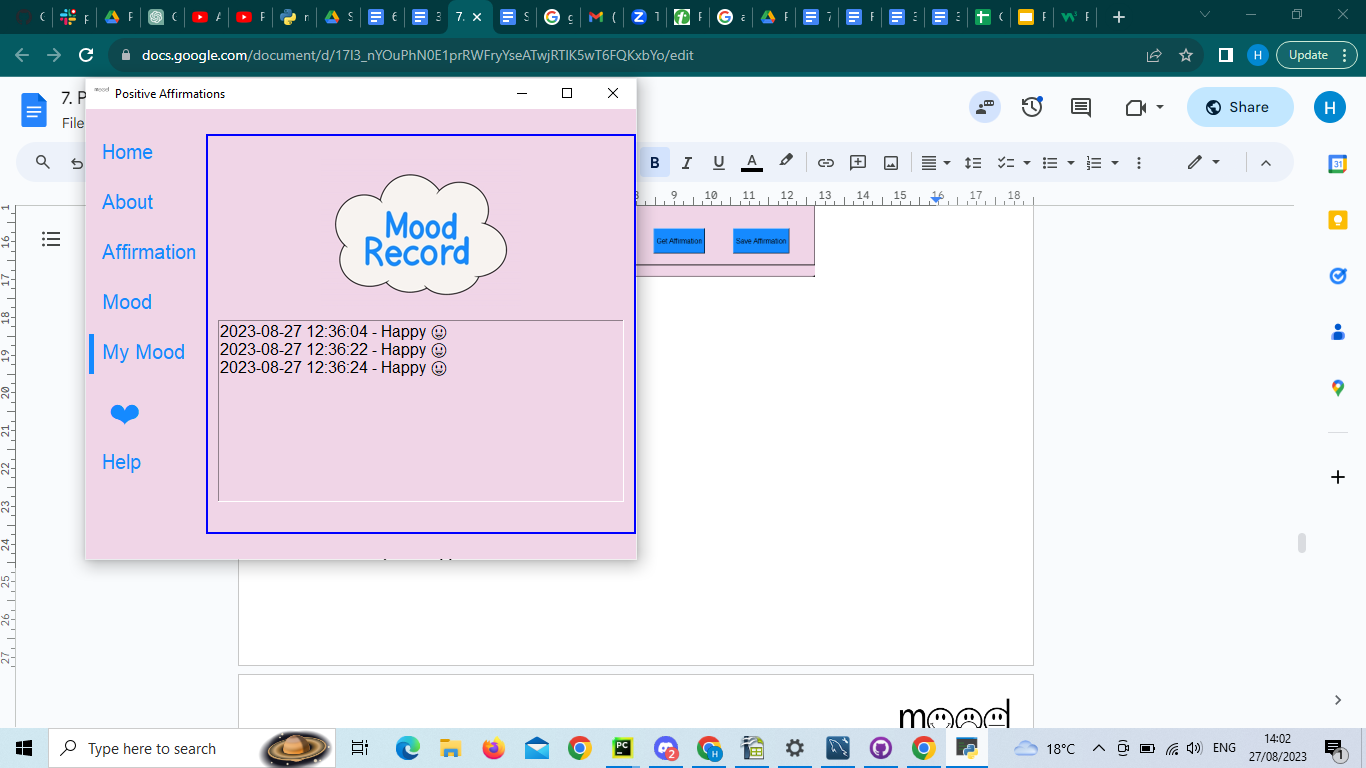
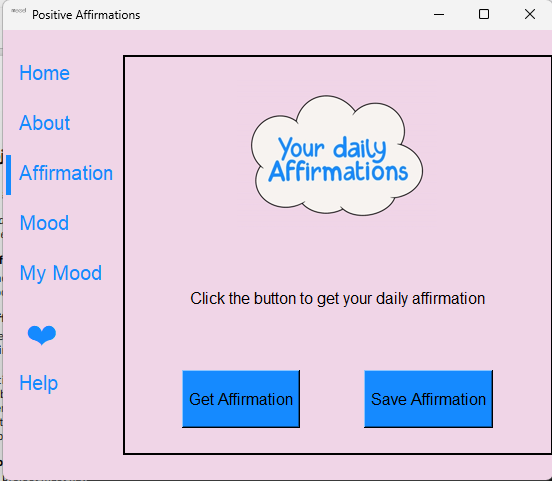
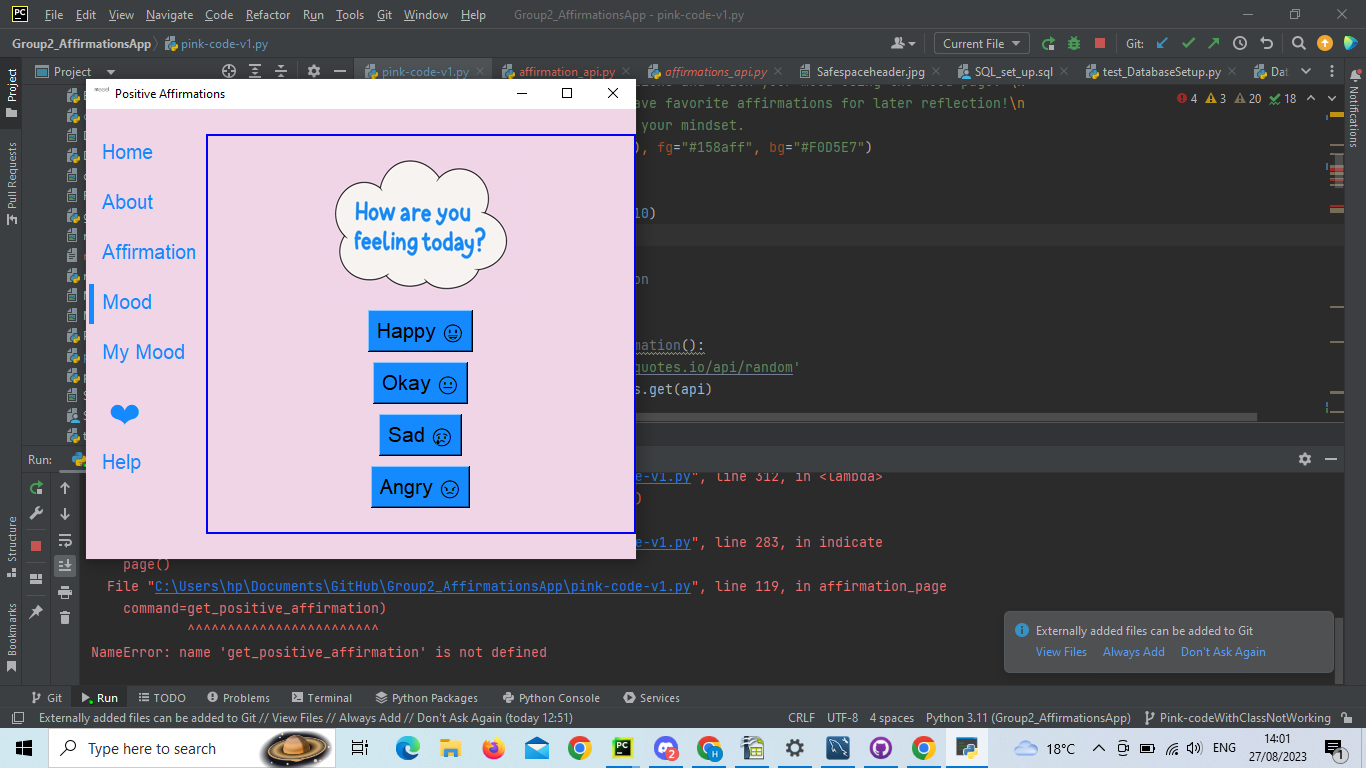
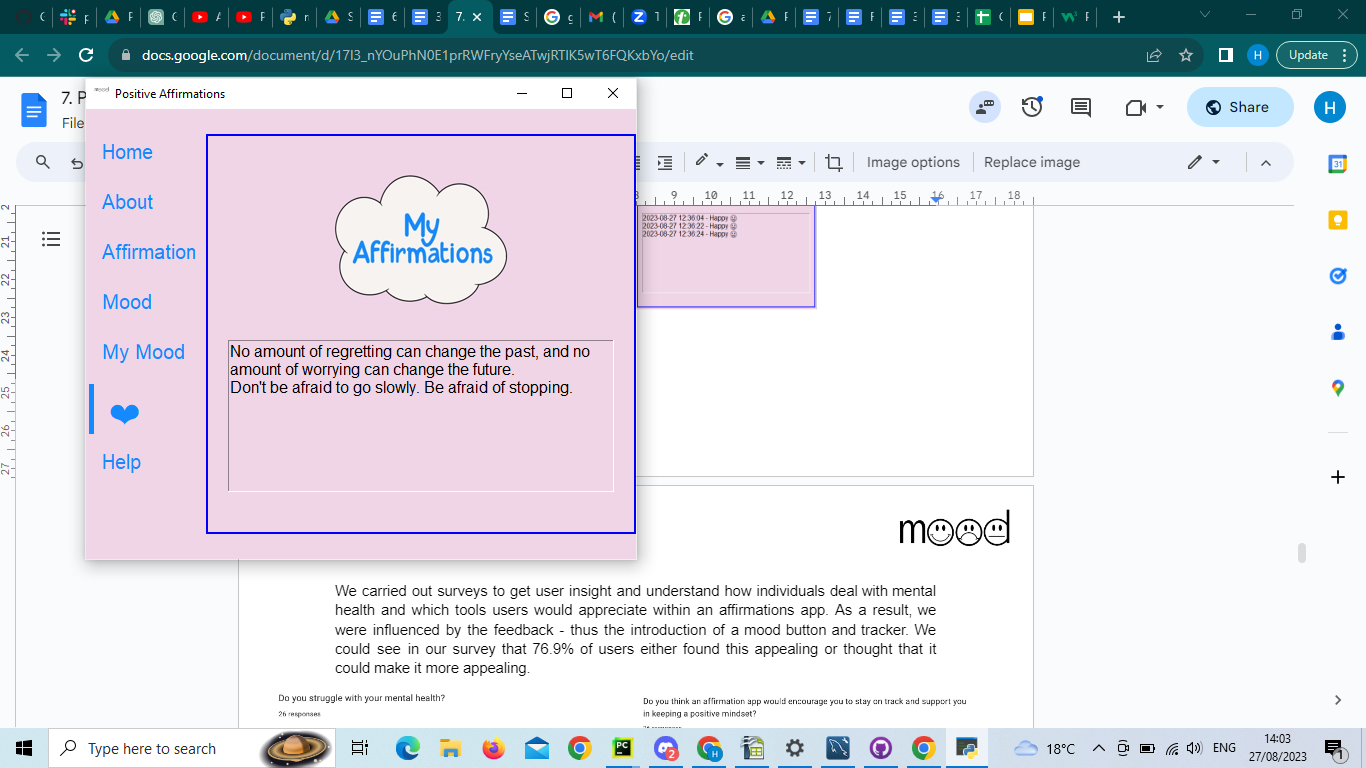
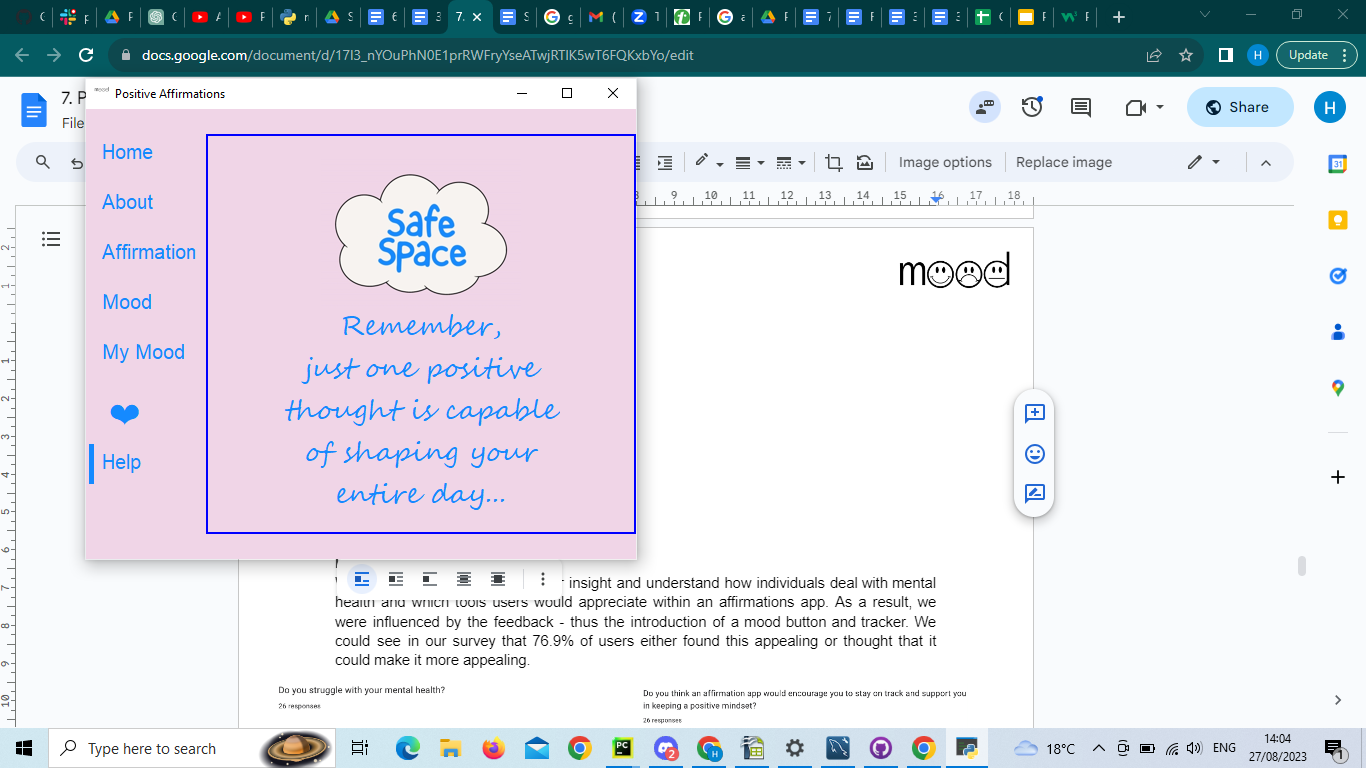
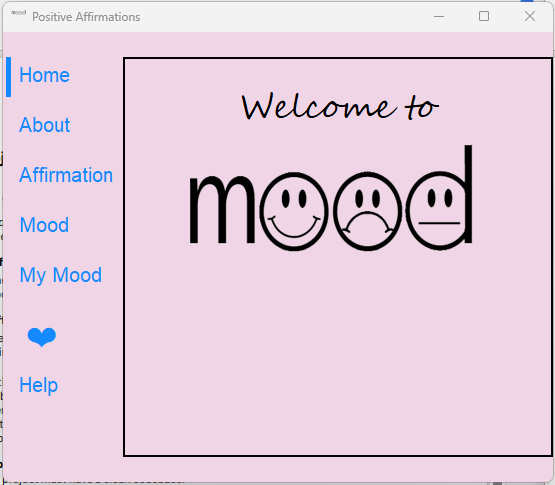
3rd version: Our next version was really fun and we decided to get really creative and experiment. We added cloud shaped buttons, changed our layout slightly and played with colours. Harmony managed to adjust the header. We were extremely happy with our creation. At this point we had met all our deadlines and realised we had some extra time to experiment a little bit more. Despite our app functioning, the layout had some room for improvement. Some buttons were unnecessary and we agreed that it didn’t make sense with all the items being on the same page. Code used at this point: *B-code\_updated.py*

However, with extra time Layefa suggested a toggle system for better user navigation.

