Team number: 24
Project Title: MindSpace

Milestone 1	Activities	Planned Outputs	Achieved Outputs
Restate the milestone from your Draft plan.	Restate the key activities from your draft plan.	Restate the planned outputs from your draft work plan.	Outline the actual outputs compared to what was projected (or type “same as planned”)
This milestone will create a prototype and demo of MindSpace application. By this milestone, the application shall finish the prototype design. Implement a simple version of frontend of ‘Landing page’, ‘Permission’ and ‘Home’ and full functionalities of ‘Login’, ‘Sign up’ page. The application shall be implemented to utilise be able to display all output emotion analyse results on a single page with client’s sample data.	Create empty pages with functioning routing and navigation	A demo mobile app with ‘landing’, ‘login’, ‘signup’, ‘permission alert’, ‘home’, ‘explore’, ‘insights’, ‘resources and ‘timeline’ pages with empty content and buttons with navigation working. Buttons on the ‘landing’ page will trigger navigation to ‘login’ and ‘signup’ pages. A Button on ‘login’ and ‘signup’ page will trigger navigation to the ‘permission alert’ page. A Button on the ‘permission alert’ page will trigger navigation to the ‘home’ page. A Button on the ‘explore’ page will trigger navigation to the ‘resources’ page. There will be tabs at the bottom of screen for ‘home’, ‘explore’, ‘insights’, and ‘timeline’ pages for shifting view from one to another.	Same as planned.
	Create the ‘Landing Page’ as the first screen after opening the application	A Landing Page screen with a login button and a signup button so that users will be redirected to the ‘Login’ screen after tapping the login button, and will be redirected to the ‘Create your Account’ screen after tapping the ‘Sign Up’ button	Same as planned.
	Implement the ‘Login’ and ‘Sign up’ screen	A ‘Login’ screen and ‘sign-up’ screen including a Google auth button, a Facebook auth button, so that users can later login with Google authentication, Facebook authentication. Users will be redirected the ‘Explore’ screen after authentication. A form handling email-password authentication that allows users to create new account with email and set up password, or email-password authentication.	Email auth and Google auth are finished. Facebook auth is cancelled and removed from the project scope. Added email verification function after signing up with email. Added a dashboard page to display user information including username profile after successfully login. Added Demo user login button for debugging in development stage.
	Frontend design of ‘Permission’ alert page	A page for requesting permission to access users’ keyboard inputs data. The permission functionalities will not be implemented for this milestone.	The alert function is finished but not triggered in demo in this stage.

	Implement the display and frontend functionalities of the 'Home' screen using sample data provided by clients	A page that displays client's NLP's output sample data into Map and List. Map is a component representing the Top 5 emotions in different colours and sizes based on their frequency. Below the Map, a list will be used to show all other identified emotions with frequency number in a day.	Same as planned.
	[new] Setting up backend server	No planned output since it's a new activity.	Created DockerFile, docker-compose.yml and that allows Docker to create a Docker container of python-alpine image with Django server, MySQL database and other dependencies installed and running. Also, a "super admin" user has been set up for accessing database with Django admin page.
	[new] Implement the backend endpoint '/api/token' that allows client to exchange information received from Google authentication server with a token key for authentication with backend server	No planned output since it is a new activity.	Created the endpoint "/api/token" of the backend server which accepts http POST request with payload of a Google user id and a token key received from the Google authentication server. The backend server returns another token key for authentication only when the information in the payload is independently verified by the Google authentication server.
	[new] Implement to import the emotion record data provided by the client into the backend server database	No planned output since it is a new activity.	All emotion record data provided by the client can be imported into the MySQL server by running the command: "python manage.py import data"
	[new] Implement the backend endpoint '/api/record/emotions' for retrieving emotional record data from the backend server	No planned output since it is a new activity.	All emotion record data of an authenticated user can be retrieved by making a http GET request to the end point "/api/record/emotions" with Bearer token in the Authorization header.

Team reflection on progress	Provide some comments below regarding the completion of this milestone specifically around: 1. How is the project progressing? 2. Are there any differences between projected and actual outputs/outcomes?

1. Project progress

Our project progressed smoothly and delivered the result as planned in milestone 1 plan except some minor changes due to client's new requirements. From week 1 to week 5, we designed the interface and layout of the application and communicate on client's requirement of the first demo application. From week 6 to week 7 (included a 2-week mid-semester break), we finished the front-end development of the homepage, the basic structure of application, demo backend server with Django and Google authorization test with Firebase. Below is how we progressed and managed the project.

- Client communication - we had weekly client meetings in order to have a well understanding of client's requirement.
- Project management - weekly internal group meetings were held for progress checking, review and problem-solving. As for normal communication, we use Slack for a more efficient communication on each task since Slack allows us to have separate communication room for each task.
- Learning how to use the client's required frameworks - we spent much effort in learning Ionic framework with Angular JS, Django, and Firebase through Udemy courses and online courses.

2. Differences between projected and actual outputs

Overall, we have achieved activities as we planned except the function of login with Facebook.

After we developed the function of login and signup with Google using Firebase and Django as our clients requested, we decided to cancel the implementation of login and signup with Facebook. Since we found that it requires copious time and effort to implement it with Facebook, and we have already put a lot of efforts and time to learn and implement the Google authorization with Firebase and Django server built with docker for database, we would be behind schedule if we insist on implementing the login with Facebook. Therefore, the Facebook auth function is removed from the project scope.

After experiencing a tight schedule and heavy workload in milestone 1, we have learned to be more prudent for the estimation of workload for the next milestone, and to review our project scope to evaluate if there is any unnecessary functionality, so that we would have a more feasible plan for final milestone and the whole project.

Team reflection on managing problems	Have you encountered any problems to date? If so, how have you managed them?

Problem 1 - Deep learning curve

Docker, Django, Angular, Ionic framework, and Firebase are new to my team. We need to learn everything very fast in order to complete the tasks listed in milestone plan 1.

We were managed to learn the technology framework in a short time by following many useful online resources and courses on Udemy. Courses on Udemy provide many hands-on examples for novices like us to quickly write some codes for build mobile apps and Django backend server.

Problem 2 - Dependency problem encountered when setting up the backend server

When building a Docker image with dependency for our backend server, the standard python-alpine environment does not support installing the grpcio tool required for installing the firebase-admin tool used in our Django server.

We were managed to solve this problem by finding another Docker image (basisai/python-alpine-grpcio) available in Docker hub, so that we do not need to install it during the installation process of the firebase-admin tool.

Problem 3 - Failed to update UI with data retrieved from the backend server

A particular problem we encountered in ionic app development is that we fail to get the UI update with data retrieved from backend server. It is because the http request method provided in ionic-angular package return a JavaScript Observable object, which by default is a synchronous process and can lead to error when the subsequent line of code is executed before the http request finished running.

To solve this problem, we need to learn the mechanism of Observable, and use the BehaviorSubject class provided in RxJS package to store the data retrieved from the database, which can be converted into an Observable object to be subscribed by other parts of the program. When putting the subsequent lines of code into the call back function of the returned Observable object, it can ensure those code is executed only after the data is updated after each http request, which can in turn solve the problem of failing to update UI with data retrieved from the backend server.

Supervisor assessment	Please, rate your team (1) effort, (2) project progress and (3) their self-reflection for milestone 1 Rating scale 1-10 as per standard marking scheme, i.e. 5 is a Pass and 7 is a credit. Add some comments to explain your rating