Computer programming

1. Final task assignment

The task consists in designing the Mobile software for memorizing words.

2. Allocated time: 6 hours

The contest duration is 6 hours 00 minutes

3. Requirements

- Contestants should use all following technologies:
  - React Native (RN) in TypeScript (TS) language: https://reactnative.dev
  - Amplify for Backend part of the task: https://docs.amplify.aws
    - Authentication : https://bit.ly/3IFCLrJ
    - API (GraphQL) : https://bit.ly/3ErPpYH
  - These two framework tools must be used in conjunction to complete the programming.
- Contestants should use 16.19 version node
- Contestants should use all following node packages:
  - Styled-components (SC) for CSS style: https://styled-components.com
  - React Navigation (RNR) for routing screens: https://reactnavigation.org
  - These two node packages must be used in conjunction to complete the programming.
- Contestants should make this functions
  - **Model of word data should have name and meaning properties.** Contestants can add additional properties. Also contestants add other data models as they want.
  - **Add word**
  - **Modify word**
  - **Remove word**
  - **View the word list**

- **Study the words**
- **Mark some word as memorized or not**
- **Check score of memorizing words**
- **Authentication**
    - Sign up
    - Sign in
    - Sign out
    - Withdrawal membership

- The app should manage the data in DynamoDB by Amplify API (GraphQL)
- The data should depend on the user. So the user can manage only their own data.
- Contestants should make screens for the functions.
- Contestants should not bring any kind of storage devices like USB memory or etc.
- The following programs will be installed on every computer.
    - Visual Studio Code (VS code)
        - For the text editing part of the task. Contestants should code by it.
        - Contestants should not sign in their accounts in VS code.
    - iterm2
        - Default terminal program.
        - Zsh is default shell.
        - Contestants should use iterm2 to do some terminal jobs for RN.
    - Android Studio
        - To run Android emulator.
    - Xcode
        - To run iOS simulation.
    - Contestants should not use other programs on their pc.
- Contestants can use some of the following VS code extensions.
    - Extensions
        - JavaScript and TypeScript Nightly
        - Prettier - Code formatter
        - vscode-styled-components

        - Contestants can ask for more extensions before the contest. But they can use them only if the chief judge allows it. And other contestants also can use the same extensions if the chief judge allows it.
- Contestants can use some of the following config files if they want.
    - Config files
        - .editorconfig
        - .eslintignore
        - .eslintrc.js
        - .prettierrc.js
        - .stylelintrc
        - tsconfig.json
    - Contestants should submit the files before the contest if they want.
- Contestants should submit a node package list that they will use to develop the app.
    - Contestants can't add some node packages in the middle of the contest.
    - Contestants have to separate dev packages from normal packages.

```
e.g)

[normal packages]
- styled-components
- react-native-gesture-handler
...

[dev packages]
- eslint
- prettier
...
```

- Contestants should submit Appsync GraphQL schema.
    - Contestants can't update the schema in the middle of the contest.

```
e.g)
```

```
# path: amplify/backend/api/[PROJECT NAME]/schema.graphql


type Word
{
   name: String!
   meaning: String!
}
```

- RN project folder will be prepared on every computer.
  - The project will be created by following command.

    ```
    npx react-native init [PROJECT NAME] --template react-native-template-typescript
    ```
  - The config files that contestants submit will be added.
  - The node package list that contestants submit will be added.
  - Default amplify configuration will be prepared. So contestants don't need to input their aws account information.
  - Amplify Auth and API(GraphQL) resources will be prepared.
  - Contestants should code only in the **src** folder. Default **App.tsx** file will be moved into the **src** folder,
  - Contestants can make sub directories and files in the **src** folder.
- Amplify Auth will be prepared like the following configuration.
  - **Default configuration**
  - Users can login by **Username**
  - **No advanced settings**
- Amplify API will be prepared like the following configuration.
  - **GraphQL**
  - Authorization modes: **Amazon Cognito User Pool**
  - **No additional auth types**
  - **schema.graphql** : Contestants should submit this file
  - GraphQL API files will be prepared in TS
    - src/graphql/**/*.ts, src/API.ts
    - for all possible GraphQL operations