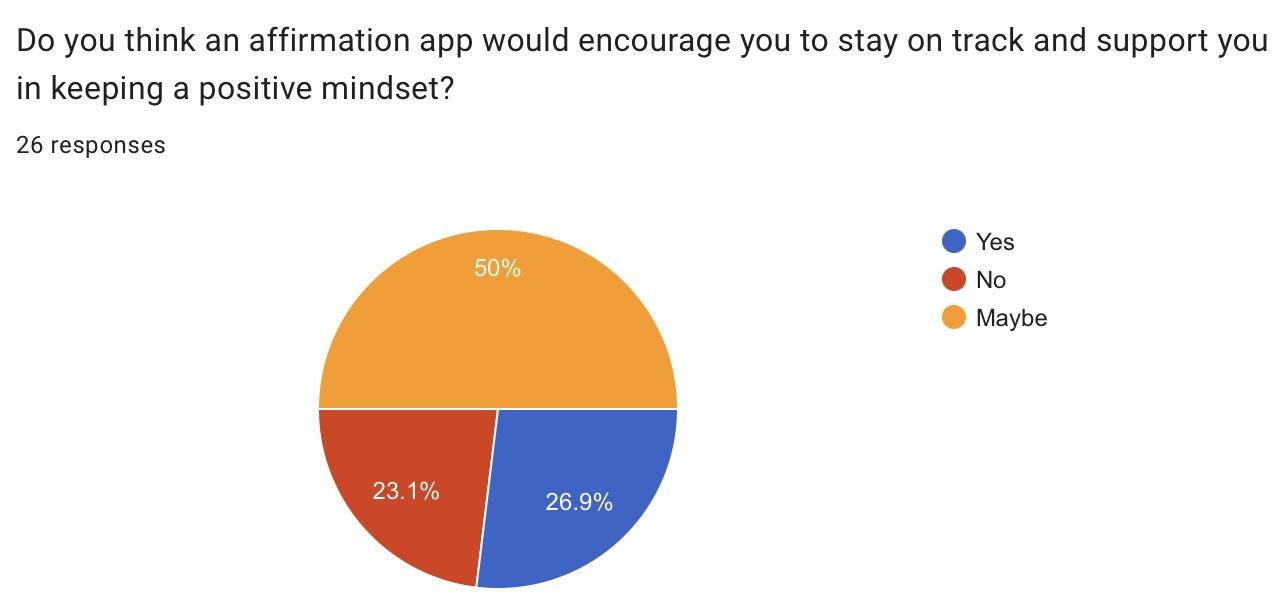
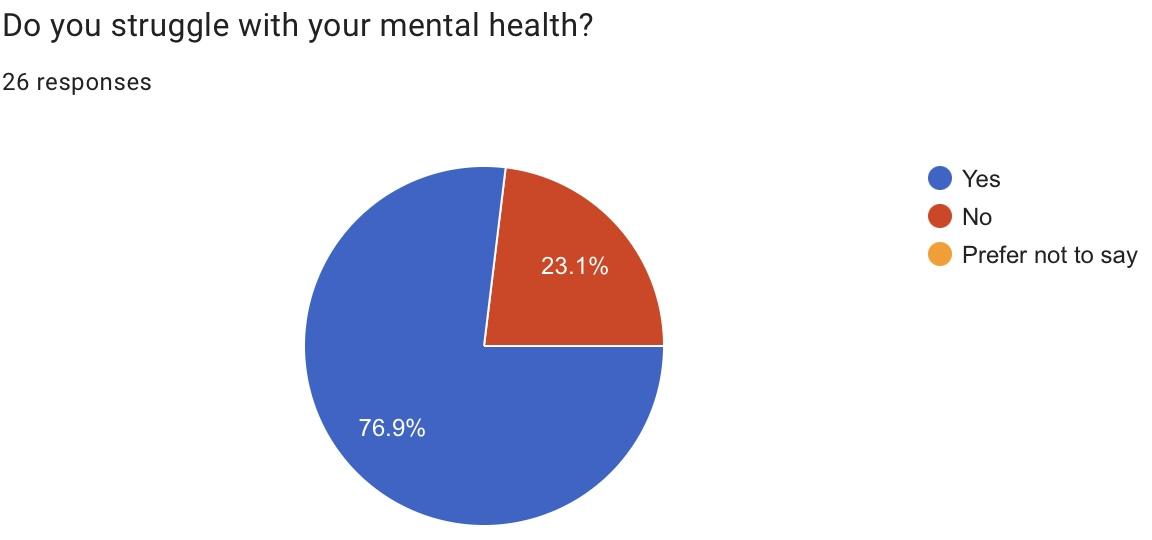
Everyone in the group contributed to some of the code and those that could review code did so. On occasions code didn't seem to work and so we teamed up to comb through the code and find out why things weren't working. On one occasion Harmony couldn't figure out why her code wasn't working. Someone else managed to spot that the file that was being imported was empty for an unknown reason. Harmony copied the correct code back into the file and it worked again.

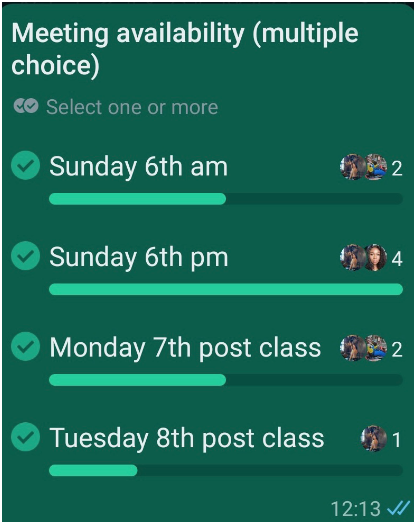
We coded together using shared screens when we're stuck, otherwise we got on with code independently and uploaded our updated versions to GitHub so that we could all check that the code is running on all of our devices meaning it will work when shared with our instructor. We also created a google doc so that members could keep updated as to what has been done/ discussed and what still needs to be done.

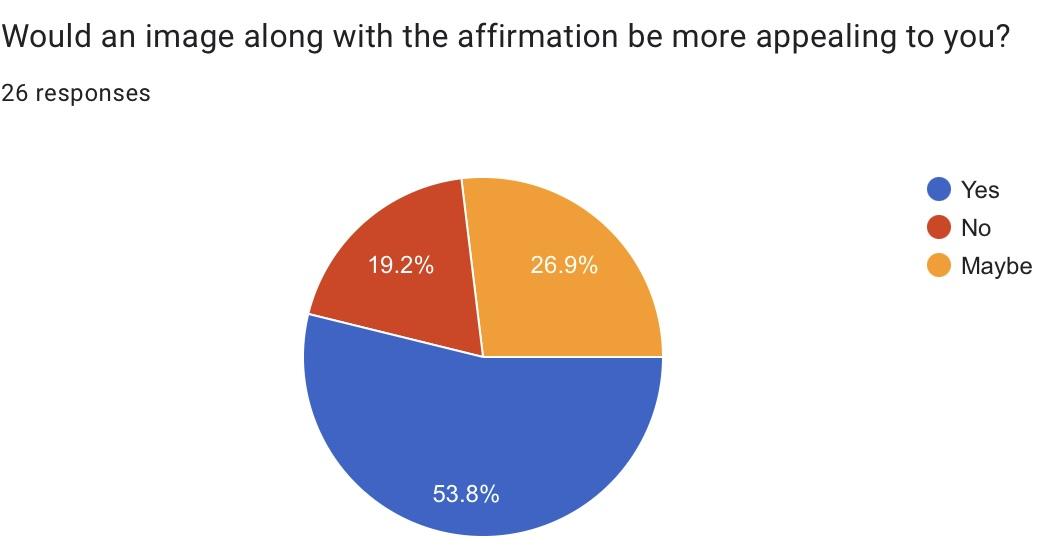
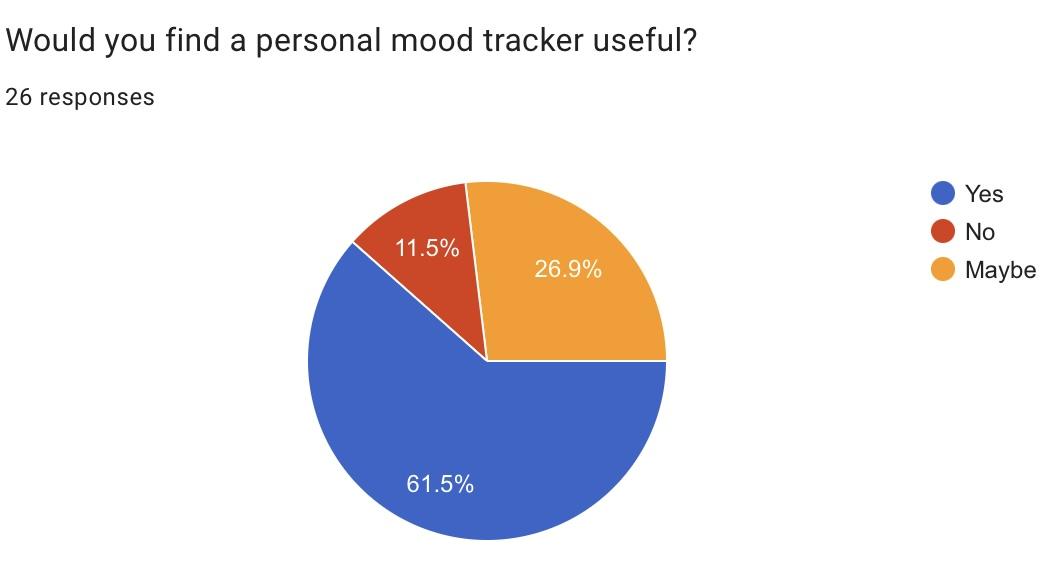
We feedback continually but have also received valuable feedback from others as displayed further into this document. The feedback has influenced our design and the function of being able to review affirmations rather than just generate them and record mood. Feedback has also allowed us to reflect on the things we would love to do to make our app even better but on this occasion we have been restricted with time

In an initial meeting we had to understand our strengths and weaknesses so that we could utilise our strengths accordingly. Layefa and Bobbie showed their creativity by designing the mood icon and cloud headers. Harmony showed her love of design through the final versions layout:

**Development approach** We carried out surveys to get user insight and understand how individuals deal with mental health and which tools users would appreciate within an affirmations app. As a result, we were influenced by the feedback - thus the introduction of a mood button and tracker. We could see in our survey that 76.9% of users either found this appealing or thought that it could make it more appealing.