        - depth 2
- Contestants can't use the internet during the contest.
- Contestants can use iOS simulation or Android emulator to test. But they have to build an android APK for the result.
    - Android signing configuration will be prepared to build APK.
- All Macs will be equipped with AZERTY keyboards. However, contestants may install their own keyboard before the beginning of the competition.
- The jury will collect all electronic means of communication contestants may have at the beginning of each module. They will be returned to their owners once each module is finished.
- The juries except the chief judge also can't use all electronic means of communication.
- Translators should not teach or coach their contestants for cheating.
- Any contestant caught cheating, talking to someone from the public or using a communication device will suffer a penalty of 5 points for the first transgression. A second transgression will lead to an exclusion from the contest.
- Contestants should keep their project files on their pc after the contest finishes. The juries will judge on each their pc.
- Contestants should not modify their project files after the contest.

4. Procedure

Day -1 (March 23rd): The day before the contest, contestants will be met by the judges at the competition stand. There will be a briefing on the organization of the competition and the safety rules. Due to the large number of registered contestants, the competition will be organized in two groups. Contestants from a same delegation will inevitably be placed in the same group. Each contestant will receive a schedule with the detailed working hours. Contestants will draw lots to be assigned to a work place where they may drop off their tools. The jury will not take account of any problem coming from the contestants' personal equipment.

Composition of the groups

Group 1: 4 contestants	**Group 2: 3 contestants**
China, 1 contestant	Japan, 2 contestants
Azerbajdan, 1 contestant	Korea, 1 contestant
Ecuador, 1 contestant	
Chinese Taipei, 1 contestant	

Day 1 (March 24th): Group n°1 will have 6 hours to complete the task assignment. Group n°2 will go on an organized trip.

Day 2 (March 25th): Group n°2 will have 6 hours to complete the task assignment. Group n°1 will go on an organized trip

5. Scoring criteria

N°	Items to be evaluated	O/S	Scoring scale
01	(Auth) Can sign in on some screen?	O	4
02	(Auth) Can sign out on some screen?	O	4
03	(Auth) Can sign up on some screen?	O	4
04	(Auth) Can withdraw membership on some screen?	O	4
05	(Auth) Has custom UI for authentication with all auth flows(sign in, sign out, sign up, withdrawal)?	S	2
06	(API) Data depends on the user?	O	2
07	(API) Can add new word on some screen without bugs?	O	4
08	(API) Can modify some word on some screen without bugs?	O	4
09	(API) Can remove some word on some screen without bugs?	O	4
10	(API) Can see the word list on some screen without bugs?	O	4
11	(API) Can study the words on some screen without bugs?	O	4
12	(API) Can mark some word as memorized or not on some screen without bugs?	O	4
13	(API) Can check score of memorizing words on some screen without bugs?	O	4
14	(RN) Used useContext without bugs suitably?	S	3
15	(RN) Used useState without bugs suitably?	S	3

16	(RN) Used useRef without bugs suitably?	S	3
17	(RN) Used useMemo without bugs and dependency problems?	S	3
18	(RN) Used useCallback without bugs and dependency problems?	S	3
19	(RN) Used useEffect without bugs and dependency problems?	S	3
20	(SC) Used styled-components without bugs?	S	3
21	(SC) Used theming of styled-components suitably?	S	3
22	(RNR) Used routing for screens without bugs?	S	3
23	(RNR) Used ParamList type for screens suitably?	S	3
24	(RNR) Used passing parameters to routes without bugs?	S	3
25	Optimization of the coding - measure score by code lines of files in src folder except amplify files. - It will be 0 if the contestant doesn't finish it - The less lines they use, the higher they get the score.	S	5
26	Optimization of the coding (no redundancy in the code)	S	5
27	Has additional good features?	S	4
28	General aesthetic aspect of the mobile application	S	5
TOTAL POINTS			**100**