**Tools and Libraries**

We worked with tkinter and so many of the libraries we accessed were within that. Tkinter allowed us to build our app structure. MySQL was used as our database to store information, ’datetime’ to accompany data recording, ‘webbrowser’ for URL access, PIL to import our images produced in house on pro create and photoshop

**Challenges:**

* Creating Classes. The issue was that we continued building on the initial code which was made, rather than using it as a reference to create a program with classes. We found that trying to create classes around a functioning program was difficult. This created a lot of errors in places that we had fixed throughout the development process.
* Limited quote supply from API due free membership, users will only receive 6 quotes a minute. This isn’t hugely damaging to our app however it would be nice if we had unlimited access. This could easily be solved by creating a membership
* Connecting to SQL: Layefa had a problem with the SQL workbench as it seemed the password she initially had was no longer working. Trying to fix this took a lot of time due to the restriction of time the baton was passed over to Poppy who was able to rectify the issue.
* To find a time when we were all available. We were able to meet once a week and discuss our findings and progress. This was useful but it was challenging to only meet once a week because of the time restriction. We made the best of the situation but it was very challenging and worrying. We got past this by also meeting up mid week in smaller groups
* Not knowing one another. As this was the first time we were working together as a group we shared a fear that it could be difficult to communicate or work together. However we all got on superb and everyone pulled their weight
* Despite worries that ideas would clash, a challenge for us was picking an idea because we were all very supportive and loved all the ideas. To solve this issue, we set up a voting system (similar to the one above which was made for planning meetings) which allowed us to come to a verdict swiftly
* Complications: It was important for the save button to be in the same file as the database\_manager which connects to SQL so that when importing one file to another it would connect easily. In fact, overall it became complicated when sharing code on Github because there were multiple versions of code with very specific files needed to run each piece of code. Although this is nothing we are not used to, it did take some adjustment working with other team members' code and external files. To help get on top of this we created ReadMe pages and also shared screens in meetings so that one person could go through the code and the rest could watch.
* We wanted to produce a downloadable app that anyone could access but soon realised 1) a mobile app was not possible using purely tkinter 2) even a computerised app seemed ambitious after further research and mentor feedback. It seemed as though kivy would have been a more suitable working environment if we wanted to create a phone app. Overall, we were happy with our decision sticking with tkinter.
* As insightful as user feedback was, it made it quite challenging moving forward because 1) we wanted to make all our users happy 2) we got so many ideas that it made it difficult to not get carried away with the features we wanted to make a part of our app. We had to regroup, reflect on our time limitations and plan a basic design which we were then able to build upon.

**TESTING AND EVALUATION:** Print statements to make sure the database was connected to the program. Printing one function at a time also allowed us to make sure everything was being utilised.

**CONCLUSION:** Every single one of us played a vital role in delivering a successful app. Our team had wonderful communication, collectively checking in on one another and our project progress. Our timely efforts and target setting meant we naturally had an iterative approach. We were able to build on a very simple idea, producing something more complex. As for future improvement, to enhance user accessibility we would incorporate a navigation icon system. Additionally, with more time and careful planning, we would have created a function whereby users could share their saved quotes across social platforms. This would allow our app to stay current, allowing us to be visible and reach a larger audience. To improve our code we would implement classes more thoroughly throughout our code along with more unit tests. We would also create a backup database of affirmations in case our API went down so that users could still benefit from using the app.